微服务中，网关会把流量分配给每个Pod节点上

\1. 如果我们直接将Pod杀死，那这部分流量就无法得到正确处理，会影响部分用户，通常来说网关或者注册中心会将我们的服务保持一个心跳，过了心跳超时之后会自动摘除我们的服务，但是有一个问题就是超时时间可能是30秒也可能是60秒，虽然不会影响我们的系统，但是会产生用户轻微抖动。

\2. 如果我们在停止前执行一条命令，通知网关或者注册中心这台主机进行下线，那么注册中心就会标记这台主机已经下线，不进行流量转发，用户就不会有任何影响，这就是优雅停止，将滚动更新影响最小化

Pod Hook
Pod Hook是由kubelet发起的，当容器中的进程启动前或者容器中的进程终止之前运行，这是包含在容器的生命周期之中。我们可以同时为Pod中的所有容器都配置hook

在k8s中，理想的状态是pod优雅释放，并产生新的Pod。但是并不是每一个Pod都会这么顺利

```
Pod卡死，处理不了优雅退出的命令或者操作
优雅退出的逻辑有BUG，陷入死循环
代码问题，导致执行的命令没有效果
```

对于以上问题，k8s的Pod终止流程中还有一个"最多可以容忍的时间"，即grace period (在pod的.spec.terminationGracePeriodSeconds字段定义)，这个值默认是30秒，当我们执行kubectl delete的时候也可以通过--grace-period参数显示指定一个优雅退出时间来覆盖Pod中的配置，如果我们配置的grace period超过时间之后，k8s就只能选择强制kill Pod

Kubernetes终止生命周期

```
1 - K8S 启动新POD。2 - K8S等待新POD进入Ready(Running) 状态。3 - K8S创建Endpoint。此时，k8s创建endpoint，将新服务纳入负载均衡。4 - 用户删除pod，Pod设置为”Terminating”状态，并从所有服务的Endpoints列表中删除。此时，Pod停止获得新的流量。但在Pod中运行的容器不会受到影响。5 - preStop Hook被执行。 preStop Hook是一个发送到Pod中的容器特殊命令或Http请求。6 - SIGTERM信号被发送到Pod。 此时，Kubernetes将向pod中的容器发送SIGTERM信号。这个信号让容器知道它们很快就会关闭。7 - Kubernetes等待优雅的终止 此时，Kubernetes等待指定的时间称为优雅终止宽限期。默认情况下，这是30秒。值得注意的是，这与preStop Hook和SIGTERM信号并行发生。Kubernetes不会等待preStop Hook完成。
```

基于PreStop环境演示

在生产环境中使用spring框架，由于服务更新过程中，服务容器被直接充值，部分请求仍被分发到终止的容器(没有配置钩子，熟悉默认环境)，导致服务出现500错误，这部分错误请求数据占用比较少，因为Pod滚动更新都是一对一。因为部分用户会产生服务器错误的情况，考虑使用优雅的终止方式，将错误请求降到最低，直至滚动更新不影响用户

Eureka是一个基于REST的服务，作为Spring Cloud服务注册中心，用于定位服务来进行中间层服务器的负载均衡和故障转移。各服务启动时，会向Eureka Server注册自己的信息(IP、端口、服务信息等)，Eureka Server会存储这些信息，微服务启动后，会周期性(默认30秒)的向Eureka Server发送心跳以续约自己的租期，并且可以从eureka中获取其他微服务的地址信息，执行相关逻辑。。

由于Eureka默认的心跳检测为30秒，当K8S下线Pod时Eureka会有30秒的异常问题，所以我们需要在Pod 停止前发送一条请求，通知Eureka进行下线操作，这样进行优雅的停止对用户的影响做到最小

```
apiVersion: v1
kind: Pod
metadata:
  name: abcdocker
  labels:
    name: abcdocker
spec:
  containers:
  - name: abcdocker
    image: nginx
    ports:
      - containerPort: 80
    lifecycle:
      preStop:
        exec:
          command:
            - bash
            - -c
            - 'curl -X POST --data DOWN http://127.0.0.1:8080/service-
registry/instance-status  -H
              "Content-Type: application/vnd.spring-
boot.actuator.v2+json;charset=UTF-8";sleep 30'

####### 参数解释
127.0.0.1:8080 #代表eureka地址
service-registry    #代表注册中心
DOWN        #执行down请求
sleep       #等待30秒
```

当我们删除Pod的时候就会执行上面的命令操作，并且等待30秒

```
[root@yzsjhl82-135 yaml]# kubectl get pod
NAME        READY   STATUS    RESTARTS   AGE
abcdocker   1/1     Running   0          2m16s
[root@yzsjhl82-135 yaml]# kubectl delete pod abcdocker
pod "abcdocker" deleted

#此刻Pod不会马上删除，而是执行Exec中的命令，并等待30秒
```

配置中添加了一个sleep时间，主要是作为服务停止的缓冲时间。

nacos心跳检测时间

Nacos 目前支持临时实例使用心跳上报方式维持活性，发送心跳的周期默认是 5 秒，Nacos 服务端会在 15 秒没收到心跳后将实例设置为不健康，

在 30 秒没收到心跳时将这个临时实例摘除。这里要注意30秒这个时间。

nacos下线应用地址举例：

```
http://192.168.0.218:8848/nacos/v1/ns/instance?serviceName=jdd-parking-cloud-admin&clusterName=DEFAULT&
\groupName=DEFAULT_GROUP&ip=172.16.246.32&port=8093&ephemeral=true&weight=1&enabled=false&namespaceId=a9076f8c-a1c7-474c-9ea4-1112677d9af7
```

说明：

- 192.168.0.218:8848 nacos注册地址
- jdd-parking-cloud-admin 注册的应用名称
- 172.16.246.32 注册的应用名称所在主机地址
- 8093 注册的应用名称使用的端口号
- enabled=false 下线，enabled=true 上线
- namespaceId 命令空间，默认使用public命名空间则不写这个

通过分析nacos下线应用地址，需要如下参数：nacos注册地址，应用名称，应用所在主机ip,应用端口号,命令空间(public不需要)

考虑到应用所在主机ip是pod ip,这个需要从pod容器中获取，因此，不能在PreStop中使用命令行的形式，也就是如下的形式

```
curl -x PUT http://192.168.0.218:8848/nacos/v1/ns/instance?serviceName=jdd-parking-cloud-admin&
\clusterName=DEFAULT&groupName=DEFAULT_GROUP&ip=172.16.246.32&port=8093&ephemeral=true& \weight=1&enabled=false&namespaceId=a9076f8c-a1c7-474c-9ea4-1112677d9af7
```

原因：nacos地址可以写死，应用名称可以写死，应用端口号可以写死，但是应用所在主机ip也就是pod ip没法获取。
1.PreStop是配置在Daeployment中的，pod的数量和ip都是不固定的。
2.就算把pod ip设置成环境变量的形式，也只能是在pod容器中使用，在PreStop中还是获取不到pod ip

综合以上分析，这里采取的办法是在构造镜像的时候入手，新增一个preStop.sh脚本，内容写上nacos下线的那个命令, 然后载PreStop命令行中执行这个脚本文件。

在这个过程中，若是有些参数值无法从环境变量中获取，则需要增加这些参数的环境变量。

preStop脚本内容

注意脚本中的sleep 45命令，这个是确保应用从nacos中下线使用的，默认是30秒，具体看开头nacos心跳检测时间，sleep设置时间大于30秒就可以，这里设置45秒
preStop.sh脚本中使用的变量有些是默认提供的，有些是需要提前设置环境变量的，取值是容器中的值

```
#!/bin/sh

# shell脚本作用
# 在更新pod时先执行这个脚本，把pod应用从nacos中下线，然后再关闭pod

#echo "输出必要的环境变量"
#echo "${NACOS_SERVER_ADDR}"
#echo "${PODNAME}"
#echo "${PODIP}"
#echo "${NACOS_NAMESPACE}"

result=$(curl -X PUT "http://${NACOS_SERVER_ADDR}/nacos/v1/ns/instance?
serviceName=${PODNAME}&clusterName=DEFAULT&groupName=DEFAULT_GROUP&ip=${PODIP}&p
ort=8093&enabled=false&namespaceId=${NACOS_NAMESPACE}")

echo "输出curl执行结果result:${result}"

if [ ${result} == "ok" ]; then
  echo "执行成功"
  sleep 45
  exit 0
else
  echo "执行失败"
  exit 1
fi
```

如上脚本中，
NACOS_SERVER_ADDR和NACOS_NAMESPACE从ConfigMap中设置中获取，
PODIP和PODNAME是在环境变量中手动设置的

然后修改Dockerfile文件，增加这个preStop脚本,设置可执行权限，注意脚本放置的路径，后面会用到

```
ADD preStop.sh /tmp/preStop.sh
RUN chmod 777 /tmp/preStop.sh
```

经过以上操作，项目中新增一个preStop脚本文件，把这个文件给添加到Dockerfile文件中，并放置到指定路径下，然后提交到gitlab,自动构建docker镜像,记住镜像标签。

然后在k8s中设置PreStop内容如下：

然后调整应用的deployment的yaml文件，把优雅终止宽限期(terminationGracePeriod限定时间)由默认的30秒调整为60秒，确保这个时间大于sleep 45的时间。

```
spec:
  containers:
  - name: nginx
    image: nginx
    ports:
    - name: http
      containerPort: 80
    lifecycle:
      preStop:
        exec:
          command: ['/bin/sh','-c','/tmp/preStop.sh']
```

同时更新应用使用的docker镜像，待镜像启动后，增加副本数，由1增加到3，同时观察nacos中注册的应用数，确认显示有3个。

然后缩减一个副本数，副本数由3变成2，注意观察nacos中的应用是否有一个ip状态的变成"上线"(显示这个表示应用是下线状态)，等待30秒后就看不到这个ip应用了。
然后观察k8s中pod的消失，等了60秒后才开始取消一个pod,最后查看事件events，发现并没有FailedPreStopHook，这是正常的，因为只有报错的情况下才会出现FailedPreStopHook，正常情况下不会出现这个。

Hook调用的日志没有暴露给Pod的Event，所以只能到通过describe命令来获取，如果是正常的操作是不会有event，如果有错误可以看到FailedPostStartHook和FailedPreStopHook这种event。并且如果Hook调用出现错误，则Pod状态不会是Running

总结

1.pod灭亡有个优雅终止宽限期(terminationGracePeriod限定时间)，默认是30秒，nacos中应用超过30秒则摘除，主要围绕这俩时间来进行处理
2.项目中新增一个preStop.sh脚本，并添加到Dockerfile文件中，确保构造的镜像中有这个sh文件
脚本内容是应用从nacos下面的命令，以及sleep时间，这个时间需要超过nacos默认的30秒 （pod镜像中确保有curl命令）
3.k8s中增加sh脚本中使用到的环境变量，以便pod中sh脚本可以从pod环境中获取这些变量的值
4.k8s中设置设置PreStop，使用命令行的方式执行如上的sh脚本
5.nacos中验证，事件events中验证

原文：https://www.cnblogs.com/fengjian2016/p/15205477.html