微服务引擎

快速入门

文档版本 01

发布日期 2024-08-30

华为云计算技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为云计算技术有限公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为云计算技术有限公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

华为云计算技术有限公司

地址： 贵州省贵安新区黔中大道交兴功路华为云数据中心 邮编：550029

网址： https://www.huaweicloud.com/

目 录

1 Spring Cloud 应用通过 Spring Cloud Huawei SDK 接入 ServiceComb 引擎

本章节通过一个demo演示如何将一个Spring Cloud应用通过Spring Cloud Huawei SDK接入ServiceComb引擎，指导您快速上手使用ServiceComb引擎。

1. **准备工作**

   您需要完成注册华为云并实名认证、为账户充值、为用户添加操作权限、创建VPC和子网、获取demo包及本地编译构建打包环境准备工作。

2. **创建ServiceComb引擎**

   在创建引擎时，您可以根据实际业务需要，选择合适的引擎规格、可用区和网络等。

3. **Spring Cloud接入ServiceComb引擎**

   本指导将演示使用一个provider服务和一个consumer服务接入ServiceComb引擎。

准备工作

1. 注册华为云并实名认证。

   如果您已有一个华为账户，请跳到下一个任务。如果您还没有华为账户，请参考以下步骤创建。

   a. 打开**华为云官网**，单击“注册”。
   b. 根据提示信息完成注册，详细操作请参见**注册华为账号并开通华为云**。

      注册成功后，系统会自动跳转至您的个人信息界面。
   c. 参考**实名认证**完成个人或企业账号实名认证。

2. 为账户充值。

   您需要确保账户有足够金额。

   - 关于ServiceComb引擎的价格，请参见**微服务引擎价格详情**。
   - 关于充值，请参见**如何给华为账户充值**。

3. 为用户添加操作权限。

   用户在创建依赖资源和ServiceComb引擎前，需要具备相应的操作权限。添加用户权限的操作，请参考**创建用户并授权使用微服务引擎**。

4. 创建VPC和子网。

ServiceComb引擎运行于虚拟私有云（VPC）中，并需要绑定具体的子网。创建ServiceComb引擎前，需保证有可用的虚拟私有云和子网。创建虚拟私有云和子网的方法，请参考**创建虚拟私有云和子网**。如果已有可用的VPC和子网，不需要再次创建。

5. 本地编译构建打包机器环境已安装了Java JDK、Maven，并且能够访问Maven中央库。
6. 下载github的**demo源码**到本地并解压。

说明

该demo的配置文件中已经完成了集成Spring Cloud Huawei的配置操作，若您需了解详细配置信息，请参考**Spring Cloud接入ServiceComb引擎**。

创建 ServiceComb 引擎

步骤1 进入**购买ServiceComb引擎专享版**页面。

步骤2 参考下表设置参数，参数前面带*号的是必须设置的参数。创建ServiceComb引擎所需参数的详细介绍请参考**创建ServiceComb引擎**。

表 1-1 创建 ServiceComb 引擎参数

参数	说明
*计费模式	选择计费方式，此处选择“按需计费”。按需计费是一种后付费模式，即先使用再付费，按照ServiceComb引擎实际使用时长计费。
*企业项目	选择ServiceComb引擎所在的企业项目。企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理。此处默认选择“default”。
*选择实例数	选择引擎规格，此处选择微服务实例数为100。
*引擎类型	引擎类型为集群，其为集群模式部署，主机级容灾。
*ServiceComb引擎名称	输入ServiceComb引擎的名称，名称以字母开头，由字母、数字和-组成，且不能以-结尾，长度为3~64个字符。例如：cse-test。
*可用区	可选择1个或3个可用区。此处选择1个可用区，可提供主机级别容灾能力。
*网络	选择已创建的虚拟私有云及子网，可在下拉框中搜索和选择合适的虚拟私有云和子网。
描述	单击  ，输入引擎描述信息，例如：体验快速创建ServiceComb引擎。
*安全认证	开启了“安全认证”的ServiceComb引擎专享版，通过微服务引擎控制台提供了基于RBAC（Role-Based Access Control，基于角色的访问控制）的系统管理功能。此处选择“关闭安全认证”，关闭安全认证功能后，可以在实例创建完成后再设置开启安全认证。

步骤3 单击“立即购买”，进入引擎信息确认界面。

步骤4 单击“提交”，等待引擎创建完毕。

说明

- 微服务引擎创建完成，大约需要31分钟。
- 微服务引擎创建成功后，“状态”为“可用”。查看微服务引擎状态，请参考**查看ServiceComb引擎信息**。
- 如果微服务引擎创建失败，可在操作日志页面上查看失败原因并处理后可进行以下操作：
  - 可在“微服务引擎信息”区域，单击“重试”重新创建。
  - 如果重试失败，可删除创建失败的微服务引擎，删除微服务引擎，请参考**删除ServiceComb引擎专享版**。

----结束

Spring Cloud 接入 ServiceComb 引擎

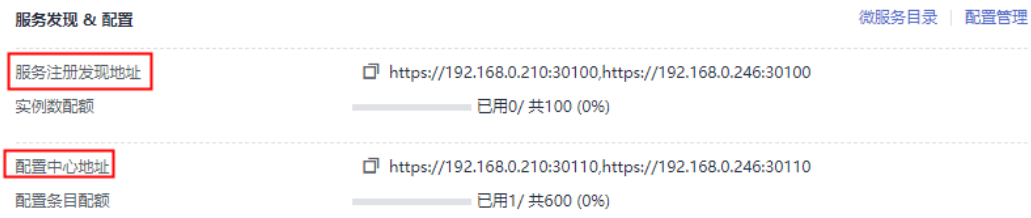
步骤1 登录**微服务引擎控制台**。

步骤2 在左侧导航栏选择“ServiceComb引擎专享版”。

步骤3 单击**创建ServiceComb引擎**中创建的ServiceComb引擎。

步骤4 获取ServiceComb引擎的注册中心地址和配置中心地址。

在“服务发现 & 配置”区域，查看获取引擎服务注册发现地址和配置中心地址。



步骤5 修改demo中的注册中心地址和配置中心地址。

1. 在下载到本地的demo源码目录下，分别找到“\basic\consumer\src\main\resources\bootstrap.yml”和“\basic\provider\src\main\resources\bootstrap.yml”文件。
2. 添加ServiceComb引擎的注册中心地址和配置中心地址到项目配置文件中（以“\basic\consumer\src\main\resources\bootstrap.yml”为例）。

```
spring:
  application:
    name: basic-consumer
  cloud:
    servicecomb:
      discovery:
        enabled: true
        watch: false
        # 注册中心地址
        address: https://192.168.0.210:30100,https://192.168.0.246:30100
        appName: basic-application
        serviceName: ${spring.application.name}
        version: 0.0.1
        healthCheckInterval: 30
      config:
        # 配置中心地址
        serverType: kie
        serverAddr: https://192.168.0.210:30110,https://192.168.0.246:30110
```

说明

使用ServiceStage部署的场景，服务注册中心地址和配置中心地址在部署过程中会自动注入，无需额外手工添加。

步骤6 打包demo源码成jar包。

1. 在demo源码根目录下，打开cmd命令，执行**mvn clean package**命令，对项目进行打包编译。
2. 编译成功后，生成如**表1-2**所示的两个Jar包。

表 1-2 软件包列表

软件包所在目录	软件包名称	说明
basic\consumer\target	basic-consumer-1.0-SNAPSHOT.jar	服务消费者
basic\provider\target	basic-provider-1.0-SNAPSHOT.jar	服务生产者

步骤7 部署应用。

使用ServiceStage部署微服务provider和consumer。

1. 将**步骤6**生成的JAR包上传至OBS。
2. 参考**快速创建Kubernetes集群**创建CCE集群，并同ServiceComb引擎实例属于同一VPC。
3. 参考**创建环境**在引擎实例所在VPC下创建ServiceStage环境，并对ServiceComb引擎和CCE资源进行纳管。
4. 参考**创建并部署组件**部署provider和consumer微服务。

步骤8 确认部署结果。

1. **可选:** 在微服务引擎控制台页面，在左侧导航栏选择“ServiceComb引擎专享版”，单击**创建ServiceComb引擎**中创建的ServiceComb引擎。
2. 选择“微服务目录 > 微服务列表”，查看微服务basic-consumer和basic-provider的实例数量。
   - 若实例数量值不为0，则表示已经成功接入ServiceComb引擎。
   - 若实例数量为0，或者找不到basic-consumer和basic-provider服务名，则表示微服务应用接入ServiceComb引擎失败。

----结束

2 Spring Cloud 应用通过 Spring Cloud SDK 接入 Nacos 引擎

本章节通过一个demo进行全流程的微服务应用接入Nacos引擎操作演示，帮助您快速了解如何接入Nacos引擎。

1. **准备工作**
   您需要完成注册华为云并实名认证、为账户充值、为用户添加操作权限、创建VPC和子网、获取demo包及准备好本地编译构建打包环境准备工作。
2. **创建注册配置中心**
   在创建引擎时，您可以根据实际业务需要，选择合适的引擎规格、可用区和网络等。
3. **Spring Cloud应用接入Nacos引擎**
   本指导将使用一个provider服务和一个consumer服务接入Nacos引擎。

准备工作

1. 注册华为云并实名认证。
   如果您已有一个华为账户，请跳到下一个任务。如果您还没有华为账户，请参考以下步骤创建。
   a. 打开**华为云官网**，单击“注册”。
   b. 根据提示信息完成注册，详细操作请参见**注册华为账号并开通华为云**。
      注册成功后，系统会自动跳转至您的个人信息界面。
   c. 参考**实名认证**完成个人或企业账号实名认证。
2. 为账户充值。
   您需要确保账户有足够金额。
   - 关于注册配置中心的价格，请参见**微服务引擎价格详情**。
   - 关于充值，请参见**如何给华为账户充值**。
3. 为用户添加操作权限。
   用户在创建依赖资源和Nacos引擎前，需要具备相应的操作权限。添加用户权限的操作，请参考**创建用户并授权使用微服务引擎**。

   说明

   创建注册配置中心需要保证用户具有CSE FullAccess、DNS FullAccess权限。

4. 创建VPC和子网。

   Nacos引擎运行于虚拟私有云（VPC）中，并需要绑定具体的子网。创建Nacos引擎前，需保证有可用的虚拟私有云和子网。创建虚拟私有云和子网的方法，请参考**创建虚拟私有云和子网**。如果已有可用的VPC和子网，不需要再次创建。

5. 本地编译构建打包机器环境已安装了Java JDK、Maven，并且能够访问Maven中央库。

6. 下载github的**demo源码**到本地并解压。

创建注册配置中心

步骤1 进入**购买注册配置中心**页面。

步骤2 左侧导航栏选择“注册配置中心”。

步骤3 在注册配置中心页面，单击“购买注册配置中心”。

步骤4 参考下表设置参数，参数前面带*号的是必须设置的参数。创建注册配置中心所需参数的详细介绍请参考**创建注册配置中心**。

表 2-1 创建注册配置中心参数

参数	说明
*计费模式	选择计费方式，此处选择“按需计费”。按需计费是一种后付费模式，即先使用再付费，按照注册配置中心实际使用时长计费。
*企业项目	选择注册配置中心Nacos所在的企业项目。企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理。此处默认选择“default”。
*引擎名称	输入Nacos引擎的名称，名称以字母开头，由字母、数字和-组成，且不能以-结尾，长度为3~64个字符。如输入nacos-test。
*注册配置中心类型	支持的注册配置中心类型为“Nacos”。 **说明** 注册配置中心集群的节点会尽可能均分到不同的可用区中，单节点故障不影响对外业务功能。注册配置中心不支持AZ级故障的容灾，可提供主机级别容灾能力。
*选择实例数	选择需要购买的容量规格。此处选择实例数为500，其容量单元为10。
版本	只能创建最新版本。
*网络	选择已创建的虚拟私有云及子网，可在下拉框中搜索和选择合适的虚拟私有云和子网。 虚拟私有云可以为您的引擎构建隔离的、用户自主配置和管理的虚拟网络环境。

步骤5 单击“立即购买”，注册配置中心开始创建，当“运行状态”为“可用”时，注册配置中心创建完成。

----结束

Spring Cloud 应用接入 Nacos 引擎

步骤1 登录**微服务引擎控制台**。

步骤2 获取Nacos引擎注册发现地址。

1. 在左侧导航栏选择“注册配置中心”，单击创建的Nacos引擎实例。
2. 在“基础信息”页面的“连接信息”区域，获取注册发现地址。

步骤3 修改demo中的配置中心地址和服务注册中心地址和微服务名。

1. 在“bootstrap.properties”中配置Nacos配置中心。

   在下载到本地的demo源码目录中找到“nacos-examples-master\nacos-spring-cloud-example\nacos-spring-cloud-discovery-example\nacos-spring-cloud-consumer-example\src\main\resources”添加“bootstrap.properties”文件，配置Naocs配置中心：

   ```
   spring.cloud.nacos.config.server-addr=XXX.nacos.cse.com:8848 //Nacos的配置中心地址
   spring.cloud.nacos.config.prefix=example  //配置文件名前缀
   spring.cloud.nacos.config.file-extension=properties  //配置文件名后缀
   spring.cloud.nacos.config.group=XXX  //配置文件所属分组，不填默认DEFAULT_GROUP
   spring.cloud.nacos.config.namespace=XXX  //配置文件所属命名空间ID，不填默认public
   ```

   更多配置详情，请参考**Nacos配置参考**。

2. 在“application.properties”中配置Nacos的服务注册发现地址和微服务名。
   - 在下载到本地的demo源码目录中找到“nacos-examples-master\nacos-spring-cloud-example\nacos-spring-cloud-discovery-example\nacos-spring-cloud-consumer-example\src\main\resources\application.properties”文件，配置consumer服务：

     ```
     server.port=8080
     spring.application.name=service-consumer //微服务名
     spring.cloud.nacos.discovery.server-addr= XXX.nacos.cse.com:8848 //Nacos的服务注册发现地址
     spring.cloud.nacos.discovery.group=XXX //微服务所属分组，不填则默认DEFAULT_GROUP
     spring.cloud.nacos.discovery.namespace=XXX //微服务所属命名空间ID，不填则默认public
     spring.cloud.nacos.discovery.cluster-name=XXX //微服务所属集群名称，不填则默认DEFAULT
     ```

   - 在下载到本地的demo源码目录中找到“nacos-examples-master\nacos-spring-cloud-example\nacos-spring-cloud-discovery-example\nacos-spring-cloud-provider-example\src\main\resources\application.properties”文件，配置provider服务：

     ```
     server.port=8070
     spring.application.name=service-provider //微服务名
     spring.cloud.nacos.discovery.server-addr= XXX.nacos.cse.com:8848 //Nacos的服务注册发现地址
     spring.cloud.nacos.discovery.group=XXX //微服务所属分组，不填则默认DEFAULT_GROUP
     spring.cloud.nacos.discovery.namespace=XXX //微服务所属命名空间ID，不填则默认public
     spring.cloud.nacos.discovery.cluster-name=XXX //微服务所属集群名称，不填则默认DEFAULT
     ```

   更多配置详情，请参考**Nacos注册发现**。

步骤4 打包demo源码成jar包。

1. 在demo源码根目录下，打开cmd命令，执行**mvn clean package**命令，对项目进行打包编译。
2. 编译成功后，生成如**表2-2**所示的两个Jar包。

表 2-2 软件包列表

软件包所在目录	软件包名称	说明
\nacos-spring-cloud-consumer-example\target	nacos-spring-cloud-consumer-example-0.2.0-SNAPSHOT.jar	服务消费者
\nacos-spring-cloud-provider-example\target	nacos-spring-cloud-provider-example-0.2.0-SNAPSHOT.jar	服务生产者

步骤5 部署Spring Cloud应用。

将微服务Provider和Consumer部署至Nacos引擎所在VPC的ECS节点。

1. 请参考**购买并登录Linux弹性云服务器**在引擎实例所属VPC下创建一台ECS节点并登录。
2. 安装JRE，为服务提供运行环境。
3. 将**步骤4**生成JAR包上传至ECS节点。
4. 执行命令：**java -jar {对应jar包}**，运行生成的JAR包。

步骤6 确认部署结果。

1. **可选:** 在微服务引擎控制台页面，在左侧导航栏选择“注册配置中心”，单击**创建注册配置中心**中创建的Nacos引擎。
2. 选择“服务管理”，查看微服务service-consumer和service-provider的实例数量。
   - 若实例数量值不为0，则表示已经成功接入Nacos引擎。
   - 若实例数量为0，或者找不到service-consumer和service-provider服务名，则表示微服务应用接入Nacos引擎失败。

----结束

3 应用网关添加 ServiceComb 引擎中的服务并为其配置路由策略

CSE应用网关是各类应用的流量入口，是基于Envoy项目增强的云上托管类网关产品，实现Ingress与微服务网关合一的全新形态。您可以为微服务创建一个应用网关，通过从ServiceComb引擎、Nacos引擎、固定地址或CCE中添加服务，然后在应用网关中为服务创建路由策略，以便该服务通过网关对外提供服务。

1. **准备工作**

   您需要完成注册华为云并实名认证、为账户充值、为用户添加操作权限、创建VPC、子网和独享型负载均衡器、ServiceComb引擎创建、获取demo包和本地构建编译打包环境的准备工作。

2. **创建应用网关**

   在创建网关时，您可以根据实际业务需要，选择合适的网关规格、可用区和网络等。

3. **应用网关为服务配置路由策略**

   本指导以通过从ServiceComb引擎添加服务并为该服务配置路由策略。

准备工作

1. 注册华为云并实名认证。

   如果您已有一个华为账户，请跳到下一个任务。如果您还没有华为账户，请参考以下步骤创建。

   a. 打开**华为云官网**，单击“注册”。

   b. 根据提示信息完成注册，详细操作请参见**注册华为账号并开通华为云**。

      注册成功后，系统会自动跳转至您的个人信息界面。

   c. 参考**实名认证**完成个人或企业账号实名认证。

2. 为账户充值。

   您需要确保账户有足够金额。

   - 关于ServiceComb引擎和应用网关的价格，请参见**微服务引擎价格详情**。
   - 关于充值，请参见**如何给华为账户充值**。

3. 为用户添加操作权限。

   用户在创建依赖资源和应用网关前，需要具备相应的操作权限。添加用户权限的操作，请参考**创建用户并授权使用微服务引擎**。

说明

创建应用网关需要保证用户具有ELB FullAccess权限。

4. 创建ServiceComb引擎、VPC、子网和独享型负载均衡器。
   - 应用网关、ServiceComb引擎运行于虚拟私有云（VPC）中，并需要绑定具体的子网。创建应用应用网关和ServiceComb引擎前，需保证有可用的虚拟私有云和子网。创建虚拟私有云和子网的方法，请参考**创建虚拟私有云和子网**。如果已有可用的VPC和子网，不需要再次创建。

     说明

     ServiceComb引擎同应用网关所在VPC相同。
   - 应用网关可以从ServiceComb添加服务，因此需要有创建好的ServiceComb引擎。创建ServiceComb引擎具体操作请参考**创建ServiceComb引擎**。
   - 应用网关的运行依赖于独享型负载均衡器，创建应用网关前，需保证已创建了独享型负载均衡器，创建独享型负载均衡器请参考**创建独享型负载均衡器**。
5. 本地编译构建打包机器环境已安装了Java JDK、Maven，并且能够访问Maven中央库。
6. 下载github的jar包：**unit-controller.jar**、**unit-consumer.jar**、**unit-provider.jar**，参考**Spring Cloud应用通过Spring Cloud Huawei SDK接入ServiceComb引擎**将应用接入ServiceComb引擎，如下图所示。

图 3-1 应用接入 ServiceComb

创建应用网关

步骤1 进入**购买应用网关**页面。

步骤2 参考下表设置参数，参数前面带*号的是必须设置的参数。创建应用网关所需参数的详细介绍请参考**创建应用网关**。

表 3-1 创建应用网关参数

参数	说明
*企业项目	选择应用网关所在的企业项目。企业项目是一种云资源管理方式，企业项目管理服务提供统一的云资源按项目管理，以及项目内的资源管理、成员管理。此处默认选择“default”。
*引擎名称	输入应用网关名称，名称以字母开头，由字母、数字和-组成，且不能以-结尾，长度为3~64个字符。如：microGateway-test。
产品版本	目前只支持专业版。

参数	说明
*规格	选择实例规格，此处选择“小型”规格。
可用区	您可根据实际情况选择可用区。
版本	只能创建最新版本。
节点数	选择网关的节点数量，节点数量不少于2个。
入口网络	选择已创建的虚拟私有云及子网，可在下拉框中搜索和选择合适的虚拟私有云和子网。此VPC子网用于选择可用的弹性负载均衡，需与后端VPC子网网络互通，具体操作请参考**对等连接**。 **说明** 当应用网关创建完成后，不支持变更虚拟私有云。
*弹性负载均衡	选择已创建的弹性负载均衡。此弹性负载均衡与入口网络属于同一个VPC子网。 **说明** • 当前仅支持独享型网络型负载均衡器。会使用所选ELB的80、443端口，如果选择的负载均衡器已经占用了部分端口，会创建失败。 • ELB被应用网关使用后，为其配置的用于应用网关使用的监听器（即监听器名称中包含应用网关名称）及其关联的后端服务器组和后端服务器均不允许被删除。
后端网络	虚拟私有云可以为您的引擎构建隔离的、用户自主配置和管理的虚拟网络环境。 选择已创建的虚拟私有云及子网，可在下拉框中搜索和选择合适的虚拟私有云和子网。默认同入口网络一致。若同入口网络不一致时，需要打通网络。 **说明** • 当应用网关创建完成后，不支持变更虚拟私有云。 • 各VPC网段不冲突。 • 当前端ELB与后端网络不同VPC时，需开启ELB的跨VPC后端，具体操作请参考**跨VPC后端**。

步骤3 单击“立即购买”，网关开始创建，当“运行状态”为“可用”时，应用网关创建完成。

说明

应用网关创建成功后，“运行状态”为“可用”。查看应用网关状态，请参考**查看应用网关基本信息**。

----结束

应用网关为服务配置路由策略

步骤1 登录**微服务引擎控制台**。

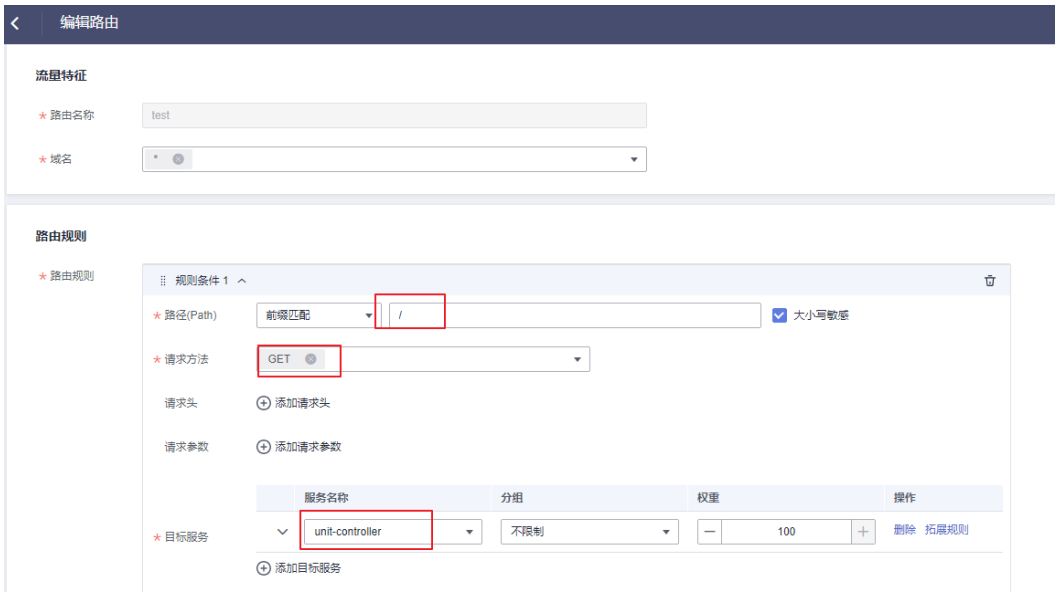
步骤2 在左侧导航栏选择“应用网关”。

步骤3 单击创建好的且状态为可用的应用网关实例，进入应用网关基础信息界面。

步骤4 选择“ 路由管理 > 服务来源”，单击“ 创建来源”，选择“来源类型”为“CSE ServiceComb引擎”，输入“来源名称”，选择已部署了组件的ServiceComb引擎，单击“确定”。

步骤5 选择“ 路由管理 > 服务管理”，单击“ 创建服务”，选择“来源类型”为“CSE ServiceComb引擎”，选择**步骤4**中创建的服务来源，“服务列表”选择“unit-controller”，单击“确定”。

步骤6 选择“ 路由管理 > 路由配置”，单击“ 创建路由”，输入路由名称，勾选域名“*”，在“路径（Path）”中，匹配规则选择“前缀匹配”，服务地址输入“/”，“请求方法”选择“GET”，“目标服务”选择“unit-controller”，“权重”输入100，单击“保存并发布”。



步骤7 SSH远程登录CCE集群的节点，进行curl命令检查：“curl http://{网关ELB的内网IP}/{路径}”。

其中参考**查看应用网关信息**获取ELB内网IP。

图 3-2 ELB 内网 IP

执行命令**curl http://192.168.0.194/unit-controller/hello** 返回结果包含controller组件信息。

图 3-3 获取组件信息

----结束

4 入门实践

本文介绍CSE常见的使用实践，帮助您更好的使用CSE。

实践	描述
托管Spring Cloud应用	Spring Boot、Spring Cloud广泛应用于构建微服务应用。使用ServiceComb引擎托管Spring Cloud应用，主要目的是使用高可靠的商业中间件替换开源组件，对应用系统进行更好地管理和运维，改造过程应尽可能降低对业务逻辑的影响。
托管Java Chassis应用	Java Chassis是Apache基金会管理的开源微服务开发框架，最早由CSE捐献，目前有上百个开发者为项目做出贡献。 它提供了如下独特的功能： • 灵活高性能的RPC实现。Java Chassis基于Open API，给出了不同RPC开发方式的统一描述，让微服务接口管理更加规范，同时保留了灵活的开发者使用习惯。Java Chassis基于Reactive，实现了高效的REST、Highway等通信协议，同时保留了传统Servlet等通信协议的兼容。 • 丰富的服务治理能力和统一的治理职责链。负载均衡、流量控制、故障隔离等常见的微服务治理能力都可以开箱即用，同时提供了统一的治理职责链，让新的治理功能的开发变得简单。
Spring Cloud应用实现优雅上下线功能	在应用使用过程中，应用的发布、重启、扩缩容操作无法避免，为了保证应用正确上下线、流量不丢失，微服务引擎基于Sermant Agent提供了一套优雅上下线的方案，包括预热、延迟下线等，避免了请求超时、连接拒绝、流量丢失等问题的发生。

实践	描述
Spring Cloud应用实现标签路由功能	在微服务存在多个版本、多个实例的情况下，需要管理服务之间的路由，达到无损升级、应用拨测等业务目的。Sermant Agent提供了标签路由的能力，标签路由通过匹配http请求头中的信息，把符合规则的流量转发到对应的标签应用中，从而实现标签路由的功能。
Spring Cloud应用实现全链路灰度	在微服务架构下，有些开发需求会使微服务调用链路上的多个微服务同时发生了改动，通常每个微服务都会有灰度环境或分组来接收灰度流量。此时希望通过进入上游灰度环境的流量，也能进入下游灰度的环境中，确保一个请求始终在灰度环境中传递，即使这个调用链路上有一些微服务没有灰度环境，这些应用请求在下游的时候依然能够回到灰度环境中。通过Sermant Agent提供的全链路灰度能力，可以在不需要修改任何您的业务代码的情况下，能够轻松实现上述能力。