**k8s部署之打包发布镜像**

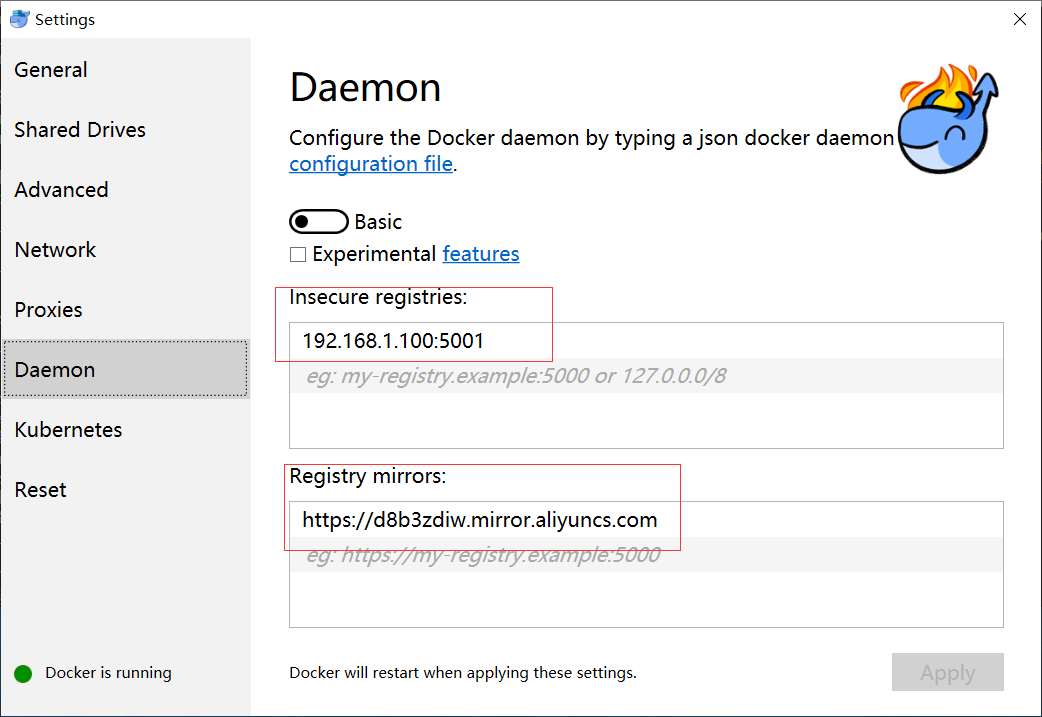
**前言**

以下操作在Node节点上进行

**1、下载Docker for Windows**

地址：https://www.docker.com/products/docker-desktop

**2、本地安装Docker for Windows并配置**



Insercure registries:私有库

Registry mirrors:镜像加速

**3、创建测试项目**

使用vs新建项目**TestWebApi**，并【添加Docker支持】生成Dockerfile文件,另外新建批处理文件build-push-dev.bat，两个文件内容如下：

【**Dockerfile**】

FROM mcr.microsoft.com/dotnet/core/aspnet:3.0-buster-slim AS base

WORKDIR /app

EXPOSE 80

EXPOSE 443

FROM mcr.microsoft.com/dotnet/core/sdk:3.0-buster AS build

WORKDIR /src

COPY ["TestWebApi/TestWebApi.csproj", "TestWebApi/"]

RUN dotnet restore "TestWebApi/TestWebApi.csproj"

COPY . .

WORKDIR "/src/TestWebApi"

RUN dotnet build "TestWebApi.csproj" -c Release -o /app/build

FROM build AS publish

RUN dotnet publish "TestWebApi.csproj" -c Release -o /app/publish

FROM base AS final

WORKDIR /app

COPY --from=publish /app/publish .

ENTRYPOINT ["dotnet", "TestWebApi.dll"]

【**build-push-dev.bat**】

for /f "tokens=1,2,3 delims=/- " %%a in ("%date%") do @set D=%%a%%c%%b

for /f "tokens=1,2,3 delims=:." %%a in ("%time%") do @set T=%%a%%b%%c

set T=%T: =0%

set currenttime=%D%%T%

set imagetag=testwebapi:%currenttime%

set dockerserver=192.168.1.100:5001

set projectname=dev

set image=%dockerserver%/%projectname%/%imagetag%

docker build --rm -t %imagetag% -f Dockerfile ..

docker tag %imagetag% %image%

docker login -u=dev -p=Dev@123456 %dockerserver%

docker push %image%

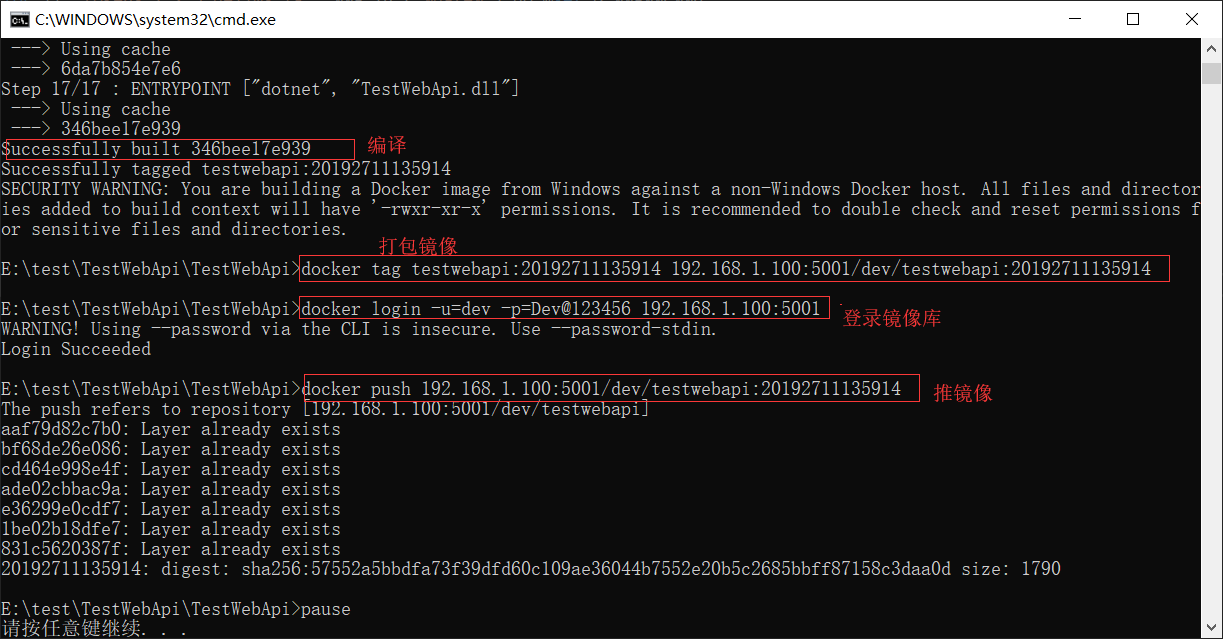
pause

projectname：上到镜像库的项目名dev,需要事先创建好，并设置账号

dockerserver：私有镜像库Harbor地址

docker login：这里设置Harbor登录账号

执行批处文件如下：



登录到harbor镜像库，可以看到已经成功将本地镜像打包并推送到私有库：



**4、创建Secret**

参考：<https://www.kubernetes.org.cn/secret>

用kubectl命令来创建用于docker registry认证的secret：

kubectl create secret docker-registry harbor.registry --docker-server=192.168.1.100:5001 --docker-username=dev --docker-password=Dev@123456 --docker-email=yangliangbin@hyihk.com



登录registry

docker login 192.168.1.100:5001

Username: dev

Password:

Error response from daemon: Get https://192.168.1.100:5001/v2/: http: server gave HTTP response to HTTPS client

出现这个错误，原因是客户端采用https，docker registry未采用https服务所致。处理如下：

在”/etc/docker/“目录下，创建”daemon.json“文件。在文件中写入：

{

    "insecure-registries": [

        "192.168.1.100:5001"

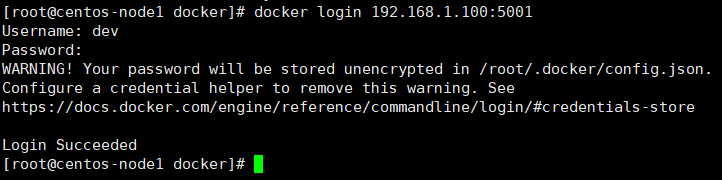
    ]

}

systemctl daemon-reload #守护进程重启

systemctl restart docker #重启docker服务

重启登录



**5、定义并创建Pod**

定义test-webapi.yaml文件,为了方便测试这里定义hostPort: 8001访问pod，规范的部署流程是定义service方式访问Pod

apiVersion: v1

kind: Pod

metadata:

  name: test-webapi

  labels:

    app: test-webapi

spec:

  containers:

    - name: testwebapi

      image: dev/testwebapi

      ports:

        - containerPort: 80

hostPort: 8001

  imagePullSecrets:

    - name: harbor.registry

执行创建pod

kubectl create -f test-webapi.yaml



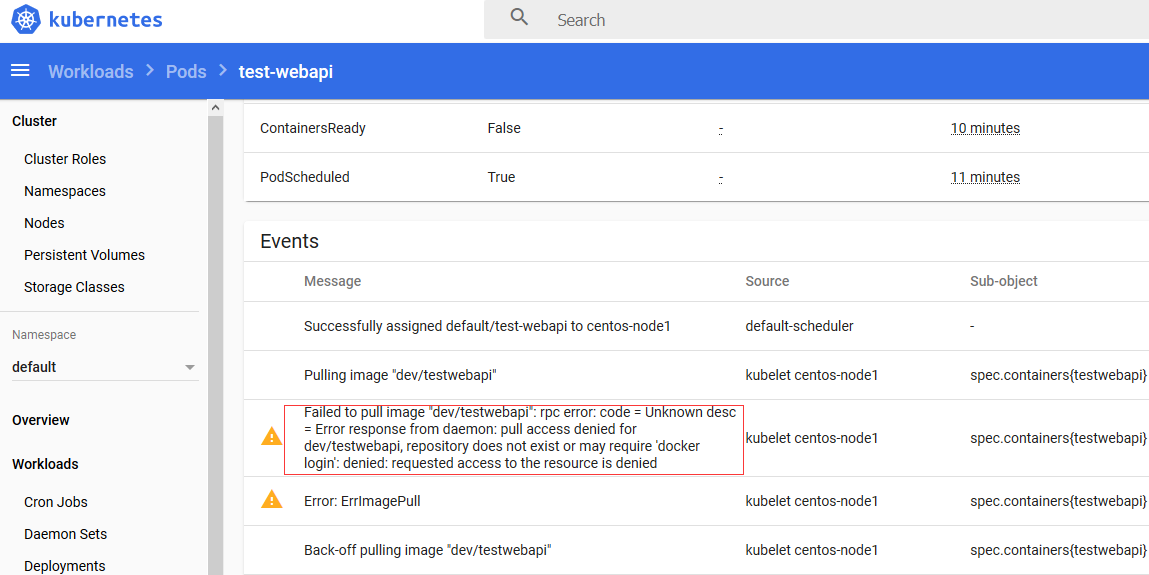
查看pod

kubectl get pods | grep test-webapi



pod状态为ImagePullBackOff,镜像拉取失败了，看一下日志，发现是找不到镜像，检查到镜像名搞错了

**dev/testwebapi** 改为 **192.168.1.100:5001/dev/testwebapi:20192711135914**



修改后文件如下：

apiVersion: v1

kind: Pod

metadata:

  name: test-webapi

  labels:

    app: test-webapi

spec:

  containers:

    - name: testwebapi

      image: 192.168.1.100：5001/dev/testwebapi:20192711135914

      ports:

        - containerPort: 8001

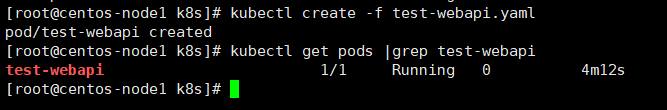
  imagePullSecrets:

    - name: harbor.registry

删除并重新创建Pod

kubectl delete pod test-webapi

kubectl get pods | grep test-webapi



最后，我们访问一下 test

#内部访问

curl 127.0.0.1:8001/WeatherForecast

#外部访问：http://192.168.1.101:8001/WeatherForecast

