应用管理与运维平台

常见问题

文档版本 01

发布日期 2025-07-07

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 应用开发问题咨询指引

开源社区链接

servicecomb-java-chassis、spring-cloud-huawei等SDK作为开源框架，版本的发布和维护都由开源团队负责，目前已经在开源社区积累了诸多问题的解决经验，若在开发使用过程中遇到问题，请通过如下链接检索历史问题、提交新问题和进行问题咨询：

- servicecomb-java-chassis开源框架

  开源社区：**https://github.com/apache/servicecomb-java-chassis/issues**

  开发者指导：**https://servicecomb.apache.org/references/java-chassis/zh_CN/**

- spring-cloud-huawei开源框架

  开源社区：**https://github.com/huaweicloud/spring-cloud-huawei/issues**

  参考文档：**https://support.huaweicloud.com/intl/zh-cn/devg-servicestage/cse_04_0009.html**

说明

- 搜索小技巧：

  使用站内搜索，提升准确性。比如服务报错：Load balancer does not have available server for client: default，可以使用如下搜索方式：

  Load balancer site:bbs.huaweicloud.com或者Load balancer site:github.com/apache/servicecomb-java-chassis/issues

- servicecomb-java-chassis, spring-cloud-huawei均为开源软件，ServiceStage团队对其进行了插件扩展以接入云服务，如果遇到框架使用问题而非插件问题，请到对应开源社区提问。

关键信息

为了方便问题的快速定位，提issue时务必提供详细的关键信息，提供可以复现问题的Demo。

以servicecomb-java-chassis为例，请提供如下关键信息：

1. 框架相关日志：默认框架日志会与业务日志一起打印，并且会在根目录下生成cse.log。若业务侧使用了log4j2或者logback等日志框架，需根据自定义的日志策略查找关键信息。

   a. 服务启动类问题关键信息：

表 1-1 服务启动类问题关键信息

关键词	描述
choose org.apache.servicecomb	servicecomb-java-chassis支持两种rest通信通道，需要根据日志判断使用的通信通道。 choose org.apache.servicecomb.transport.rest.vertx.VertxRestTransport框架默认使用Rest over Vertx通信通道，即使用vertx作为http服务器。 choose org.apache.servicecomb.transport.rest.servlet.ServletRestTransport同时支持Rest over Servlet通信通道，即使用其他HTTP服务器，比如tomcat。
endpoint to publish	服务发布地址。
Register microservice instance success	服务实例注册成功的标志。

b. 服务调用类问题关键信息：

表 1-2 服务调用类问题关键信息

关键词	描述
find instances	消费端（发起调用的服务）在调用服务端（被调用的服务）之前，会先从微服务引擎的服务中心查询服务端的实例
accesslog	accesslog会记录调用该服务的请求源，API，状态码等，默认情况下不会开启该功能

accesslog的打印受通信通道和日志框架影响。若使用Rest over Vertx通信通道，accesslog是由Vertx记录，详细配置可以参考：**https://servicecomb.apache.org/references/java-chassis/zh_CN/build-provider/access-log-configuration.html**。

推荐accesslog日志格式：

```
servicecomb.accesslog.pattern: "%h - - %t cs-uri %s %B %D %H %SCB-traceId"
```

默认情况下会在根目录生成access.log，若业务侧使用了log4j2或者logback等日志框架，日志框架的切换可参考：**https://servicecomb.apache.org/references/java-chassis/zh_CN/build-provider/access-log-configuration.html**。

若使用Rest over Servlet通信通道，accesslog是由使用的HTTP服务器记录，需自行搜索资料开启。

例如springboot内置的tomcat，可以通过如下配置开启。

```
server:
  tomcat:
    accesslog:
      enabled: true
```



```
    pattern: '%h %l %u %t "%r" %s %b %D'
    directory: accesslogs
    buffered: false
  basedir: ./logs
```

2. 微服务引擎的版本和SDK的版本。微服务引擎版本可单击引擎名称查看。SDK版本可搜索groupId为org.apache.serivcecomb的依赖。

2 环境管理

2.1 微服务和其他平台服务有什么不同?

微服务是构建应用系统的架构模式，平台服务是云平台提供给平台用户使用的中间件服务。

在使用上，平台服务需要进行订购流程进行服务开通。对于微服务，需自己开发并使用平台提供的服务发现能力进行服务发现。

3 应用管理

3.1 如何查看应用组件部署失败的原因?

问题描述

应用组件部署完成后，状态显示为“未就绪”，表示应用组件部署失败。

解决方法

步骤1 登录ServiceStage控制台。

步骤2 选择以下任意方式进入组件“实例列表”页面：

- 在“应用管理”页面，单击组件所属应用名称，在“组件列表”单击待操作组件名称，在左侧导航栏单击“实例列表”。
- 在“组件管理”页面，单击待操作组件名称，在左侧导航栏单击“实例列表”。

步骤3 单击实例列表待操作实例名称前的。

步骤4 选择“事件”页签，在事件列表中，查看事件描述信息，判断应用组件部署失败的原因。

----结束

3.2 实例长期处于创建中怎么办?

应用组件部署以后，如果服务实例状态长期处于“未就绪”状态时，可以进入服务实例列表，展开实例详细信息，在“事件”页签查看详细信息，显示内存不足，如下图所示。

可以通过新增节点解决该问题，操作方法请参考。

3.3 如何解决 Docker 运行 node 应用程序时的依赖问题?

问题描述

在微服务docker里面运行一个node程序，这个程序依赖一个node-gyp，怎么在程序运行之前安装好这些依赖。

解决办法

可以定制自己的Dockerfile，在Dockerfile里面添加node-gyp依赖。

3.4 如何定制 Tomcat Context path?

在创建和部署Tomcat应用时需要设置Tomcat配置，使用默认server.xml配置，上下文路径是"/"，没有指定应用路径。

- 选择开启“公网访问”，应用访问地址为：http://${应用公网域名}:${应用访问端口号}。例如，http://example_domain.com:30317。
- 未开启“公网访问”，应用访问地址为：http://${VPC内网访问地址}:${应用访问端口号}。例如，http://192.168.0.168:30317。

在部署组件的组件配置过程中，可在设置“Tomcat 配置”时根据具体业务实际自定义应用路径：

1. 勾选“配置参数”。
2. 单击“使用示例模板”，根据业务要求编辑模板文件。
3. 参考如下示例修改Context path的内容，例如修改为"app-path" ，自定义应用路径。则应用访问地址被修改为http://example_domain.com:30317/app-path或者http://192.168.0.168:30317/app-path。

```
<Host name="localhost"  appBase="webapps"
    unpackWARs="true" autoDeploy="true" >
  <Context path="app-path" docBase="ROOT.war"/>
```

3.5 如何固定应用组件 IP?

问题描述

在部署应用组件的过程中，如果不设置“TCP/UDP路由配置”，那么当容器重启时，应用的访问IP会发生变化。这种情况会为您的某些配置造成困扰。

解决方法

创建部署应用组件时或者部署应用组件后设置一下“TCP/UDP路由配置”即可。以下三种方式均可解决该问题：

- 集群内访问：应用暴露给同一集群内其他应用访问的方式，可以通过集群内部域名访问。
- VPC内网访问：应用可以让同一VPC内其他应用程序访问，通过集群节点的IP或者私网弹性负载均衡ELB的服务地址访问。

- 公网访问：通过弹性IP从公网访问应用，一般用于系统中需要暴露到公网的服务。该访问方式需要给集群内任一节点绑定弹性IP，并设置一个映射在节点上的端口。

3.6 如何处理虚拟机类型环境下创建和部署组件时遇到 ECS 错误?

问题描述

在ServiceStage虚拟机类型环境下创建和部署组件时可能会遇到ECS服务不可用问题。

例如，在组件部署时调用ECS接口超时，查看日志详情报错如下：

```
{
   "statusCode": 500,
   "jsonBody": {
      "error_code": "SVCSTG.VMAPP.5001002",
      "error_msg": "read ECS host 471ff77a-c827-41d5-941d-4fea8aaa56ef fail TIMEOUT."
   }
}
```

解决方法

步骤1 重新部署组件，查看部署是否成功。

- 是，处理结束。
- 否，执行步骤2。

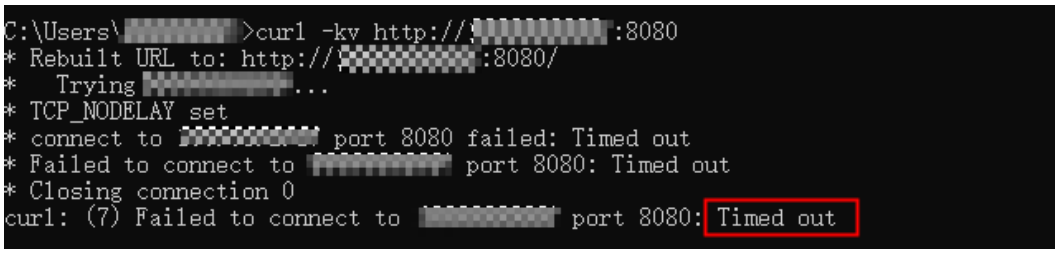
步骤2 联系技术支持工程师协助解决。

----结束

3.7 如何处理虚拟机类型环境下创建和部署组件时端口访问不通的错误?

问题描述

在ServiceStage虚拟机类型环境下创建和部署组件时可能会遇到容器端口无法访问的问题。使用**curl -kv http://**${部署应用组件的弹性云服务器节点IP}:${容器端口}命令访问容器端口时会提示访问超时。



解决方法

步骤1 登录云服务器控制台，单击“弹性云服务器”。

步骤2 在弹性云服务器列表选择部署组件的弹性云服务器，单击名称进入“基本信息”。

步骤3 在“安全组”页签，单击“更改安全组”。

- 如果该端口规则已经在已有安全组中存在，可直接勾选该安全组。
- 如果该端口规则在已有安全组中不存在，请单击“新建安全组”，创建安全组，自行配置相关规则后再勾选该新建安全组。

步骤4 再次使用**curl -kv http://**${部署应用组件的弹性云服务器节点IP}**:**${端口}命令访问容器端口，确认问题是否已解决。

----结束

3.8 虚拟机类型环境下部署的应用组件支持在什么目录写文件?

虚拟机类型环境下部署的应用组件只支持在应用组件的运行目录下写文件，比如写日志文件或者解压包等。

应用组件的运行目录为安装虚拟机应用组件的ECS上的/opt/application/${appName}/${appVersion}/${instanceId}目录路径下。其中:

- ${appName}为组件实例名。
- ${appVersion}为组件实例的版本号。
- ${instanceId}为实例id。

同时，虚拟机类型环境下部署的应用组件只支持在应用组件的运行目录下写文件只针对新安装的组件实例或者升级后的组件实例生效，对于原先已经部署的组件实例的目录还是维持原来的权限。

3.9 如何处理虚拟机类型环境下部署的组件删除失败报 host status is not active 错误的问题?

问题描述

虚拟机类型环境下部署组件删除失败，在任务详情界面单击“查看详情”，看到的错误信息示例如下：

```
{
   "statusCode": 400,
   "jsonBody": {
      "error_code": "SVCSTG.VMAPP.4001020",
      "error_msg": "4001020",
      "error_detail": "host status is not active: abb3d0a4-f715-4932-
b7ec-6dd917f65778,4f68e35b-6e08-48d0-bd3a-1151be19efa5"
   }
}
```

其中：

- 错误码为：SVCSTG.VMAPP.4001020。
- 详细错误信息为：host status is not active: abb3d0a4-f715-4932-b7ec-6dd917f65778 ，“abb3d0a4-f715-4932-b7ec-6dd917f65778,4f68e35b-6e08-48d0-bd3a-1151be19efc6”是部署组件的两个弹性云服务器的ID。

解决方法

步骤1 登录云服务器控制台，单击“弹性云服务器”。

步骤2 在弹性云服务器列表，使用报错信息中的弹性云服务器ID搜索部署组件的弹性云服务器。

步骤3 查看弹性云服务的状态是否为“运行中”。

- 是，执行**步骤2**，搜索报错信息中的下一个弹性云服务器。
- 否，执行**步骤4**。

步骤4 根据弹性云服务器的状态：

- 在“操作”列选择“更多 > 开机”或者“更多 > 重启”，将弹性云服务器的状态恢复为“运行中”。
- 如果不再使用该弹性云服务器，在“操作”列选择“更多 > 删除”，删除该弹性云服务器。

步骤5 待报错信息中所有弹性云服务器已按照**步骤2**~**步骤4**执行完操作后，重新在ServiceStage界面执行删除组件操作。

----结束

3.10 如何体验 ServiceStage 的源码部署功能?

如**表3-1**所示，ServiceStage基于GitHub提供了一些不同语言的demo。

您可以Fork特定语言的demo源码到自己的GitHub代码仓库中，参考**创建并部署组件**去体验ServiceStage的源码部署功能。

表 3-1 ServiceStage 提供的 demo 源码及 GitHub 地址说明

demo名称	语言类型	GitHub代码仓库地址
ServiceComb-SpringMVC	Java	**https://github.com/servicestage-template/ServiceComb-SpringMVC**
ServiceComb-JAX-RS	Java	**https://github.com/servicestage-template/ServiceComb-JAX-RS**
ServiceComb-POJO	Java	**https://github.com/servicestage-template/ServiceComb-POJO**
SpringBoot-WebService	Java	**https://github.com/servicestage-template/SpringBoot-WebService**
SpringBoot-Webapp-Tomcat	Java	**https://github.com/servicestage-template/SpringBoot-Webapp-Tomcat**
nodejs-express	Node.js	**https://github.com/servicestage-template/nodejs-express-4-16**
nodejs-koa	Node.js	**https://github.com/servicestage-template/nodejs-koa-2-5-2**
php-laravel	PHP	**https://github.com/servicestage-template/php-laravel-v5-6-28**
php-slim	PHP	**https://github.com/servicestage-template/php-slim-3-10-0**

3.11 如何处理使用 ServiceStage 灰度发布升级组件失败?

问题描述

使用ServiceStage灰度发布升级组件失败，可能出现如下报错信息：

- query microservice info failed, microservices should be registered.
- The grayscale service must be a new version.

解决办法

步骤1 根据报错信息内容，确认失败原因。

- 如果报错信息为“query microservice info failed, microservices should be registered.”，可能是由于灰度发布的组件实例不是微服务类型的组件实例或者组件实例未注册到CSE上。

- 如果报错信息为“The grayscale service must be a new version.”，是由于注册到CSE的灰度版本组件实例不是新版本。

步骤2 在“部署记录”页面上的“部署记录”列表，选择灰度发布失败的部署记录。

步骤3 单击“回滚”，回滚至升级前版本。

步骤4 根据**步骤1**确认的失败原因，解决问题。

步骤5 重新执行组件灰度发布，确认是否成功。

- 是，操作结束。
- 否，联系技术支持工程师解决问题。

----结束

3.12 如何通过挂载配置项方式修改容器部署的组件配置文件?

问题描述

使用ServiceStage容器部署方式部署的组件，由于ServiceStage技术栈中提供的配置文件可能无法满足您的实际业务要求，可以通过挂载配置项的方式修改组件配置文件。

解决方案

步骤1 登录ServiceStage控制台。

步骤2 单击“环境管理”，找到组件对应的环境后单击环境名称。

步骤3 在“资源配置”下左侧列表，选择“计算”资源类型下的资源名称“云容器引擎CCE”。

步骤4 选择“配置项”页签，单击“创建配置项”。

步骤5 选择“创建方式”，输入“配置名称”，选择和组件相同的“所属集群”和“命名空间”，“配置数据”输入对应的“键”和“值”，单击“创建配置项”。

步骤6 单击“组件管理”，单击待修改组件配置文件的组件名称，进入组件“概览”页面。

步骤7 单击“升级”，选择组件“升级类型”，单击“下一步”。

步骤8 选择“高级设置 > 部署配置 > 数据存储 > 本地磁盘”。

步骤9 单击“挂载本地磁盘”，选择“配置项挂载”， 选择**步骤5**已创建好的“配置项”，“挂载路径”输入文件路径（建议路径为一个不存在的路径，否则可能会导致容器中的文件路径被覆盖造成路径污染），单击“确定”。

说明

配置项挂载的文件通常为只读状态，配置为不存在的路径时，可以通过复制文件到指定路径解决无法修改的问题。

步骤10 选择“启动命令”页签，输入对应的“运行命令”、“运行参数”，单击“升级”。

----结束

3.13 如何处理应用组件接入微服务引擎后在微服务治理下看到的应用名称和 ServiceStage 应用管理下的不同?

问题描述

应用组件接入微服务引擎后在微服务治理下看到的应用名称和ServiceStage应用管理下的不同。例如，在“应用管理”下的“canary-application”应用下创建并部署的组件，接入微服务引擎后，在引擎的“微服务目录 > 微服务列表”下，查看到该组件实例的“所属应用”名称为“canary-application-batch”，两者并不相同。

原因分析

在ServiceStage中，应用是指一个功能相对完备的业务系统，由一个或多个特性相关的组件组成，以应用维度组织多个组件。

在微服务中，可以将应用理解为完成某项完整业务场景的软件系统。应用一般由多个微服务组成，应用里面的微服务能够相互发现和调用。

- 在Spring Cloud微服务架构开发的项目中，应用名称通常在项目下各组件的“bootstrap.yaml”配置文件中定义。
- 在Java Chassis微服务架构开发的项目中，应用名称通常在项目下各组件的“microservice.yaml”配置文件中定义。

配置文件一般都存放于您当前项目各组件目录下的“/src/main/resources/”路径下。

ServiceStage应用下的各组件实例接入微服务引擎后，其微服务实例“所属应用”名称就是各组件下的配置文件中所定义的应用名称。

解决办法

如果您需要使微服务引擎的“微服务目录 > 微服务列表”下查看到该组件实例的“所属应用”名称和ServiceStage“应用管理”下的应用名称保持一致：

- Spring Cloud微服务架构，请将项目各组件的配置文件中应用名称参数项的值修改为：${CAS_APPLICATION_NAME:basic-application}。
- Java Chassis微服务架构，请将项目各组件的配置文件中应用名称参数项的值修改为：${CAS_APPLICATION_NAME}。

4 持续交付

4.1 ServiceStage 怎么管理 IDEA 上的代码?

IDEA是本地的IDE，在IDE上编码后上传代码库，在ServiceStage上选择源码部署。

如果是基于ServiceComb框架开发的话，创建ServiceComb应用，选择源码部署，指定引擎，就可以治理了。

4.2 如何添加构建服务器地址到 GitLab 服务器安全组?

背景介绍

若您的GitLab服务搭建在公有云内部网络，并且公网无法直接访问，则需将构建服务的相关地址添加到您的GitLab服务器安全组中，以保证构建任务顺利执行。

操作步骤

步骤1 将ServiceStage所在网段加到GitLab私有仓库所在节点的安全组中，构建服务使用该IP访问GitLab服务的接口。

操作方法，请参考**设置安全组**。

说明

ServiceStage所在网段，请联系技术支持工程师获取。

步骤2 获取构建镜像的集群名称和过滤节点标签信息。

- 应用组件构建，请参考**编辑源码构建工程**，获取“构建集群”和“过滤节点标签”。
- 构建任务构建，请参考**创建源码构建任务**，获取“构建集群”和“过滤节点标签”。

图 4-1 获取集群名称和标签信息

步骤3 获取该集群下有该标签的节点弹性IP。

- 应用组件构建
  a. 在ServiceStage控制台，选择“持续交付 > 构建”，进入构建页面。
  b. 单击构建集群的名称，进入集群详情页面。
  c. 单击“节点管理”，获取该集群下有该标签的节点弹性IP。
- 构建任务构建
  a. 在ServiceStage控制台，选择“持续交付 > 构建”，进入构建页面。
  b. 选择构建任务，单击该构建任务的构建集群名称，进入集群详情页面。
  c. 单击“节点管理”，获取该集群下有该标签的节点弹性IP。

步骤4 将**步骤3**中获取的构建镜像的运行节点添加到GitLab私有仓库所在节点的安全组中，构建时会访问GitLab服务获取代码。

操作方法，请参考**设置安全组**。

----结束

4.3 如何添加构建服务器地址到 Maven 服务器安全组?

背景信息

将构建集群的构建节点弹性IP添加到Maven私有服务所在节点的安全组中，以便构建服务访问Maven私有服务器下载依赖包。

操作步骤

步骤1 获取构建镜像的集群名称和过滤节点标签信息。

- 应用组件构建，请参考**编辑源码构建工程**，获取“构建集群”和“过滤节点标签”。
- 构建任务构建，请参考**创建源码构建任务**，获取“构建集群”和“过滤节点标签”。

图 4-2 获取集群名称和标签信息

步骤2 获取该集群下有该标签的节点弹性IP。

- 应用组件构建
  a. 在ServiceStage控制台，选择“持续交付 > 构建”，进入构建页面。
  b. 单击构建集群的名称，进入集群详情页面。
  c. 单击“节点管理”，获取该集群下有该标签的节点弹性IP。
- 构建任务构建
  a. 在ServiceStage控制台，选择“持续交付 > 构建”，进入构建页面。
  b. 选择构建任务，单击该构建任务的构建集群名称，进入集群详情页面。
  c. 单击“节点管理”，获取该集群下有该标签的节点弹性IP。

步骤3 将构建集群的构建节点弹性IP添加到Maven私有服务所在节点的安全组中。

操作方法，请参考**设置安全组**。

----结束

4.4 如何解决使用 ServiceStage 构建失败的问题?

软件工程构建失败的原因很多，可以按照以下场景排查定位。

构建任务调度到 CCE 集群的 Containerd 容器引擎节点后构建报错

问题描述

构建失败，构建日志显示如下错误提示信息中的一种：

- /proc/sys/user/max_user_namespaces needs to be set to non-zero.
- /proc/sys/user/max_user_namespaces=100 may be low. Consider setting to >= 1024.

原因分析

由于构建任务被调度到了CCE集群的Containerd容器引擎节点，需要使用rootless来保证构建的安全性。在此过程中需要创建namespaces，节点虚机的相应设置需要能够满足构建要求，而部分虚机镜像的max_user_namespaces默认设置为0或过小，导致构建失败报错。

解决方法

步骤1 使用root用户登录执行构建任务的CCE集群下的所有Containerd容器引擎节点。

步骤2 执行如下命令将虚机镜像的max_user_namespaces默认设置为1024并确认。

```
echo 1024 > /proc/sys/user/max_user_namespaces
cat /proc/sys/user/max_user_namespaces
```

步骤3 重新启动构建任务，请参考**启动构建任务**。

- 构建成功，操作结束。
- 构建失败，请联系技术支持工程师协助处理。

----结束

CCE 集群节点配置污点和容忍策略后构建报错

问题描述

构建失败，构建日志显示如下错误提示信息：

```
0/1 nodes are available: 1 node(s) had untolerated taint {node.kubernetes.io/route-unschedulable: }. preemption: 0/1 nodes are available: 1 Preemption is not helpful for scheduling.
```

原因分析

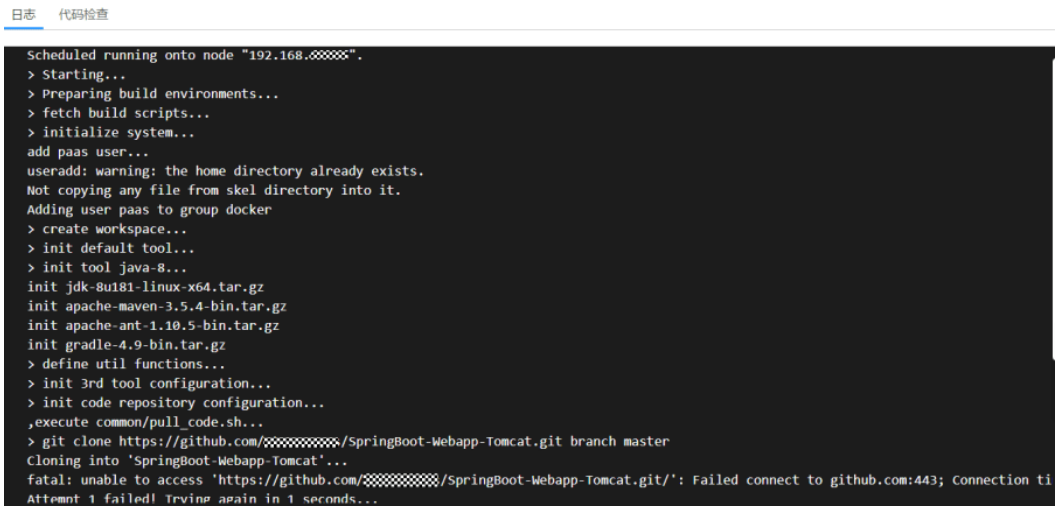
由于构建任务被调度到了CCE集群的受限调度节点上，该节点进行了污点管理配置（如报错信息中显示的node.kubernetes.io/route-unschedulable）。污点能够使节点排斥某些特定的Pod，从而避免Pod调度到该节点上，同时该CCE集群中没有其他节点可供调度，导致构建失败报错。

解决方法

请参考**管理节点污点（Taint）**，去除受限调度节点上的污点，保证CCE集群下至少有一个节点可供调度。

出现拉取不到代码的场景

如下图所示：



一般原因可能有三种。

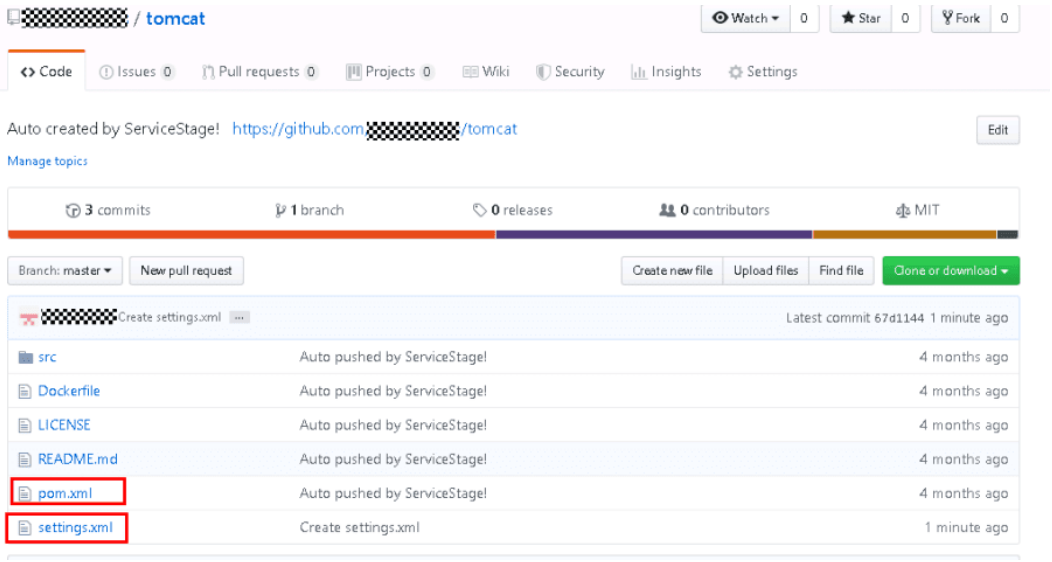
- 如果是在自己的节点上构建应用，可能是该节点没有绑定弹性ip，如上图所示"192.168.x.x"的节点没有绑定弹性ip，解决方法：去该节点**绑定弹性IP**。
- 授权信息过期，代码源的私人令牌权限范围不够，或者授权信息已被移除等，例如：CodeArts的代码源，拉取不到代码，可能是创建授权的时候，用户名对应的密码输错了，导致拉取不到代码。解决方法：**重新授权**即可。
- 自己搭建的代码源仓库，和构建的节点网络不通，例如：在集群A的某一个节点上搭建了一个私有的bitbucket，使用集群B构建，但是集群B和集群A不是同一个vpc，内网不通，导致构建拉取不到代码。解决方法：打通网络。

构建的代码依赖自己的私有 maven 仓库

有以下两种解决方案。

- 在自己的项目的根目录下增加settings.xml文件，在settings.xml指定自己的私有maven仓库地址（如果自己的私有maven仓库是需要认证的，则需要在settings.xml配置上自己的认证信息，用户名密码等）。
- 在自己项目的pom.xml文件中，指定自己的私有maven仓库。

settings.xml和pom.xml所在路径示例如下。



基于源码构建 Dockerfile 设置错误

如何写Dockerfile，可以参考**官网**，也可以参照ServiceStage**模板生成的demo**。

项目代码依赖 CSE 的 SDK 和搭建在 CodeArts 上的私有 maven 仓库

解决方法如下：

步骤1 在自己项目根目录下增加一个“settings.xml”文件。

步骤2 登录CodeArts的私有依赖库，在左侧仓库列表选择指定的maven私有依赖库。

步骤3 在右上角单击 ，在弹出菜单选择“配置指导”。

步骤4 单击“下载配置文件”，下载“settings.xml”文件。

步骤5 按照如下方法修改下载的“settings.xml”文件：

1. 在<mirrorOf>中增加一个!HuaweiCloudSDK。

```
<mirror>
    <id>z_mirrors</id>
    <mirrorOf>*,!releases,!snapshots,!HuaweiCloudSDK</mirrorOf>
    <url>https://repo.hauweicloud.com/repository/maven</url>
</mirror>
```

2. <profiles>的<repositories>下面增加一个maven仓库。

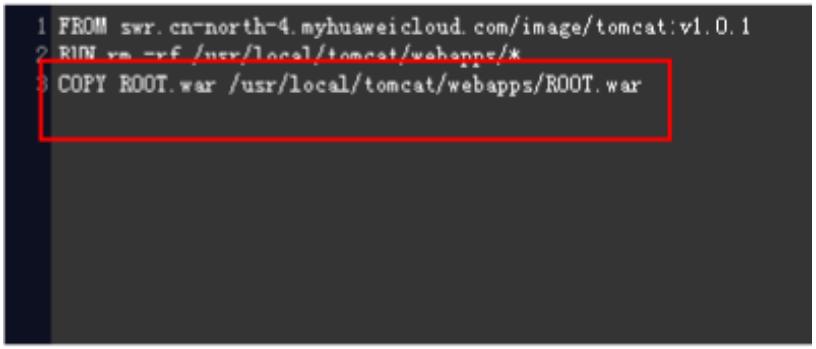
```
<repository>
     <id>HuaweiCloudSDK</id>
     <url>https://repo.huaweicloud.com/repository/maven/huaweicloudsdk/</url>
     <releases>
     <enabled>true</enabled>
     </releases>
     <snapshots>
     <enabled>true</enabled>
     </snapshots>
</repository>
```

步骤6 将修改好的“settings.xml”放在自己项目的根目录后，进行构建即可。

----结束

创建软件包构建工程时自定义 Dockerfile

选择软件包后，后台会自动上传到镜像的当前工作目录。Dockerfile的示例如下图所示：



4.5 如何使用 VPC 终端节点在构建镜像时访问依赖的服务?

问题描述

VPC终端节点（VPC Endpoint），能够将VPC私密地连接到终端节点服务，使VPC中的云资源无需弹性公网IP就能够访问终端节点服务，提高了访问效率，为您提供更加灵活、安全的组网方式。

在Kubernetes类型环境下创建并部署组件时的组件镜像构建过程中，可以通过已创建的VPC终端节点与OBS、SWR服务通信，并使用APIG通过配置好的内网域名调用ServiceStage的功能接口。

说明

仅“华东二”、“华南-广州”、“亚太-新加坡”、“非洲-约翰内斯堡”、“华东-上海一”、“华北-北京四”、“拉美-墨西哥城二”区域的ServiceStage支持通过VPC终端节点访问依赖的服务。

解决方法

步骤1 登录ServiceStage控制台。

步骤2 在区域列表选择您的业务所在区域，例如“亚太-新加坡”。

步骤3 在浏览器地址栏，获取“region”字段对应的值。

region取值示例如下面加粗内容所示：

https://console-intl.huaweicloud.com/servicestage/?agencyId=d6**********41&region=**ap-southeast-3**&locale=zh-cn#/overview

步骤4 分别为SWR、APIG服务创建VPC终端节点，请参考**购买连接“接口”型终端节点服务的终端节点**。

1. 区域：请选择**步骤2**所选择的区域。
2. 服务类别：选择“按名称查找服务”。
3. 服务名称：请参考**表4-1**填写各云服务对应的终端节点服务名称。

   说明

   请将下表中的${region}替换为**步骤3**获取到的值。

   表 4-1 连接“接口”型终端节点服务说明

云服务	终端节点服务名称
SWR	com.myhuaweicloud.${region}.swr **说明** 如果您选择的是“华南-广州”区域，SWR的终端节点服务名称是：swr.cn-south-1.myhuaweicloud.com。
APIG	com.myhuaweicloud.${region}.api

4. 勾选“创建内网域名”。
5. 虚拟私有云：根据您的实际业务需要，为**表4-1**所示的所有终端节点服务选择同一个虚拟私有云。
6. 子网：根据您的实际业务需要，为**表4-1**所示的终端节点服务分别选择子网。
7. 其他参数请根据实际业务需要进行设置。

步骤5 为OBS服务创建VPC终端节点，请参考**购买连接“网关”型终端节点服务的终端节点**。

需要为**表4-2**所示的OBS服务的终端节点服务分别创建VPC终端节点。

1. 区域：请选择**步骤2**所选择的区域。
2. 服务类别：选择“按名称查找服务”。
3. 服务名称：请参考**表4-2**填写OBS服务对应的终端节点服务名称。

表 4-2 连接“网关”型终端节点服务说明

区域	终端节点服务名称
华南-广州	cn-south-1.com.myhuaweicloud.v4.obsv2
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz05
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz08
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz09
华东二	com.myhuaweicloud.cn-east-4.obslb01.v4.obsv2.vxlan
	com.myhuaweicloud.cn-east-4.obslb01.v6.obsv2.vxlan
亚太-新加坡	ap-southeast-3.com.myhuaweicloud.v4.obsv2
	ap-southeast-3.com.myhuaweicloud.v6.obsv2
非洲-约翰内斯堡	af-south-1.myhuaweicloud.v4.obsv2
	af-south-1.myhuaweicloud.v6.obsv2
华东-上海一	cn-east-3.com.myhuaweicloud.v4.global.obsv2
	cn-east-3.v4obsv2_new.58c9f146-63f0-4f07-98d3-18fe4874086b
	cn-east-3.myhuaweicloud.v4.obsv2.lz03
	cn-east-3.myhuaweicloud.v4.obsv2.lz04
	cn-east-3.com.myhuaweicloud.v4.obsv2.lz09
	cn-east-3.com.myhuaweicloud.v4.obsv2.lz11
华北-北京四	cn-north-4.com.myhuaweicloud.v4.obsv2.OBSCluster9
	cn-north-4.com.myhuaweicloud.v4.byte.obsv2
	cn-north-4.com.myhuaweicloud.v4.CBG.obsv2
	cn-north-4.com.myhuaweicloud.v4.obsv2.lz11
	cn-north-4.com.myhuaweicloud.v4.storage.lz13
	cn-north-4.com.myhuaweicloud.v4.obsv2.lz25
拉美-墨西哥城二	la-north-2.com.myhuaweicloud.v4.obsv2

4. 虚拟私有云：为**表4-2**所示的所有终端节点服务选择**步骤4**所选择的虚拟私有云。
5. 子网：根据您的实际业务需要，为**表4-2**所示的终端节点服务分别选择子网。
6. 其他参数请根据实际业务需要进行设置。

步骤6 在终端节点列表，获取**步骤4**中为APIG、SWR服务创建的VPC终端节点对应的服务地址。

其中，${region}为**步骤3**获取到的值。

步骤7 创建内网域名，请参考**创建内网域名**。

1. 域名：请分别填写以下内网域名。

   说明

   请将以下内网域名中的${region}替换为**步骤3**获取到的值。

   - servicestage.${region}.myhuaweicloud.com
   - swr-api.${region}.myhuaweicloud.com
   - swr.${region}.myhuaweicloud.com
2. VPC：选择**步骤4**所选择的虚拟私有云。
3. 其他参数请根据您的实际业务需要进行设置。

步骤8 为**步骤7**创建的所有内网域名添加记录集。

1. 记录类型：选择“A – 将域名指向IPv4地址”。
2. 记录值：参考下表填写。

   说明

   请将下表中的${region}替换为**步骤3**获取到的值。

内网域名	记录值
servicestage.${region}.myhuaweicloud.com	输入**步骤6**获取到的com.myhuaweicloud.${region}.api终端节点服务对应的服务地址。
swr-api.${region}.myhuaweicloud.com	
swr.${region}.myhuaweicloud.com	输入**步骤6**获取到的com.myhuaweicloud.${region}.swr终端节点服务对应的服务地址。

3. 其他参数请根据您的实际业务需要进行设置。

----结束

5 软件中心

5.1 如何解决上传软件包失败的问题?

问题描述

上传满足系统要求的软件包后，系统提示“无权限访问，请联系管理员”。

解决方法

步骤1 在谷歌浏览器中，选择“更多工具 > 清除浏览数据”。

图 5-1 清除浏览数据

步骤2 在弹出的清除浏览数据窗口中，保持默认勾选，单击“清除数据”。

----结束

5.2 如何解决 Docker 客户端 push 镜像失败的问题

问题描述

在后台Docker登录成功以后，使用Docker客户端上传镜像包时，例如执行如下命令上传：

docker push *10.125.54.133:20202/test1*/**busybox:latest**

说明

- 10.125.54.133:20202为租户或用户准备上传仓库的IP和端口号。
- test1为namespace。

上传失败，Docker客户端出现如下提示：

```
unauthorized: authentication required
```

解决方法

步骤1 租户或用户以正确的AK/SK登录Docker客户端。

步骤2 向本租户或用户下有操作权限的namespace上传镜像，或者更换到新的namespace。

- 执行如下命令，向本租户或用户下有操作权限的namespace上传镜像。
  docker push *10.125.54.133:20202/test2*/**busybox:latest**

  说明

  - 10.125.54.133:20202为租户或用户准备上传仓库的IP和端口号。
  - test2为该租户或用户下有操作权限的namespace。

- 执行如下命令，更换到新的namespace。
  docker push *10.125.54.133:20202/test3*/**busybox:latest**

  说明

  - 10.125.54.133:20202为租户或用户准备上传仓库的IP和端口号。
  - test3为新的namespace。

步骤3 上传成功后，显示如下：

```
The push refers to a repository [10.125.54.133:20202/test2/busybox]
6a749002dd6a: Pushed
latest: digest: sha256:ecb3f3e96e003af6e02f0f47ac4d25a3b0585db54de0a82bb070f8cb78a79bc7 size: 527
```

出现异常，请联系技术支持工程师。

----结束

5.3 如何处理下载 SWR 软件包失败?

问题描述

创建tomcat应用时，界面显示创建失败。经查后台tomcat日志，原因为下载swr中的软件包时认证失败。手工下载swr中的软件包，报401。

解决方法

把镜像设置为公开即可，私有包会导致因权限问题无法被拉取。

6 基础设施

6.1 退订服务器是否影响现有程序运行?

问题描述

退订服务器是否影响现有程序运行?

解决方法

- 容器部署，退订服务器后服务实例将会在CCE集群内重新调度。
- 虚拟机部署，退订服务器后部署在该虚机上的服务实例将不可用，不会被重新调度。

6.2 如何使用 VPC 终端节点在安装虚拟机 Agent 时访问依赖的服务?

问题描述

VPC终端节点（VPC Endpoint），能够将VPC私密地连接到终端节点服务，使VPC中的云资源无需弹性公网IP就能够访问终端节点服务，提高了访问效率，为您提供更加灵活、安全的组网方式。

为虚拟机类型环境纳管的ECS安装虚拟机Agent时，可以通过已创建的VPC终端节点与LTS、AOM、OBS、SWR服务通信，并使用APIG通过配置好的内网域名调用ServiceStage、ECS、VPC、AOM的功能接口。

说明

仅“华东二”、“华南-广州”、“亚太-新加坡”、“非洲-约翰内斯堡”、“华东-上海一”、“华北-北京四”、“拉美-墨西哥城二”区域的ServiceStage支持通过VPC终端节点访问依赖的服务。

解决方法

步骤1 登录ServiceStage控制台。

步骤2 在区域列表选择您的业务所在区域，例如“亚太-新加坡”。

步骤3 在浏览器地址栏，获取“region”字段对应的值。

region取值示例如下面加粗内容所示：
https://console-intl.huaweicloud.com/servicestage/?agencyId=d6**********41&region=**ap-southeast-3**&locale=zh-cn#/overview

步骤4 分别为**表6-1**所示服务创建VPC终端节点，请参考**购买连接“接口”型终端节点服务的终端节点**。

1. 区域：请选择**步骤2**所选择的区域。
2. 服务类别：选择“按名称查找服务”。
3. 服务名称：请参考**表6-1**填写各云服务对应的终端节点服务名称，然后单击“验证”。

   说明

   请将下表中的${region}替换为**步骤3**获取到的值。

   表 6-1 连接“接口”型终端节点服务说明

云服务	终端节点服务名称
LTS	com.myhuaweicloud.${region}.lts-access
AOM	com.myhuaweicloud.${region}.aom-access
SWR	com.myhuaweicloud.${region}.swr **说明** 如果您选择的是“华南-广州”区域，SWR的终端节点服务名称是：swr.cn-south-1.myhuaweicloud.com。
APIG	com.myhuaweicloud.${region}.api

4. 勾选“创建内网域名”。
5. 虚拟私有云：根据您的实际业务需要，为**表6-1**所示的所有终端节点服务选择同一个虚拟私有云。
6. 子网：根据您的实际业务需要，为**表6-1**所示的所有终端节点服务分别选择子网。
7. 其他参数请根据实际业务需要进行设置。

步骤5 为OBS服务创建VPC终端节点，请参考**购买连接“网关”型终端节点服务的终端节点**。

需要为**表6-2**所示的OBS服务的终端节点服务分别创建VPC终端节点。

1. 区域：请选择**步骤2**所选择的区域。
2. 服务类别：选择“按名称查找服务”。
3. 服务名称：请参考**表6-2**填写对应区域OBS服务的终端节点服务名称。

表 6-2 连接“网关”型终端节点服务说明

区域	终端节点服务名称
华南-广州	cn-south-1.com.myhuaweicloud.v4.obsv2
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz05
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz08
	cn-south-1.com.myhuaweicloud.v4.obsv2.lz09
华东二	com.myhuaweicloud.cn-east-4.obslb01.v4.obsv2.vxlan
	com.myhuaweicloud.cn-east-4.obslb01.v6.obsv2.vxlan
亚太-新加坡	ap-southeast-3.com.myhuaweicloud.v4.obsv2
	ap-southeast-3.com.myhuaweicloud.v6.obsv2
非洲-约翰内斯堡	af-south-1.myhuaweicloud.v4.obsv2
	af-south-1.myhuaweicloud.v6.obsv2
华东-上海一	cn-east-3.com.myhuaweicloud.v4.global.obsv2
	cn-east-3.v4obsv2_new.58c9f146-63f0-4f07-98d3-18fe4874086b
	cn-east-3.myhuaweicloud.v4.obsv2.lz03
	cn-east-3.myhuaweicloud.v4.obsv2.lz04
	cn-east-3.com.myhuaweicloud.v4.obsv2.lz09
	cn-east-3.com.myhuaweicloud.v4.obsv2.lz11
华北-北京四	cn-north-4.com.myhuaweicloud.v4.obsv2.OBSCluster9
	cn-north-4.com.myhuaweicloud.v4.byte.obsv2
	cn-north-4.com.myhuaweicloud.v4.CBG.obsv2
	cn-north-4.com.myhuaweicloud.v4.obsv2.lz11
	cn-north-4.com.myhuaweicloud.v4.storage.lz13
	cn-north-4.com.myhuaweicloud.v4.obsv2.lz25
拉美-墨西哥城二	la-north-2.com.myhuaweicloud.v4.obsv2

4. 虚拟私有云：为**表6-2**所示的所有终端节点服务选择**步骤4**所选择的虚拟私有云。
5. 子网：根据您的实际业务需要，为**表6-2**所示的终端节点服务分别选择子网。
6. 其他参数请根据您的实际业务需要进行设置。

步骤6 在终端节点列表，获取**步骤4**中为APIG、SWR服务创建的VPC终端节点对应的服务地址。

其中，${region}为**步骤3**获取到的值。

步骤7 创建内网域名，请参考**创建内网域名**。

1. 域名：请分别填写以下内网域名。

   说明

   请将以下内网域名中的${region}替换为**步骤3**获取到的值。

   - servicestage.${region}.myhuaweicloud.com
   - ecs.${region}.myhuaweicloud.com
   - vpc.${region}.myhuaweicloud.com
   - aom.${region}.myhuaweicloud.com
   - swr-api.${region}.myhuaweicloud.com
   - swr.${region}.myhuaweicloud.com
2. VPC：选择**步骤4**所选择的虚拟私有云。
3. 其他参数请根据您的实际业务需要进行设置。

步骤8 为**步骤7**创建的所有内网域名添加记录集。

1. 记录类型：选择“A – 将域名指向IPv4地址”。
2. 记录值：参考下表填写。

   说明

   请将下表中的${region}替换为**步骤3**获取到的值。

内网域名	记录值
servicestage.${region}.myhuaweicloud.com	输入**步骤6**获取到的com.myhuaweicloud.${region}.api终端节点服务对应的服务地址。
ecs.${region}.myhuaweicloud.com	
vpc.${region}.myhuaweicloud.com	
aom.${region}.myhuaweicloud.com	
swr-api.${region}.myhuaweicloud.com	

内网域名	记录值
swr.${region}.myhuaweicloud.com	输入步骤6获取到的com.myhuaweicloud.${region}.swr终端节点服务对应的服务地址。

3. 其他参数请根据您的实际业务需要进行设置。

----结束

6.3 如何处理虚拟机 Agent 安装成功但是界面仍然显示缺少 Agent?

问题描述

虚拟机安装Agent完成后，显示安装成功，提示“Install agent success!”。

登录ServiceStage控制台，查看虚拟机状态仍然为“缺少Agent，请先安装”。

解决方法

步骤1 登录虚拟机Agent离线的弹性云服务器，请参考登录弹性云服务器。

步骤2 执行如下命令查看安装虚拟机Agent时选择的“授权模式”。

cd /opt/servicestage-agent

cat servicestage-agent.conf

- 如果返回结果中AK、SK的值为空，则“授权模式”为“委托授权”，请执行步骤3。
- 如果返回结果中AK、SK的值不为空，则“授权模式”为“AK/SK”，请执行步骤4。

步骤3 “授权模式”为“委托授权”，请执行以下操作：

1. 登录云服务器控制台。
2. 在左侧导航栏选择“弹性云服务器”，单击虚拟机Agent离线的弹性云服务器名称。
3. 在“基本信息”页签的“管理信息”区域，查看该弹性云服务器绑定的IAM委托名称。
4. 登录统一身份认证服务控制台。
5. 在左侧导航栏选择“委托”，单击**步骤3.3**获取到的委托名称。
   a. 选择“基本信息”页签，查看“云服务”是否为ECS服务。
   b. 选择“授权记录”页签，查看“权限”是否为Tenant Administrator。

   如果以上全部为是，请执行**步骤5**。

   如果以上任意一项为否，请先**修改委托**，然后执行**步骤3.6**。
6. 登录虚拟机Agent离线的弹性云服务器，请参考**登录弹性云服务器**。
7. 执行以下命令完成agent的重启，其中x.x.x要替换成环境中servicestage-agent的实际版本。

   cd /opt/servicestage-agent/servicestage-agent-x.x.x

   su agent ./servicestage-agent.sh start
8. 查看Agent状态是否在线。
   - 是，操作结束。
   - 否，请执行**步骤5**。

步骤4 “授权模式”为“AK/SK”，请执行以下操作：

1. 获取权限正确的或者创建新的AK、SK，请参考**访问密钥**。
2. 登录虚拟机Agent离线的弹性云服务器，请参考**登录弹性云服务器**。
3. 执行以下命令修改配置文件中的AK和SK的值，修改完后保存退出。

   cd /opt/servicestage-agent

   vi servicestage-agent.conf
4. 执行以下命令完成agent的重启，其中x.x.x要替换成环境中servicestage-agent的实际版本：

   cd /opt/servicestage-agent/servicestage-agent-x.x.x

   su agent ./servicestage-agent.sh start
5. 查看Agent状态是否在线。
   - 是，操作结束。
   - 否，请执行**步骤5**。

步骤5 如果以上方法不能解决问题，请联系技术支持工程师。

----结束

6.4 如何处理虚拟机 Agent 离线?

问题描述

Agent已经安装，但处于离线状态，不能正常工作。

解决方法

步骤1 登录虚拟机Agent离线的弹性云服务器，请参考**登录弹性云服务器**。

步骤2 执行如下命令查看安装虚拟机Agent时选择的“授权模式”。

cd /opt/servicestage-agent

cat servicestage-agent.conf

- 如果返回结果中AK、SK的值为空，则“授权模式”为“委托授权”，请执行**步骤3**。
- 如果返回结果中AK、SK的值不为空，则“授权模式”为“AK/SK”，请执行**步骤4**。

步骤3 “授权模式”为“委托授权”，请执行以下操作：

1. 登录云服务器控制台。
2. 在左侧导航栏选择“弹性云服务器”，单击虚拟机Agent离线的弹性云服务器名称。
3. 在“基本信息”页签的“管理信息”区域，查看该弹性云服务器绑定的IAM委托名称。
4. 登录统一身份认证服务控制台。
5. 在左侧导航栏选择“委托”，单击**步骤3.3**获取到的委托名称。
   a. 选择“基本信息”页签，查看“云服务”是否为ECS服务。
   b. 选择“授权记录”页签，查看“权限”是否为Tenant Administrator。

   如果以上全部为是，请执行**步骤5**。

   如果以上任意一项为否，请先**修改委托**，然后执行**步骤3.6**。
6. 登录虚拟机Agent离线的弹性云服务器，请参考**登录弹性云服务器**。
7. 执行以下命令完成agent的重启，其中x.x.x要替换成环境中servicestage-agent的实际版本。

   cd /opt/servicestage-agent/servicestage-agent-x.x.x

   su agent ./servicestage-agent.sh restart

步骤4 “授权模式”为“AK/SK”，请执行以下操作：

1. 获取权限正确的或者创建新的AK、SK，请参考**访问密钥**。
2. 登录虚拟机Agent离线的弹性云服务器，请参考**登录弹性云服务器**。
3. 执行以下命令修改配置文件中的AK和SK的值，修改完后保存退出。

   cd /opt/servicestage-agent

   vi servicestage-agent.conf
4. 执行以下命令完成agent的重启，其中x.x.x要替换成环境中servicestage-agent的实际版本：

   cd /opt/servicestage-agent/servicestage-agent-x.x.x

   su agent ./servicestage-agent.sh restart

步骤5 如果以上方法不能解决问题，请联系技术支持工程师。

----结束

6.5 如何解决虚拟机 Agent 亚健康?

问题描述

在环境详情查看纳管的虚拟机Agent状态为亚健康。

原因分析

虚拟机网络问题。

解决方法

步骤1 登录ServiceStage控制台。

步骤2 单击“环境管理”，进入“环境管理”页面。

步骤3 单击问题虚拟机所在的环境名称，进入环境详情页面。

步骤4 在“全部资源”页签下左侧列表，选择“计算”资源类型下的“弹性云服务器 ECS”或者“自定义服务器”资源名称。

步骤5 在右侧已纳管的资源列表，获取“Agent 状态”为“亚健康”虚拟机名称。

步骤6 登录虚拟机。

步骤7 执行如下命令打开虚拟机agent-critical日志：

```
vi /var/log/servicestage-agent/servicestage-agent-critical.log
```

步骤8 在日志中搜索上报告警的时间，根据时间定位到错误请求的日志信息。

步骤9 根据日志中的错误信息去解决问题，若无法解决问题请联系技术支持工程师。

----结束

6.6 如何处理开启了安全认证的微服务引擎专享版开启 IPv6 后服务注册失败?

问题描述

基于Java Chassis开发的微服务注册到开启了安全认证的微服务引擎专享版，微服务的注册发现地址使用微服务引擎服务注册发现的IPv4地址，可以注册成功并正常启动。

如果修改微服务的注册发现地址为微服务引擎注册发现的IPv6地址后，注册失败并报错“java.net.SocketException: Protocol family unavailable”。

可能原因

创建微服务引擎专享版时，当选择开启了IPv6的VPC网络时，创建引擎支持IPv6网络。当部署服务使用IPv6网段且选择容器部署时，选择的CCE集群需要开启IPv6双栈开关。

如果选择的CCE集群资源没有开启IPv6开关，就会导致服务网络不通，报错“java.net.SocketException: Protocol family unavailable”。

解决方法

步骤1 修改部署了微服务应用的环境，添加开启了“IPv6双栈”开关的CCE集群。

修改环境，请参考**修改环境**。

步骤2 重新部署应用，请参考**创建并部署组件**。

----结束

6.7 如何处理操作微服务引擎专享版时遇到非微服务引擎本身错误?

问题描述

在对微服务引擎专享版执行创建、删除、升级等操作时，可能会遇到非微服务引擎本身的错误。

例如，在创建微服务引擎专享版时，集群部署失败，报错如下：

{"error_code":"SVCSTG.00500400","error_message":"{\"kind\":\"Status\",\"apiVersion\":\"v1\",\"metadata\":{},\"status\":\"Failure\",\"code\":400,\"errorCode\":\"CCE.01400013\",\"errorMessage\":\"Insufficient volume quota.\",\"error_code\":\"CCE_CM.0307\",\"error_msg\":\"Volume quota is not enough\",\"message\":\"volume quota checking failed as [60/240] insufficient volume size quota\",\"reason\":\"QuotaInsufficient\"}"}

解决方法

页面上展示的错误信息中已经包含相应服务的错误码，根据错误码和错误信息联系相应服务的技术支持工程师提供支持。

6.8 如何将组件构建集群由共享构建集群切换为私有构建集群?

问题描述

ServiceStage提供的共享构建集群为所有用户共用，从安全和资源隔离的角度考虑，并不安全。

建议您在共享构建集群下线前，尽快将共享构建集群上的应用配置到私有构建集群，私有构建集群会提供更加安全可靠的服务。

前置操作

1. 已经创建集群，请参考**购买集群**操作。
2. 已为CCE集群节点绑定弹性IP，请参考**将弹性公网IP绑定至实例**。

解决方法

步骤1 登录ServiceStage控制台，选择“持续交付 > 构建”。

步骤2 选择“所有项目”、“所有创建类别”、“共享集群”和“所有状态”进行过滤，根据过滤结果判断是否需要切换。

- 如果过滤结果为空，操作结束。
- 如果过滤结果不为空，执行**步骤3**。

步骤3 进入编辑构建工程界面：

- “创建类别”为“系统创建”：选择“应用管理 > 应用组件”，单击待操作的组件名称，选择“构建 > 编辑”。
- “创建类别”为“用户创建”：在当前页面待操作构建名称所在列操作行，选择“更多 > 编辑”。

步骤4 编辑构建工程。

- 以源码部署的组件，请参考**编辑源码构建工程**编辑构建工程。
- 以软件包部署的组件，请参考**编辑软件包构建工程**。

其中：

1. “构建集群”选择“使用自己的构建集群”。
2. 在“构建集群”下拉列表选择私有构建集群。

----结束

6.9 如何解决微服务引擎创建过程中处理接入地址步骤失败?

问题描述

创建引擎过程中，处理接入地址步骤失败，报错提示：

```
{"error_code":"SVCSTG.00500404","error_message":"{"code":"VPC.0202","message":"Query resource by id xxx  fail.the subnet could not be found."}"}
```

问题原因

用户的项目未对CSE云服务进行委托授权。

解决方案

- 当您使用从ServiceStage发放的微服务引擎实例时，如想在CSE中发放新实例，需要对CSE云服务进行授权，具体操作可参考**授权使用微服务引擎**。
- 当您没有授予任何权限时，由于CSE使用依赖VPC云服务，因此需要先参考**创建委托**创建云服务委托cse_admin_trust，将操作权限委托给CSE。

6.10 微服务专享版引擎版本从 1.x 升级到 2.x 时有哪些注意事项?

微服务引擎专享版从1.x升级到2.x的过程中及升级完以后可能会出现的现象及解决方法如下：

- **现象1**：在微服务引擎专享版从1.x版本升级至2.x版本的过程中，使用接口获取配置或更新配置失败，报connection refused或Connection was closed，出现错误信息示例如下：

  ```
  [ERROR] Config update from xxx.xxx.xxx.xx failed. Error message is [Connection refused: xxx.xxx.xxx.xx]. org.apache.servicecomb.config.client.ConfigCenterClient$ConfigRefresh.lambda$null $13(ConfigCenterClient.java:428)
  ```

  或

  ```
  [ERROR]Config update from xxx.xxx.xxx.xx failed. Error message is [Connection was closed]. org.apache.servicecomb.config.client.ConfigCenterClient$ConfigRefresh.lambda$null $13(ConfigCenterClient.java:428)
  ```

  解决方法：微服务引擎专享版1.x版本升级至2.x版本时配置中心会有短暂的重启，重启期间获取配置或更新配置会报错断连。因此引擎升级过程中避免更新配置，升级完成后该问题即可解决。

- **现象2**：使用引擎版本为1.x配置中心接入的用户，无法使用“业务场景治理”功能。

  解决办法：由于引擎版本为2.x的配置中心换成了kie，需要将配置中心接入方式切换为kie，具体切换方式详见**Spring Cloud使用配置中心**中相关内容。

- **现象3**：在使用版本为2.x的微服务引擎时，使用导入配置文件功能，存在原配置中心格式的文件无法导入，提示文件为空或者格式错误。

  解决办法：将配置文件的配置项格式修改为2.x引擎要求的配置文件格式，新的配置文件为json文件，内容格式如下：

  ```
  {
     "data":[
        {
           "key":"xxx",
           "labels":{
              "environment":"xxx",
        "service": "xxx",
        "app": "xxx",
        "version": "xxx"
           },
           "value":"xxx",
           "value_type":"text",
           "status":"enabled"
        },
        {
           "key":"xxx",
           "labels":{
              "environment":"xxx"
           },
           "value":"xxx",
           "value_type":"text",
           "status":"enabled"
        },
        {
           "key":"xxx",
           "labels":{
              "environment":"xxx",
        "service": "xxx"
           },
           "value":"xxx",
           "value_type":"text",
           "status":"enabled"
        },
        {
           "key":"xxx",
           "labels":{
  ```

```
            "environment":"xxx",
        "service": "xxx",
        "app": "xxx"
            },
            "value":"xxx",
            "value_type":"text",
            "status":"enabled"
        }
    ]
}
```

其中：

- key和value是配置项对应的键和值，其为必填。
- labels是配置范围，其为必填，通过填写environment，service，app，version等字段来确定配置范围。
- value_type是配置项类型，其为必填，可以选择ini、json、text、yaml、properties、xml，默认为text。
- status是配置是否启用，其为选填，可以选择enabled（开启），disabled（关闭），默认关闭。

- **现象4**：若在微服务引擎1.x版本的配置中心设置了全局配置，当升级到2.x之后，全局配置根据配置中心升级后的范围会相应的自动调整作用范围environment=${environmentName}，environmentName取值可以为空、development、testing、acceptance或production。此时如果SDK调整以kie作为配置中心时，需要在项目配置文件中增加自定义标签以获取该部分配置，以下以environment=production为例展示：

  spring-cloud-huawei框架：

```
spring:
  cloud:
    servicecomb:
      config:
        serverType: kie
        kie:
          customLabel: environment
          customLabelValue: production
```

  servicecomb-java-chassis框架：

```
servicecomb:
  kie:
    customLabel: environment
    customLabelValue: production
```

6.11 获取配置失败

问题现象

微服务在接入相应的微服务开发框架（如spring-cloud-huawei、java-chassis）后，微服务通过SDK调用查询配置接口到微服务引擎获取配置项失败。

问题原因

微服务与注册中心间的连接因网络、CPU等其他因素发生抖动时，可能会导致请求异常。

解决方案

微服务框架具有自愈能力，拉取配置失败后会自动进行重试，一般情况下不会导致业务异常。您可以查看下次获取配置是否成功，若否，请联系技术支持人员。

7 应用运维

7.1 为什么 ServiceStage 中看不到日志?

ServiceStage中看不到日志，可能是由于待查看日志的主机未安装ICAgent或者用户业务日志输出位置为非标准位置导致的。

解决方法

- 待查看日志的主机未安装ICAgent

  ServiceStage的日志查看能力是由AOM服务提供的。主机是否安装ICAgent是使用AOM的日志能力的前提，否则将无法查看ServiceStage的日志。ICAgent是AOM的采集器，分别运行在每台主机上用于实时采集指标、日志和应用性能数据。

  如何为待查看日志的主机安装ICAgent，请参考**安装ICAgent**。

- 用户业务日志输出位置为非标准位置

  由于用户配置了日志策略，导致用户程序业务日志未输出到标准的输出位置。需参考如下方法进行排查处理：

  - 虚拟机部署

    排查配置的日志策略，是否把用户程序业务日志输出位置写到ServiceStage默认指定的虚拟机日志目录（ /var/log/application/${组件名}-${环境名}-${随机字符串}/${版本号}/${实例ID}/start_app.log ）外的其他目录。

    请查询业务代码，对日志策略进行调整。

  - 容器部署

    排查配置的日志策略，是否把业务日志输出到除标准输出外的其他地方。请参考**设置应用日志策略**进行相关配置。

7.2 替换弹性 IP 后应用访问方式失效后怎么办?

问题描述

当外网负载均衡绑定到应用时，如果把负载均衡的弹性IP替换掉，则应用访问方式上无法自动更新。

解决方法

需要手动删除之前的记录，重新添加新更换的IP，以新IP为访问地址的ELB访问方式。

7.3 如何处理启动一个新服务后导致一个节点内存使用率过高的问题?

问题描述

启动一个新服务导致一个节点内存使用率过高，怎么处理?

解决方法

设置**调度策略亲和性**，可以使服务实例按亲和性节点部署即可。

7.4 如何彻底把某个服务卸载关闭不再使用?

步骤1 登录ServiceStage控制台。

步骤2 选择“应用管理”。

步骤3 单击待操作的应用名称，进入“应用概览”页面。

步骤4 勾选所有组件，单击“批量删除”。

步骤5 在弹出对话框单击“确定”，完成组件删除。

步骤6 选择“微服务引擎 > 引擎实例”，进入微服务引擎列表。

步骤7 在页面上方“微服务引擎”下拉列表，选择注册了待删除服务的微服务引擎。

步骤8 选择“服务目录 > 微服务列表”。

- 未开启安全认证的微服务引擎，请执行**步骤10**。
- 开启安全认证的微服务引擎，请执行**步骤9**。

步骤9 在弹出的“安全认证”对话框输入账号名及其密码，单击“确定”。

说明

- 首次连接微服务引擎，请输入root账号名及**创建微服务引擎**时输入的密码。
- 创建账号请参考**新增账号**。

步骤10 删除该服务下对应的微服务，即可把服务彻底卸载关闭。

说明

删除资源需要对应的权限，如果提示没有权限，参考**ServiceStage权限说明**申请权限。

----结束

8 应用开发问题

8.1 微服务和普通应用有什么不同?

微服务是一种架构模式，其核心是将一个单体应用分成多个部分进行开发。所以微服务架构的应用程序，其本质上是一个分布式应用。

基于微服务架构构建的应用程序，可以让业务变化更快，整体系统可靠性更高。

类型	微服务	普通应用
开发	每个微服务的体量相对较小，业界的two pizza团队和“2周即可全部重写全部代码”等都可以作为微服务划分的参考。在开发时期，需注意服务接口的定义以与周边微服务进行配合，“基于契约”的开发方式是非常推荐的。 微服务开发，请参考**开发微服务应用**。	普通应用逻辑复杂、模块耦合、代码臃肿、修改难度大、版本迭代效率低下。
部署	微服务组成的应用系统通常比较复杂，在一次性部署的时候，需要进行编排部署。 微服务应用部署，请参考**创建并部署组件**。	普通应用可能会比较大，构建和部署时间也相应地比较长，不利于频繁部署，阻碍持续交付。在移动应用开发中，这个问题会显得尤为严重。
运维	在原来的指标监控、日志收集之外还非常强调治理。其核心理念是在运行时期通过对线上系统的各种调整以达到系统整体健康度要求的效果。 应用运维，请参考**组件运维**。	普通应用线上问题修复周期长，任何一个线上问题修复都需要对整个应用系统进行全面升级。

8.2 如何解决微服务注册失败问题（针对 java chassis）？

微服务部署成功后，需要将微服务注册到服务中心和配置中心，才能使用注册发现和微服务治理能力。如果注册失败，可能由于以下因素导致的：

- AK/SK未配置或者配置不正确。
- 服务中心或配置中心地址配置不正确。
- 网络不通。
- 域名解析失败。
- 监听端口被占用。

排查步骤

- 异常消息如下时，问题原因为AK/SK没有正确设置和携带到请求头里。
  ```
  {"errorCode":"401002","errorMessage":"Request unauthorized","detail":"Invalid request, header is invalid, ak sk or project is empty."}
  ```
  检查方法：
  a. 检查项目中是否依赖如下认证模块（间接依赖也可以，比如项目依赖了cse-solution-service-engine）。
  ```
  <groupId>com.huawei.paas.cse</groupId>
  <artifactId>foundation-auth</artifactId>
  ```
  b. 检查microservice.yaml文件中AK/SK是否配置正确，可以参考**下载AK/SK**获取正确的AK/SK。
  ```
  cse:
    credentials:
      accessKey: your access key
      secretKey: your secret key
      akskCustomCipher: default
  ```
- 异常消息如下时，问题原因为AK/SK配置不正确。
  ```
  {"errorCode":"401002","errorMessage":"Request unauthorized","detail":"Get service token from iam proxy failed,{\"error\":\"validate ak sk error\"}"}
  ```
  检查方法：
  检查microservice.yaml文件中AK/SK是否配置正确，可以参考**下载AK/SK**获取正确的AK/SK。
  ```
  cse:
    credentials:
      accessKey: your access key
      secretKey: your secret key
      akskCustomCipher: default
  ```
- 异常消息如下时，问题原因为Project配置不正确。
  ```
  {"errorCode":"401002","errorMessage":"Request unauthorized","detail":"Get service token from iam proxy failed,{\"error\":\"get project token from iam failed. error:http post failed, statuscode: 400\"}"}
  ```
  检查方法：
  检查microservice.yaml文件中Project是否配置正确，可以参考**查看项目名**获取正确的Project。
  ```
  cse:
    credentials:
      accessKey: your access key
      secretKey: your secret key
      akskCustomCipher: default
      project: cn-north-1
  ```
- 异常消息如下时，问题原因为没有足够的额度增加服务实例。


```
{"errorCode":"400100","errorMessage":"Not enough quota","detail":"no quota to create instance, ..."}
```

检查方法：

登录公有云，在微服务引擎概览页面，可以看到实例个数的额度。如果发现有剩余额度，需要检查下代码配置的服务中心地址和区域信息。需要注意的是您需要检查实例所在区域的额度。

- 当微服务链接不上服务中心或者配置中心时，微服务会打印如下信息。由于微服务并没有连上服务中心或配置中心，因此在服务中心或配置中心侧，看不到errorCode异常。

```
Connection refused: no further information
```

检查方法：

a. 检查microservice.yaml文件中服务中心和配置中心地址是否配置正确，如果不正确，请配置为正确的地址。

```
cse:
  service:                  //服务中心信息，其中address为服务中心地址
    registry:
      address: https://cse.cn-north-1.myhuaweicloud.com
      instance:
        watch: false
  config:                   //配置中心信息，其中address为配置中心地址
    client:
      serverUri: https://cse.cn-north-1.myhuaweicloud.com
      refreshMode: 1
      refresh_interval: 5000
```

b. 如果服务中心和配置中心地址配置正确，请执行以下命令检查网络是否良好：

ping *<servicecenter ip>*

ping *<configurationcenter ip>*

如果能**ping**通，表示网络良好。

> **说明**
>
> 当服务中心或配置中心配置的地址为域名时，需要将IP地址修改为域名，再执行**ping**命令。

c. 如果网络良好，执行以下命令获取服务中心或配置中心地址：

ping *<domain name>*

系统显示如下，请将获取到的IP地址和域名配置到本地“/etc/hosts”文件中。

```
10.153.78.18   cse.cn-north-1.myhuaweicloud.com
```

- 另外，微服务端口号被占用时，微服务可能无法正常启动，可执行如下命令查看服务监控端口是否被占用。

netstat -ano | findstr *8080*

如果端口被其他应用占用，请修改microservice.yaml文件，将端口修改为未被占用的端口。

```
rest:
  address: 0.0.0.0:8087  //微服务端口，请确保该端口号无冲突
```

8.3 部署在云上的微服务如何进行排错?

对于问题的定界，可以使用微服务仪表盘，通过仪表盘可以看到系统内所有微服务及其实例的实时运行情况，找到没有正常工作的节点。

找到问题节点后，可以通过APM查看问题节点的应用日志来分析具体问题。

8.4 如何决定使用 SDK 构建微服务还是使用 ServiceMesh?

- SDK方式适合完全自治的微服务，方便线下调试，但是需要引入SDK，基于SDK进行开发。
- Mesher的方式需要在部署的时候准备Mesher环境，开发方便，无需引入其他的SDK。

使用 Mesher 场景

- 将非Java语言编写的业务代码改造为微服务。
- 老旧的Java服务改造微服务。
- 想将非Java SDK开发的服务与Java SDK编写的服务对接。

使用 JAVA SDK 场景

- 使用了分布式事务。
- Java语言编写的微服务，尤其是新的微服务项目。
- mesher目前只支持http1.1，所以需要其他协议支持可以选择SDK。

8.5 如何解决获取依赖失败的问题?

配置maven镜像源之后，获取依赖失败，如图8-1所示。

图 8-1 获取依赖失败

对于企业内部需要使用代理访问外网的情况，需要配置Maven代理，可以在用户目录（windows中如*C:|Users|yang****|*）下的.m2目录中setting.xml（用户配置）或Maven安装目录下的conf目录中setting.xml（系统全局配置）里配置代理来实现。

找到setting.xml文件中的标签对，在其内配置代理信息，参考如下样例。

```
<proxies>
  <!-- proxy
   | Specification for one proxy, to be used in connecting to the network.
   |
      -->
  <proxy>
    <id>自定义proxyid，多个proxy配置间不可重复</id>
    <active>true</active>
    <protocol>http</protocol>
    <username>proxy认证账号</username>
    <password>proxy认证密码</password>
    <host>企业的代理地址</host>
```



```
    <port>代理地址端口号</port>
  </proxy>

  <proxy>
    <id>自定义proxyid，多个proxy配置间不可重复</id>
    <active>true</active>
    <protocol>http</protocol>
    <username>proxy认证账号</username>
    <password>proxy认证密码</password>
    <host>企业的代理地址</host>
    <port>代理地址端口号</port>
  </proxy>

</proxies>
```

8.6 服务名重复校验范围是什么?

问题描述

服务名重复校验范围是什么?

解决方法

服务名重复校验范围是微服务名称、微服务应用、微服务版本和微服务环境。

是一个微服务的主键，标识一个唯一的微服务。

请确保主键不重复。

8.7 为什么一定要定义服务契约?

企业级系统规模普遍较大，微服务组件众多，所以对服务间接口进行统一管理是企业的关键需求。微服务引擎通过契约管理满足这一需求。

管理角度：通过契约管理，企业中的接口管理者可以统一定义微服务的契约文件（符合接口描述标准的接口定义文件），从而做到规范并协调多个开发团队的接口开发，降低沟通成本且避免后期的混乱。

开发角度：在微服务开发的时候，不同团队甚至不同ISV间，可以基于统一的契约文件开发同一应用或系统，从而方便整体系统一致性的维护。具体表现在，单体应用中模块间是代码级调用，在编译期就可以解决API不兼容问题，修复成本也极低。微服务解耦后，服务间变为了远程调用，接口不一致通常发现时间较晚，会造成更大的修复成本。有了契约可以保证架构师设计契约，严格审查变更，并反向生成代码，保证兼容性。

另外，对于规模较小、统一管理要求不高的系统，产品支持从接口代码自动生成契约文件。

8.8 如何解决微服务应用开发过程中微服务开发框架同 netty 版本不匹配的问题?

问题描述

开发微服务应用时，运行日志提示如下错误：

```
"Caused by: java.lang.NoSuchMethodError:
io.netty.handler.codec.http.websocketx.WebSocketClientHandshakerFactory.newHandshaker(Ljava/net/
URI;Lio/netty/handler/codec/http/websocketx/WebSocketVersion;Ljava/lang/String;ZLio/netty/handler/codec/
http/HttpHeaders;IZZJ)Lio/netty/handler/codec/http/websocketx/WebSocketClientHandshaker;"
```

原因分析

通常是由于某个第三方软件引入了不匹配的版本依赖。

解决方法

可在开发环境下使用**mvn dependency:tree**命令查看依赖树，排查微服务开发框架同netty版本是否匹配。

例如，ServiceComb 2.0.1开发框架所匹配的netty依赖版本为4.1.45.Final。

使用maven管理复杂依赖关系，请参考：**https://servicecomb.apache.org/cn/docs/maven_dependency_management/**。

8.9 如何将 Java 或者 Tomcat 应用打包成压缩包用于虚拟机部署方式部署组件?

使用虚拟机部署方式部署组件时，ServiceStage支持将Java或者Tomcat应用打包成zip或者tar.gz压缩包用于部署。

应用压缩包内目录总体结构说明

- Java应用压缩包示例：**https://github.com/servicestage-demo/example/tree/master/servicestage-vm-demo/demoJavaExample.zip**
- Tomcat应用压缩包示例：**https://github.com/servicestage-demo/example/tree/master/servicestage-vm-demo/demoTomcatExample.zip**

以Java应用压缩包demoJavaExample.zip为例，应用压缩包内目录总体结构说明如下：

```
demoJavaExample/
|
├── scripts/
|   |   ├── pre-install.sh
|   |   ├── pre-stop.sh
|   |   ├── stop.sh
|   |   ├── uninstall.sh
|   |   ├── install.sh
|   |   ├── start.sh
|   |   ├── check.sh
|   |   ├── post-start.sh
├── packages/
```

```
│   │   ├── weather-1.0.0.jar
├── config/
│   │   ├── system.cfg
├── appspec.yml
```

压缩包名前缀必须和解压后的文件目录名一致。例如压缩包名为demoJavaExample.zip，解压后文件目录必须为demoJavaExample。

应用压缩包内各个目录及文件的作用说明如下：

- **scripts**：必选目录，存储的是应用各个生命周期执行的脚本文件。
- **packages**：必选目录，存储的是应用的jar包或者war包。
- **config**：必选目录，存储应用的配置信息。Java应用，存储的是system.cfg文件；Tomcat应用，存储的是system.cfg、logging.properties、server.xml文件。
- **appspec.yml**：必选文件，记录了生命周期的定义，也可以指定健康检查等信息。

appspec.yml 文件说明

如下所示，appspec.yml文件定义了整个部署的流程以及部署过程中使用到的环境变量和健康检查等内容。

```
spec:
  # 应用运行自定义用户
  deps:
    - name: "@os/linux/user@1.0"
      user: www
      group: www
      home: /home/www

  # 直接引入应用lifecycle脚本环境变量
  env:
  - name: APP_ENV
    value: "{{app.env}}"

  # 以文件形式引入保存于/opt/application/${appName}/${appVersion}/${instanceId}/servicestage-vmapp/
application.conf的应用lifecycle脚本环境变量
  value:
  - name: APP_VALUE
    value: "{{app.value}}"

  # 应用健康检查
  probes:
    # 接口健康检查
    # health:
    #   exec:
    #     method: GET
    #     request: http://127.0.0.1:8080/healthcheck
    #     timeout: 5

    # 脚本命令健康检查
    liveness:
      exec:
        command:
          - ps -ef | grep ${APP_HOME}/apache-tomcat-*/bin/bootstrap.jar | grep -v grep   # 目前只支持
APP_HOME可以取到环境变量
        timeout: 300
        runas: www
    retries: 10
    interval: 6

  # 应用lifecycle脚本
  lifecycle:
    pre-install:
```

```
  - command: scripts/pre-install.sh
    timeout: 300
install:
  - command: scripts/install.sh
    timeout: 300
check:
  - command: scripts/check.sh
    timeout: 300
    runas: www
start:
  - command: scripts/start.sh
    timeout: 300
    runas: www
post-start:
  - command: scripts/post-start.sh
    timeout: 300
    runas: www
pre-stop:
  - command: scripts/pre-stop.sh
    timeout: 300
    runas: www
stop:
  - command: scripts/stop.sh
    timeout: 300
    runas: www
uninstall:
  - command: scripts/uninstall.sh
    timeout: 300
```

- deps：此字段定义了应用运行自定义用户，建议参考如下示例固定。

  ```
  deps:
    - name: "@os/linux/user@1.0"
      user: www
      group: www
      home: /home/www
  ```

- env：此字段记录了各脚本运行时的环境变量，在脚本运行时可以直接通过环境引入。
- value：此字段记录的值会保存在${APP_HOME}/servicestage-vmapp/application.conf中。如果需要脚本中使用这些值，可以在脚本开始执行如下命令：

  ```
  #!/bin/bash
  . ${APP_HOME}/servicestage-vmapp/application.conf
  # dosomething
  ```

- probes：健康检查方式，可以是health、liveness。两种健康检查方式同时使用时，需要两种方式返回状态都正常，才能表明组件处于存活状态；只使用其中一种健康检查方式时，只需要使用的方式返回状态正常，就表明组件处于存活状态；两种健康检查方式都不使用时，除了停止、删除组件的场景，其余场景默认组件存活。
- health：访问健康检查接口，查看组件状态。接口返回状态为200到400（不包括400）之间的状态码时，表明组件处于存活状态。
  - method：查看组件状态时访问接口使用的方法，支持HTPP、HTTPS。
  - request：访问的地址。
  - body：访问的请求体。
  - timeout：超时时间，单位为秒。
- liveness：执行指定命令，查看组件状态。命令执行无报错且返回不为空时，表明组件处于存活状态。
  - command：查看组件状态使用的命令。
  - runas：执行查看组件状态命令时的Linux用户。

  - timeout：命令执行超时时间，单位为秒。
- retries：健康检查的重试次数。应用启动时生效，超过此次数应用仍未启动时，认为组件启动失败。默认重试次数为10次，支持自定义。
- interval：健康检查方式的重试间隔。本次重试失败后，距离下次重试开始的时间间隔为：已重试次数*默认重试间隔。默认重试间隔为6s，支持自定义。
- lifecycle：各个生命周期执行的脚本或者命令。
  - command：指定执行的脚本文件。必须是一个文件相对于${APP_HOME}文件夹所在的相对路径。
  - timeout：超时时间，单位为秒。
  - runas：执行的用户身份。

生命周期执行顺序：

- 部署的顺序：ServiceStage安装技术栈+pre-install.sh —>install.sh —>start.sh—>check.sh—>post-start.sh
- 升级/回滚的顺序：ServiceStage安装技术栈+pre-install.sh —>install.sh—>pre-stop.sh—>stop.sh—>uninstall.sh+ServiceStage卸载技术栈—>start.sh—>check.sh—>post-start.sh
- 删除的顺序：pre-stop.sh—>stop.sh—>uninstall.sh+ServiceStage卸载技术栈

config 说明

- system.cfg文件

  a. 设置config参数

  在**使用虚拟机部署方式创建并部署组件**时，您可以参考**添加配置项**设置一些配置项，ServiceStage会将设置的配置项以键值对的形式存储在${APP_HOME}/config/system.cfg文件中。您也可以预置一些配置参数在config里面，格式参考示例如下：

```
# system.cfg文件中存储内容的格式
key1=value1
key2=value2
```

  b. 引用config参数

  引用config参数的方法示例如下：

```
#!/bin/bash
 . ${APP_HOME}/config/system.cfg
 . ${APP_HOME}/config/user_config.cfg
 echo ${key1}
```

- server.xml文件

  请参考**tomcat的server.xml说明**。

- logging.properties文件

  请参考**tomcat的logging.properties说明**。

packages 目录说明

您需要将执行的jar包、war包等内容保存在此文件夹中，在脚本中使用命令进行执行。

scripts 目录说明

此文件夹用于存储应用各个生命周期执行的脚本。

系统默认配置

系统默认配置均在${APP_HOME}/servicestage-vmapp/application.conf文件中定义，并不会直接出现在环境变量中。application.conf文件中定义的环境变量需要在脚本中增加如下命令后才能引用。

```
. ${APP_HOME}/servicestage-vmapp/application.conf
```

可直接引用的脚本环境变量包括APP_HOME、在appspec.yml中指定的环境变量和**添加组件环境变量**时指定的环境变量。

文件内容示例如下：

```
export LOG_PATH=/var/log/application/zqb-4-vm-wqd-2-7f6fbc/3c719644-f9f5-46b4-a06a-61fcf163e5b5
export APP_HOME=/opt/application/zqb-4-vm-wqd-2-7f6fbc/2023.1207.11314/3c719644-f9f5-46b4-a06a-61fcf163e5b5
export TOMCAT_STACK_HOME=/opt/application/zqb-4-vm-wqd-2-7f6fbc/2023.1207.11314/3c719644-f9f5-46b4-a06a-61fcf163e5b5/apache-tomcat-8.5.82
export JRE_STACK_HOME=/opt/application/zqb-4-vm-wqd-2-7f6fbc/2023.1207.11314/3c719644-f9f5-46b4-a06a-61fcf163e5b5/jre1.8
export APP_VALUE="{{app.value}}"
export APP_USER=www
export APP_GROUP=www
```

- LOG_PATH：日志记录的地址，需要统一将应用日志打印到LOG_PATH下的日志文件中，命名为{lifecycle}_app.log，方便后续的日志上报。
- APP_HOME：记录了当前应用运行的环境。
- TOMCAT_STACK_HOME：Tomcat的主目录。
- JRE_STACK_HOME：Jre的主目录。
- APP_VALUE：在appspec.yml中指定的环境变量。
- APP_USER：文件所属的用户。建议固定为www，也支持自定义用户。
- APP_GROUP：文件的属组。建议固定为www，也支持自定义用户属组。

如果自定义了用户及其属组，需要执行如下命令赋予该自定义用户运行脚本的权限：

```
sudo -- /bin/bash -c "chmod -R 755 ${APP_HOME}/scripts/* ${LOG_PATH};chown -R ${APP_USER}:${APP_GROUP} ${APP_HOME}/scripts/start.sh;chown -R ${APP_USER}:${APP_GROUP} ${APP_HOME}/scripts/stop.sh;chown -R ${APP_USER}:${APP_GROUP} ${LOG_PATH} ${JRE_STACK_HOME}/"
```

脚本编写说明

Servicestage在执行脚本时的目录结构如下所示：

```
APP_HOME/
|
├── scripts/
|   |       ├── pre-install.sh
|   |       ├── pre-stop.sh
|   |       ├── stop.sh
|   |       ├── uninstall.sh
|   |       ├── install.sh
|   |       ├── start.sh
|   |       ├── check.sh
|   |       ├── post-start.sh
|   ├──packages/
|   |       ├── my-app.jar/my-app.war
├── config/
|   |       ├── system.cfg
├── servicestage-vmapp/
|   |       ├── application.conf
├── jre1.8
├── apache-tomcat-8.5.82
```

- 在编写脚本时，可以先进入${APP_HOME}目录，获取其他文件的位置。
- 若想获取jre或者tomcat的位置，可以在脚本开始位置执行。${APP_HOME}/servicestage-vmapp/application.conf加载环境变量，再通过${JRE_STACK_HOME}或${TOMCAT_STACK_HOME}引用获取。
- Servicestage会将${LOG_PATH}目录下的日志上报，因此需要统一将应用日志打印到${LOG_PATH}目录下的日志文件中，命名为{lifecycle}_app.log，方便后续日志上报。
- 建议在使用时，脚本都是使用如下几行作为开头，用于获取环境变量、用户配置、指定技术栈HOME、指定日志输出路径、指定当前执行地址等。

```
#!/bin/bash

# 获取用户参数、指定环境变量
. ${APP_HOME}/config/system.cfg
. ${APP_HOME}/servicestage-vmapp/application.conf

# 指定技术栈HOME
JRE_HOME=${JRE_STACK_HOME}
TOMCAT_HOME=${TOMCAT_STACK_HOME}

#指定日志输出地址
installLog="${LOG_PATH}/install_app.log"

#打印日志使用方法
function writeLog()
{
    msg="$1\n"
    printf "[`date '+%Y-%m-%d %H:%M:%S'`] $msg" | sudo tee -a ${installLog};
}

writeLog "-----begin!------"
```

- 如果使用自定义用户运行脚本，需要将stop.sh脚本中如下语句中的www运行用户名称替换为您的自定义运行用户名称。

  替换前：

  ```
  ps -u www -ef|grep -v grep|grep ${JRE_HOME}/bin/java|awk '{print $2}
  ```

  替换后：

  ```
  ps -u {您的自定义运行用户} -ef|grep -v grep|grep ${JRE_HOME}/bin/java|awk '{print $2}
  ```

  例如：

  ```
  ps -u app-user -ef|grep -v grep|grep ${JRE_HOME}/bin/java|awk '{print $2}
  ```

tomcat 的 server.xml 说明

- 在创建tomcat组件时可以提前准备server.xml，将server.xml复制到tomcat的目录下（这一步骤建议在install生命周期中执行）。

```
#!/bin/bash
. ${APP_HOME}/servicestage-vmapp/application.conf
cp ${APP_HOME}/conf/server.xml ${TOMCAT_STACK_HOME}/conf/server.xml
```

- ServiceStage在${TOMCAT_STACK_HOME}/conf/server.xml有一份预置的server.xml文件，其中有四个占位符，使用时需要将其正确的替换，可以参考如下代码：

```
#!/bin/bash
. ${APP_HOME}/servicestage-vmapp/application.conf
function replacePara()
{
    sWord=$1
    dWord=$2
    theFile=$3
    if [[ "$sWord" == "" || "$theFile" == "" ]]
    then
```



```
        writeLog "[ERROR] ReplacePara has empty param, \$1:$sWord, \$3:$theFile"
        return 1
    fi

    if [[ ! -f ${theFile} ]]
    then
        writeLog "[ERROR] File $theFile does not exist."
        return 1
    fi
    count=`grep -c "$sWord" ${theFile}`
    if [[ "${count}" == "0" ]];then
        return 0
    fi
    sed "s#$sWord#${dWord}#g" ${theFile} > ${theFile}.temp
    if [[ "$?" != "0" ]]
    then
        writeLog "[ERROR] Sed command in replacePara error."
        return 1
    fi
    mv -f ${theFile}.temp ${theFile}
}
replacePara "@{LOG_FILE_PATH_APP}" "${LOG_FILE_PATH_APP}" ${TOMCAT_HOME}/conf/server.xml
replacePara "@{http_port}" "${instance_port_port}" ${TOMCAT_HOME}/conf/server.xml
replacePara "@{server_port}" "${server_port}" ${TOMCAT_HOME}/conf/server.xml
replacePara "@{APP_PACKAGE_NAME}" "examples.war" ${TOMCAT_HOME}/conf/server.xml
```

- @{server_port}：tomcat服务器的端口号。
- @{APP_PACKAGE_NAME}：war包相对于${TOMCAT_STACK_HOME}目录的位置。
- @{http_port}： tomcat的Connector的端口号。
- @{LOG_FILE_PATH_APP}：tomcat日志打印的路径。

tomcat 的 logging.properties 说明

您需要自行准备logging.properties，将其复制到tomcat的目录下，复制代码示例如下：

```
#!/bin/bash
. ${APP_HOME}/servicestage-vmapp/application.conf
cp ${APP_HOME}/conf/logging.properties ${TOMCAT_STACK_HOME}/conf/logging.properties
```

logging.properties可以参考如下内容，使用时将@{LOG_FILE_PATH_APP}替换为真正的日志目录。

```
handlers = 1catalina.org.apache.juli.AsyncFileHandler, 2localhost.org.apache.juli.AsyncFileHandler,
3manager.org.apache.juli.AsyncFileHandler, 4host-manager.org.apache.juli.AsyncFileHandler,
java.util.logging.ConsoleHandler

.handlers = 1catalina.org.apache.juli.AsyncFileHandler, java.util.logging.ConsoleHandler

1catalina.org.apache.juli.AsyncFileHandler.level = FINE
1catalina.org.apache.juli.AsyncFileHandler.directory = @{LOG_FILE_PATH_APP}
1catalina.org.apache.juli.AsyncFileHandler.prefix = catalina.

2localhost.org.apache.juli.AsyncFileHandler.level = FINE
2localhost.org.apache.juli.AsyncFileHandler.directory = @{LOG_FILE_PATH_APP}
2localhost.org.apache.juli.AsyncFileHandler.prefix = localhost.

3manager.org.apache.juli.AsyncFileHandler.level = FINE
3manager.org.apache.juli.AsyncFileHandler.directory = @{LOG_FILE_PATH_APP}
3manager.org.apache.juli.AsyncFileHandler.prefix = manager.

4host-manager.org.apache.juli.AsyncFileHandler.level = FINE
4host-manager.org.apache.juli.AsyncFileHandler.directory = @{LOG_FILE_PATH_APP}
4host-manager.org.apache.juli.AsyncFileHandler.prefix = host-manager.
```



```
java.util.logging.ConsoleHandler.level = FINE
java.util.logging.ConsoleHandler.formatter = org.apache.juli.OneLineFormatter
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].handlers =
2localhost.org.apache.juli.AsyncFileHandler

org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/manager].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/manager].handlers =
3manager.org.apache.juli.AsyncFileHandler

org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/host-manager].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/host-manager].handlers = 4host-
manager.org.apache.juli.AsyncFileHandler

org.apache.catalina.core.ContainerBase.[Catalina].[localhost].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].handlers =
2localhost.org.apache.juli.AsyncFileHandler

org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/manager].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/manager].handlers =
3manager.org.apache.juli.AsyncFileHandler

org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/host-manager].level = INFO
org.apache.catalina.core.ContainerBase.[Catalina].[localhost].[/host-manager].handlers = 4host-
manager.org.apache.juli.AsyncFileHandler
```

FAQ

- 如何以root账户运行脚本命令?
  a. 将要以root账户运行的内容编写成一个脚本root-install.sh。
  b. 在真正要执行的install.sh执行如下语句，其中${ROOT_PASSWORD}为root账号的密码，-c后面的双引号中为执行的命令。切换用户后会丢失环境变量，可以在sh前加上所需要的环境变量。
     ```
     echo "${ROOT_PASSWORD}" | su - root -c "APP_HOME=${APP_HOME} sh root-install.sh"
     ```
  c. 若执行上述语句发生“su: Permission denied”报错，找到/etc/pam.d/su文件，将“ auth required pam_wheel.so use_uid”这一句注释掉。

