Huawei Cloud EulerOS (HCE)

用户指南

文档版本	01
发布日期	2025-09-18

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 使用概述

您可通过下列方法使用Huawei Cloud EulerOS。

- 首次创建弹性云服务器实例时，推荐使用HCE公共镜像。
- 将操作系统切换为HCE。

  如果现有的弹性云服务器配置（网卡、磁盘、VPN等配置的类型和数量）都不需要改变，仅需要修改弹性云服务器的操作系统镜像，并且您的软件和原操作系统耦合度较低，适配到HCE改动较小，建议使用**系统切换**，可快速切换到HCE。
- 将操作系统迁移为HCE。

  如果现有的弹性云服务器配置（网卡、磁盘、VPN等配置的类型和数量）都不需要改变，且希望保留操作系统软件的配置参数，可以通过操作**系统迁移**的方式迁移到HCE。

  说明

  仅支持迁移至Huawei Cloud EulerOS 2.0标准版和Huawei Cloud EulerOS 1.1CentOS兼容版，不支持迁移至其他HCE镜像版本。

区别	系统切换	系统迁移
数据备份	• 切换操作系统会清除系统盘数据，包括系统盘上的系统分区和所有其它分区。 • 切换操作系统不影响数据盘数据。	• 迁移操作系统不会清除系统盘数据，为避免系统软件的数据丢失，建议将其备份。 • 迁移操作系统不影响数据盘数据。
个性化设置	切换操作系统后，当前操作系统内的个性化设置（如DNS、主机名等）将被重置，需重新配置。	迁移操作系统后，当前操作系统内的个性化设置（如DNS、主机名等）不需重新配置。

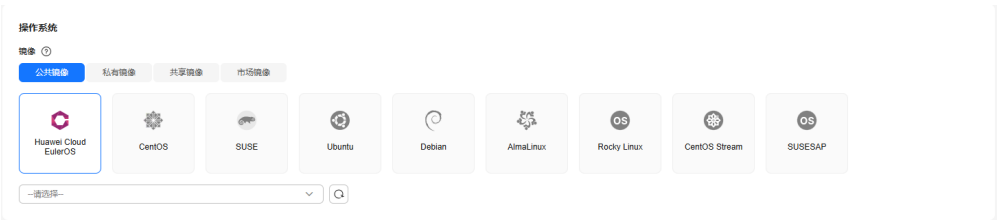
2 新创建弹性云服务器时选择 HCE

操作步骤

1. 登录控制台，进入**购买弹性云服务器**页面。
2. 选择HCE公共镜像。

   在基础配置环节选择公共镜像时，选择Huawei Cloud EulerOS操作系统和具体的镜像版本。购买弹性云服务器完整的操作步骤详见**购买ECS**。

   图 2-1 选择公共镜像



3 系统切换

约束与限制

- “包年/包月”方式购买的弹性云服务器切换操作系统时，由于所选镜像不同，当前云服务器的系统盘容量可能不足，不支持切换后的镜像使用。此时，需先卸载系统盘并进行扩容，然后再重新切换操作系统。
- 云硬盘的配额须大于0。
- 不支持BIOS启动方式与UEFI启动方式的操作系统互相切换。

切换须知

- 切换操作系统后，将不再保留原操作系统，并删除原有系统盘及清除系统盘数据，包括系统盘上的系统分区和所有其它分区，请做好数据备份。详细内容，请参考**备份弹性云服务器**。
- 切换操作系统不影响数据盘数据。
- 切换操作系统后IP地址和MAC地址不发生改变。
- 切换操作系统成功后会自动开机。
- 切换操作系统后不支持更换系统盘的云硬盘类型。
- 切换操作系统后，您的业务运行环境需要在新的系统中重新部署。
- 切换操作系统后，原操作系统内的个性化设置（如DNS、主机名等）将被重置，需重新配置。
  - 重新配置云服务器DNS信息请参考：**怎样配置弹性云服务器的DNS和NTP信息?**
  - 重新配置主机名请参考：**怎样使修改的静态主机名永久生效?**

计费规则

- 切换操作系统功能不收费。按需计费的云服务器切换成功后，系统将按照新的配置费用（系统盘、规格）进行计费。
- “按需付费”方式购买的弹性云服务器切换操作系统后，由于所选镜像不同，系统盘的容量可能会增大，由此将带来费用的变更，具体收费请参见**产品价格详情**。
- 切换操作系统时产生的费用，不支持使用代金券支付。

前提条件

- 待切换操作系统的挂载点包含系统盘。
- 如果原服务器使用的是密码登录方式，切换操作系统后使用密钥登录方式，请提前创建密钥文件。
- 如果您使用私有镜像切换操作系统，请参考**《镜像服务用户指南》**提前完成私有镜像的制作。
  - 如果需要指定云服务器的镜像，请提前使用指定云服务器创建私有镜像。
  - 如果需要使用本地的镜像文件，请提前将镜像文件导入并注册为云平台的私有镜像。
  - 如果需要使用其他区域的私有镜像，请提前复制镜像。
  - 如果需要使用其他账号的私有镜像，请提前完成镜像共享。

操作步骤

1. 登录管理控制台。
2. 单击“ ☰ ”，选择“ “计算” ->“弹性云服务器”

图 3-1 选择弹性云服务器

3. 在待切换操作系统的弹性云服务器的“操作”列下，单击“更多”->“镜像””->“切换操作系统”。

   切换操作系统前请先将云服务器关机，或根据页面提示勾选“立即关机（重装操作系统前需先将云服务器关机）”。
4. 根据需求选择需要更换的规格，包括“镜像类型”和“镜像”。

   说明

   对于“包年/包月”方式购买的，如果系统盘容量小于您选择的待切换镜像的大小，此时，您需要先卸载系统盘，并进行扩容，然后再挂载至原执行切换操作。

   扩容系统盘的操作指导，请参见**“扩容云硬盘”**章节。

图 3-2 切换操作系统

5. 设置登录方式。

   如果待切换操作系统的弹性云服务器是使用密钥登录方式创建的，请参考《弹性云服务器ECS用户指南》中的**密码和密钥对管理**章节。

6. 单击“确定”。
7. 在“切换云服务器操作系统”页面，确认切换的操作系统规格无误后，阅读并勾选相关协议或声明，单击“提交申请”。

   提交切换操作系统的申请后，弹性云服务器的状态变为“切换中”，当该状态消失后，表示切换结束。

图 3-3 查看任务详情

后续处理

- 如果切换操作系统前后都是Linux系统，且数据盘设置了开机自动挂载分区。切换操作系统后，数据盘分区挂载信息会丢失，请更新/etc/fstab配置。

  a. 在/etc/fstab写入切换后的分区信息。写入之前建议您先备份/etc/fstab文件。

     详细操作请参考**初始化Linux数据盘（fdisk）**，设置开机自动挂载磁盘分区。

  b. 挂载分区。挂载分区后即可开始使用数据盘。

     mount *diskname mountpoint*

  c. 执行以下命令，查看挂载结果。

     df -TH

- 如果操作系统切换失败，公有云平台支持重试功能，用户可重新执行步骤**2**-步骤**7**，切换操作系统。
- 重试后，如果仍未成功，可联系客服进行人工恢复。

4 系统迁移

4.1 x2hce-ca 应用兼容性评估

4.1.1 工具概述

x2hce-ca是华为云为系统迁移提供的一款免费的应用兼容性评估工具。

x2hce-ca通过对待迁移应用进行快速扫描分析，帮助您评估应用在源操作系统和目标操作系统的兼容性。

表 4-1 支持兼容性评的 x86 公共镜像

OS发行系列	源操作系统	目标操作系统
HCE	64bit：Huawei Cloud EulerOS：1.1	HCE 2.0 标准版 64位
EulerOS	64bit：EulerOS：2.11/2.10/2.9/2.5/2.2	HCE 2.0 标准版 64位
CentOS	64bit：CentOS 7：7.9/7.8/7.7/7.6/7.5/7.4/7.3/7.2/7.1/7.0 64bit：CentOS 8：8.3/8.2/8.1/8.0	HCE 2.0 标准版 64位
	64bit：CentOS 7：7.9/7.6	Huawei Cloud EulerOS 1.1 CentOS兼容版

表 4-2 支持兼容性评估的 Arm 公共镜像

OS发行系列	源操作系统	目标操作系统
EulerOS	64bit：EulerOS：2.11/2.10/2.9/2.8	HCE 2.0 标准版 64位 Arm版

4.1.2 约束限制

- 由于x2hce-ca工具安装会有额外资源包引入，不建议在业务环境中运行。x2hce-ca工具支持在HCE和CentOS的操作系统进行安装使用。
- x2hce-ca工具支持扫描的文件格式为jar、py、pyc、bin、sh、rpm、ko。其中，针对rpm格式文件只支持源码为C、C++、Java和Python语言。
- x2hce-ca工具不支持回滚，任务异常中断后会在/opt/x2hce-ca/目录下产生残留文件，并不影响工具再次使用。异常中断的任务请重新执行。
- 安装和运行x2hce-ca工具的系统参数要求如表4-3所述。

表 4-3 运行 x2hce-ca 工具的操作系统参数要求

硬件类型	说明
架构	x86_64、aarch64
CPU	双核及以上
内存	系统空闲内存要求8GB及以上
系统盘/根分区存储容量	20GB及以上

4.1.3 安装 x2hce-ca

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，HCE 2.0的配置如下。

```
[base]
name=HCE $releasever base
baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

须知

如果在CentOS上面安装x2hce-ca，无需配置hce的repo源，只需要在华为云官网 repo 源获取最新版本的 x2hce-ca-hce 工具包：**https://repo.huaweicloud.com/hce/2.0/updates/**。

2. 安装x2hce-ca。

   通过**yum install -y x2hce-ca-hce**命令安装工具。安装完成后，生成表4-4列表中的目录。

   表 4-4 用户相关目录列表

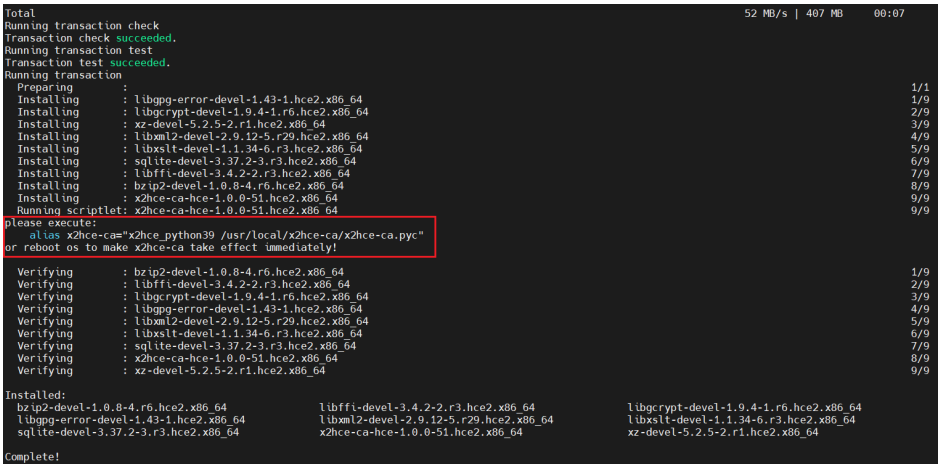
目录	说明
/var/log/x2hce-ca	存放工具日志文件的目录。
/var/log/aparser	存放配置收集器日志文件的目录。
/opt/x2hce-ca/output	报告默认输出目录。
/opt/x2hce-ca/scan	待扫描应用软件包的建议存放目录。
/etc/x2hce-ca/config	存放静态配置文件的目录。
/etc/x2hce-ca/database_2.0.0.630	存放数据库文件的目录。
/usr/local/x2hce-ca	程序文件存放路径。
/usr/local/x2hce-python-3	工具自带python安装目录

   须知

   如果在CentOS上面安装x2hce-ca，直接使用第1步下载的 x2hce-ca-hce 工具包安装，通过**yum install -y x2hce-ca-hce-1.0.0-53.hce2.x86_64.rpm**命令安装工具，注意将版本号替换为您实际下载到的版本。

3. 为使**x2hce-ca**命令生效，请按照安装完成后的提示进行相应配置。提示样例如图4-1所示，此处是提醒用户执行命令**alias x2hce-ca="x2hce_python39 /usr/local/x2hce-ca/x2hce-ca.pyc"**或重启操作系统使**x2hce-ca**命令生效。

   图 4-1 x2hce-ca-hce 安装后提示样例



4.1.4 评估软件兼容性

扫描方式

x2hce-ca工具支持两种软件包扫描方式，请明确要使用的扫描方式和待评估的软件包。

- 扫描源操作系统上单个或多个应用软件包。
- 扫描源操作系统上单个或多个目录下的所有应用软件包。

操作步骤

1. 默认登录并切换到root用户下使用工具。
2. 使用如下命令对软件包进行兼容性扫描。
   x2hce-ca scan <option> [**-os_name** *源系统名称*] [**-target_os_name** *目标系统名称*]

   说明

   使用以下命令验证Java默认版本：

   java -version

   - 若目标机器上已安装Java1.8.0，则自动执行后续扫描。
   - 若目标机器上未安装Java1.8.0，根据操作系统的不同处理方式有所不同。
     - 操作系统是HCE 2.0时，会自动安装缺失的Java依赖：java-1.8.0-openjdk-devel、java-1.8.0-openjdk和java-1.8.0-openjdk-headless。
     - 操作系统不是HCE 2.0时，会出现报错信息提示安装缺失的Java依赖。推荐执行以下命令自行安装。

       yum -y install java-1.8.0-openjdk-devel
   - 若目标机器上存在多个Java版本，且默认版本不是1.8.0时，用户需要执行以下命令手动设置Java默认版本为1.8.0。

     update-alternatives --config java

   <option>有如下设置：

   – Dir_Name/App_Name，扫描单个应用软件包。

     以x86_64和aarch64操作系统架构为例：

     扫描/mnt/路径下的应用软件包NetworkManager-1.18.8-1.el7.x86_64.rpm。

     ```
     x2hce-ca scan /mnt/NetworkManager-1.18.8-1.el7.x86_64.rpm -os_name centos7.9 -target_os_name hce2.0
     ```

     扫描/mnt/路径下的应用软件包NetworkManager-1.18.8-1.el7.aarch64.rpm。

     ```
     x2hce-ca scan /mnt/NetworkManager-1.18.8-1.el7.aarch64.rpm -os_name EulerOSV2.0SP8arm -target_os_name hce2.0arm -arch aarch64
     ```

     说明

     其中**-arch**的默认值为x86_64。

   – Dir_Name1/App_Name1 Dir_Name2/App_Name2，扫描多个应用软件包。

     以x86_64操作系统架构为例：

     扫描/opt/x2hce-ca/scan/路径下的应用软件包grep-3.4-0.h3.r3.eulerosv2r9.x86_64.rpm和/opt/x2hce-ca/scan/rpm/路径下的应用软件包groff-1.22.4-5.h1.eulerosv2r9.x86_64.rpm。

     ```
     x2hce-ca scan /opt/x2hce-ca/scan/grep-3.4-0.h3.r3.eulerosv2r9.x86_64.rpm /opt/x2hce-ca/scan/rpm/groff-1.22.4-5.h1.eulerosv2r9.x86_64.rpm -os_name centos7.9 -target_os_name hce2.0
     ```

- -b Dir_Name，扫描单个目录下的所有应用包。
  例如，扫描directory1目录下的所有应用包。
  ```
  x2hce-ca scan -b directory1 -os_name centos7.9  -target_os_name hce2.0
  ```
- -b Dir_Name1 Dir_Name2，扫描多个目录下的所有应用包。
  例如，扫描directory1和directory2目录下的所有应用包。
  ```
  x2hce-ca scan -b directory1 directory2 -os_name centos7.9 -target_os_name hce2.0
  ```

  说明

  建议单个目录下放置不超过750个文件，且文件总大小不超过900M，过多的软件包可能会导致工具故障。
- -l rpm_Name，扫描本地安装的软件。
  例如，扫描openssl。
  ```
  x2hce-ca scan -l openssl
  ```
- -l rpm_Name1,rpm_Name2...，扫描多个本地安装的软件。
  例如，扫描openssh和openssl。
  ```
  x2hce-ca scan -l openssl,openssh
  ```

  说明

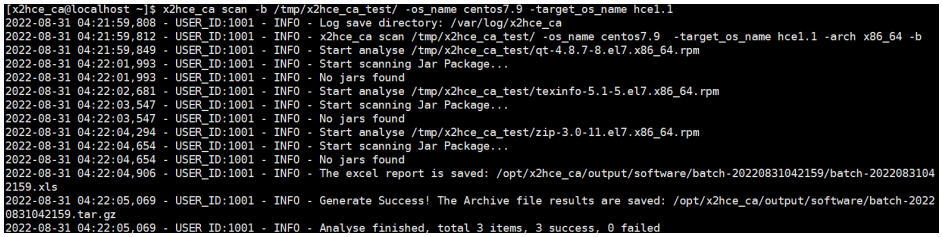
  x2hce-ca只在CentOS系统中支持-l参数。

表 4-5 参数类型

参数	参数类型	说明
-os_name	String	源操作系统。 可选参数，默认参数为centos7.9，其他参数可参考**工具概述**。 例如设置为-os_name centos8.2，指选择源操作系统为CentOS 8.2。
-target_os_name	String	目标操作系统。 可选参数， HCE 2.0中默认参数为hce2.0其他参数可参考**工具概述**。 例如设置为-target_os_name hce1.1，指选择目标操作系统为Huawei Cloud EulerOS 1.1。

3. 结果分析。
   以扫描/tmp/x2hce-ca_test目录下的三个RPM包为例，命令执行后的输出如**图4-2**所示。

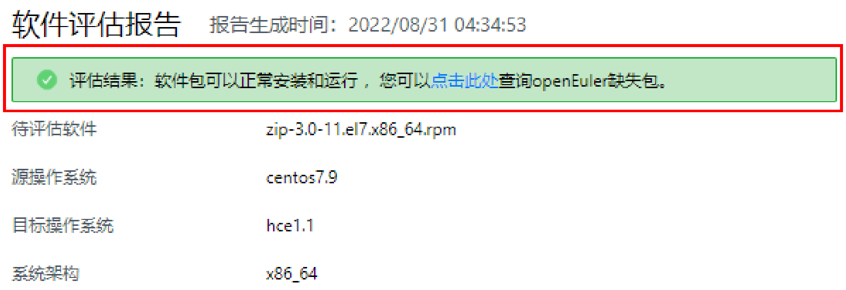
图 4-2 扫描结果



– 软件包的兼容性评估报告保存在/opt/x2hce-ca/output/software/目录下，请自行下载查看具体评估结果。

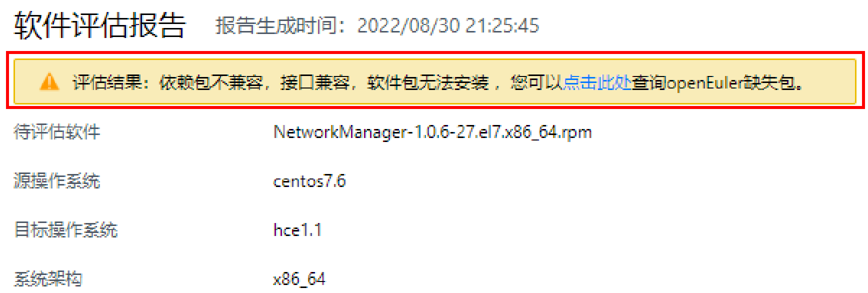
  ▪ 每个软件包有同名HTML格式文件，软件包兼容的评估结果如图4-3。

图 4-3 软件包兼容的评估结果



软件包不兼容的评估结果如图4-4。

图 4-4 软件包不兼容的评估结果



  ▪ 详细的依赖包兼容性、接口兼容性等信息可在软件包同名xlsx格式文件中查看。

图 4-5 依赖包兼容性和接口兼容性信息

待评估软件	源操作系统	目标操作系统	系统架构	评估项	报告运行时	报告生成时间	直接依赖包数	待确认包	调用外部接口数	待确认接口	接口兼容百分	依赖包兼容百分	源软件包	目标软件包
zip-3.0-11.el7.x86_64.rpm	centos7.9	hce1.1	x86_64	direct dependence, function interface	0.477	20220831043453	2	0	100	0	100%	100%	zip-3.0-11.el7.x86_64.rpm	zip-3.0-11.hce1c.x86_64.rpm
zip-3.0-11.el7.x86_64.rpm	centos7.9	hce1.1	x86_64	direct dependence, function interface	0.477	20220831043453	2	0	100	0	100%	100%	zip-3.0-11.el7.x86_64.rpm	zip-3.0-11.hce1c.x86_64.rpm
zip-3.0-11.el7.x86_64.rpm	centos7.9	hce1.1	x86_64	direct dependence, function interface	0.477	20220831043453	2	0	100	0	100%	100%	zip-3.0-11.el7.x86_64.rpm	zip-3.0-11.hce1c.x86_64.rpm

– 对于扫描失败的应用软件包，请在/opt/x2hce-ca/output/software/目录下查看对应Excel报告。

4.2 将操作系统迁移至 HCE 2.0

4.2.1 约束限制

- 操作系统迁移过程中涉及RPM包卸载、安装及更新，操作系统存在异常重启的风险。在迁移前会自动备份操作系统的系统盘数据，也可以通过快速创建云服务器备份实现手动备份。
- 建议操作系统可用内存大于128MB，系统盘可用空间大于5GB（指迁移工具运行需要的系统盘空间，不包含数据备份的空间），boot分区可用空间大于200MB。

- 请避免自定义的RPM包和操作系统组件RPM重名。否则迁移时，自定义的RPM包会被迁移工具删除。
- 迁移操作系统后不支持更换系统盘的云硬盘类型。
- 系统迁移过程中，待迁移系统中存在部分冲突包。迁移工具会自动删除冲突包以完成系统迁移。冲突包列表详见**冲突包列表**。
- 系统迁移过程中会使用dnf组件，如果系统原有的dnf组件版本过低会影响升级过程，需升级原系统的dnf组件。

4.2.2 迁移操作

迁移前须知

1. 在进行迁移操作前，请仔细阅读**系统迁移与系统切换的相关介绍**，并根据您的实际业务情况来评估是选择系统迁移还是系统切换方案；系统切换的详细指南参见；

2. 不同的软件、版本对操作系统的兼容性存在差异，在迁移开始前，请充分测试您所使用的软件与HCE系统的兼容性，确保迁移后业务正常；

3. 迁移前请做好必要的系统与数据备份，防止因非预期原因导致业务受损。

准备迁移工具依赖的软件包

在系统迁移过程中，迁移工具对特定的基础软件和系统参数存在依赖，本节介绍软件包和系统参数的准备工作。

1. 远程连接待迁移的操作系统。

   根据弹性云服务器控制台操作指导，远程登录到待迁移虚拟机内部，远程登录的具体操作，请参见**连接方式概述**，并确保虚拟机能够与互联网建立正常通信。

2. 检查待迁移系统网络是否能够正常访问HCE的repo源，确保迁移工具可以获取到依赖的软件（来自HCE的repo源）。

   执行命令**curl https://repo.huaweicloud.com/hce/2.0/os/x86_64/**命令检测是否能够访问HCE的repo源。若有类似如下输出信息，则能正常访问HCE的repo源。

   ```
   % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
   Dload  Upload   Total   Spent    Left  Speed
   100  3417    0  3417    0     0    373      0 --:--:--  0:00:09 --:--:--   696
   <!doctype html>
   <html>
   <head>
   <meta charset="utf-8">
   <title></title>
   <link rel="stylesheet"  href="/repository/static/css/style.css" type="text/css"/>
   <style>
   * {
   font-family: 'Verdana', sans-serif;
   margin: 0;
   padding: 0;
   -webkit-box-sizing: border-box;
   -moz-box-sizing: border-box;
   box-sizing: border-box;
   }
   ......
   ```

3. 配置repo源（指原操作系统的repo源），确保迁移工具可以获取到依赖的软件。

   各操作系统的repo源地址不同，请配置正确的repo源地址。

4. 安装依赖的软件包。

a. 安装python基础软件包。
```
[root@localhost ~]# yum install -y python //任意目录执行安装命令
```

b. （可选）创建软连接。

说明

CentOS 8系列及**EulerOS 2.10/2.9**版本需执行以下步骤，其他操作系统版本请忽略。

i. 安装python3基础软件包。
```
[root@localhost ~]# yum install -y python3 //任意目录执行安装命令
```

ii. 执行下述命令检查是否存在python软链接。
```
[root@localhost]# ll /usr/bin/ | grep python
```

- 若显示“python -> /usr/bin/python3”，但不显示“python3 -> python3.9”，则代表python软链接已存在，但没有链接至python3，需执行如下命令删除原有python软链接，再执行**创建python软链接**。
```
[root@localhost]# unlink /usr/bin/python
```
- 若显示“python -> /usr/bin/python3”，并且显示“python3 -> python3.9”，请继续执行**安装迁移工具并检查迁移条件**。
- 若无上述显示，则代表python软链接不存在，执行**创建python软链接**。

iii. 创建python软链接。
```
[root@localhost]# python
-bash: /usr/bin/python: No such file or directory   //python软链接不存在的提示信息
[root@localhost]# cd /usr/bin/      //切换目录至/usr/bin下
[root@localhost bin]# ln -s python3 python   //创建软件python软链接
[root@localhost bin]# python
Python 3.6.8 (default, Apr 16 2020, 01:36:27)
[GCC 8.3.1 20191121 (Red Hat 8.3.1-5)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
//通过ctrl+D退出上述界面
```

安装迁移工具并检查迁移条件

1. 从**华为云开源镜像站**下载最新版本的迁移工具安装包centos2hce2-*.rpm。

   - [root@localhost test]# **yum download centos2hce2 --nogpgcheck --repofrom=centos2hce2_repo,https://repo.huaweicloud.com/hce/2.0/updates/x86_64/** //下载centos2hce2-*.rpm
     [root@localhost test]# **ls** //检查是否下载成功，此处软件的版本号可能会不同
     centos2hce2-1.0.0-0.0.82.hce2.x86_64.rpm
   - [root@localhost test]# **echo -e "[centos2hce2]\nname=centos2hce2_repo\nbaseurl=https://repo.huaweicloud.com/hce/2.0/updates/x86_64/\nenabled=1\ngpgcheck=0" > /etc/yum.repos.d/centos2hce2.repo && yum install centos2hce2 --downloadonly --downloaddir=. && rm -f /etc/yum.repos.d/centos2hce2.repo && yum makecache** //下载centos2hce2-*.rpm
     [root@localhost test]# **ls** //检查是否下载成功，此处软件的版本号可能会不同
     centos2hce2-1.0.0-0.0.82.hce2.x86_64.rpm

2. 安装迁移工具。
```
[root@localhost test]# rpm -ivh centos2hce2-1.0.0-0.0.82.hce2.x86_64.rpm --nodeps // 注意将版本号替换为您实际下载到的版本
warning: centos2hce2-1.0.0-0.0.82.hce2.x86_64.rpm: Header V4 RSA/SHA256 Signature, key ID a8def926: NOKEY
Verifying...                          ################################# [100%]
Preparing...                          ################################# [100%]
Updating / installing...
1:centos2hce2-1.0.0-0.0.6.hce2     ################################# [100%]
```

3. 配置待迁移系统的系统软件数据的备份路径。

在系统切换前，迁移工具将自动备份系统软件的所有数据至备份路径。

执行**vim /etc/centos2hce2.conf**命令，在**centos2hce2.conf**配置文件中配置backup_dir字段，配置备份路径。backup_dir默认为/mnt/sdb/.osbak。

```
# backup dir
backup_dir = "/mnt/sdb/.osbak"    #配置原系统软件数据的备份路径
```

说明

- 为避免迁移过程中系统数据的丢失，建议配置备份目录。
- 在系统迁移时，迁移工具会自动检查备份目录的空间。建议配置单独的数据盘（如/dev/sdb/，并将该分区挂载到/mnt/sdb/），避免因为空间不足导致的检查失败。
- 请勿将tmpfs类型的文件系统（如/dev、/run等）作为备份目录，系统重启后tmpfs类型文件系统内的文件会丢失。

4. 设置系统迁移参数。

   a. 设置web迁移方式。

   web迁移方式通过下载RPM包进行系统迁移，因此要求在下载RPM包的过程中不能断网。

   在centos2hce2.conf配置文件中，参考参数说明进行设置：

```
[repo_relation]
.....
# default yum source, val: web or iso
default_yum_source = 'web'
.....
# if web as source, web link config as follow
web_link_dir = "https://repo.huaweicloud.com/hce/2.0/os/x86_64/;https://
repo.huaweicloud.com/hce/2.0/updates/x86_64/"
```

表 4-6 主要参数说明

参数	说明
default_yum_source	迁移方式，设置为'web'。
web_link_dir	HCE的base repo源和updates repo源地址，多个repo源之间需用英文分号隔开。 设置为https://repo.huaweicloud.com/hce/2.0/os/x86_64/;https://repo.huaweicloud.com/hce/2.0/updates/x86_64/ 在此配置的HCE repo源地址会在迁移过程中暂时替换掉当前旧系统的repo源。迁移完成后，恢复被替换的repo源，此时系统中将存在HCE的repo源文件与旧系统的repo源文件，建议将/etc/yum.repos.d/目录下不再使用的repo源文件进行清理，只保留HCE的repo源文件。

   b. 配置isclose_modules参数，仅CentOS 8系列需要配置。

   CentOS 8系列支持将RPM包集成为module的方式批量安装RPM包。HCE不支持此种安装方式。因此系统迁移前，须关闭module功能。

   - “yes”表示系统迁移前会自动关闭系统上的modules，默认为“yes”。

- “no”表示系统迁移前不会自动关闭系统上的modules，且若检测到有modules开启时，迁移操作中断。

```
[system]
# whether close modules, if value is no, system may be not migrate
isclose_modules = "yes"
```

说明

- 执行命令**dnf module list**可查看待迁移系统中所有运行的module。
- 执行命令**dnf module list | grep '\[e\]'**可查看待迁移系统开启的module。

5. 执行**centos2hce2.py --check all**命令，检查当前系统配置是否满足迁移条件。
   - 提示“Environment check passed!”时，表示满足迁移条件，可直接执行迁移操作。
   - 提示“call migration failed”时，表示不满足迁移条件，请根据步骤6自动处理相关异常信息。Error Number及其对应错误信息请参见表4-7。

表 4-7 Error Number 对应关系

Error Number	错误信息
10001	非root用户下执行迁移工具命令时，需要切换至root。
10002	URL存在问题，/etc/centos2hce2.conf配置文件中web_link_dir、web_link_tar参数填写有误导致无法下载对应repo文件、rpm文件或者无法连通。
10003	基础命令缺失，例如rpm、yum、yumdownloader命令。
10004	空间检查失败，磁盘空间不足或者内存大小不足。
10005	原系统无本地yum源或者yum源不通，需要重新配置。
10006	目标系统yum源配置有误，检查/etc/centos2hce2.conf配置文件中web_link_dir参数填写是否有误。
10007	安装sut失败，检查/etc/centos2hce2.conf配置文件中web_link_dir参数填写是否有误。
10008	sut检查失败。
10009	依赖检查失败，需要先执行centos2hce2.py --install all安装依赖。
10010	chroot升级方式，清理原有chroot文件夹失败，文件夹路径见/etc/centos2hce2.conf配置文件中的chroot_path。
10011	chroot路径配置错误，/etc/centos2hce2.conf配置文件中chroot_path配置有误。
10012	chroot升级方式，并且配置了预构建环境tar包下载地址，解压tar包失败，检查/etc/centos2hce2.conf配置文件中的web_link_tar参数是否有误。
10013	/etc/ld.so.conf检查失败，需要清理/etc/ld.so.conf文件中除“include ld.so.conf.d/*.conf”之外的字段。

Error Number	错误信息
10014	文件系统存在损坏或异常，需要修复。
10015	/etc/fstab文件挂载目录不符合标准，需要将/etc/fstab文件中非LVM卷格式的文件系统分区以UUID进行挂载。
10016	开启文件属性检查后，系统内存在Immutable/Append_Only属性的文件，对于检查出的文件，需要加入到/etc/centos2hce2.conf配置文件中的exclude_dir字段。
10017	/etc/sysconfig/ntpd文件存在-u ntp:ntp配置，需要删除/etc/sysconfig/ntpd文件中-u ntp:ntp字段。
10018	/etc/ssh/sshd_config配置文件存在HCE 2.0不支持的算法，需要按照提示删除这些算法。
10019	系统中含有重复的rpm包，请先卸载不再使用的低版本rpm包后再次检查。（如果不想卸载重复包例如多内核场景kernel、kernel-devel等，直接进行升级，可以通过配置/etc/centos2hce2.conf文件中的extra_check_switch = false选项，跳过额外检查。）

6. 安装迁移工具依赖的软件。

   执行**centos2hce2.py --install all**命令，迁移工具会先进行备份，接着系统自动安装迁移工具依赖的软件包，并进行迁移前相关预处理操作。

   以下提示表明，已安装依赖的软件包及相关预处理操作，需再次执行步骤**5**进行环境检查。

   ```
   2022-08-19 03:12:58,373-INFO-centos2hce2.py-[line:832]: Dependency packages already exist!
   2022-08-19 03:12:58,373-INFO-centos2hce2.py-[line:891]: migrate install depend options finished
   ```

   在centos2hce2.conf配置文件中，进行如下设置：

   ```
   [check_config]
   ...
   # backup switch before upgrade
   backup_available = ""
   ```

   其中backup_available字段控制是否在install阶段进行备份动作，默认配置为空。

   - 当backup_available为空或不为false的任意值时，在install阶段进行备份。
   - 当backup_available为false(false不区分大小写)时，install阶段不进行备份，在upgrade阶段备份。
   - 对于centos8.0以上的系统，执行**centos2hce2.py --install all**命令，迁移工具会先进行备份，接着关闭centos的模块软件包管理机制，如果开启了此配置项，执行回滚后的系统内模块软件管理机制仍然是关闭的。

7. (可选) 重复备份。

   执行**centos2hce2.py --backup force** 命令，迁移工具会根据步骤**3**中配置的备份路径，对当前系统中的文件进行备份。

   说明

   步骤**6**中安装的工具依赖软件包，在执行此命令之后也会被备份。

迁移系统至 HCE

1. 执行迁移命令**centos2hce2.py --upgrade all** 进行系统迁移。

   出现migrate success提示信息，表明系统迁移成功。迁移后支持回退至原系统，详见系统回退的步骤1。

   图 4-6 系统迁移

   ```
   [root@localhost ~]# centos2hce2.py --upgrade all
   2022-08-19 03:19:21,060-INFO-centos2hce2.py-[line:1233]: start migration
   2022-08-19 03:19:21,075-INFO-centos2hce2.py-[line:425]: config sut succeed
   2022-08-19 03:19:21,096-INFO-centos2hce2.py-[line:901]: SElinux service switches to permissive mode and has been temporarily closed!
   ```

   图 4-7 迁移成功

   ```
   [ INFO ] - [initramfs]: set command line value done
   2022-08-19 03:30:35,117-INFO-centos2hce2.py-[line:1032]: migrate success
   2022-08-19 03:30:35,467-INFO-centos2hce2.py-[line:989]: python link is /usr/bin/python3.9
   2022-08-19 03:30:35,482-INFO-centos2hce2.py-[line:994]: create python link succeed
   2022-08-19 03:30:35,482-INFO-centos2hce2.py-[line:1044]: migrate excute finished
   ```

   说明

   - 迁移命令不能设置为Linux后台方式执行。
   - 可附加--simple_name参数，使得迁移后的grub菜单中显示Huawei Cloud EulerOS的简称。
   - 在升级过程中如果遇到因为网络中断、软件包冲突等情况导致的升级失败，可以通过重新执行迁移命令再次进行系统迁移。
   - 在升级过程中如果遇到报错如图4-8所示，表明升级过程因冲突包中断，需要处理冲突包后再次执行升级。冲突包处理请参见**冲突包列表**。

   图 4-8 冲突包报错

   ```
   warning: Converting database from bdb_ro to ndb backend
   Unable to detect release version (use '--releasever' to specify release version)
   Error: Transaction test error:
     file /usr/share/squid/errors/zh-cn from install of squid-7:4.9-20.hce2.x86_64 conflicts
   with file from package squid-7:3.5.20-2.2.h10.x86_64
     file /usr/share/squid/errors/zh-tw from install of squid-7:4.9-20.hce2.x86_64 conflicts
   with file from package squid-7:3.5.20-2.2.h10.x86_64
   ```

2. 系统迁移完毕后，执行**reboot**命令（若**reboot**无响应，执行**reboot -f**）使系统完成切换。

   系统重启后，执行**cat /etc/hce-release**命令查看迁移后的操作系统信息，执行**uname -a**命令查看系统内核信息。

   若显示Huawei Cloud EulerOS操作系统，则迁移成功；否则迁移失败，请联系技术工程师咨询。

   图 4-9 查看迁移后操作系统信息及系统内核信息

   ```
   [root@localhost ~]# cat /etc/hce-release
   Huawei Cloud EulerOS release 2.0 (West Lake)
   [root@localhost ~]# uname -a
   Linux localhost.localdomain 5.10.0-60.18.0.50.h425_1.hce2.x86_64 #1 SMP Thu Aug 18 16:31:04 UTC 2022 x86_64 x86_64 x86_64 GNU/Linux
   ```

   说明

   待系统迁移到HCE后，控制台仍然显示迁移前的操作系统名称。您可手动更新控制台操作系统名称，具体操作详见**迁移系统后，如何更改控制台操作系统名称?**

3. 清理旧版本组件的文件。

   待系统迁移到HCE后，新版本组件替换旧版本组件，但此时旧版本组件的文件仍然保存在系统中。执行命令**centos2hce2.py --precommit upgrade**可清理旧版本组件的文件。

   返回信息中提示“upgrade precommit success”表示环境清理成功。

   图 4-10 清理旧版本组件的文件

   ```
   2022-08-19 03:52:32,871-INFO-centos2hce2.py-[line:1112]: remove geolite2-city-20180605-1.el8.noarch succeed
   2022-08-19 03:52:32,984-INFO-centos2hce2.py-[line:1112]: remove subscription-manager-rhsm-certificates-1.26.16-1.el8.0.1.x86_64 succeed
   2022-08-19 03:52:33,543-INFO-centos2hce2.py-[line:1112]: remove cockpit-ws-211.3-1.el8.x86_64 succeed
   2022-08-19 03:52:33,543-INFO-centos2hce2.py-[line:1113]: handle clean rpms finished
   2022-08-19 03:52:37,902-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libasan.a
   2022-08-19 03:52:37,906-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libatomic.a
   2022-08-19 03:52:37,910-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libgomp.so
   2022-08-19 03:52:37,915-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libitm.a
   2022-08-19 03:52:37,919-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libquadmath.a
   2022-08-19 03:52:37,923-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/lib/gcc/x86_64-linux-gnu/10.3.1/32/libubsan.a
   2022-08-19 03:52:37,927-INFO-centos2hce2.py-[line:1124]: remove useless link /usr/share/doc/e2fsprogs/RELEASE-NOTES
   2022-08-19 03:52:37,929-INFO-centos2hce2.py-[line:1126]: remove useless link finished
   2022-08-19 03:52:37,930-INFO-centos2hce2.py-[line:1132]: clean system finished and migrate succeed
   2022-08-19 03:52:37,930-INFO-centos2hce2.py-[line:1200]: upgrade precommit success
   ```

   说明

   清理动作可执行多次。

4. （可选）修改Cloud-init相关配置。
   - 若迁移之前的操作系统中存在Cloud-init，服务状态正常，且Cloud-init为rpm包形式，请跳过此步骤。
   - 若迁移之前的操作系统存在Cloud-init服务，服务状态正常，且Cloud-init为某个文件非rpm包形式（如CentOS 7系列），迁移后请对/etc/cloud/cloud.cfg文件进行如下配置。

   a. 设置开放root密码远程登录并开启root用户的ssh权限。

      设置“disable_root”为“0”不禁用root用户；“ssh_pwauth”为“1”启用密码远程登录；“lock_passwd”为“False”不锁住用户密码。

      ```
      users:
       - name: root
         lock_passwd: False

      disable_root: 0
      ssh_pwauth: 1
      ```

   b. 执行**/usr/bin/cloud-init init --local**命令，无错误发生，说明Cloud-init配置成功。

      正确安装的Cloud-init会显示Cloud-init的版本详细信息，并且无任何错误信息。

      图 4-11 Cloud-init 配置成功

      ```
      [root@localhost ~]# /usr/bin/cloud-init init --local
      Cloud-init v. 21.4 running 'init-local' at Fri, 22 Jul 2022 07:43:21 +0000. Up 602150.81 seconds.
      [root@localhost ~]#
      ```

   c. 如果在升级完成后出现cloud-init不可用的情况，需要重新安装cloud-init软件，具体操作参照**安装Cloud-Init工具**。

5. （可选）因迁移时会自动关闭selinux服务，如迁移后需启用selinux，执行**centos2hce2.py --precommit upg-selinux**命令。此命令分为两个阶段，每次执行后都需重启系统（若迁移前未开启selinux请忽略此步骤）。

   a. 执行**centos2hce2.py --precommit upg-selinux**命令。

      ```
      [root@localhost ~]# centos2hce2.py --precommit upg-selinux
      2022-08-21 23:46:23,891-INFO-centos2hce2.py-[line:1239]: precommit migration
      2022-08-21 23:46:23,891-INFO-centos2hce2.py-[line:1149]: begin to set selinux
      2022-08-21 23:46:23,892-INFO-centos2hce2.py-[line:1157]: grub path is /boot/grub2/grub.cfg
      ```

```
2022-08-21 23:46:23,895-INFO-centos2hce2.py-[line:1162]: sed selinux succeed
2022-08-21 23:46:23,897-INFO-centos2hce2.py-[line:1167]: create autorelabel file succeed
2022-08-21 23:46:23,901-INFO-centos2hce2.py-[line:1172]: modify selinux config succeed
2022-08-21 23:46:23,901-INFO-centos2hce2.py-[line:1174]: create phase 1 flag file succeed
2022-08-21 23:46:23,901-INFO-centos2hce2.py-[line:1184]: selinux has been set, please reboot
now
2022-08-21 23:46:23,901-INFO-centos2hce2.py-[line:1206]: upgrade precommit selinux success
[root@localhost ~]# reboot
```

b. 系统重启后，再次执行**centos2hce2.py --precommit upg-selinux**命令。

```
[root@localhost ~]# centos2hce2.py --precommit upg-selinux
2022-08-21 23:57:07,576-INFO-centos2hce2.py-[line:1239]: precommit migration
2022-08-21 23:57:07,576-INFO-centos2hce2.py-[line:1176]: now begin to set selinux phase 2
2022-08-21 23:57:07,580-INFO-centos2hce2.py-[line:1181]: modify selinux config succeed
2022-08-21 23:57:07,580-INFO-centos2hce2.py-[line:1183]: create phase 2 flag file succeed
2022-08-21 23:57:07,580-INFO-centos2hce2.py-[line:1184]: selinux has been set, please reboot
now
2022-08-21 23:57:07,580-INFO-centos2hce2.py-[line:1206]: upgrade precommit selinux success
[root@localhost ~]# reboot
```

c. 第二次重启后，执行**getenforce**查看selinux状态，Enforcing表明selinux为开启状态。

```
[root@localhost ~]# getenforce
Enforcing
```

6. （可选）确认迁移完毕后，清理原系统数据。

迁移操作完成后，原系统的系统数据仍然保留在新系统中，并占用较大内存。建议执行**centos2hce2.py --commit all**命令清理数据。

执行命令后，系统会自动清理原系统的系统数据，包括步骤3中备份路径下的系统数据。

须知

执行命令后，操作系统无法回退。

```
[root@localhost ~]# centos2hce2.py --commit all
2022-08-22 04:45:32,601-INFO-centos2hce2.py-[line:1242]: commit migration
```

系统回退

1. 系统回退。

迁移操作支持系统回退，您可根据需要决定是否回退至原操作系统。

a. 执行**centos2hce2.py --rollback all**命令进行系统回退。回退后，执行**reboot**命令对系统重启。

图 4-12 系统回退&重启

```
[root@localhost ~]# centos2hce2.py --rollback all
2022-08-22 04:03:12,107-INFO-centos2hce2.py-[line:1236]: Start rollback
[ INFO ] - [sut]: ================== Pre Rollback Stage ==================
[ INFO ] - [sut]: skip system version check during rollback
[ INFO ] - [sut]: start to do rollback before reboot
[ INFO ] - [sut]: set dracut module for rollback.
[ INFO ] - [sut]: reboot aborted.You need reboot as soon as possbile
[root@localhost ~]# reboot
```

b. 执行**centos2hce2.py --precommit rollback**命令，恢复环境。

图 4-13 恢复环境

```
[root@localhost ~]# centos2hce2.py --precommit rollback
2022-08-22 04:36:13,902-INFO-centos2hce2.py-[line:1239]: precommit migration
2022-08-22 04:36:13,904-INFO-centos2hce2.py-[line:1071]: /opt/migrate//rsync_backup is not exists, skip it
2022-08-22 04:36:13,905-INFO-centos2hce2.py-[line:483]: sut not backup no need rollback
2022-08-22 04:36:17,996-INFO-centos2hce2.py-[line:1194]: rollback precommit success
```

2. （可选）若迁移前已开启selinux，迁移时会自动关闭selinux服务。如有需要，回退后请手动恢复selinux状态。

   a. 执行**centos2hce2.py --precommit rbk-selinux**命令。

   ```
   [root@localhost ~]# centos2hce2.py --precommit rbk-selinux
   2022-09-05 03:58:37,015-INFO-centos2hce2.py-[line:1401]: precommit migration
   2022-09-05 03:58:37,047-INFO-centos2hce2.py-[line:1319]: now begin to set selinux
   2022-09-05 03:58:37,051-INFO-centos2hce2.py-[line:1324]: modify selinux config succeed
   2022-09-05 03:58:37,051-INFO-centos2hce2.py-[line:1325]: selinux has been set, please reboot now
   2022-09-05 03:58:37,051-INFO-centos2hce2.py-[line:1340]: set rollback selinux succeed
   2022-09-05 03:58:37,051-INFO-centos2hce2.py-[line:1365]: upgrade precommit selinux success
   ```

   b. 执行**reboot**命令，进行系统重启。

   ```
   [root@localhost ~]# reboot
   ```

   c. 系统重启后，可查看到selinux状态为开启状态。

   ```
   [root@localhost ~]# getenforce
   Enforcing
   ```

3. 清理系统数据。

   执行**centos2hce2.py --commit all**命令清理数据。

   执行命令后，系统会自动清理目标系统和原系统的系统数据，包括步骤3中备份路径下的系统数据。

   ```
   [root@localhost ~]# centos2hce2.py --commit all
   2022-08-22 04:45:32,601-INFO-centos2hce2.py-[line:1242]: commit migration
   ```

4.2.3 冲突包列表

说明

- 冲突包列表表示在原系统中存在与HCE系统冲突的软件包，会影响升级过程。
- 冲突包会在升级过程中自动卸载并且不会再安装上，升级前请评估原系统依赖的软件包是否在此冲突列表内，以免造成升级完成后软件缺失。
- 如果发现升级后软件包丢失，可以通过yum命令安装新版本软件包。
- 如果升级过程中遇到其他冲突问题，可以修改/etc/centos2hce2.conf配置文件，参考本章节冲突包列表增加自定义的冲突包名称。

表 4-8 CentOS 8 系列冲突包列表

CentOS 版本	冲突包列表
CentOS8.0	rust-doc;intel-gpu-tools;netcf-libs;redhat-rpm-config;asciidoc;gnuplot-common;perf;tigervnc-icons;libpq-devel;paratype-pt-sans-caption-fonts;scala-apidoc;java-11-openjdk-devel;java-11-openjdk-headless;java-1.8.0-openjdk-headless;dovecot;systemd-journal-remote;pcp-manager;pcp-webapi;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-demo;ant;tigervnc-server-applet;java-atk-wrapper;java-11-openjdk;guava20;javapackages-tools;jboss-jaxrs-2.0-api;maven-shared-utils;tagsoup;cdi-api;libbase;geronimo-annotation;pentaho-reporting-flow-engine;maven-resolver-api;apache-commons-codec;maven-lib;jansi-native;maven-wagon-provider-api;libguestfs-java;apache-commons-cli;istack-commons-tools;jline;plexus-cipher;istack-commons-runtime;jcl-over-slf4j;apache-commons-io;maven-resolver-spi;maven-wagon-file;httpcomponents-core;icedtea-web;glassfish-el-api;aopalliance;hawtjni-runtime;plexus-containers-component-annotations;flute;jboss-annotations-1.2-api;liblayout;java-1.8.0-openjdk;postgresql-jdbc;mariadb-java-client;plexus-sec-dispatcher;google-guice;libformula;jdeparser;ant-lib;maven-wagon-http-shared;jboss-logging;plexus-classworlds;slf4j;librepository;ongres-scram-client;sisu-plexus;libfonts;plexus-interpolation;java-1.8.0-openjdk-src;plexus-utils;scala-swing;maven-wagon-http;ongres-scram;maven-resolver-impl;libloader;httpcomponents-client;atinject;apache-commons-logging;maven-resolver-connector-basic;jansi;jsoup;maven-resolver-util;jboss-interceptors-1.2-api;libreoffice-ure;byteman;sac;apache-commons-lang3;libserializer;scala;maven-resolver-transport-wagon;jboss-logging-tools;sisu-inject;libreoffice-core;java-1.8.0-openjdk-devel

CentOS版本	冲突包列表
CentOS8.1	kernel-rpm-macros;intel-gpu-tools;netcf-libs;redhat-rpm-config;asciidoc;gnuplot-common;perf;tigervnc-icons;libpq-devel;paratype-pt-sans-caption-fonts;java-1.8.0-openjdk-headless;java-11-openjdk-headless;java-11-openjdk-devel;pcp-pmda-rpm;pcp-pmda-podman;scala-apidoc;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-demo;ant;tigervnc-server-applet;java-atk-wrapper;java-11-openjdk;jansi-native;hawtjni-runtime;ongres-scram;jboss-annotations-1.2-api;liblayout;atinject;plexus-utils;istack-commons-tools;jline;apache-commons-io;ongres-scram-client;maven-shared-utils;maven-resolver-impl;libfonts;jsoup;apache-commons-codec;glassfish-el-api;jdeparser;maven-resolver-util;scala-swing;tagsoup;google-guice;istack-commons-runtime;jcl-over-slf4j;pentaho-reporting-flow-engine;maven-resolver-api;maven-resolver-connector-basic;libloader;slf4j;apache-commons-cli;maven-wagon-provider-api;maven-resolver-transport-wagon;byteman;httpcomponents-client;jna;java-1.8.0-openjdk-devel;maven-lib;libreoffice-core;java-1.8.0-openjdk-src;javapackages-tools;plexus-cipher;cdi-api;jboss-logging;sisu-inject;httpcomponents-core;guava20;sac;libbase;jboss-jaxrs-2.0-api;java-1.8.0-openjdk;libserializer;plexus-containers-component-annotations;jboss-interceptors-1.2-api;jboss-logging-tools;libguestfs-java;ant-lib;libreoffice-ure;maven-resolver-spi;maven-wagon-file;jansi;maven-wagon-http-shared;apache-commons-lang3;postgresql-jdbc;mariadb-java-client;plexus-sec-dispatcher;sisu-plexus;scala;plexus-classworlds;flute;maven-wagon-http;icedtea-web;libformula;plexus-interpolation;aopalliance;geronimo-annotation;librepository;apache-commons-logging

CentOS版本	冲突包列表
CentOS8.2	python-psycopg2-doc;exiv2;llvm-googletest;adwaita-qt;llvm-static;rust-doc;intel-gpu-tools;netcf-libs;flatpak-session-helper;asciidoc;perf;tigervnc-icons;paratype-pt-sans-caption-fonts;java-1.8.0-openjdk-headless;java-11-openjdk-devel;java-11-openjdk-headless;scala-apidoc;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-demo;ant;tigervnc-server-applet;java-atk-wrapper;java-11-openjdk;jboss-annotations-1.2-api;cdi-api;ongres-scram;maven-resolver-util;apache-commons-codec;istack-commons-tools;icedtea-web;plexus-classworlds;plexus-utils;maven-wagon-http-shared;atinject;javapackages-tools;istack-commons-runtime;jline;geronimo-annotation;jansi;jdeparser;byteman;liblayout;maven-resolver-transport-wagon;jmc-core;ant-lib;libreoffice-core;jansi-native;jcl-over-slf4j;slf4j;ee4j-parent;libfonts;maven-wagon-http;jboss-logging;jboss-interceptors-1.2-api;tagsoup;httpcomponents-client;plexus-containers-component-annotations;apache-commons-lang3;jaf;java-1.8.0-openjdk-src;jsoup;guava20;flute;apache-commons-cli;libbase;ongres-scram-client;jboss-logging-tools;plexus-interpolation;libloader;librepository;libreoffice-ure;scala-swing;jboss-jaxrs-2.0-api;maven-resolver-spi;maven-lib;apache-commons-io;hawtjni-runtime;google-guice;aopalliance;libguestfs-java;postgresql-jdbc;jna;glassfish-el-api;maven-resolver-impl;java-1.8.0-openjdk;directory-maven-plugin;mariadb-java-client;httpcomponents-core;maven-wagon-file;maven-wagon-provider-api;owasp-java-encoder;libserializer;maven-shared-utils;plexus-cipher;java-1.8.0-openjdk-devel;plexus-sec-dispatcher;pentaho-reporting-flow-engine;maven-resolver-api;sac;scala;libformula;sisu-inject;apache-commons-logging;maven-resolver-connector-basic;sisu-plexus;centos-logos-httpd;pcp-pmda-rpm

CentOS版本	冲突包列表
CentOS8.3	netcf-libs;rust-doc;git-credential-libsecret;texlive-context;intel-gpu-tools;flatpak-session-helper;asciidoc;perf;tigervnc-icons;paratype-pt-sans-caption-fonts;java-1.8.0-openjdk-headless;java-11-openjdk-devel;java-11-openjdk-headless;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-demo;ant;tigervnc-server-applet;java-atk-wrapper;java-11-openjdk;exiv2;llvm-googletest;adwaita-qt;llvm-static;python-psycopg2-doc;scala-apidoc;libXau;libappstream-glib;jmc-core;byteman;libfonts;jaf;jcl-over-slf4j;mariadb-java-client;tagsoup;libguestfs-java;jsoup;apache-commons-cli;sisu-inject;jansi-native;jna;apache-commons-lang3;flute;librepository;javapackages-tools;cdi-api;ongres-scram;java-1.8.0-openjdk-devel;sisu-plexus;istack-commons-runtime;jboss-logging;guava20;java-1.8.0-openjdk-src;maven-resolver-util;geronimo-annotation;hawtjni-runtime;jboss-annotations-1.2-api;ongres-scram-client;maven-resolver-connector-basic;slf4j;sac;apache-commons-codec;atinject;maven-wagon-http;libreoffice-ure;plexus-cipher;jboss-interceptors-1.2-api;jline;pentaho-reporting-flow-engine;httpcomponents-core;liblayout;istack-commons-tools;jdeparser;maven-wagon-provider-api;ee4j-parent;apache-commons-io;maven-resolver-spi;jboss-logging-tools;plexus-sec-dispatcher;plexus-containers-component-annotations;jboss-jaxrs-2.0-api;scala;libbase;libreoffice-core;httpcomponents-client;directory-maven-plugin;java-1.8.0-openjdk;libformula;maven-wagon-file;maven-shared-utils;aopalliance;glassfish-el-api;owasp-java-encoder;postgresql-jdbc;libloader;google-guice;plexus-classworlds;ant-lib;maven-resolver-api;plexus-interpolation;java-1.8.0-openjdk-slowdebug;maven-resolver-impl;java-1.8.0-openjdk-headless-slowdebug;prometheus-jmx-exporter;maven-resolver-transport-wagon;jolokia-jvm-agent;maven-wagon-http-shared;maven-lib;jansi;HdrHistogram;apache-commons-logging;plexus-utils;icedtea-web;libserializer;scala-swing

CentOS版本	冲突包列表
CentOS8.4	python-psycopg2-doc;anaconda-install-env-deps;hwloc-gui;python3-lit;exiv2;cups-filters;cups-filters-libs;gutenprint;adwaita-qt;cups;cups-lpd;hplip-common;hwloc-libs;gutenprint-doc;gutenprint-libs;gutenprint-libs-ui;hwloc;foomatic-db-ppds;foomatic-db;python39-pip;python39-setuptools;python39-numpy;python39-chardet;python39-psutil;python39-urllib3;python39-requests;python39-wheel;libasan6;paratype-pt-sans-caption-fonts;python39-six;python39-idna;python39-ply;python39-pyyaml;python39-pycparser;python39-lxml;python39-pysocks;rust-doc;netcf-libs;git-credential-libsecret;texlive-context;flatpak-session-helper;asciidoc;intel-gpu-tools;tigervnc-icons;jmc-core;byteman;libfonts;jaf;jcl-over-slf4j;mariadb-java-client;tagsoup;libguestfs-java;jsoup;apache-commons-cli;sisu-inject;jansi-native;jna;apache-commons-lang3;flute;librepository;javapackages-tools;cdi-api;ongres-scram;java-1.8.0-openjdk-devel;sisu-plexus;istack-commons-runtime;jboss-logging;guava20;java-1.8.0-openjdk-src;maven-resolver-util;geronimo-annotation;hawtjni-runtime;jboss-annotations-1.2-api;ongres-scram-client;maven-resolver-connector-basic;slf4j;sac;apache-commons-codec;atinject;maven-wagon-http;libreoffice-ure;plexus-cipher;jboss-interceptors-1.2-api;jline;pentaho-reporting-flow-engine;httpcomponents-core;liblayout;istack-commons-tools;jdeparser;maven-wagon-provider-api;ee4j-parent;apache-commons-io;maven-resolver-spi;jboss-logging-tools;plexus-sec-dispatcher;plexus-containers-component-annotations;jboss-jaxrs-2.0-api;scala;libbase;libreoffice-core;httpcomponents-client;directory-maven-plugin;java-1.8.0-openjdk;libformula;maven-wagon-file;maven-shared-utils;aopalliance;glassfish-el-api;owasp-java-encoder;postgresql-jdbc;libloader;google-guice;plexus-classworlds;ant-lib;maven-resolver-api;plexus-interpolation;java-1.8.0-openjdk-slowdebug;maven-resolver-impl;java-1.8.0-openjdk-headless-slowdebug;prometheus-jmx-exporter;maven-resolver-transport-wagon;jolokia-jvm-agent;maven-wagon-http-shared;maven-lib;jansi;HdrHistogram;apache-commons-logging;plexus-utils;icedtea-web;libserializer;scala-swing;java-1.8.0-openjdk-headless;java-11-openjdk-devel;java-11-openjdk-headless;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-demo;ant;java-atk-wrapper;java-11-openjdk;scala-apidoc;libappstream-glib;PackageKit-gtk3-module;gnome-software;flatpak-libs;PackageKit-glib;PackageKit-gstreamer-plugin;libpq-devel;poppler;perf

CentOS版本	冲突包列表
CentOS8.5	bluez;python-psycopg2-doc;perl-Devel-Peek;OpenIPMI-libs;anaconda-install-env-deps;postfix-mysql;perl-Devel-SelfStubber;metacity;bluez-libs;libicu;vte-profile;qt5-qttools-examples;exiv2;cups-filters;cups-filters-libs;gutenprint;gnome-session;cups;cups-lpd;hplip-common;hwloc;gnome-session-wayland-session;gutenprint-doc;gutenprint-libs;gutenprint-libs-ui;gnome-session-xsession;foomatic-db-ppds;foomatic-db;gnome-classic-session;gnome-shell-extension-apps-menu;gnome-shell-extension-auto-move-windows;gnome-shell-extension-drive-menu;gnome-shell-extension-launch-new-instance;gnome-shell-extension-native-window-placement;gnome-shell-extension-places-menu;gnome-shell-extension-screenshot-window-sizer;gnome-shell-extension-user-theme;gnome-shell-extension-window-list;gnome-shell-extension-windowsNavigator;gnome-shell-extension-workspace-indicator;python39-six;python39-idna;python39-ply;python39-pyyaml;python39-pycparser;python39-psutil;python39-urllib3;python39-lxml;python39-pysocks;xorg-x11-server-Xwayland;compat-hwloc1;bluez-obexd;bluez-hid2hci;netcf-libs;git-credential-libsecret;texlive-context;flatpak-session-helper;asciidoc;intel-gpu-tools;tigervnc-icons;libasan6;paratype-pt-sans-caption-fonts;pcp-pmda-podman;jmc-core;byteman;libfonts;jaf;jcl-over-slf4j;mariadb-java-client;tagsoup;libguestfs-java;jsoup;apache-commons-cli;sisu-inject;jansi-native;jna;apache-commons-lang3;flute;librepository;javapackages-tools;cdi-api;ongres-scram;java-1.8.0-openjdk-devel;sisu-plexus;istack-commons-runtime;jboss-logging;guava20;java-1.8.0-openjdk-src;maven-resolver-util;geronimo-annotation;hawtjni-runtime;jboss-annotations-1.2-api;ongres-scram-client;maven-resolver-connector-basic;slf4j;sac;apache-commons-codec;atinject;maven-wagon-http;libreoffice-ure;plexus-cipher;jboss-interceptors-1.2-api;jline;pentaho-reporting-flow-engine;httpcomponents-core;liblayout;istack-commons-tools;jdeparser;maven-wagon-provider-api;ee4j-parent;apache-commons-io;maven-resolver-spi;jboss-logging-tools;plexus-sec-dispatcher;plexus-containers-component-annotations;jboss-jaxrs-2.0-api;scala;libbase;libreoffice-core;httpcomponents-client;directory-maven-plugin;java-1.8.0-openjdk;libformula;maven-wagon-file;maven-shared-utils;aopalliance;glassfish-el-api;owasp-java-encoder;postgresql-jdbc;libloader;google-guice;plexus-classworlds;ant-lib;maven-resolver-api;plexus-interpolation;java-1.8.0-openjdk-slowdebug;maven-resolver-impl;java-1.8.0-openjdk-headless-slowdebug;prometheus-jmx-exporter;maven-resolver-transport-wagon;jolokia-jvm-agent;maven-wagon-http-shared;maven-lib;jansi;HdrHistogram;apache-commons-logging;plexus-utils;icedtea-web;libserializer;scala-swing;java-1.8.0-openjdk-headless;java-11-openjdk-devel;java-11-openjdk-headless;libguestfs-java-devel;libguestfs-javadoc;icedtea-web-javadoc;systemtap-runtime-java;java-1.8.0-openjdk-accessibility;java-1.8.0-openjdk-

CentOS 版本	冲突包列表
	demo;ant;java-atk-wrapper;java-11-openjdk;scala-apidoc;libappstream-glib;PackageKit-gtk3-module;gnome-software;flatpak-libs;PackageKit-glib;PackageKit-gstreamer-plugin;coreos-installer-bootinfra;OpenIPMI;rust;cargo;perf;flatpak;hplip-libs;nautilus;gutenprint-cups;libgtop2;PackageKit;libsane-hpaio;PackageKit-command-not-found;xorg-x11-drv-wacom-serial-support;clutter;clutter-gtk;clutter-gst3;cheese-libs;cheese;gnome-initial-setup;gnome-control-center;clutter-gst2

表 4-9 CentOS 7 系列冲突包列表

CentOS 版本	冲突包列表
CentOS7.0	texlive-kpathsea-lib;libdhash;libref_array;libbasicobjects;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;tomcat-servlet-3.0-api;gnuplot-common;postgresql-devel;tigervnc-icons;squid;perf;dovecot;dovecot-mysql;dovecot-pgsql;dovecot-pigeonhole;lvm2-cluster
CentOS7.1	texlive-kpathsea-lib;libdhash;libref_array;qemu-kvm-tools;texlive-dvipdfm-bin;tomcat-servlet-3.0-api;gnuplot-common;squid;tigervnc-icons;postgresql-devel;perf;dovecot;dovecot-mysql;dovecot-pgsql;dovecot-pigeonhole;lvm2-cluster;texlive-dvipdfm;libcacard
CentOS7.2	texlive-kpathsea-lib;libdhash;qemu-kvm-tools;rdma-ndd;texlive-dvipdfm;texlive-dvipdfm-bin;dstat;tomcat-servlet-3.0-api;gnuplot-common;perf;squid;tigervnc-icons;tigervnc-icons;postgresql-devel;dovecot;dovecot-pgsql;dovecot-pigeonhole;lvm2-cluster;ipa-server-trust-ad
CentOS7.3	spice-glib;texlive-kpathsea-lib;libdhash;qemu-kvm-tools;rdma-ndd;texlive-dvipdfm;texlive-dvipdfm-bin;dstat;tomcat-servlet-3.0-api;gnuplot-common;perf;squid;tigervnc-icons;postgresql-devel;dovecot;dovecot-mysql;dovecot-pgsql;dovecot-pigeonhole;lvm2-cluster;pcp-pmda-kvm;pcp-pmda-rpm;spice-gtk3;vinagre;ipa-server;ipa-server-trust-ad
CentOS7.4	spice-glib;texlive-kpathsea-lib;libdhash;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;gnuplot-common;perf;squid;tigervnc-icons;postgresql-devel;lvm2-cluster;spice-gtk3;vinagre
CentOS7.5	spice-glib;texlive-kpathsea-lib;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;gnuplot-common;perf;squid;tigervnc-icons;postgresql-devel;lvm2-cluster;spice-gtk3;vinagre

CentOS版本	冲突包列表
CentOS7.6	shim-x64;spice-glib;adwaita-gtk2-theme;texlive-kpathsea-lib;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;gnuplot-common;cockpit-ws;perf;squid;tigervnc-icons;postgresql-devel;java-11-openjdk-headless;lvm2-cluster;spice-gtk3;vinagre
CentOS7.7	shim-x64;spice-glib;openmpi;adwaita-gtk2-theme;exiv2;texlive-kpathsea-lib;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;cockpit-ws;gnuplot-common;perf;squid;tigervnc-icons;postgresql-devel;java-11-openjdk-headless;lvm2-cluster;spice-gtk3;openmpi-devel;vinagre
CentOS7.8	shim-x64;spice-glib;openmpi;adwaita-gtk2-theme;exiv2;texlive-kpathsea-lib;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;cockpit-ws;gnuplot-common;perf;squid;tigervnc-icons;postgresql-devel;java-11-openjdk-headless;lvm2-cluster;spice-gtk3;openmpi-devel;vinagre
CentOS7.9	spice-glib;openmpi;adwaita-gtk2-theme;exiv2;gnuplot-common;texlive-kpathsea-lib;perf;qemu-kvm-tools;texlive-dvipdfm-bin;texlive-dvipdfm;dstat;tomcat-servlet-3.0-api;cockpit-ws;squid;tigervnc-icons;postgresql-devel;java-11-openjdk-headless;lvm2-cluster;spice-gtk3;openmpi-devel

表 4-10 HCE 冲突包列表

HCE	冲突包列表
HCE 1.1	spice-glib;openmpi;exiv2;sg3_utils;spice-gtk3;openmpi-devel;kernel-hcek;tomcat-servlet-3.0-api;kernel-hcek-devel;dstat;gnuplot-common;cockpit-ws;perf;squid;postgresql-devel;java-11-openjdk-headless;lvm2-cluster;fcoe-utils;libblockdev;udisks2;python-blivet;device-mapper-multipath;device-mapper-multipath-libs;libblockdev-crypto;libblockdev-fs;libblockdev-loop;libblockdev-mdraid;libblockdev-nvdimm;libblockdev-part;libblockdev-swap;libblockdev-utils;NetworkManager-team;NetworkManager-bluetooth;NetworkManager-wifi;libstorage-uio-static;kiwi-dlimage

表 4-11 EulerOS 冲突包列表

EulerOS	冲突包列表
EulerOS 2.9	euleros-release;euleros-latest-release;kiwi-systemdeps;python3-kiwi;NetworkManager-team;NetworkManager-bluetooth;NetworkManager-wifi;libstorage-uio-static;kiwi-dlimage;systemd-udev-compat

EulerOS	冲突包列表
EulerOS 2.10	euleros-release;euleros-latest-release;kiwi-systemdeps;python3-kiwi;NetworkManager-team;NetworkManager-bluetooth;NetworkManager-wifi;libstorage-uio-static;kiwi-dlimage;systemd-udev-compat

表 4-12 特殊冲突包配置项

配置选项	冲突包列表
specific_conflict	openssl110f-libs;openssl110h-libs;openssl111d-libs

特殊冲突包配置项会在执行centos2hce2.py --check all命令时进行校验，如果系统中存在上述列表的软件，迁移过程中与HCE 的软件包产生冲突，迁移工具会提示先进行卸载该软件包再进行升级。

4.3 将操作系统迁移至 HCE

4.3.1 约束限制

- 对于HCE 1.1镜像，仅支持从CentOS7.9迁移到HCE 1.1，并且不支持配置图形化界面的CentOS7.9系统的迁移。
- 操作系统迁移过程中涉及RPM包卸载、安装及更新，操作系统存在异常重启的风险。在迁移前会自动备份操作系统的系统盘数据，也可以通过**快速创建云服务器备份**实现手动备份。
- 建议操作系统内存剩余大于128MB，系统盘空间剩余大于1GB。

4.3.2 迁移操作

本节介绍从CentOS7.9迁移到HCE 1.1的操作过程。

准备迁移工具依赖的软件包

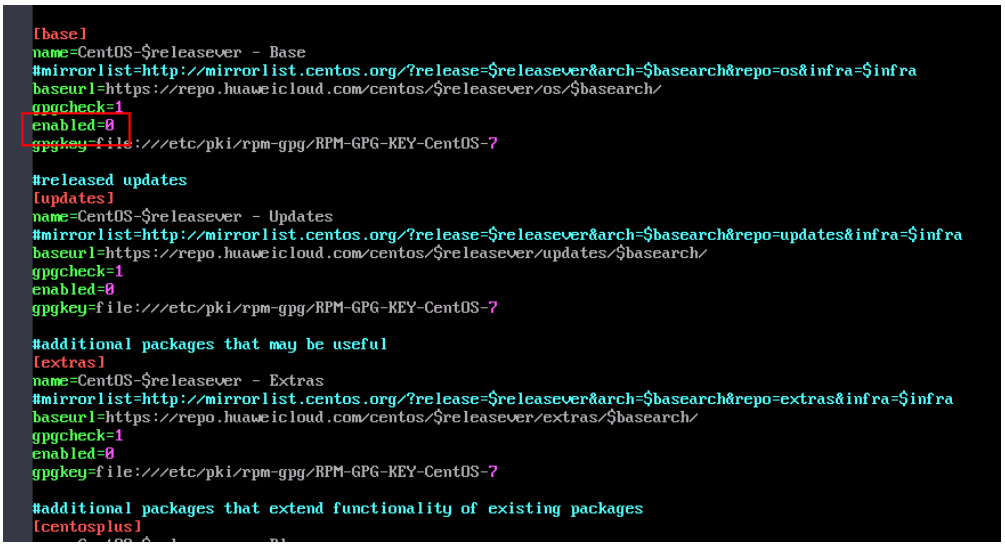
1. 远程连接待迁移的操作系统。

   根据弹性云服务器控制台操作指导，远程登录到待迁移虚拟机内部，远程登录的具体操作，请参见**连接方式概述**，并确保虚拟机能够与互联网建立正常通信。
2. 先关闭CentOS系统/etc/yum.repos.d下的所有的repo配置，确保CentOS的repo源不与HCE的repo源发生冲突。

   图 4-14 关闭 repo 配置

以Centos_Base.repo为例，将里面的每个子项原始的repo源里面添加enabled=0的配置项，如下图所示。

图 4-15 添加 enabled=0 的配置项



3. 配置HCE的repo源。

   将如下内容添加到hce.repo中，并将hce.repo配置文件存放在/etc/yum.repos.d/目录下。

```
[centos7_everything]
name=centos7_everything
baseurl=https://repo.huaweicloud.com/hce/1.1/os/x86_64/
enable=1
gpgcheck=0
priority=1

#released updates
[updates]
name=hce1_updates
baseurl=https://repo.huaweicloud.com/hce/1.1/updates/x86_64/
gpgcheck=0
enabled=1
gpgkey=
```

4. 检查CentOS7.9系统网络是否能够正常访问HCE的repo源。

   执行命令**curl https://repo.huaweicloud.com/hce/1.1/os/x86_64/**命令检测是否能够访问HCE的repo源。若有类似如下输出信息，则能正常访问HCE的repo源。

```
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
Dload  Upload   Total   Spent    Left  Speed
100  3417    0  3417    0     0    373      0 --:--:--  0:00:09 --:--:--   696
<!doctype html>
<html>
<head>
<meta charset="utf-8">
<title></title>
<link rel="stylesheet"  href="/repository/static/css/style.css" type="text/css"/>
<style>
* {
font-family: 'Verdana', sans-serif;
margin: 0;
padding: 0;
-webkit-box-sizing: border-box;
-moz-box-sizing: border-box;
box-sizing: border-box;
```

```
}
......
```

5. 执行如下命令安装python3。

```
[root@localhost ~]# yum install -y python3 //任意目录执行安装命令
```

说明

如果CentOS 7.9系统已经安装python3，请忽略此步骤。

6. 关闭selinux。

为了保证迁移前后系统配置文件一致，需要关闭selinux。

a. 修改/etc/selinux/config文件，将config文件中SELINUX的值设置成disabled。

```
SELINUX=disabled
```

b. 重启操作系统使selinux配置生效。

安装迁移工具

1. 从**华为云开源镜像站**下载迁移工具安装包centos2hce1-*.rpm。

*表示迁移工具版本，本节以centos2hce1-1.0.0-0.0.2.x86_64.rpm安装包示例。

```
[root@localhost test]# wget https://repo.huaweicloud.com/hce/1.1/updates/x86_64/Packages/centos2hce1-1.0.0-0.0.2.x86_64.rpm   //下载centos2hce1-*.rpm
[root@localhost test]# ls  //检查是否下载成功
centos2hce1-1.0.0-0.0.2.x86_64.rpm
```

2. 安装迁移工具。

工具安装完成后，系统自动生成/etc/centos2hce1.conf配置文件。

```
[root@localhost ~]# rpm -ivh centos2hce1-1.0.0-0.0.2.x86_64.rpm
```

3. 配置centos2hce1.conf文件。

配置HCE的repo源地址，用于检测repo源是否能够正常访问，并更新RPM包。

```
#iso as yum source link
[repo_info]
base_yum_url =https://repo.huaweicloud.com/hce/1.1/os/x86_64/

#iso as yum source
repostr_hce1_1 =
    [base]
    name=hceversion
    baseurl=https://repo.huaweicloud.com/hce/1.1/os/x86_64/
    gpgcheck=0
    enabled=1
    #released updates
    [updates]
    name=hce1_updates
    baseurl=https://repo.huaweicloud.com/hce/1.1/updates/x86_64/
    gpgcheck=0
    enabled=1
    gpgkey=
```

说明

centos2hce1.conf配置文件说明详见**附录：conf配置文件说明**。

系统迁移

1. 备份操作系统。

系统迁移至HCE1.1不支持回滚，请备份CentOS整体操作系统（包括系统盘和数据盘）。

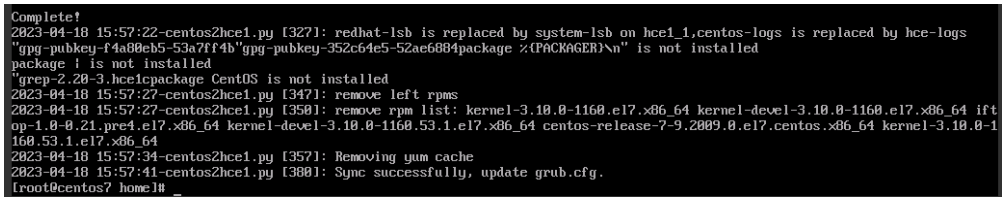
2. 执行**centos2hce1.py**命令，进行系统迁移。

系统迁移的耗时受更新的RPM包数量、大小和从repo源下载速度等影响，一般会在20分钟到1个小时左右完成，具体时间视实际环境确定，执行操作时注意预留足够的时间。

```
[root@localhost home]# centos2hce1.py
```

有如下回显信息，表示迁移完成。若迁移失败请使用备份数据恢复。

图 4-16 回显信息



说明

CentOS含有某些HCE 1.1不提供的RPM包，执行**centos2hce1.py**命令迁移系统后，迁移工具会自动清除这些RPM包。如果您想保留这些RPM包，请使用**-s skip**参数进行系统迁移。

3. （可选）删除无用的RPM包。

   如下两个RPM包在迁移过程中并没有使用，也不会对系统运行产生任何影响。在此对您可能产生的疑惑进行解释。

图 4-17 无用的 RPM 包

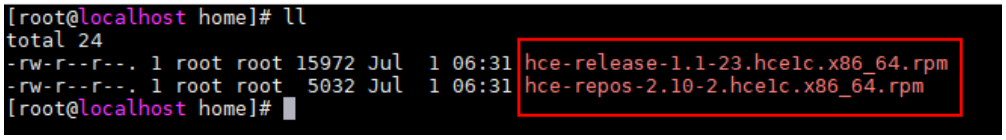
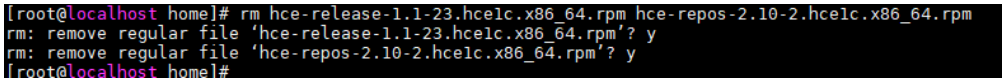


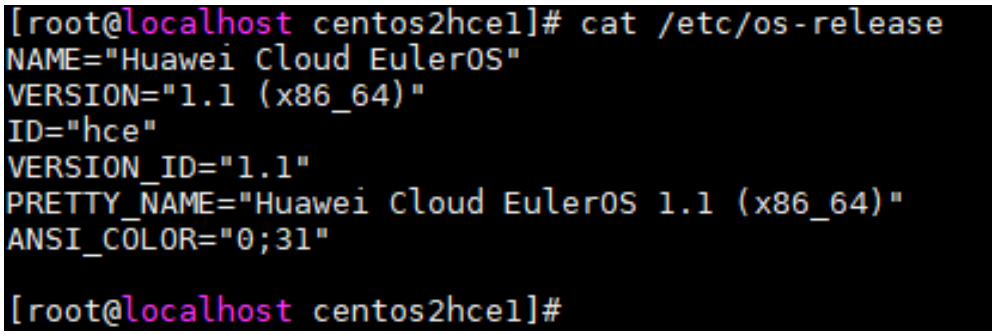
图 4-18 删除无用的 RPM 包



4. 执行**reboot**命令重启操作系统。
5. 执行**cat /etc/os-release**命令检查是否迁移成功。

   显示如下Huawei Cloud EulerOS信息表示迁移成功。

图 4-19 迁移成功



6. （可选）开启selinux。

   系统迁移前关闭了selinux，请根据需要选择是否开启selinux。

a. 修改/etc/selinux/config文件，将config文件中SELINUX的值设置成enforcing
```
SELINUX=enforcing
```
b. 重启操作系统使selinux配置生效。

附录：conf 配置文件说明

```
#rpm lists for os migration
[rpm_lists]
#origin system must need rpms
baserpms_list = "basesystem initscripts hce-logos plymouth grub2 grubby"  //系统迁移依赖的RPM包

#old rpm and default conflict rpms  //迁移过程中，原系统可能存在的冲突包
oldrpms_list =  centos-backgrounds centos-release-cr desktop-backgrounds-basic \
centos-release-advanced-virtualization centos-release-ansible26 centos-release-ansible-27 \
centos-release-ansible-28 centos-release-ansible-29 centos-release-azure \
centos-release-ceph-jewel centos-release-ceph-luminous centos-release-ceph-nautilus \
centos-release-ceph-octopus centos-release-configmanagement centos-release-dotnet centos-release-fdio \
centos-release-gluster40 centos-release-gluster41 centos-release-gluster5 \
centos-release-gluster6 centos-release-gluster7 centos-release-gluster8 \
centos-release-gluster-legacy centos-release-messaging centos-release-nfs-ganesha28 \
centos-release-nfs-ganesha30 centos-release-nfv-common \
centos-release-nfv-openvswitch centos-release-openshift-origin centos-release-openstack-queens \
centos-release-openstack-rocky centos-release-openstack-stein centos-release-openstack-train \
centos-release-openstack-ussuri centos-release-opstools centos-release-ovirt42 centos-release-ovirt43 \
centos-release-ovirt44 centos-release-paas-common centos-release-qemu-ev centos-release-qpid-proton \
centos-release-rabbitmq-38 centos-release-samba411 centos-release-samba412 \
centos-release-scl centos-release-scl-rh centos-release-storage-common \
centos-release-virt-common centos-release-xen centos-release-xen-410 \
centos-release-xen-412 centos-release-xen-46 centos-release-xen-48 centos-release-xen-common \
python3-syspurpose python-oauth sl-logos yum-rhn-plugin centos-indexhtml \
libreport-centos libreport-web libreport-plugin-mantisbt libreport-plugin-rhtsupport \
libreport hunspell-en-US hunspell-en policycoreutils-gui libcanberra-gtk2 cups \
NetworkManager-libreswan-gnome plymouth-graphics-libs avahi cups-lpd pinentry-qt \
librsvg2-devel libcanberra-gtk3 gnome-themes-standard wodim gsettings-desktop-schemas-devel \
avahi-ui-gtk3 freerdp-libs pulseaudio-utils gstreamer1-plugins-bad-free-gtk ghostscript-cups \
setools-console libxkbcommon-x11 cups plymouth-plugin-two-step pulseaudio-module-x11 ImageMagick-c++ \
cups-devel policycoreutils-sandbox PackageKit-gstreamer-plugin gtk3-immodule-xim avahi-glib avahi-autoipd \
mesa-libGLES foomatic libcanberra-devel plymouth-plugin-label PackageKit-gtk3-module colord avahi-gobject \
pinentry-qt4 avahi-ui-gtk3 plymouth-plugin-two-step ghostscript-cups ImageMagick-perl firewall-config \
plymouth-plugin-label redhat-redhat-lsb-corelsb vim-X11 dbus-x11 pulseaudio PackageKit-command-not-found libproxy-mozjs \
pinentry-gtk nm-connection-editor gtk2-immodule-xim wireshark-gnome pulseaudio-module-bluetooth pidgin-sipe freerdp kmod-kvdo  \
redhat-lsb-core

#The following list contains the same symbol as centos/redhat
dstrpms_list = "hce-release hce-repos"

[log_conf]
# migration tool log common dir
migrate_common_dir = "/var/log/migrate-tool/"    //日志存放路径

# migration tool classification log dir
migrate_classification_dir = %(migrate_common_dir)s/centos2hce1/

#iso as yum source link
[repo_info]
base_yum_url =https://repo.huaweicloud.com/hce/1.1/os/x86_64/  //基础yum源路径，用于检查网络状态


#iso as yum source
repostr_hce1_1 =     //提供迁移模式的源路径
   [base]
   name=hceversion
   baseurl=https://repo.huaweicloud.com/hce/1.1/os/x86_64/   //基础yum源路径，用于获取RPM包
```



```
gpgcheck=0
enabled=1
gpgkey=

#released updates
[updates]
name=hce1_updates
baseurl=
gpgcheck=0
enabled=0
gpgkey=

#additional packages that may be useful
[extras]
name=hce1_extras
baseurl=
gpgcheck=0
enabled=0
gpgkey=

# plus packages provided by Huawei Linux dev team
[plus]
name=hce1_plus
baseurl=
gpgcheck=0
enabled=0
gpgkey=
```


5 更新 HCE 系统和 RPM 包

5.1 升级概述

HCE提供操作系统和RPM包的更新维护，包括部署在HCE上的RPM包、安全更新涉及的RPM包和漏洞修复。为了操作系统和RPM包的使用更加安全，请及时升级。

HCE支持使用dnf/yum命令和OSMT工具两种升级方式。

- Linux自身支持dnf/yum命令，可对RPM包进行升级和回退，升级操作简单。
- OSMT是华为云提供的对HCE系统及RPM包升级和回退的工具，可自定义升级范围和定时检查、延迟重启。

两种升级方式区别如下：

表 5-1 升级方式区别

项目	使用dnf或yum命令升级	使用OSMT工具升级
RPM包升级	• 支持无差别升级所有待更新的RPM包，包括安全更新涉及的RPM包和漏洞修复。 • 支持仅升级安全更新涉及的RPM包。	• 支持无差别升级所有待更新的RPM包，包括安全更新涉及的RPM包和漏洞修复。 • 支持自定义升级范围： – 升级不需要重启的RPM包。 – 升级需要重启的RPM包。 – 升级自定义黑名单或白名单列表中的RPM包。 – 升级安全更新涉及的RPM包。 – 漏洞修复。 – 升级新增功能的RPM包。 – 更新新增的RPM包。 • 支持自定义时间自动更新RPM包、延迟重启。

项目	使用dnf或yum命令升级	使用OSMT工具升级
系统版本升级	不支持系统版本升级。	支持系统版本升级。
支持升级的版本	支持HCE 1.1从低版本向高版本的升级。	支持HCE 2.0从低版本向高版本的升级。
回退	支持回退所有历史操作。	系统或RPM包仅支持最近一次升级的回退。

5.2 使用 dnf 或 yum 命令升级和回退

本节介绍HCE 1.1从低版本向高版本的升级和回退操作。dnf和yum命令的使用方法相同，本节以dnf命令为例。

说明

- HCE 2.0及以上版本支持yum和dnf命令。
- HCE 1.1版本仅支持yum命令。

背景信息

DNF (Dandified YUM) 和 YUM (Yellowdog Updater Modified) 都是基于 RPM 的包管理工具，用于安装、更新和删除软件包。YUM 是较早的工具，而 DNF 是 YUM 的继任者，旨在提供更好的性能和更多的功能，如模块化支持和并行下载。HCE为保证更好的兼容性，同时提供yum命令和dnf命令。

升级步骤

1. 检查待更新的RPM包。
   - 执行**dnf list updates**命令查看所有待更新的RPM包列表。

     ```
     [root@localhost bin]# dnf list updates
     Last metadata expiration check: 6:49:11 ago on Tue 28 Jun 2022 01:55:35 PM CST.
     hce-config.x86_64                        3.0-66.hce2
     hce-latest-release.x86_64                  2.0-1656179342.2.0.2206.B032.hce2
     irqbalance.x86_64                        3:1.8.0-7.h9.hce2
     kernel.x86_64                           5.10.0-60.18.0.50.h316_1.hce2
     kernel-tools.x86_64                      5.10.0-60.18.0.50.h316_1.hce2
     kernel-tools-libs.x86_64                 5.10.0-60.18.0.50.h316_1.hce2
     kexec-tools.x86_64                       2.0.23-4.h8.hce2
     libcurl.x86_64                          7.79.1-2.h4.hce2
     libssh.x86_64                           0.9.6-2.h3.hce2
     libstdc++.x86_64                         10.3.1-10.h10.hce2
     libxml2.x86_64                           2.9.12-5.h5.hce2
     openssh.x86_64                           8.8p1-2.h12.hce2
     openssh-clients.x86_64                     8.8p1-2.h12.hce2
     openssh-server.x86_64                      8.8p1-2.h12.hce2
     Obsoleting Packages
     dnf-data.noarch                          4.10.0-3.h6.hce2
     dnf.noarch                              4.10.0-3.h5.hce2
     dnf-data.noarch                          4.10.0-3.h6.hce2
     dnf-data.noarch                          4.10.0-3.h5.hce2
     ```

   - 执行**dnf list updates --security**命令，仅查看安全更新涉及的RPM包。

     ```
     [root@localhost bin]# dnf list updates --security
     Last metadata expiration check: 0:00:03 ago on Fri 08 Jul 2022 04:45:56 PM CST.
     No security updates needed, but 2 updates available
     ```

2. 升级待更新的RPM包。
   - 执行**dnf update**命令升级所有待更新的RPM包，包括安全更新涉及的RPM包和漏洞修复。执行命令输出信息中会显示组件的目标版本信息(Version列）。

```
[root@localhost bin]# dnf update
Last metadata expiration check: 7:12:18 ago on Tue 28 Jun 2022 01:55:35 PM CST.
Dependencies resolved.
================================================================================
====================
 Package                  Arch      Version                          Repo     Size
================================================================================
====================
Installing:
 kernel                   x86_64    5.10.0-60.18.0.50.h316_1.hce2          hce2     47 M
Upgrading:
 hce-config               x86_64    3.0-66.hce2                      hce2     13 k
 hce-latest-release       x86_64    2.0-1656179342.2.0.2206.B032.hce2      hce2    5.2 k
 kernel-tools             x86_64    5.10.0-60.18.0.50.h316_1.hce2          hce2    230 k
 kernel-tools-libs        x86_64    5.10.0-60.18.0.50.h316_1.hce2          hce2     62 k
 kexec-tools              x86_64    2.0.23-4.h8.hce2                 hce2    400 k
 libcurl                  x86_64    7.79.1-2.h4.hce2                 hce2    284 k
 libssh                   x86_64    0.9.6-2.h3.hce2                  hce2    194 k
 libstdc++                x86_64    10.3.1-10.h10.hce2                     hce2    535 k
 libxml2                  x86_64    2.9.12-5.h5.hce2                 hce2    659 k
 logrotate                x86_64    3.18.1-1.h2.hce2                 hce2     60 k
 mdadm                    x86_64    4.1-5.h2.hce2                    hce2    331 k
 nftables                 x86_64    1:1.0.0-1.h3.hce2                hce2    303 k
 perl                     x86_64    4:5.34.0-3.h5.hce2               hce2    3.2 M
 perl-libs                x86_64    4:5.34.0-3.h5.hce2               hce2    1.8 M
Installing dependencies:
 grub2-tools-efi          x86_64    1:2.06-3.h5.hce2                       hce2    472 k

 Transaction Summary
================================================================================
====================
Install   2 Packages
Upgrade  72 Packages
Total download size: 105 M
Is this ok [y/N]:
```

   - 执行**dnf update --security**命令，仅升级安全更新涉及的RPM包。

```
[root@localhost bin]# dnf update --security
Last metadata expiration check: 7:15:16 ago on Tue 28 Jun 2022 01:55:35 PM CST.
No security updates needed, but 73 updates available Dependencies resolved.
Nothing to do.
Complete!
```

3. 升级成功后，请及时确认业务运行情况。

回退步骤

1. 执行**dnf history**命令，查询需要回退的历史操作ID号。

图 5-1 查询需要回退的历史操作 ID 号

```
[root@localhost ~]# dnf history
ID     | Command line                | Date and time    | Action(s)      | Altered
-------------------------------------------------------------------------------
     5 | upgrade chrony              | 2022-10-09 11:38 | Upgrade        |    1
     4 | history undo 3              | 2022-10-09 11:37 | Downgrade      |    1
     3 | upgrade chrony              | 2022-10-09 11:36 | Upgrade        |    1
     2 | install createrepo          | 2022-10-09 11:30 | Install        |    2
     1 |                             | 2022-10-09 11:15 | Install        |  421 EE
[root@localhost ~]#
```

2. 执行**dnf history undo** <ID号>命令回退此条历史操作。

5.3 使用 OSMT 工具升级

5.3.1 概述

OSMT是华为云提供的对HCE版本及RPM包升级和回退的工具。OSMT可自定义配置RPM包的升级范围，并支持周期性定时升级、在指定的时间段单次升级、延时升级并重启等功能。

- **版本升级和回退**：介绍对整体HCE系统的升级及回退操作。
- **更新RPM包**：介绍仅对RPM包的升级和回退操作。

说明

OSMT仅支持针对HCE 2.0及以上的版本进行升级和回退。该工具会周期访问repo源以获取软件更新信息，从而产生网络流量。您可通过systemctl stop osmt-agent命令停止该服务，并通过systemctl disable osmt-agent命令禁用该服务自启动。

5.3.2 约束限制

- 升级和回退的耗时受更新的RPM包数量、大小和从repo源下载速度等影响，一般会在30分钟内完成，具体时间视实际环境确定，执行操作时注意预留足够的时间。
- OSMT工具仅支持对官方repo源中的RPM包进行升级，请确保repo源配置的正确性。修改repo源后需要执行**systemctl restart osmt-agent**重启osmt-agent服务。

  创建**/etc/yum.repos.d/hce.repo**文件并配置如下内容：

```
[base]
name=HCE $releasever base
baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

- 建议通过**osmt config**命令来修改配置文件，使用其他方式修改配置文件，可能导致OSMT功能异常。
- 升级操作必须使用root用户。
- 系统或RPM包的升级回退对剩余空间的要求：
  - 剩余内存至少512M。
  - 根分区剩余空间至少1.5G。
  - 备份内容的存储目录（store_path）剩余空间至少8G。

  - 系统中/boot分区剩余空间至少100M。

    说明

    - 升级范围、目标版本不同，所需存储空间不同。升级时OSMT工具会自动估算升级所需空间，如果剩余空间不足，会给出相应的错误提示。
    - 以上对内存、空间的阈值要求，是基于升级过程中最佳实践的值，不建议用户修改默认配置，如果修改的阈值过小，当系统剩余内存、空间过少时可能会导致升级过程失败。

- 版本升级、回退对selinux状态的影响：
  - 版本升级对selinux状态不产生影响，即版本升级前后，selinux状态一致。
  - 若回退前系统的selinux状态为enforcing，回退后系统自动将selinux状态变为permissive。
  - 若要开启selinux，请手动修改selinux的状态为enforcing并再次重启系统，可恢复到enforcing状态。
  - 若回退前selinux状态为disabled，回退后仍然为disabled状态。此时系统回退对selinux状态不产生影响。
  - 若要开启selinux，需要将selinux的状态先设置为permissive，并在根目录下创建.autorelabel文件并且重启，再将selinux状态修改为enforcing并且重启。
- 升级开始前，OSMT工具会对系统健康状态进行评估，以确保升级过程正常。如升级前检查失败，请根据提示信息进行处理。您也可以手动执行检查命令进行检查，详见**OSMT命令帮助信息**。
- OSMT工具升级过程依赖dnf工具，为了确保升级过程的稳定，OSMT工具会将dnf及其依赖RPM包升级至最新。如需回退这些RPM包，可参见**回退步骤**。
- 如果在RPM包更新后，系统配置被修改（**sysctl -a**可查询系统配置），则存在无法使用OSMT工具升级的情况。可用sysctl命令刷新系统配置，**sysctl -p** *<file>*可指定生效的配置文件。**sysctl --system**可应用所有系统目录下的配置文件，如果使用该命令，需要提前确认所有系统目录下的内核配置文件。
- 内核、内核热补丁分别不能超过5个版本。如果其中一个超过5个版本，则在OSMT检查阶段会检查失败，请您卸载不需要的版本后重新进行升级检查。
- 在chrony服务和ntp服务共存，并且chrony服务处于active状态的情况下，OSMT检查阶段会不通过，请您终止chrony服务或卸载chrony、ntp其中一个服务后重新升级。

5.3.3 版本升级和回退

本节介绍对整体HCE版本的升级及回退操作。

版本升级或回退时，会同时将RPM包更新到目标系统对应的RPM包版本，和osmt.conf配置文件中的黑白名单rpm列表无关。

版本升级

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

   ```
   [base]
   name=HCE $releasever base
   baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
   enabled=1
   ```

```
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

说明

- 错误的配置内容可能会导致OSMT升级失败或非预期的升级行为。

2. 更新OSMT软件版本。

   OSMT软件版本和HCE版本存在配套关系。HCE默认安装当前系统的OSMT工具，系统升级时，需要将OSMT更新至目标系统版本对应的OSMT版本。

   执行**dnf update osmt -y --releasever** [*系统目标版本号*] 更新OSMT软件版本。

   说明

   若误将OSMT删除，执行**dnf install osmt -y --releasever** [*系统目标版本号*] 进行安装。例如，将当前系统为HCE 2.0，则执行**dnf install osmt -y --releasever 2.0**命令安装OSMT2.0版本。

3. 升级HCE系统版本。

   osmt update --releasever [*系统目标版本号*] **--reboot_config** [*重启配置*]

   请根据是否需要立刻重启，选择合适的升级方式。更多的升级选项，详见**osmt update -h**。

   - 将HCE 2.0升级到目标版本，如HCE 3.0。

     osmt update --releasever 3.0

     升级后，须执行**reboot**命令重启系统，目标系统版本才能生效。

   - 将HCE 2.0升级到目标版本，如HCE 3.0，并立刻重启。

     osmt update --releasever 3.0 --reboot_config always

   - 将HCE 2.0升级到目标版本，如HCE 3.0，并指定重启时间，如“2022-12-30 23:00:00”。

     osmt update --releasever 3.0 --reboot_config "2022-12-30 23:00:00"

4. 重启完成后，检查是否升级成功。

   执行**cat /etc/hce-latest**查看**hceversion**字段，若此字段中版本部分是**--releasever**指定的版本号，表示升级成功。

5. （可选）删除升级备份文件。

   确认升级后功能正常后，执行**osmt remove**删除备份文件。

   说明

   请确认升级无异常后再执行**osmt remove**。执行**osmt remove**将删除所有升级备份数据，执行后无法再执行回退。

版本回退

1. 请根据是否需要立刻重启，选择合适的回退方式。
   - 回退至原系统，不立刻重启。
     osmt rollback
   - 回退至原系统并立刻重启。使用此方式，请忽略步骤2。
     osmt rollback --reboot_config always
2. 执行**reboot**命令重启系统。
   必须重启系统才能回退到HCE的原系统版本。
3. 检查是否回退成功。
   可执行**cat /etc/hce-latest**查看**hceversion**字段，若此字段中版本部分是升级前的版本号，表示已回退成功。

5.3.4 更新 RPM 包

5.3.4.1 准备工作

RPM包的更新方法有两种：使用**osmt update**命令更新和使用后台osmt-agent服务自动更新。此两种方法，都须先执行本节操作。

1. 确认repo源配置正常。
   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

```
[base]
name=HCE $releasever base
baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

   说明
   - 错误的配置内容可能会导致OSMT升级失败或非预期的升级行为。
2. 执行**dnf update osmt -y**命令更新OSMT升级工具。
3. 配置/etc/osmt/osmt.conf文件。
   OSMT根据配置文件osmt.conf的设置，对RPM包进行更新。请根据需要配置osmt.conf文件。

```
[auto]
#if auto_upgrade is True, the osmt-agent will auto upgrade rpms use osmt.conf and reboot between time interval we specified
#the value of cycle_time means the  osmt-agent will check upgrade every cycle_time seconds, default
```

```
86400s(1 day)
#When a configuration item has a line break, you need to leave a space or tab at the beginning of
the line
auto_upgrade = False
cycle_time = 3600
minimal_interval = 3600
auto_upgrade_window = "22:00-05:00"
auto_upgrade_interval = 1

[Package]
# There are three rules of filters, all enabled by default. severity will be effect only when the types
contain security, it is the subtype of security.
# The following are the three rules:
#    1. white list has the highest priority, if whitelist is configured then ignore other rules and filter out
the whitelist packages from the full list of packages to be upgrade
#    2. Filter the update range by types, when the types contain security, further filter the severity of
security updates severity, only upgrade the severity level of security.
#    3. Filter blacklist to remove packages in blacklist from types filter results, and packages which
depend on packages in blacklist will also be removed.
# filters must contain at least one types rule, if the types rule is empty, the -a option will not upgrade
any packages (by default all 3 filters are enabled).

filters = "types, blacklist"
whitelist = ""

# types include: security, bugfix, enhancement, newpackage, unknown/other
# if types is empty, no package will be upgrade
types = "security"
# severity is the subtype of security, include: low, moderate, important, critical
severity = "important, critical"
blacklist = ""
# The rpm package that requires a system reboot to take effect after the upgrade
need_reboot_rpms = "kernel,kernel-debug,kernel-debuginfo,kernel-debuginfo-common,kernel-
devel,kernel-headers,kernel-ori,kernel-tools,kernel-tools-libs,glibc,glibc-utils,glibc-static,glibc-
headers,glibc-devel,glibc-common,dbus,dbus-python,dbus-libs,dbus-glib-devel,dbus-glib,dbus-
devel,systemd,systemd-devel,systemd-libs,systemd-python,systemd-sysv,grub2,grub2-efi,grub2-
tools,openssl,openssl-devel,openssl-libs,gnutls,gnutls-dane,gnutls-devel,gnutls-utils,linux-
firmware,openssh,openssh-server,openssh-clients,openssh-keycat,openssh-askpass,python-
libs,python,grub2-pc,grub2-common,grub2-tools-minimal,grub2-pc-modules,grub2-tools-extra,grub2-
efi-x64,grub2-efi-x64-cdboot,kernel-cross-headers,kernel-source,glibc-all-langpacks,dbus-
common,dbus-daemon,dbus-tools,systemd-container,systemd-pam,systemd-udev,grub2-efi-
aa64,grub2-efi-aa64-cdboot,grub2-efi-aa64-modules,openssl-perl,openssl-pkcs,kernel-tools-libs-
devel,glibc-debugutils,glibc-locale-source,systemd-help,grub2-efi-ia32-modules,grub2-efi-x64-
modules,grub2-tools-efi,grub2-help,openssl-pkcs11,grub2-efi-ia32-cdboot,osmt"
preinstalled_only = False
# Due to security requirements, the following packages need to be uninstalled during the upgrade
need_uninstall_rpm_list = "elfutils-extra,gcc,make,tcpdump,binutils-extra,strace,gdb,gdb-
headless,cpp,rpm-
build,cups,ypserv,telnet,ypbind,libtool,appict,kmem_analyzer_tools,mcpp,flex,cmake,llvm,rpcgen,wiresh
ark,netcat,nmap,ethereal"

[backup]
store_path = /var/log
backup_dir = /etc,/usr,/boot,/var,/run
exclude_dir =
recover_service =

#the minimum resources required(MB)
[resource_needed]
#min_req_boot_space = 100
#min_req_backup_space = 8192
#min_req_root_space = 1536
#min_req_memory = 512

[cmdline]
cmdline_value =
skip_swap = True

[conflict]
```

```
#conflict_rpm = test1,test2
# These rpms conflict with the upgrade and must be removed if installed.
conflict_rpms_list = "esc,initial-setup,python3-crypto,setroubleshoot,setroubleshoot-
legacy,setroubleshoot-server,setroubleshoot-plugins,openresty-openssl111,openresty-openssl111-
asan,openresty-openssl111-asan-devel,openresty-openssl111-debug,openresty-openssl111-debug-
devel,openresty-openssl111-devel,openresty-zlib,openresty-zlib-asan,openresty-zlib-asan-
devel,openresty-zlib-devel,dleyna-connector-dbus,dleyna-connector-dbus-devel,dleyna-core,dleyna-
core-devel,dleyna-server,tracker,tracker-devel,tracker-miners,kabi-dw,sblim-
sfcb,apull,elara,secpaver,ksh,python3-sssd,python3-editor,python3-Flask-SQLAlchemy,python3-bind"

[strategy]
timeout_action = "stop"
timeout_action_before = 0

[check]
daemon_whitelist = "sysstat-collect.service, sysstat-summary.service, man-db-cache-update.service,
systemd-tmpfiles-clean.service"
check_systemd_running_jobs = True
# the timeout of query systemd services
query_timeout = 30
check_rpm_packages = True
check_file_attr = True

[chroot_config]
chroot_switch = False
chroot_path = "/root/sut_chroot"
rpm_tar_name = "hce-upgrade_pack"
sut_config_file = "/etc/sut/sut.conf"
web_link_tar =
```

表 5-2 osmt.conf 主要配置项

配置项	说明
[auto]	● auto_upgrade：指定更新RPM包更新方式。默认为False。 – True：使用**osmt update**命令更新和使用后台osmt-agent服务自动更新两种方式都支持。 – False：仅支持使用**osmt update**命令更新RPM包。 ● auto_upgrade为True时，配套如下参数。 – cycle_time：检查是否有待更新软件包的周期，单位是秒。默认值为3600秒。 – minimal_interval：指定**osmt update -b**命令参数中开始时间和截止时间的最小时间间隔，单位是秒。默认值为3600秒。 – auto_upgrade_window：配置后台osmt-agent服务自动升级的时间窗，格式为"HH:MM-HH:MM"，表示升级的开始时间和截止时间。 如果截止时间小于开始时间，则表示本次升级时间段跨越自然日。如“22:00-05:00”表示升级时间段为当日22:00到次日凌晨5:00。 – auto_upgrade_interval：指定两次自动升级之间的最小间隔（单位：天）。 ● auto_upgrade为False时，仅配套如下参数，执行配置文件时[auto]配置项中其他参数不生效。 – cycle_time：检查是否有待更新软件包的周期，单位是秒。默认值为3600秒。 – minimal_interval：指定**osmt update -b**命令参数中开始时间和截止时间的最小时间间隔，单位是秒。默认值为3600秒。 ● motd_setup: 设置登录提示是否开启。默认为True。 – True：开启登录提示。 – False：关闭登录提示，设置后会立刻删除登录提示，并且不会再次生成。如果重新开启，需要使用**osmt update -s**或任意升级命令重新触发生成。

配置项	说明
[Package]	• filters：指定更新的范围，包括types、blacklist、whitelist。例如filters = "blacklist"，指升级时不更新黑名单中的RPM包。 – types：待更新的RPM包类型。 – blacklist：RPM包黑名单列表。 设置黑名单后，不对黑名单中的RPM包进行更新。如果某RPM包依赖黑名单中的包，则不会升级此RPM包。 – whitelist：RPM包白名单列表。 如果不设置白名单或者黑名单列表，默认更新所有可更新的RPM包。 白名单优先级大于黑名单，如果某RPM包同时配置在白名单和黑名单，则会升级此RPM包。 **说明** 如果使用命令指定了黑名单和白名单，以命令指定的黑名单和白名单为准，不再根据配置文件中的黑名单和白名单列表更新RPM包。 • need_reboot_rpms：升级后要重启的RPM包。 后台osmt-agent服务自动升级时，不会更新need_reboot_rpms中的RPM包。若要更新need_reboot_rpms列表中的RPM包，必须使用**osmt update --auto --reboot_config always**或**osmt update --auto --reboot_config “重启时间”**命令更新。 • preinstalled_only：当设置成True时，只升级/etc/osmt/preinstalled.list中的RPM包。
[backup]	• store_path：在该目录下创建备份目录。 升级过程中，OSMT会在store_path下创建.osbak目录，如果原先存在该目录，则需要先使用**osmt remove**将其删除。 • backup_dir：需要备份的目录。其中/etc、/usr、/boot、/var、/run为默认的需要备份的目录，且不可以删减。这些目录会在执行版本升级功能的时候自动进行备份。注意不能与exclude_dir中配置的目录相同，否则升级工具会在目录处理时产生冲突而终止升级流程。 • exclude_dir：不需要备份的目录。默认没有配置，一般建议配置成业务的数据目录，注意不能与backup_dir中配置的目录相同，否则升级工具会在目录处理时产生冲突而终止升级流程。 • recover_service：OSMT会检查升级前后此列表中服务的启用状态是否一致，如果服务的启用状态被修改，OSMT会尝试恢复此服务的状态。 **说明** [backup]中的路径需要配置为绝对路径的形式。
[cmdline]	cmdline_value：指定升级之后系统的启动项。请配置正确的启动项，确保系统能够正常启动。默认值为HCE的默认启动项。
[conflict]	conflict_rpm：指定升级过程中冲突的软件包，升级时系统将自动删除冲突的软件包。

配置项	说明
[check]	• check_systemd_running_jobs：设置是否在升级前检查系统中有正在启动或正在停止的服务，默认为True。 – True：升级前检查系统中是否有正在启动或正在停止的服务。 – False：不检查系统中是否有正在启动或正在停止的服务。 • check_rpm_packages：设置是否在升级前检查系统中rpm包的状态，包括包的依赖是否缺失、是否存在重复的包。默认为True。 – True：升级前检查rpm包的状态。 – False：升级前不检查rpm包的状态。 • check_file_attr：设置是否在升级前检查系统文件是否有只读的属性。默认为True。 – True：升级前检查系统中文件是否有只读属性。 – False：升级前不检查系统中文件是否有只读属性。
[chroot_config]	• chroot_switch：是否开启turbo方式的版本升级，默认为False。 – True：版本升级流程中开启turbo方式的升级。 – False：版本升级流程中不开启turbo方式的升级。 • chroot_path：设置turbo升级方式流程中的chroot环境目录。 • rpm_tar_name：设置turbo升级方式中预构建的tar包的名字。 • sut_config_file：设置sut的配置文件路径，默认是/etc/sut/sut.conf。 • web_link_tar：设置turbo升级方式中下载预构建的tar包的url地址。 **说明** [chroot_config]中的所有配置仅适用于管理面的HCE版本升级方式，turbo方式指的是使用预构建出来的目标版本的tar包通过chroot方式在指定目录下进行版本升级。

说明

其他配置项不建议修改，详情请参见**/etc/osmt/osmt.conf配置文件说明**。

5.3.4.2 osmt update 命令更新

手动更新RPM包有两种方式。

- 根据配置文件中的**filters**字段更新RPM包。
  osmt update --auto --reboot_config [*重启配置*]

表 5-3 重启配置参数说明

参数	说明
never	若未指定重启配置参数，或指定为never时，更新结束后，不重启。 此种方式下，**need_reboot_rpms**列表中的RPM包不会被升级。这种情况下，若要升级**need_reboot_rpms**列表中的RPM包，请将**filters**字段配置为"whitelist"且将对应的RPM包配置到"whitelist"选项中，操作命令如下。在这种情况下，升级完成后，请手动重启系统，使RPM包更新生效。 **osmt config -k filters -v "whitelist"** **osmt config -k whitelist -v** "*rpm1, rpm2, rpm3*"
always	更新RPM包后（包括**need_reboot_rpms**列表中的RPM包），立即自动重启。
<specific time>	更新RPM包后（包括**need_reboot_rpms**列表中的RPM包），在指定的重启时间自动重启。时间格式如 "2020-02-02 2:02:02"。

- 使用黑名单和白名单的方式更新RPM包。

  osmt update --pkgs [*rpm1 rpm2 rpm3 ...*] **--exclude_pkgs** [*rpm4 rpm5 rpm6 ...*] --**reboot_config** [*重启配置*]

  - --pkgs：可选，指待更新的白名单列表，RPM包以空格分隔。

    例如，执行如下命令更新白名单中的hce-logos、hce-lsb、tomcat包。

    osmt update --pkgs hce-logos hce-lsb tomcat
  - --exclude_pkgs：可选，指待更新的黑名单列表，RPM包以空格分隔。

    例如，执行如下命令不更新黑名单中的ongres-scram、llvm-static包。

    osmt update --exclude_pkgs ongres-scram llvm-static
  - --reboot_config [*重启配置*]：可选，指定重启方式，包含always、never、重启时间。
    - always：更新结束后，若有重启才能生效的RPM包，则立即自动重启；若没有，则不重启。
    - never ： 更新结束后，不重启。
    - <specific time>：指定重启时间。更新结束后，若有重启才能生效的RPM包，则在指定时间自动重启；若没有，则不重启。时间格式如 "2020-02-02 2:02:02"。

  说明

  - 如果使用黑名单和白名单的更新方式，--pkgs和--exclude_pkgs至少使用一个。
  - 如果使用命令指定了黑名单和白名单，以命令指定的黑名单和白名单为准，不再根据配置文件中的黑名单和白名单列表更新RPM包。

5.3.4.3 osmt-agent 服务自动更新

osmt-agent服务支持周期性检查是否有待更新的RPM包，并自动更新RPM包。检查的周期和执行更新的时间段可以自定义设置。

1. 执行以下命令，确保osmt.conf文件auto_upgrade字段为True。

   osmt config -k auto_upgrade -v True

2. 执行**systemctl status osmt-agent.service**命令确认osmt-agent服务是否正常开启。
   - Active为active (running)状态，表示osmt-agent正常开启。
   - 如果osmt-agent没有处于active (running)状态，请执行**systemctl start osmt-agent.service**命令启动osmt-agent。

   图 5-2 确认 osmt-agent 服务是否正常开启

   ```
   ● osmt-agent.service - osmt-agent - The agent that manages HCE OS.
        Loaded: loaded (/usr/lib/systemd/system/osmt-agent.service; enabled; vendor preset: disabled)
        Active: active (running) since Sat 2022-12-24 18:32:42 CST; 2 days ago
      Main PID: 1421 (python)
         Tasks: 3 (limit: 21113)
        Memory: 72.7M
        CGroup: /system.slice/osmt-agent.service
                ├─  1421 python /usr/bin/osmt_server
                ├─  1474 /bin/bash /usr/bin/osmt-agent
                └─ 32445 sleep 3600
   ```

3. 执行如下命令设置自动更新RPM包的时间段与升级周期。
   - 指定可自动升级的时间段。

     命令格式：osmt config -k auto_upgrade_window -v "auto_upgrade_window"

     auto_upgrade_window：配置后台osmt-agent服务自动升级的时间窗，格式为"HH:MM-HH:MM"，表示升级的开始时间和截止时间。

     如果截止时间小于开始时间，则表示本次升级时间段跨越自然日。如“22:00-05:00”表示升级时间段为当日22:00到次日凌晨5:00。

     例如，配置当日23:00到次日01:00时间段为可升级时间窗：

     osmt config -k auto_upgrade_window -v "23:00-01:00"
   - 指定检查升级的时间间隔。

     命令格式：osmt config -k auto_upgrade_interval -v auto_upgrade_interval

     auto_upgrade_interval：指定两次自动升级之间的最小间隔（单位：天）。

     例如，配置每隔1天进行自动升级的命令为：

     osmt config -k auto_upgrade_interval -v 1

5.3.5 升级后续操作

1. 升级成功后，请及时确认业务运行情况，如果业务运行正常，请在合适的时候执行**osmt remove**命令删除备份内容，删除后将无法回退本次升级内容。
2. 安全规范要求chronyd服务在安装/升级后默认处于disabled状态，所以从HCE-2.0.2206版本升级至新版本后，chronyd服务会处于disabled状态。如有需要，您可通过**systemctl enable chronyd**使能该服务，并通过**systemctl start chronyd**启动该服务。

5.3.6 回退 RPM 包

执行**osmt rollback --reboot_config always**命令回退RPM包。仅支持回退最近一次更新的RPM包。

--reboot_config always：可选，若上次升级有need_reboot_rpms列表中的RPM包，请使用此参数。

如果不带--reboot_config always，上次升级有涉及need_reboot_rpms列表中的RPM包，需要手动重启后才能回退生效。

说明

请使用最新版本的OSMT工具进行操作，不建议通过OSMT工具回退OSMT自身版本。

5.4 附录

5.4.1 OSMT 命令帮助信息

- 执行**osmt -h**命令，显示OSMT的帮助信息。

```
[root@localhost SOURCES]# osmt -h
usage: osmt [-h] {update,rollback,remove,config,job} ...

positional arguments:
  {update,rollback,remove,config,job}
    update              update packages
    rollback            rollback last upgrade
    remove              remove backup files in store path
    config              modify config file by command line
    job                 handle upgrade task.

options:
  -h, --help            show this help message and exit
```

表 5-4 osmt 参数说明

参数	说明
update	升级操作系统或RPM包。
rollback	回退操作系统或RPM包。
remove	删除存储路径中的备份文件。
config	查询或修改配置文件。
job	查询或管理升级任务。
-h, --help	可选参数，提供**osmt**命令的帮助信息。

- 执行**osmt update -h**命令，显示升级操作系统或RPM包的帮助信息。

```
[root@localhost SOURCES]# osmt update -h
usage: osmt update [-h] [--nosignature] [-s] [--all] [--security] [--version] [-a] [-p  PKGS [PKGS ...]] [-e  EXCLUDE_PKGS [EXCLUDE_PKGS ...]] [-v  RELEASEVER] [-r  REBOOT_CONFIG]
                   [-b BETWEEN] [-j] [-c]
optional arguments:
  -h, --help            show this help message and exit
  --nosignature         ignore the signature of packages
  -s , --show           show updateinfo
  --all                 show all pkgs which can update, 'osmt update --show --all'
  --security            show security pkgs which can update
  --version             show all version can update to
  -a , --auto           auto update use config file
  -p  PKGS [PKGS ...], --pkgs PKGS [PKGS ...]
                        specify the packages to upgrade
  -e  EXCLUDE_PKGS [EXCLUDE_PKGS ...], --exclude_pkgs EXCLUDE_PKGS [EXCLUDE_PKGS ...]
                        specify the packages not to upgrade
  -v  RELEASEVER, --releasever RELEASEVER
```

```
                    specify the release version to upgrade
 -r  REBOOT_CONFIG, --reboot_config REBOOT_CONFIG
                    you can choose between always, never or a specific time. 'always': reboot os after
update ends if need. 'never': never reboot os automatically. '<specific time>':
                    reboot at specified time, format like "2020-02-02 2:02:02".
 -b BETWEEN, --between BETWEEN
                    start upgrade time and end upgrade time, format like: '2020-02-02
2:02:02','2020-02-02 4:02:02'
 -j , --job          run upgrade in background.
 -c, --check          check upgrade task.
 -V, --verbose        show more log to screen
 -o, --preinstalled-only
                    upgrade preinstalled packages only
 -t, --retry          retry previous upgrade action
 --nocheck            do not check before upgrade
 -f, --fix           auto fix some system problems checked out
```

表 5-5 osmt update 主要参数说明

参数	说明
-h, --help	提供**osmt update**命令的帮助信息。
--nosignature	升级时不根据包的签名筛选待更新的RPM包。
-s,--show	显示当前系统可用的升级或更新信息。 ● --all：显示所有待更新的RPM包。 ● --security：显示待更新的安全包。 ● --version：显示所更新到的版本号。
-a,--auto	**指定更新RPM包更新方式**，与-v、-p、-e互斥。
-p,--pkgs	指定需要更新的RPM包白名单列表，与-v、-a互斥。
-e,--exclude_pkgs	指定不需要更新的RPM包黑名单列表，与-v、-a互斥。
-v , --releasever	指定需要升级到的HCE版本号。
-r, --reboot_config	指定重启方式。 ● always：更新结束后，若有重启才能生效的RPM包，则立即自动重启；若没有，则不重启。 ● never ： 更新结束后，不重启。 ● <specific time>：指定重启时间。更新结束后，若有重启才能生效的RPM包，则在指定时间自动重启；若没有，则不重启。时间格式如 "2020-02-02 2:02:02"。
-b, --between	指定osmt执行更新的开始时间和结束时间，格式为"HH:MM-HH:MM"。 如果截止时间小于开始时间，则表示本次升级时间段跨越自然日。如“22:00-05:00”表示升级时间段为当日22:00到次日凌晨5:00。
-j, --job	以后台进程方式进行本次升级。

参数	说明
-c, --check	升级前检查系统状态，确认系统是否能升级。该检查操作是可选的，在实际升级时会再次执行升级前检查。 建议在升级命令增加-c参数，对升级命令进行检查。例如，执行**osmt update -a**进行升级前，执行**osmt update -a -c**检查系统状态。
-V, --verbose	可选参数，显示详细的升级信息。
-o, --preinstalled-only	可选参数，仅升级/etc/osmt/preinstalled.list中的RPM包列表，该选项仅对版本升级有效。
-t, --retry	可选参数，进行升级重试。
--nocheck	可选参数，升级前不做任何检查，直接进入升级流程。
-f, --fix	可选参数，版本升级流程中自动修复部分环境检查的问题。

- 执行**osmt rollback -h**命令，显示回退操作系统或RPM包的帮助信息。

```
usage: osmt rollback [-h] [-r  {never,always}] [-V] [-t] [--nocheck]

optional arguments:
  -h, --help            show this help message and exit
  -r  {never,always}, --reboot_config {never,always}
                        whether to reboot after rollback
  -V, --verbose         show more log to screen
  -t, --retry           retry previous upgrade action
  --nocheck             do not check before rollback
```

表 5-6 osmt rollback 参数说明

参数	说明
-h, --help	提供**osmt rollback**命令的帮助信息。
-r, --reboot_config	指定是否允许重启。
-V, --verbose	可选参数，显示详细的过程日志。
-t, --retry	可选参数，进行回退重试。
--nocheck	可选参数，回退前不进行任何检查，直接进入回退阶段。

- 执行**osmt config -h**命令，显示修改配置项或显示配置项的帮助信息。

```
usage: osmt config [-h] [-k KEY] [-v VALUE] [-V]

optional arguments:
  -h, --help            show this help message and exit
  -k KEY, --key KEY     key of config item
  -v VALUE, --value VALUE
                        value of config item
  -V, --verbose         show more log to screen
```

表 5-7 osmt config 参数说明

参数	说明
-h, --help	提供**osmt config**命令的帮助信息。
-k, --key	指定要查询或修改的key值。
-v, --value	指定对应修改key值的value值。
-V, --verbose	可选参数，显示详细的过程日志。

说明

建议只通过**osmt config**命令来修改配置文件，使用其他方式修改配置文件，可能导致OSMT功能异常。

- 指定**osmt job -h**命令，显示任务管理的帮助信息。

```
usage: osmt job [-h] [-s] [-c] [-d DELAY] [-y]

optional arguments:
  -h, --help            show this help message and exit
  -s, --show            show task info.
  -c, --cancel          cancel current task.
  -d DELAY, --delay DELAY
                        delay task
  -y, --yes             never ask for yes.
  -V, --verbose         show more log to screen
```

表 5-8 osmt job 参数说明

参数	说明
-h, --help	提供**osmt job**命令的帮助信息。
-s, --show	显示后台任务信息。
-d, --delay	允许将当前等待重启的任务延迟一段时间，具体格式为“1:50:00”，表示将重启时间推迟1小时50分。
-c, --cancel	取消当前job。
-y, --yes	默认用户同意本次操作。
-V, --verbose	可选参数，显示详细的过程日志。

5.4.2 /etc/osmt/osmt.conf 配置文件说明

本节对OSMT工具的配置文件osmt.conf不建议修改的配置项进行说明。

```
[auto]
# if auto_upgrade is True, the osmt-agent will auto upgrade rpms use osmt.conf and reboot between time interval we specified
# the value of cycle_time means the  osmt-agent will check upgrade every cycle_time seconds, default 86400s(1 day)
# When a configuration item has a line break, you need to leave a space or tab at the beginning of the line
auto_upgrade = False
cycle_time = 3600
minimal_interval = 3600
```

```
auto_upgrade_window = "22:00-05:00"
auto_upgrade_interval = 1
[Package]
# There are three rules of filters, all enabled by default. severity will be effect only when the types contain security, it is the subtype of security.
# The following are the three rules:
#    1. whitelist has the highest priority, if whitelist is configured then ignore other rules and filter out the whitelist packages from the full list of packages to be upgrade
#    2. Filter the update range by types, when the types contain security, further filter the severity of security updates severity, only upgrade the severity level of security.
#    3. Filter blacklist to remove packages in blacklist from types filter results, and packages which depend on packages in blacklist will also be removed.
# filters must contain at least one types rule, if the types rule is empty, the -a option will not upgrade any packages (by default all 3 filters are enabled).

filters = "types, blacklist"
whitelist = ""

# types include: security, bugfix, enhancement, newpackage, unknown
# if types is empty, no package will be upgrade
# types = security, bugfix, enhancement, newpackage, unknown
types = "security"
# severity is the subtype of security, include: low, moderate, important, critical
severity = "important, critical"
blacklist = "mysql"
# 升级后需要重启系统才能生效的rpm包
need_reboot_rpms = kernel,kernel-debug,glibc,glibc-utils,dbus,dbus-python…
preinstalled_only = False
[backup]
store_path = /var/log
backup_dir = /etc,/usr,/boot,/var,/run
exclude_dir =
recover_service =
[resource_needed]
#the minimum resources required(MB)
#min_req_boot_space = 100
#min_req_backup_space = 8192
#min_req_root_space = 1536
#min_req_memory = 512
[cmdline]
cmdline_value = crashkernel=512M resume=/dev/mapper/hce-swap rd.lvm.lv=hce/root rd.lvm.lv=hce/swap
crash_kexec_post_notifiers panic=3 nmi_watchdog=1 rd.shell=0
[conflict]
#conflict_rpm = test1,test2

[strategy]
timeout_action = "stop"
timeout_action_before = 0

[check]
daemon_whitelist=sysstat-collect.service, sysstat-summary.service, systemd-tmpfiles-clean.service
# the timeout of query systemd services
check_systemd_running_jobs = True
query_timeout = 30
check_rpm_packages = True
check_file_attr = True

[chroot_config]
chroot_switch = False
chroot_path = "/root/sut_chroot"
rpm_tar_name = "hce-upgrade_pack"
sut_config_file = "/etc/sut/sut.conf"
web_link_tar =
```

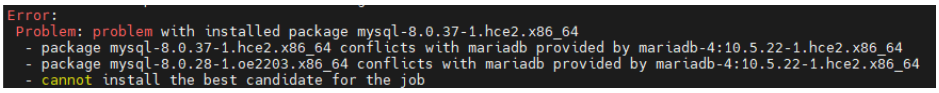
表 5-9 osmt.conf 不建议修改的配置项

配置项	说明
types	按照security、 bugfix、enhancement、newpackage、unknown五个配置项指定RPM包的更新范围，不建议修改。如有特殊需要可以根据系统实际情况进行修改。
severity	升级安全更新，不建议修改。默认升级安全更新。如有特殊需要可以根据系统实际情况进行修改。
[resource_needed]	指升级或更新前进行检查的系统资源限制值，不建议修改。如有特殊需要可以根据系统实际情况进行修改。
[chroot_config]	用于配置是否进行turbo方式升级的相关配置项，仅用于管理面HCE升级，租户面不适用，请保持默认配置。

5.4.3 常见问题

1. 使用yum/dnf/osmt工具在安装或升级过程中，出现报错，错误信息为：**package *<a>* conflicts with *<b>* provided by *<c>*,** 如：

   图 5-3 错误信息

   问题原因：系统中安装了同一软件功能的两个不同软件包，这两个软件包之间存在冲突（如mysql和mariadb这两个软件包之间存在冲突）。

   解决方案：通过rpm -e/ yum remove/ dnf remove命令 删除其中一个软件包，只保留需要的软件包。

2. 使用yum/dnf/osmt工具在安装或升级过程中，出现报错，错误信息为：**file *<x>* from install of *<y>* conflicts with file from *<z>***，如：

   图 5-4 错误信息

   问题原因：两个不同的软件包中存在路径完全相同的文件，导致冲突（如mysql-config和mariadb-config两个包中都包含了/etc/my.cnf这个文件）。

   解决方案：通过rpm -e/ yum remove/ dnf remove命令删除其中一个软件包，只保留需要的软件包。

6 对 HCE 进行安全更新

6.1 安全更新概述

本节主要介绍如何使用yum或dnf命令查询并安装HCE中的安全更新。

各版本对yum和dnf命令的支持情况不同，本节以yum命令为例介绍。

说明

dnf作为yum的替代者，提供更好的性能，dnf和yum命令的使用方法相同。

- HCE 2.0及以上版本支持yum和dnf命令。
- HCE 2.0之前版本仅支持yum命令。

6.2 关于通用漏洞披露（CVE）

CVE（Common Vulnerabilities and Exposures）是已公开披露的各种计算机安全漏洞，所发现的每个漏洞都有一个专属的CVE编号。HCE为保障系统安全性，紧密关注业界发布的CVE信息，并会及时修复系统内各类软件漏洞，增强系统的安全性。您可在HCE的安全公告中查看安全更新记录。

- **Huawei Cloud EulerOS 1.1安全公告**
- **Huawei Cloud EulerOS 2.0安全公告**

根据CVSS（Common Vulnerability Scoring System）评分，HCE将安全更新分为四个等级：

- Critical（高风险，必须安装）
- Important（较高风险，强烈建议安装）
- Moderate（中等风险，推荐安装）
- Low（低风险，可选安装）

本章节以HCE 2.0为例举例说明。在对HCEOS进行安全更新时，需要部署远端repo源并添加安全公告，以及配置客户端访问repo源，具体操作请参考**HCE的REPO源配置**章节。

说明

HCE版本不同，部分显示可能与本文档存在差异，以实际显示为准。

6.3 yum 命令参数

命令格式：yum <command> [option]

表 6-1 command 的主要参数

参数	说明
help	显示帮助信息。
updateinfo	显示软件包更新信息摘要。
upgrade	升级软件包。
check-update	检查可用的软件包更新。

表 6-2 option 的主要参数

参数	说明
-h, --help, --help-cmd	显示命令帮助信息。
--security	显示安全相关的软件包信息。
--advisory ADVISORY	指定ADVISORY相关的软件包，可用于安装、查询或更新。 多个软件包之间使用英文逗号分隔。
--cve CVES	指定CVE相关的软件包，可用于安装、查询或更新。 多个软件包之间使用英文逗号分隔。
--sec-severity {Critical,Important,Moderate,Low}	指定安全更新等级相关的软件包，可用于安装、查询或更新。 {}中的安全更新等级参数可任意组合。

说明

更多详细信息，请使用**yum --help**获取帮助信息。

6.4 查询安全更新

命令格式：yum updateinfo <command> [option]

- 执行**yum updateinfo**命令，查询全部可用的安全更新信息。

```
[root@localhost ~]# yum updateinfo
Last metadata expiration check: 0:03:05 ago on Thu 08 Sep 2022 05:30:23 PM CST.
```

```
Updates Information Summary: available
12 Security notice(s)
4 Critical Security notice(s)
6 Important Security notice(s)
2 Moderate Security notice(s)
```

- <command>的主要参数。
  - list：查询当前可用的安全更新列表。

    ```
    [root@localhost ~]# yum updateinfo list
    Last metadata expiration check: 0:03:32 ago on Thu 08 Sep 2022 05:30:23 PM CST.
    HCE2-SA-2022-0006 Critical/Sec.  curl-7.79.1-2.h6.hce2.x86_64
    HCE2-SA-2022-0011 Moderate/Sec.  gnupg2-2.2.32-1.h6.hce2.x86_64
    HCE2-SA-2022-0002 Important/Sec. kernel-5.10.0-60.18.0.50.h425_2.hce2.x86_64
    ```

  - info <SA ID>：查询指定SA ID的安全更新详情。

    ```
    [root@localhost ~]# yum updateinfo info HCE2-SA-2024-0262
    Last metadata expiration check: 0:01:07 ago on Wed 26 Mar 2025 11:08:19 AM CST.
    ================================================================================
      An update for wget is now available for HCE 2.0
    ================================================================================
      Update ID: HCE2-SA-2024-0262
           Type: security
        Updated: 2024-09-23 18:09:48
           CVEs: CVE-2024-38428
    Description: Security Fix(es):
                 :
                 : url.c in GNU Wget through 1.24.5 mishandles semicolons in the userinfo subcomponent
    of a URI, and thus there may be insecure behavior in which data that was supposed to be in the
    userinfo subcomponent is misinterpreted to be part of the host subcomponent.
    (CVE-2024-38428)
       Severity: Critical
    ```

- [option]的主要参数。
  - --sec-severity={Critical,Important,Moderate,Low}：指定安全更新级别，{}中的安全更新等级参数可任意组合。

    例如，使用--sec-severity=Critical查询某个安全更新级别。

    ```
    [root@localhost ~]# yum updateinfo list --sec-severity=Critical
    Last metadata expiration check: 0:10:15 ago on Thu 08 Sep 2022 05:30:23 PM CST.
    HCE2-SA-2022-0006 Critical/Sec. curl-7.79.1-2.h6.hce2.x86_64
    HCE2-SA-2022-0003 Critical/Sec. libarchive-3.5.2-1.h2.hce2.x86_64
    HCE2-SA-2022-0006 Critical/Sec. libcurl-7.79.1-2.h6.hce2.x86_64
    ......
    ```

    例如，使用--sec-severity={Critical,Moderate}查询多个安全更新级别。

    ```
    [root@localhost ~]# yum updateinfo list --sec-severity={Critical,Moderate}
    Last metadata expiration check: 0:11:07 ago on Thu 08 Sep 2022 05:30:23 PM CST.
    HCE2-SA-2022-0006 Critical/Sec. curl-7.79.1-2.h6.hce2.x86_64
    HCE2-SA-2022-0011 Moderate/Sec. gnupg2-2.2.32-1.h6.hce2.x86_64
    HCE2-SA-2022-0003 Critical/Sec. libarchive-3.5.2-1.h2.hce2.x86_64
    ......
    ```

  - --cve=<CVE ID>：查询指定的CVE ID。

    ```
    [root@localhost ~]# yum updateinfo info --cve=CVE-2024-38428
    Last metadata expiration check: 0:11:10 ago on Wed 26 Mar 2025 11:08:19 AM CST.
    ================================================================================
      An update for wget is now available for HCE 2.0
    ================================================================================
      Update ID: HCE2-SA-2024-0262
           Type: security
        Updated: 2024-09-23 18:09:48
           CVEs: CVE-2024-38428
    Description: Security Fix(es):
                 :
                 : url.c in GNU Wget through 1.24.5 mishandles semicolons in the userinfo subcomponent
    ```

```
of a URI, and thus there may be insecure behavior in which data that was supposed to be in the
userinfo subcomponent is misinterpreted to be part of the host subcomponent.
(CVE-2024-38428)
  Severity: Critical
```

说明

更多详细信息，请使用**yum updateinfo --help**获取帮助信息。

6.5 检查安全更新

- 执行**yum check-update --security**命令，检查系统当前可用的安全更新。

```
[root@localhost ~]# yum check-update --security
Last metadata expiration check: 0:13:32 ago on Wed 26 Mar 2025 11:08:19 AM CST.

c-ares.x86_64                          1.18.1-1.r5.hce2                    hceversion
curl.x86_64                            7.79.1-2.r34.hce2                   hceversion
dnsmasq.x86_64                           2.86-1.r20.hce2                     hceversion
expat.x86_64                           2.4.1-5.r8.hce2                     hceversion
......
```

- 执行**yum check-update --sec-severity={Critical,Important,Moderate,Low}**命令，检查指定级别的安全更新。

  {}中的安全更新等级参数可任意组合。

```
[root@localhost ~]#  yum check-update --sec-severity=Moderate
Last metadata expiration check: 0:15:20 ago on Wed 26 Mar 2025 11:08:19 AM CST.

c-ares.x86_64                          1.18.1-1.r5.hce2                    hceversion
curl.x86_64                            7.79.1-2.r34.hce2                   hceversion
expat.x86_64                           2.4.1-5.r8.hce2                     hceversion
gnutls.x86_64                          3.7.2-2.r31.hce2                    hceversion
gnutls-utils.x86_64                      3.7.2-2.r31.hce2                    hceversion
......
```

6.6 安装安全更新

- 执行**yum upgrade --security**命令，安装全部安全更新。

```
[root@localhost ~]# yum upgrade --security
Last metadata expiration check: 0:16:41 ago on Wed 26 Mar 2025 11:08:19 AM CST.
Dependencies resolved.
================================================================
Package        Arch        Version                 Repository    Size
================================================================
Installing:
Kernel         x86_64        5.10.0-60.18.0.50.h498_2.hce2    hceversion        49 M
Upgrading:
Curl           x86_64        7.79.1-2.h6.hce2           hceversion        147 k
......
Transcription Summary
================================================================
Install   1 Package
Upgrade  22 Packages
Total download size: 69 M
Is this ok [y/N]:
```

- 执行**yum upgrade --sec-severity={Critical,Important,Moderate,Low}**命令，安装指定级别的安全更新。

  {}中的安全更新等级参数可任意组合。

```
[root@localhost ~]# yum upgrade --sec-severity=Moderate
Last metadata expiration check: 0:32:27 ago on Thu 08 Sep 2022 05:30:23 PM CST.
Dependencies resolved.
================================================================
Package                 Architecture          Version           Repository       Size
```

```
================================================================
Upgrading:
gnupg2              x86_64              2.2.32-1.h6.hce2      hceversion          2.2 M
python3-unbound     x86_64              1.13.2-3.h2.hce2      hceversion          96 k
unbound-libs        x86_64              1.13.2-3.h2.hce2      hceversion          505 k
Transaction Summary
================================================================Upgrade      3
Packages
Total download size: 2.8 M
Is this ok [y/N]:
```

- 执行**yum upgrade --advisory=<SA ID>**命令，安装指定SA ID的安全更新。

  多个软件包之间使用英文逗号分隔。

```
root@localhost ~]# yum upgrade --advisory=HCE2-SA-2022-0033
Last metadata expiration check: 1:48:44 ago on Tue 13 Sep 2022 09:43:13 AM CST.
Dependencies resolved.
================================================================
Package             Architecture          Version               Repository          Size
================================================================
Upgrading:
python3-rpm         x86_64       4.17.0-8.h13.hce2       hceversion       87 k
rpm         x86_64       4.17.0-8.h13.hce2       hceversion       498 k
rpm-libs       x86_64       4.17.0-8.h13.hce2       hceversion       376 k
Transaction Summary
================================================================
Upgrade  3 Packages
Total download size: 962 k
Is this ok [y/N]:
```

- 执行**yum upgrade --cve=<CVE ID>**命令，安装指定CVE ID的安全更新。

  多个软件包之间使用英文逗号分隔。

```
[root@localhost ~]# yum upgrade --cve=CVE-2021-28861
Last metadata expiration check: 5:16:36 ago on Tue 13 Sep 2022 09:43:13 AM CST.
Dependencies resolved.
================================================================
Package               Architecture    Version           Repository    Size
================================================================
Upgrading:
python3             x86_64          3.9.9-7.h10.hce2          hceversion          8.0 M
python3-unversioned-command    x86_64          3.9.9-7.h10.hce2          hceversion          3.9 k
Transaction Summary
================================================================
Upgrade  2 Packages
Total download size: 8.0 M
Is this ok [y/N]:
```

7 HCE 获取 openEuler 扩展软件包

HCE默认不加载开源社区openEuler的repo源，避免openEuler的软件包和HCE的软件包冲突。

当前HCE 2.0版本仅兼容openEuler 22.03 LTS版本。本节介绍HCE 2.0版本如何获取openEuler 22.03 LTS的扩展软件包。获取方法有两种，请根据需要选择合适的方法。

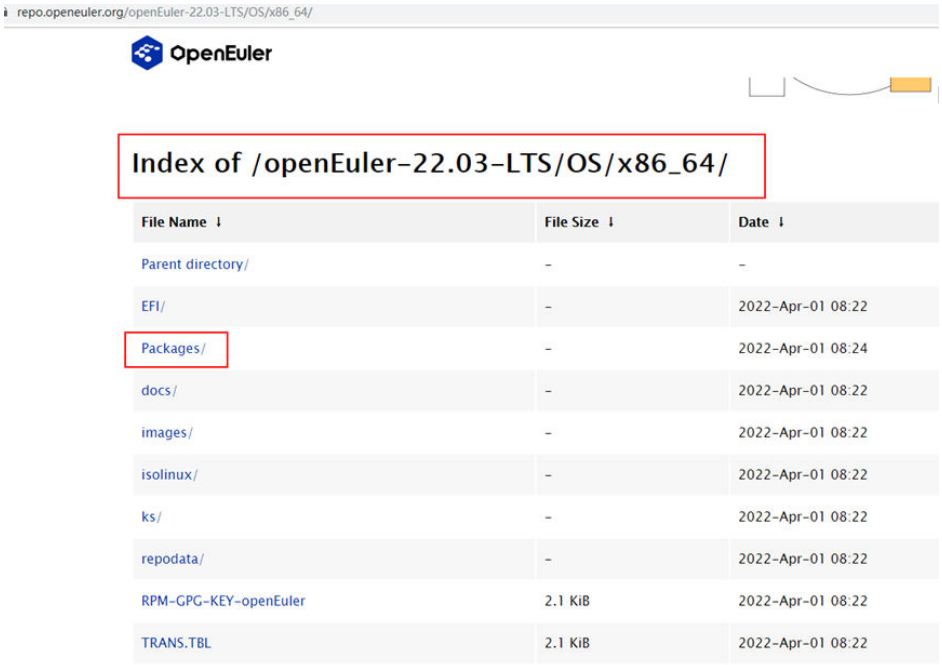
获取方法	适用场景	安装依赖RPM包	repo文件的备份及恢复
通过wget命令下载RPM包	适用于少量RPM包的安装。	需要手动下载并安装依赖的RPM包。	不涉及
通过repo文件批量下载RPM包	适用于较多RPM包的安装。	自动安装依赖的RPM包，无需再次下载。	须先将/etc/yum.repos.d目录下原有的repo文件进行备份，并删除此目录下原有的repo文件。安装RPM包后，须再次恢复这些repo文件。

通过 wget 命令下载 RPM 包

本节以下载hadoop-3.1-common-3.1.4-4.oe2203.noarch.rpm为例，介绍使用**wget**命令下载并安装RPM包。

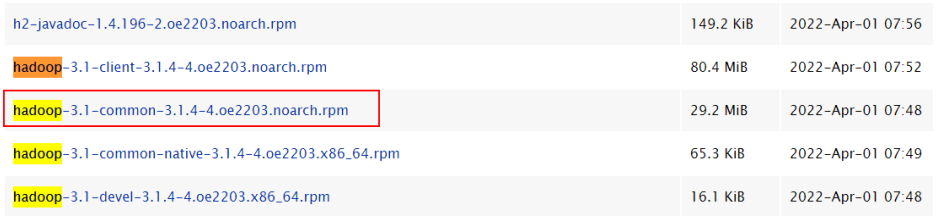
1. 单击**这里**登录openEuler社区。
2. 在“OS/”或“everything/”目录下，选择“aarch64/”或者“x86_64/”系统架构目录，并打开“Packages/”目录。

图 7-1 Packages 目录示例



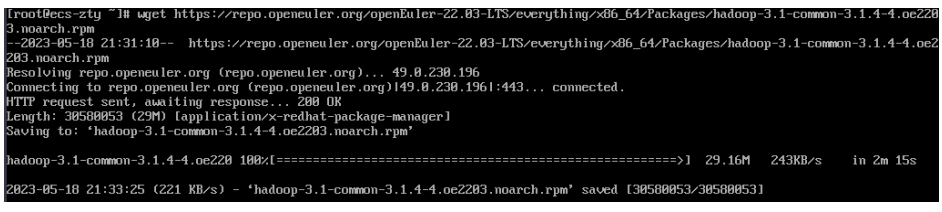
3. 查找所需要的RPM包，例如hadoop-3.1-common-3.1.4-4.oe2203.noarch.rpm。

图 7-2 hadoop 的 rpm 包示例



4. 选择此包后右击复制下载链接，执行**wget**命令下载RPM包。

图 7-3 wget 命令下载示例



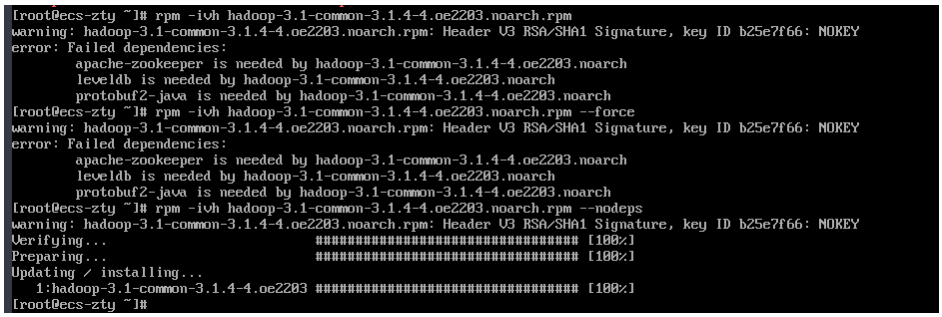
5. 检查是否下载成功。如图7-4所示表示下载成功。

图 7-4 成功下载示例

6. 使用**rpm -ivh hadoop-3.1-common-3.1.4-4.oe2203.noarch.rpm**命令安装RPM包，如图7-5所示表示安装成功。

如果安装过程中提示需要依赖其他的安装包，请根据同样的操作步骤先安装所依赖的安装包。

图 7-5 rpm 包安装示例



通过 repo 文件批量下载 RPM 包

本节以openEuler-22.03-LTS的x86_64架构为例，介绍如何下载openEuler-22.03-LTS的x86_64架构的RPM包并使用yum命令安装。

1. 首先确保虚拟机能访问**https://repo.openeuler.org/openEuler-22.03-LTS/**网址。
2. 配置yum源。

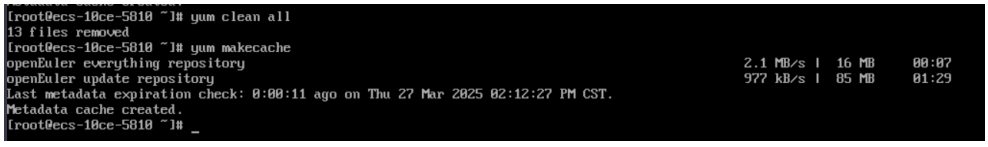
   进入/etc/yum.repos.d目录，新建一个openEuler.repo文件，并将以下内容复制到该文件里面。

   说明

   - 由于**repo.openeuler.org/openEuler-22.03-LTS/update/x86_64/**的openEuler.repo文件和HCE系统的repo文件有冲突，请先将/etc/yum.repos.d目录下HCE原有的repo文件进行备份，并删除HCE原有的repo文件，再创建openEuler.repo文件。

   ```
   [openEuler-everything]
   name=openEuler everything repository
   baseurl=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64
   gpgcheck=1
   enabled=1
   priority=3
   gpgkey=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64/RPM-GPG-KEY-openEuler
   [openEuler-update]
   name=openEuler update repository
   baseurl=https://repo.openeuler.org/openEuler-22.03-LTS/update/x86_64/
   gpgcheck=1
   enabled=1
   priority=3
   gpgkey=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64/RPM-GPG-KEY-openEuler
   ```
3. 执行**yum clean all**清除原来yum源的缓存信息。
4. 执行**yum makecache**连接新配置的源 ，如图7-6所示表示repo源连接成功。

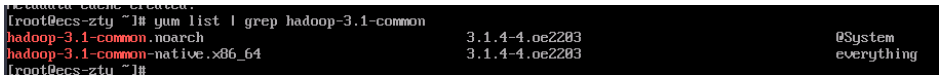
图 7-6 yum 命令执行示例



5. 安装RPM包，以hadoop-3.1-common包为例。

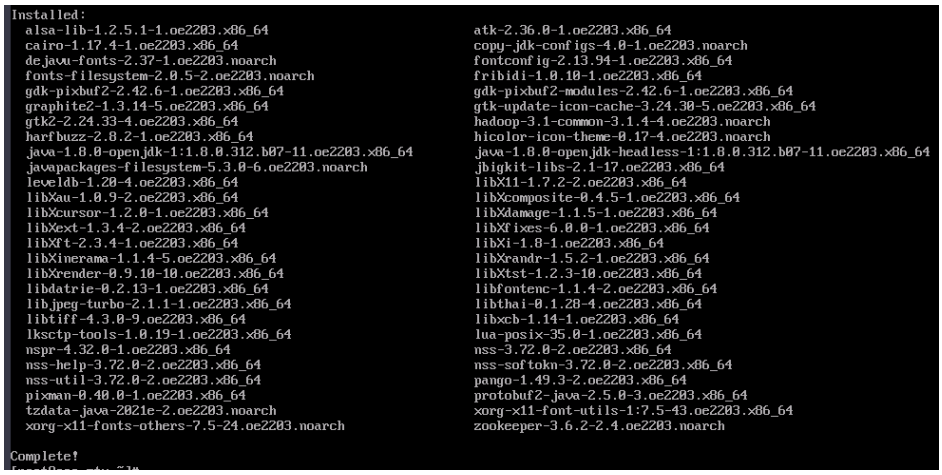
a. 执行**yum list | grep hadoop-3.1-common**命令查看是否存在该包。

图 7-7 查询 hadoop-3.1-common 包是否存在示例



b. 执行**yum -y install hadoop-3.1-common**命令来安装此包，如图7-8所示表示该包已经安装成功。

图 7-8 yum 安装 hadoop-3.1-common 包成功示例



6. 恢复repo文件。

   安装所需的openEuler包后，删除openEuler.repo文件，并将步骤2中删除的repo文件通过备份恢复。

8 制作 Docker 镜像并启动容器

本节以HCE 2.0为例介绍在HCE上制作HCE的Docker镜像并启动容器。

约束限制

- 运行容器镜像的HCE系统版本和制作的HCE容器镜像版本须保持一致。

制作镜像归档文件

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

   ```
   [base]
   name=HCE $releasever base
   baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
   enabled=1
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

   [updates]
   name=HCE $releasever updates
   baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
   enabled=1
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

   [debuginfo]
   name=HCE $releasever debuginfo
   baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
   enabled=0
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
   ```

2. 新建一个全新的目录作为Docker镜像的根系统文件目录，以/tmp/docker_rootfs目录为例，并将软件包安装到该目录。

   ```
   mkdir -p /tmp/docker_rootfs
   yum --setopt=install_weak_deps=False --installroot /tmp/docker_rootfs --releasever 2.0 install bash yum coreutils security-tool procps-ng vim-minimal tar findutils filesystem hce-repos hce-rootfiles cronie -y
   ```


> **⚠ 注意**
>
> 上述yum命令中，默认会安装当前 HCE 系统版本所需的软件包，如需安装指定HCE版本的软件包，可通过--releasever参数显示指定目标版本。例如，上述命令用于安装HCE 2.0所需的软件包。

3. chroot进入临时目录。

```
chroot /tmp/docker_rootfs
```

4. 配置临时目录。

   a. 使用security-tool.sh关闭不必要服务。

```
export EULEROS_SECURITY=0
echo "export TMOUT=300" >> /etc/bashrc
/usr/sbin/security-tool.sh -d / -c /etc/hce_security/hwsecurity/hce_security_install.conf -u /etc/hce_security/usr-security.conf -l /var/log/hce-security.log -s
```

   执行过程中，有如图8-1错误打印均为正常现象，报错的原因为：

   - 缺少服务文件。在chroot文件系统下没有启动服务导致。
   - 缺少引导系统的文件/etc/sysconfig/init。工具在系统启动阶段关闭服务，镜像rootfs不涉及系统启动。
   - 缺少/proc/sys/kernel/sysrq，这是系统启动后生成的调用节点，在chroot文件系统下不存在。

   图 8-1 报错图示

```
[root@localhost /]# /usr/sbin/security-tool.sh -d / -c /etc/hce_security/hwsecurity/hce_
security_install.conf -u /etc/hce_security/usr-security.conf -l /var/log/hce-security.lo
g -s
Failed to disable unit, unit avahi-daemon.service does not exist.
Failed to disable unit, unit avahi-daemon.socket does not exist.
Failed to disable unit, unit snmpd.service does not exist.
Failed to disable unit, unit squid.service does not exist.
Failed to disable unit, unit smb.service does not exist.
Failed to disable unit, unit vsftpd.service does not exist.
Failed to disable unit, unit tftp.service does not exist.
Failed to disable unit, unit nfs-server.service does not exist.
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
Failed to disable unit, unit rpcbind.service does not exist.
Failed to disable unit, unit slapd.service does not exist.
Failed to disable unit, unit dhcpd.service does not exist.
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
Failed to enable unit, unit haveged.service does not exist.
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
Failed to disable unit, unit postfix.service does not exist.
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
Failed to disable unit, unit chronyd.service does not exist.
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
System has not been booted with systemd as init system (PID 1). Can't operate.
Failed to connect to bus: Host is down
tail: cannot open '//etc/sysconfig/init' for reading: No such file or directory
package at is not installed
[security-tool.sh:839] [error] package at does not exist
cronie-1.5.7-1.r2.hce2.x86_64
sysctl: cannot stat /proc/sys/kernel/sysrq: No such file or directory
Removed /etc/systemd/system/multi-user.target.wants/hce-security.service.
```

   b. 卸载security-tool、cronie、systemd软件包及其依赖软件包。

```
cp -af /etc/pam.d /etc/pam.d.bak
rm -f /etc/yum/protected.d/sudo.conf /etc/yum/protected.d/systemd.conf
yum remove -y security-tool cronie systemd
rpm -e --nodeps logrotate crontabs
rm -rf /etc/pam.d
```

```
mv /etc/pam.d.bak /etc/pam.d
sh -c 'shopt -s globstar; for f in $(ls /**/*.rpmsave); do rm -f $f; done' &> /dev/null
[ -d /var/lib/dnf ] && rm -rf /var/lib/dnf/*
[ -d /var/lib/rpm ] && rm -rf /var/lib/rpm/__db.*
```

c. 移除/boot目录。

```
rm -rf /boot
```

d. 设置容器镜像语言为en_US。

```
cd /usr/lib/locale;rm -rf $(ls | grep -v en_US | grep -vw C.utf8 )
rm -rf /usr/share/locale/*
```

e. 移除共享文件 man、doc、info和mime。

```
rm -rf /usr/share/{man,doc,info,mime}
```

f. 移除缓存日志文件。

```
rm -rf /etc/ld.so.cache
[ -d /var/cache/ldconfig ] && rm -rf /var/cache/ldconfig/*
[ -d /var/cache/dnf ] && rm -rf /var/cache/dnf/*
[ -d /var/log ] && rm -rf /var/log/*.log
```

g. 移除java安全证书。

```
rm -rf /etc/pki/ca-trust/extracted/java/cacerts /etc/pki/java/cacerts
```

h. 移除/etc/machine-id。

```
rm -rf /etc/machine-id
```

i. 移除/etc/mtab。

```
rm -rf /etc/mtab
```

5. 退出chroot。

```
exit
```

6. 打包压缩临时目录，生成Docker镜像归档文件hce-docker.x86_64.tar.xz。

   归档路径为/tmp/docker_rootfs/hce-docker.x86_64.tar.xz

```
pushd /tmp/docker_rootfs/
tar cvf hce-docker.x86_64.tar .
xz hce-docker.x86_64.tar
popd
```

使用镜像归档文件启动容器

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

```
[base]
name=HCE $releasever base
baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

2. 安装docker软件包。

```
yum install docker -y
```

3. 使用镜像归档文件创建容器镜像。

```
mv /tmp/docker_rootfs/hce-docker.x86_64.tar.xz .
docker import hce-docker.x86_64.tar.xz
```

执行**docker images**命令可查看到容器镜像ID，例如6cfefae3a541。

图 8-2 查看容器镜像 ID

```
[root@localhost /]# mv /tmp/docker_rootfs/hce-docker.x86_64.tar.xz .
[root@localhost /]# docker import hce-docker.x86_64.tar.xz
sha256:6cfefae3a5410e3b48208f8a9e0a28fc08fa1b3a62ad39b27196a742969d5bfc
[root@localhost /]#
[root@localhost /]# docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
<none>              <none>              6cfefae3a541        6 seconds ago       181MB
```

说明

创建镜像可使用如下命令指定镜像的REPOSITORY和TAG参数。

docker import [OPTIONS] file|URL|- [REPOSITORY[:TAG]]

4. 使用指定的镜像运行容器并进入其bash环境。

运行如下命令后，如果shell视图改变，表示成功进入容器的bash环境。6cfefae3a541为上述镜像ID。

```
docker run -it 6cfefae3a541 bash
```

9 工具类

9.1 毕昇编译器

毕昇编译器（bisheng compiler）是华为提供的一款提供高性能、高可信及易扩展的编译器工具链。毕昇编译器引入了多种编译技术，支持C/C++/Fortran编译语言。

约束限制

HCE原生的clang编译器和毕昇编译器提供的clang编译器不能同时使用。如果您已经安装原生的clang编译器并需要使用它，就不能安装毕昇编译器。

在安装了毕昇编译器之后，如果需要使用原生的clang编译器，可执行**rpm -e bisheng-compiler**命令删除毕昇编译器，然后打开新终端。在新终端中，就可以使用原生的clang编译器。

安装毕昇编译器

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

   ```
   [base]
   name=HCE $releasever base
   baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
   enabled=1
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

   [updates]
   name=HCE $releasever updates
   baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
   enabled=1
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

   [debuginfo]
   name=HCE $releasever debuginfo
   baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
   enabled=0
   gpgcheck=1
   gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
   ```


2. 执行**yum -y install bisheng-compiler**命令安装工具。
3. 执行**source /usr/local/bisheng-compiler/env.sh**命令，导入环境变量。

   说明

   如果打开了新的终端，需要在新的终端重新导入环境变量才能正常使用毕昇编译器。
4. 检查工具是否安装成功。

   执行**clang -v**查看工具的版本号。若返回结果包含毕昇编译器版本信息，表示工具安装成功。

使用毕昇编译器

1. 编译运行C/C++程序，以C程序为例
   a. 添加以下内容到hello.c文件中。
   ```
   #include <stdio.h>

   int main() {
       printf("Hello, World! This is a C program.\n");
       return 0;
   }
   ```
   b. 编译hello.c。
   ```
   clang hello.c -o hello
   ```
   c. 执行hello.o。
   ```
   ./hello
   ```
2. 编译运行Fortran程序
   a. 添加以下内容到hello.f90文件中。
   ```
   program hello
       print *, "Hello, World! This is a Fortran program."
   end program hello
   ```
   b. 编译hello.f90。
   ```
   flang hello.f90 -o hello.o
   ```
   c. 执行hello.o。
   ```
   ./hello.o
   ```
3. 指定链接器。

   毕昇编译器指定的链接器是LLVM的lld，若不指定它则使用默认的ld。
   ```
   clang -fuse-ld=lld hello.c -o hello.o
   ./hello.o
   ```

升级回退

1. 升级
   ```
   yum upgrade bisheng-compiler
   ```
2. 回退
   ```
   yum downgrade bisheng-compiler
   ```
   由于毕昇编译器跨版本升级，可能发生由于文件冲突导致回退失败的情形。可以手动删除冲突文件，再尝试执行上述回退命令。

9.2 应用加速工具

9.2.1 概述

应用加速工具是华为云提供的一款对应用进行性能优化的工具。

应用加速工具优化应用程序有两种方式。

- 静态加速：

  静态加速只需要在应用程序运行时采集所在CPU上的pmu监控信息，基于采集到的监控信息将应用程序做静态重新制作，生成新的高性能应用程序二进制。该过程不需要对应用程序代码做修改或者仅对编译器参数做调整。静态加速有两种优化方式。

  - **使用原生的BOLT工具优化应用程序**：只能使用固定参数组合优化应用。
  - **使用hce-wae-auto命令优化应用程序**：可以根据自定义参数范围，生成不同的参数组合分别来优化应用。
- 动态加速：

  动态加速工具直接对目标应用进程进行加速，无需中断业务，在业务无感知的情况下完成优化工作。

表 9-1 静态加速和动态加速优缺点

应用加速方式	优点	缺点
静态加速	以二进制可执行文件为粒度进行优化，无需修改程序代码。	优化后需要重启应用程序。
动态加速	以进程为粒度进行优化，无需重启应用程序，并能够生成应用快照保存优化结果。同时保证二进制文件溯源能力，能够不断迭代优化应用进程，直至达到性能优化瓶颈。	当前仅支持插桩方式采集数据且仅能够进行一次优化。

除此之外，针对鲲鹏920 V200芯片，还提供CPU硬件特性设置工具。用户可以根据自身业务场景，合理配置芯片特性，软硬件结合进行优化。

约束限制

- 仅HCE x86架构支持使用静态加速和动态加速。
- 仅HCE arm架构支持在鲲鹏920 V200上使用CPU硬件特性设置工具。
- 仅root用户支持使用应用加速工具。

操作流程

1. 步骤一：**安装应用加速工具**。
2. 步骤二：采用**静态加速**或者**动态加速**方式优化应用。

9.2.2 安装工具

1. 确认repo源配置正常。

   请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

   ```
   [base]
   name=HCE $releasever base
   baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
   ```



```
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

2. 执行**yum -y install hce-wae**命令安装加速工具。
3. 检查工具是否安装成功。

   执行以下命令如果有如图9-1输出信息，表示工具安装成功。

   ```
   llvm-bolt --help | more
   ```

图 9-1 输出信息示例

```
[root@localhost sdb]# llvm-bolt --help | more
OVERVIEW: BOLT - Binary Optimization and Layout Tool

USAGE: llvm-bolt [options] <executable> <executable>

OPTIONS:

BOLT generic options:

  --bolt-id=<string>                          - add any string to tag this execution in the output binary via bolt info section
  --data=<string>                             - <data file>
  --data2=<string>                            - <data file>
  --deterministic-debuginfo                   - disables parallel execution of tasks that may producenondeterministic debug info
  --dwarf-output-path=<string>                - Path to where .dwo files or dwp file will be written out to.
  --dyno-stats                                - print execution info based on profile
  --enable-bat                                - write BOLT Address Translation tables
  --hot-data                                  - hot data symbols support (relocation mode)
  --hot-functions-at-end                      - if reorder-functions is used, order functions putting hottest last
```

说明

不同版本的输出可能有所差异，具体以实际输出为准。

9.2.3 静态加速

准备工作

1. 执行如下命令检查待优化的二进制文件中是否可以重新定位。可以重新定位表示可以进行应用优化。

   其中“application”可替换为待检查的二进制文件。

   ```
   readelf -a application | grep .rela.text
   ```

图 9-2 readelf 命令执行示例

```
[root@hce2 shard]# readelf -a gemini-redis-server | grep .rela.text
  [15] .rela.text         RELA            0000000000000000  0750a490
Relocation section '.rela.text' at offset 0x750a490 contains 241690 entries:
[root@hce2 shard]#
```

   - 如果二进制文件中.rela.text段存在，表示可以重新定位。
   - 如果不存在，为了允许BOLT在程序中重新排列函数，需要将选项--emit-relocs或-q添加到构建二进制文件的命令中。

2. 采集应用运行时的日志数据。

   部署并预热应用后，即可使用**llvm-bolt -instrument -o -instrumentation-file**命令配置应用的日志采集方式。

   例如，配置test.so文件运行后每隔30秒收集一次日志，日志保存到运行时test.log文件中请使用如下命令。

   ```
   llvm-bolt tests.so -instrument -o testd.so -instrumentation-file=test.log -instrumentation-sleep-time=30 -instrumentation-no-counters-clear
   ```

   - instrument：配置完日志采集方式后生成的新的动态库文件。本例中新生成的动态库为testd.so。
   - instrumentation-file：日志保存的文件名称。本例为test.log。
   - instrumentation-sleep-time：采集日志的时间间隔，单位为秒。本例中每隔30秒采集一次日志。
   - instrumentation-no-counters-clear：表示每次日志采集后不要清除日志计数器信息，保持日志信息上下文连续

3. 运行testd.so对应的应用程序，应用程序运行日志会自动保存在test.log文件中。

   应用加速工具会根据test.log文件中的动态数据来优化应用。

优化应用程序

- 使用原生的BOLT工具优化应用程序。

  准备好test.log文件后，就可以使用它与BOLT对应用程序进行优化。举例如下。

  ```
  llvm-bolt <executable> -o <executable>.bolt -data=test.log -reorder-blocks=ext-tsp -reorder-functions=hfsort -split-functions -split-all-cold -split-eh -dyno-stats
  ```

  说明

  以上为优化参数示例，并非最优参数组合。

- 使用**hce-wae-auto**命令优化应用程序。

  hce-wae-auto命令根据自定义的**配置文件**来优化应用程序。

  命令格式为：hce-wae-auto [-h] [-c <Path>] [-s <Keyword>] [-e <Pattern>] [-l] [--free] application

  说明

  - 执行**hce-wae-auto**命令后，产生的参数集信息路径默认与配置文件中二进制输出路径一致，路径为：/data/hce-wae-auto/hce-wae-auto.data。
  - 执行**hce-wae-auto**命令后，系统自动产生日志文件，默认路径为：/var/log/hce-wae-auto/hce-wae-auto.log。

  表 9-2 参数说明

参数名	是否需要赋值	取值类型	可选/必选	参数含义
-h	否	/	可选	显示命令参数帮助信息。
-c	是	string	可选	定义配置文件路径，默认路径为/etc/hce-wae-auto.conf。

参数名	是否需要赋值	取值类型	可选/必选	参数含义
-s	是	string	可选	定义筛选的关键词，用于模糊搜索上次参数集内容。 如果没有保存的参数集，则不生效。
-e	是	string	可选	根据上次参数集内容，执行选定的参数组合。需指定序号。 ● 单个数字指定对应序号的参数组合。 ● “:”表示序号范围。例如： – “4:6”表示序号4到序号6； – “:7”表示序号1到序号7； – “5:”表示序号5到最后一个； – “:”表示全部。 ● 单个数字可与序号范围“:”一起使用，用“,”分隔。例如： – 1,4:6表示序号1、序号4、序号5和序号6。
-l	否	/	可选	根据配置文件生成参数集并打印，不自动执行命令，不保存参数集信息。
--free	否	/	可选	不校验配置文件中参数合法性。 应用加速工具会对配置文件中的参数进行简单校验，判断参数是否属于bolt的优化参数。若指定--free指令，则取消该校验，允许输入非优化参数。
application	否	/	必选	要执行的二进制文件，可以添加相对路径或绝对路径。例如： ● mysqld ● /usr/bin/mysqld ● ../bin/mysqld

说明

-c、-s、-e、-l参数存在冲突时，优先级为-s > -l > -e > -c。

hce-wae-auto命令使用示例如下。

表 9-3 hce-wae-auto 命令使用示例表

示例	说明
hce-wae-auto mysqld	从默认路径读取配置，根据配置参数生成参数集，并自动执行命令。命令执行完成后自动保存参数集信息。

示例	说明
hce-wae-auto mysqld -c /etc/my.conf -l	从指定路径/etc/my.conf读取配置参数，生成参数集并打印生成命令。不执行命令并且不保存参数集信息。
hce-wae-auto mysqld -c /etc/my.conf	从指定路径/etc/my.conf读取配置参数，生成参数集并自动执行命令。执行完成后自动保存参数集信息。
hce-wae-auto mysqld -s align	从上次生成的参数集信息中模糊搜索关键词align，打印匹配的参数组合。
hce-wae-auto mysqld -e 4:6	从上次生成的参数集信息中，选择序号4、5、6的参数组合，生成二进制文件。
hce-wae-auto mysqld -e 1,4:6	从上次生成的参数集信息中，选择序号1、4、5、6的参数组合，生成二进制文件。
hce-wae-auto mysqld -e :	从上次生成的参数集信息中，选择所有参数组合，生成二进制文件。
hce-wae-auto mysqld -e 4:6 -l	从上次生成的参数集信息中，选择序号4、5、6的参数组合，打印生成命令，并且不执行命令。
hce-wae-auto mysqld -e : -l	从上次生成的参数集信息中，选择所有参数组合，打印生成的命令，并且不执行命令。
hce-wae-auto mysqld --free	从默认路径读取配置，不对参数进行校验，根据配置参数生成参数集，并自动执行命令。命令执行完成后自动保存参数集信息。
hce-wae-auto -h	显示指令帮助信息。

9.2.4 动态加速（只支持 HCE2.0）

准备工作

在做动态加速之前，请先做如下两个检查。两个条件都满足，才能对应用进行动态加速。

1. 执行如下命令检查待优化的二进制文件中是否可以重新定位。可以重新定位表示可以进行应用优化。

   其中“application”可替换为待检查的二进制文件。

   ```
   readelf -a application | grep .rela.text
   ```

   图 9-3 readelf 命令执行示例

   ```
   [root@hce2 shard]# readelf -a gemini-redis-server | grep .rela.text
     [15] .rela.text         RELA            0000000000000000  0750a490
   Relocation section '.rela.text' at offset 0x750a490 contains 241690 entries:
   [root@hce2 shard]#
   ```

   - 如果二进制文件中.rela.text段存在，表示可以重新定位。
   - 如果不存在，为了允许BOLT在程序中重新排列函数，需要将选项--emit-relocs或-q添加到构建二进制文件的命令中。

2. 执行**hce-wae --check /data/apps/mysql-8.0.28/bin/mysqld**命令查看应用是否支持动态加速。

   如果检查结果为3，表示可以使用动态加速工具。否则不支持动态加速。

   图 9-4 检查结果为 3 示例

```
[root@localhost ~]# hce-wae --check /data/apps/mysql-8.0.28/bin/mysqld
2023-09-14 20:41:21,968-INFO: Log level: INFO
2023-09-14 20:41:38,425-INFO: check /data/apps/mysql-8.0.28/bin/mysqld result: 3
```

操作步骤

本例以优化/data/apps/mysql-8.0.28/bin目录下的mysqld应用 ，为您介绍动态加速方式优化应用的操作。

1. 生成插桩版应用并运行。

   a. 执行命令**/data/hce-wae/dbo/gen_instrumentation /data/apps/mysql-8.0.28/bin/mysqld**生成插桩版应用。

      命令格式：**/data/hce-wae/dbo/gen_instrumentation** *应用路径*

      图 9-5 /data/hce-wae/dbo/gen_instrumentation 命令执行示例

```
[root@localhost ~]# /data/hce-wae/dbo/gen_instrumentation /data/apps/mysql-8.0.28/bin/mysqld
BOLT-INFO: Target architecture: x86_64
BOLT-INFO: BOLT version: b93bff771fd4
BOLT-INFO: first alloc address is 0x400000
BOLT-INFO: creating new program header table at address 0x4200000, offset 0x3e00000
BOLT-WARNING: debug info will be stripped from the binary. Use -update-debug-sections to keep it.
BOLT-INFO: enabling relocation mode
BOLT-INFO: forcing -jump-tables=move for instrumentation
BOLT-INFO: enabling -align-macro-fusion=all since no profile was specified
BOLT-INFO: enabling lite mode
BOLT-WARNING: split function detected on input : _ZL28delete_dictionary_tablespacev.cold/1. The support is limited in re
location mode.
BOLT-INFO: 0 out of 69377 functions in the binary (0.0%) have non-empty execution profile
BOLT-INFO: the input contains 6637 (dynamic count : 0) opportunities for macro-fusion optimization that are going to be
fixed
BOLT-INSTRUMENTER: Number of indirect call site descriptors: 37513
BOLT-INSTRUMENTER: Number of indirect call target descriptors: 87025
BOLT-INSTRUMENTER: Number of function descriptors: 68951
BOLT-INSTRUMENTER: Number of branch counters: 686408
BOLT-INSTRUMENTER: Number of ST leaf node counters: 481684
BOLT-INSTRUMENTER: Number of direct call counters: 248150
BOLT-INSTRUMENTER: Total number of counters: 1416242
BOLT-INSTRUMENTER: Total size of counters: 11329936 bytes (static alloc memory)
BOLT-INSTRUMENTER: Total size of string table emitted: 7829557 bytes in file
BOLT-INSTRUMENTER: Total size of descriptors: 70240472 bytes in file
BOLT-INSTRUMENTER: Profile will be saved to file /data/hce-wae/dbo/tmp/perf.fdata
BOLT-INFO: 425568 instructions were shortened
BOLT-INFO: removed 11295 empty blocks
BOLT-INFO: UCE removed 143092 blocks and 8769293 bytes of code.
BOLT-INFO: SCTC: patched 0 tail calls (0 forward) tail calls (0 backward) from a total of 0 while removing 0 double jump
s and removing 0 basic blocks totalling 0 bytes of code. CTCs total execution count is 0 and the number of times CTCs ar
e taken is 0.
BOLT-INFO: output linked against instrumentation runtime library, lib entry point is 0xd0616e0
BOLT-INFO: clear procedure is 0xd05fe10
BOLT-INFO: setting _end to 0x3c83b18
BOLT-INFO: setting _end to 0x3c83b18
BOLT-INFO: patched build-id (flipped last bit)
[root@localhost ~]#
```

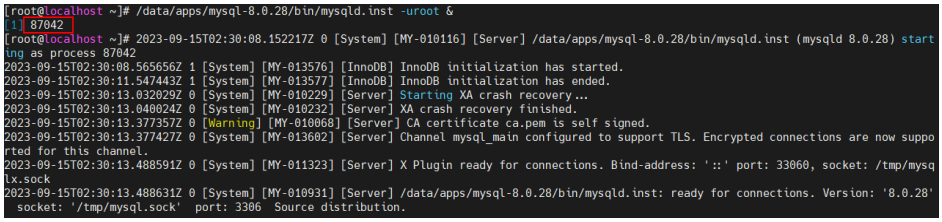
      命令运行完成后，会在当前目录生成对应的以.inst为后缀的插桩文件mysqld.inst。

      图 9-6 查询以.inst 为后缀的插桩文件示例

```
[root@localhost ~]# ls -al *.inst
-rwxrwxrwx. 1 root root 304995968 Sep 15 10:24 mysqld.inst
[root@localhost ~]#
```

   b. 运行插桩文件获取进程PID，本例为87042。

图 9-7 运行插桩文件获取进程 PID 示例



2. 创建mysqld的应用加速动态配置文件。

   每一个待优化的应用都要有一个对应的配置文件，应用加速工具根据此配置文件对应用进行动态加速。

   a. 执行如下命令复制一份默认的配置文件为**/data/hce-wae/config/mysqld.conf**。

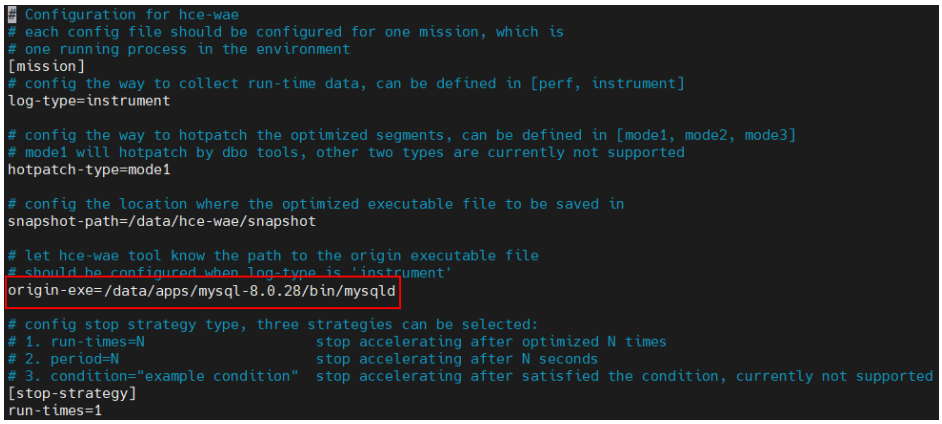
   [root@localhost]# **cp /data/hce-wae/config/hce-wae-tmp.conf /data/hce-wae/config/mysqld.conf**

   b. 设置**/data/hce-wae/config/mysqld.conf**配置文件中的origin-exe字段。

   origin-exe为待优化应用的位置，本例为**/data/apps/mysql-8.0.28/bin/mysqld**

   [root@localhost]# **vim /data/hce-wae/config/mysqld.conf**

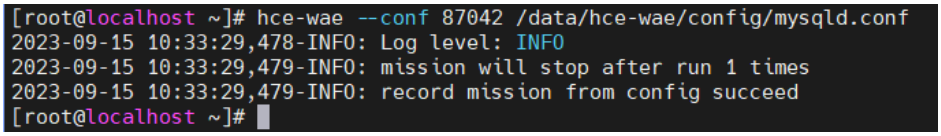
图 9-8 origin-exe 字段示例



3. 使用配置文件和对应的进程PID配置动态加速工具。

   命令格式：hce-wae --conf [PID] [/path/to/config]

图 9-9 配置动态加速工具示例



4. 启动动态加速，对插桩版mysql进行优化。

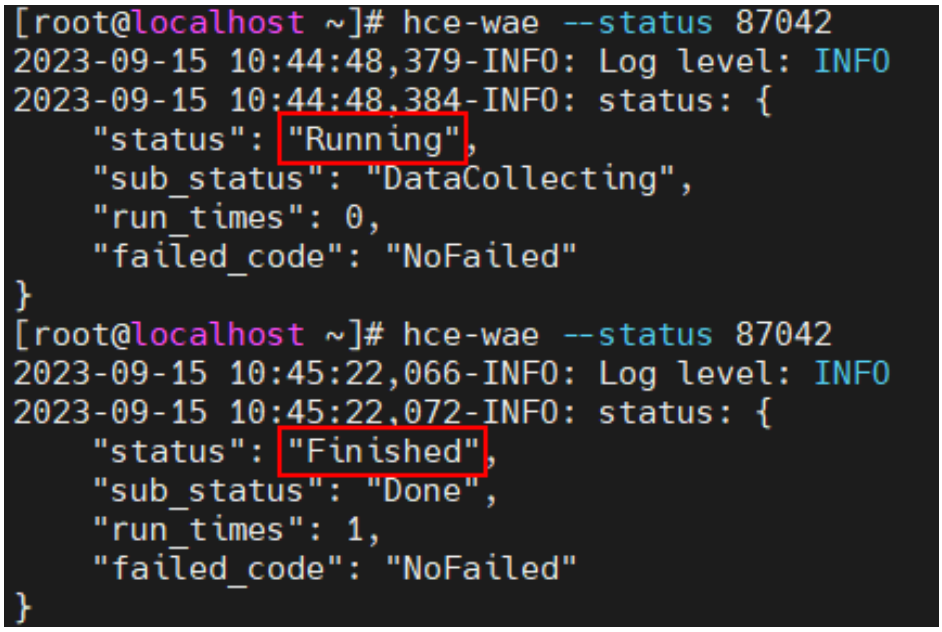
   命令格式：hce-wae --start [PID]

图 9-10 启动动态加速示例

启动后，可以通过--status参数查看当前优化状态。当状态为Running时，表示进程正在优化中；Finished时，表示进程已经优化完成。

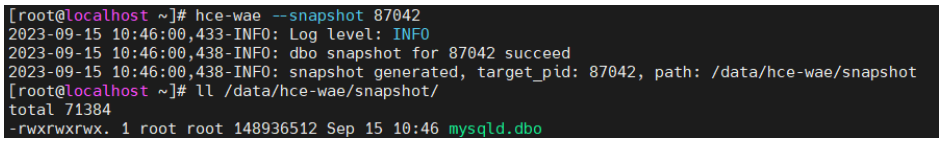
命令格式：hce-wae --status [PID]

图 9-11 查看当前优化状态示例



5. 优化后，通过--snapshot参数生成优化后的.dbo二进制快照文件，本例为mysqld.dbo。

图 9-12 生成优化后的.dbo 二进制快照文件示例



快照生成的默认路径为/data/hce-wae/snapshot/，可在配置文件中对快照位置进行修改。后续您可以直接使用此优化后的快照文件mysqld.dbo运行应用，无需重复优化。

6. 终止动态加速工具，应用优化结束。

   命令格式：hce-wae --stop [PID]

图 9-13 终止动态加速工具示例

```
[root@localhost ~]# hce-wae --stop 87042
2023-09-15 10:46:20,867-INFO: Log level: INFO
stop dbo server successfully!
2023-09-15 10:46:20,871-INFO: dbo stop for 87042 succeed
2023-09-15 10:46:20,871-INFO: mission stopped, target_pid: 87042
```

动态应用加速工具字符交互界面

动态应用加速工具支持字符交互界面，交互界面支持指令如图9-14、图9-15和表9-4所示。

图 9-14 动态应用加速工具启动界面

```
[root@localhost ~]# hce-wae --console

Welcome to hce-wae!

Version     : 1.0.0
Release     : 0.0.16.hce2
Architecture: x86_64


Type:  'h' or 'help' for help with commands
       'quit' to quit

hce-wae>
```

图 9-15 动态应用加速工具帮助界面

```
hce-wae> help

h, help                   |    Show this help message and exit

list                      |    List the process information of the current environment,
                          |  including PID, PPID, and command

check <Binary file path> |    Check if the application support static or dynamic
                          |  acceleration. 0: both not support; 1: static only; 2:
                          |  dynamic only; 3: both are supported.

show <Pid>                |    Show all non-kernel processes which contain one of the
                          |  shared-object libraries that the target process
                          |  depends on.

conf <Pid> <Config path> |    Config the target process by pid

conf <Process>            |    Set the Config of the target process

start <Pid>               |    Start dynamic acceleration for process by its pid

stop <Pid>                |    Stop dynamic acceleration for process by its pid

status <Pid>              |    Show the status of the dynamic accelerating process by
                          |    its pid

snapshot <Pid>            |    Make a snapshot for the dynamic accelerating process
                          |    by its pid

quit                      |    Exit console
hce-wae>
```

表 9-4 动态应用加速工具字符交互界面指令

字符交互界面指令	使用方式	指令描述
list	list	获取当前环境的进程信息，包含PID、PPID以及command信息。
status	status <PID>	查询当前正在进行的动态加速任务及状态。输入后在字符界面展示信息包括： • 应用进程PID • 应用进程名 • 当前已优化次数 • 当前加速状态
check	check <PID>	检测当前进程是否支持静态/动态加速，展示文字结果。
show	show <PID>	查询进程的依赖关系
conf	conf <PID>	为目标进程配置动态加速能力，输入指令后依次弹出配置选项，根据提示进行配置即可。
start	start <PID>	启动动态加速
stop	stop <PID>	终止动态加速
snapshot	snapshot <PID>	生成动态加速快照
quit	quit	退出字符交互界面
h/help	h/help	显示帮助信息

9.2.5 配置文件

本节提供了**静态加速配置文件**和**动态加速配置文件**中每个配置项的说明。

静态加速配置文件

应用加速工具默认静态加速配置信息如下，您可以自定义配置文件来优化应用。

图 9-16 应用加速工具默认静态加速配置信息示例

```
# Section `binary` defined options associated with binary file
[binary]
# Output path for generated binary files, absolute path is recommended
# default: /data/hce-wae-auto/
binary_out_path = "/data/hce-wae-auto"

# Threshold for the number of generated binary file, currently not in use
# default value: 100
binary_num_threshold = 100

# Name suffix for auto-generated binary files
# default value: `blot.auto`
binary_file_suffix = "blot.auto"

# Section `parameter` defined option associated with user defined parameter collection
[parameter]
# User defined parameter collection, parameters in this option will automatically be separated to different
# combination parameter groups, which are used as the parameter input for `llvm-blot` respectively, and
# generate different binary files.
# For those parameters which can be assigned values, use `=` connect parameter name and parameter value.
# Each line should contain one parameter and should not configure the same parameters in this option.
# example:
#   --plt=all
parameters =
    align-blocks
    frame-opt
    align-functions=1

# Section `include` defined include filter for auto-generated parameter collections
[include]
# options here should be the parameters combination that should be included for the auto-generated
# parameter group.
# Each line should contain one parameter, if the parameter is not assigned a value,
# it must end with `=`
# example:
#   frame-opt=none
align-blocks=

# Section `exclude` defined exclude filter for auto-generated parameter collections
[exclude]
# Options here should be the parameters combination that should be excluded for the auto-generated
# parameter group.
# Options has the same formate rule with the `include` section
frame-opt=none
```

表 9-5 配置信息说明

模块名	描述
binary	定义二进制文件的相关属性。binary参数详情参见表9-6。
parameter	用户定义的参数集合，工具根据此集合生成参数集。至少定义一个参数。 通过**llvm-bolt -h**命令可查看所有参数。
include	用户定义参数集中需包含的参数，允许定义多个参数，多个参数之间采用“与”逻辑。 选项以参数名为key，指定值为value。例如，frame-opt=none。 **说明** • 若key无法指定值或不需要指定值时，仍需要以“=”结束，例如frame-opt=。 • include和exclude配置包含同一个参数时，exclude的优先级大于include。
exclude	用户定义参数集中不需要包含的参数。允许定义多个参数，多个参数之间采用“与”逻辑。 **说明** • 若key无法指定值或不需要指定值时，仍需要以“=”结束，例如frame-opt=。 • include和exclude配置包含同一个参数时，exclude的优先级大于include。

表 9-6 binary 参数说明

选项名	值类型	默认值	选项描述
binary_out_path	string	"/data/hce-wae-auto"	定义自动生成的二进制文件的保存路径。
binary_file_suffix	string	"blot.auto"	定义自动生成的二进制文件名后缀。

静态加速配置文件示例

```
[binary]
binary_out_path = "/data/llvm_auto"
binary_num_threshold = 1000
binary_file_suffix = "blot.auto"

[parameter]
parameters =
    --align-blocks            # 允许添加参数前缀--
    frame-opt                 # 用户定义的参数集合中若不需要指定参数的值，则无需以=结束
    align-functions=1           # 用户定义参数集合中指定了参数对应的值，则生成的参数集中，所有参数组合中该参数的值都为1

[include]
align-blocks=                 # 参数无法指定值仍需以=结束
[exclude]
frame-opt=none                  # 指定参数及对应的值，生成的参数集中会过滤参数为frame-opt，且值为none的参数组合
indirect-call-promotion=        # 指定参数,该参数为枚举类型，则生成的参数集中会过滤所有参数为frame-opt的参数组合
```

动态加速配置文件

应用加速工具默认动态加速配置信息如下，您可以自定义配置文件来优化应用。

图 9-17 应用加速工具默认动态加速配置信息示例

```
# Configuration for hce-wae
# each config file should be configured for one mission, which is
# one running process in the environment
[mission]
# config the way to collect run-time data, can be defined in [perf, instrument]
log-type=instrument

# config the way to hotpatch the optimized segments, can be defined in [mode1, mode2, mode3]
# mode1 will hotpatch by dbo tools, other two types are currently not supported
hotpatch-type=mode1

# config the location where the optimized executable file to be saved in
snapshot-path=/data/hce-wae/snapshot

# let hce-wae tool know the path to the origin executable file
# should be configured when log-type is 'instrument'
origin-exe=/path/to/origin/executable/file

# config stop strategy type, three strategies can be selected:
# 1. run-times=N                    stop accelerating after optimized N times
# 2. period=N                       stop accelerating after N seconds
# 3. condition="example condition"  stop accelerating after satisfied the condition, currently not supported
[stop-strategy]
run-times=10
```

配置文件中模块含义：

[mission]：优化运行中的应用所要配置的参数。

[stop-strategy]：应用优化停止策略，请选择其中一种配置。

表 9-7 配置信息说明

模块名	参数名	描述
[mission]	log-type	运行时日志采集方式，当前仅支持instrument方式。
	hotpatch-type	热补丁模式，当前仅支持mode1即ptrace方式。
	snapshot-path	优化后的二进制快照文件存放的目录路径。
	origin-exe	原始应用的位置，使用instrument日志采集模式时，须指定此参数。
[stop-strategy]	run-times	指定优化次数，达到该次数时动态加速工具会停止优化，当前仅支持1次。
	period	指定优化周期，达到该时间周期时停止优化，单位为秒，取值范围为1~600。
	condition	指定优化条件，达到该条件时停止优化，当前不支持。

9.2.6 CPU 硬件特性设置

通过用户态工具 UcpuHconfig可进行CPU硬件特性设置，下表列出其配置项和使用实例等。

UcpuHconfig命令执行成功后会保存配置，os重启后自动恢复。

表 9-8 UcpuHconfig 使用说明

功能	使用说明	参数1	参数2	示例	备注
L3缓存是否溢出	观察每个L3的流量。 每个L3流量平均，建议关spill；L3流量一部分高，一部分空闲，建议开spill。	spill	[on|off]	UcpuHconfig --spill on	-
是否设置CPU缓存的write unique share状态	观察业务热点和数据流。 有连续写多cacheline的行为（例如memcpy），使能特性能优化带宽。	write_unique_share	[on|off]	UcpuHconfig --write_unique_share on	-

功能	使用说明	参数1	参数2	示例	备注
cache line替换算法	Topdown观察到后端bound 50%以上，并且有L3 bound，尝试修改替换算法，比较各个算法的实际效果。	cacheline	整数（代表策略）	UcpuHconfig --cacheline 0	0: 随机算法 1: drrip算法 2: plru 3: 随机算法
L3从ddr的预取	Topdown观察到后端bound 40%以上，并且有L3 bound，并且抓到单die DDR带宽大（20G以上）。关闭prefetch 可以节省DDR带宽。	prefetchtgt_en	[on|off]	UcpuHconfig --prefetchtgt_en on	-
缓存驱逐时是否通知ddr	Topdown观察到后端bound，再观察每个L3的流量，出现L3流量部分空闲部分繁忙，打开配置。	reg_evict_disable	[on|off]	UcpuHconfig --reg_evict_disable on	-
缓存驱逐时是否通知其他cpu	Topdown观察到后端bound 40%以上，并且有L3 bound，打开配置。	reg_evict_selfsnp_disable	[on|off]	UcpuHconfig --reg_evict_selfsnp_disable on	-
cpu从L3的预取	Topdown观察到后端bound，并且有L3 bound，如果DDR带宽大，可以尝试关预取	prefetch_l3	[on|off]	UcpuHconfig --prefetch_l3 on	-
查看	查看硬件特殊设置情况	show	[功能名| all]	UcpuHconfig --show spill UcpuHconfig --show all	-

9.3 Pod 带宽管理工具

在业务混合部署的场景下，Pod带宽管理功能根据QoS分级对资源进行合理调度，提升网络带宽利用率。HCE提供oncn-tbwm带宽管理工具，使用tbwmcli命令对收发包方向的网络限速功能，实现网络QoS。

前提条件

本功能固定使用ifb0，使用前请确定虚拟网卡ifb0未被使用，并加载ifb驱动。

约束与限制

- 仅HCE 2.0 x86架构支持使用tbwmcli命令。
- 仅允许root用户执行tbwmcli命令。
- tbwmcli命令同一时间只能在一个网卡使能Qos功能，多个网卡不支持并行使能网络QoS。
- 网卡被插拔重新恢复后，原来设置的QoS规则会丢失，需要手动重新配置网络QoS功能。
- 不支持cgroup v2。
- 升级oncn-tbwm软件包不会影响升级前的使能状态。卸载oncn-tbwm软件包会关闭对所有设备的使能。
- 仅支持识别数字、英文字母、中划线“-” 和下划线“_”四类字符类型的网卡名，其他字符类型的网卡不被识别。
- 实际使用过程中，带宽限速有可能造成协议栈内存积压，此时依赖传输层协议自行反压，对于udp等无反压机制的协议场景，可能出现丢包、ENOBUFS、限流不准等问题。
- 收包方向的网络限速依赖于TCP的反压能力，在非TCP协议的场景中，网络包已经收至目标网卡，不支持对于收包方向的网络限速。
- 不支持tbwmcli、tc命令和网卡命令混用，只能单独使用tbwmcli工具进行限速。例如，某个网卡上已经设置过tc qdisc规则的情况下，对此网卡使能网络QoS功能可能会失败。

使用方法

1. 安装oncn-tbwm软件包。
   a. 确认repo源配置正常。

      请检查默认的/etc/yum.repos.d/hce.repo配置文件中参数是否正确，正确的配置如下。

      ```
      [base]
      name=HCE $releasever base
      baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
      enabled=1
      gpgcheck=1
      gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

      [updates]
      name=HCE $releasever updates
      baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
      enabled=1
      gpgcheck=1
      ```

```
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

b. 执行**yum install oncn-tbwm**命令安装oncn-tbwm软件包。

c. 验证oncn-tbwm软件包正确性。

- 执行**tbwmcli -v**命令，正确安装则结果显示如下，具体版本以实际为准。
  ```
  version: 1.0
  ```
- 确认以下oncn-tbwm服务组件，正常情况下以下服务组件均存在。
  ```
  /usr/bin/tbwmcli
  /usr/share/tbwmcli
  /usr/share/tbwmcli/README.md
  /usr/share/tbwmcli/bwm_prio_kern.o
  /usr/share/tbwmcli/tbwm_tc.o
  ```

2. 根据需要执行tbwmcli命令。

表 9-9 tbwmcli 命令说明

命令	说明
tbwmcli -e ethx tbwmcli -d ethx egress	使能/关闭某个网卡的发包方向Qos功能。 示例：使能eth0网卡的发包方向Qos功能。 **tbwmcli -e eth0** enable eth0 egress success 示例：关闭eth0网卡的发包方向Qos功能。 **tbwmcli -d eth0 egress** disable eth0 egress success
tbwmcli -i ethx online/offline tbwmcli -d ethx ingress	使能/关闭某个网卡的收包方向Qos功能。 示例：使能eth0网卡收包方向Qos功能，并设置为在线网卡。 **tbwmcli -i eth0 online** enable eth0 ingress success, dev is online 示例：使能eth0网卡收包方向Qos功能，并设置为离线网卡。 **tbwmcli -i eth0 offline** enable eth0 ingress success, dev is offline **说明** 收包方向不支持同时设置为多个离线网卡的情况，支持一个离线网卡和多个在线网卡。 示例：关闭eth0网卡收包方向Qos功能。 **tbwmcli -d eth0 ingress** disable eth0 ingress success
tbwmcli -d ethx	强制关闭某个网卡的所有Qos功能，并关闭ifb功能。 示例：强制关闭eth0网卡的所有Qos功能，并关闭ifb功能。 **tbwmcli -d eth0** disable eth0 success

命令	说明
tbwmcli -p istats/estats	打印收/发包方向内部统计信息。 示例：打印收包方向内部统计信息。 **tbwmcli -p istats** offline_target_bandwidth: 94371840online_pkts: 3626190offline_pkts: 265807online_rate: 0offline_rate: 13580offline_prio: 0 示例：打印发包方向内部统计信息。 **tbwmcli -p estats** offline_target_bandwidth: 94371840online_pkts: 4805452offline_pkts: 373961online_rate: 0offline_rate: 19307offline_prio: 1
tbwmcli -s path <prio> tbwmcli -p path	设置/查询cgroup的QoS优先级。 当前仅支持设置离线和在线两个QoS优先级。 ● 0：设置cgroup为在线QoS优先级。 ● -1：设置cgroup为离线QoS优先级。 示例：设置test_online cgroup的优先级为0。 **tbwmcli -s /sys/fs/cgroup/test_online 0** set prio success 示例：查询test_online cgroup的优先级。 **tbwmcli -p /sys/fs/cgroup/test_online** prio is 0
tbwmcli -s bandwidth <low，high> tbwmcli -p bandwidth	设置/查询离线带宽范围。 示例：设置离线宽带范围为30mb~100mb。 **tbwmcli -s bandwidth 30mb,100mb** set bandwidth success 示例：查询离线带宽范围。 **tbwmcli -p bandwidth** bandwidth is 31457280(B),104857600(B)
tbwmcli -s waterline <val> tbwmcli -p waterlin	设置/查询在线网络带宽水线。 示例：设置在线网络带宽水线为20mb。 **tbwmcli -s waterline 20mb** set waterline success 示例：查询在线网络带宽水线。 **tbwmcli -p waterline** waterline is 20971520 (B)
tbwmcli -p devs	查看系统上所有网卡的使能状态。 **tbwmcli -p devs** lo Egress : disabled lo Ingress : disabled eth0 Egress : disabled eth0 Ingress : enabled, it's offline ifb0 Egress : enabled
tbwmcli -c	强制删除所有网卡的QoS，谨慎使用。
modprobe ifb numifbs=1	加载ifb。
rmmod ifb	卸载ifb。

9.4 硬件兼容性测试工具

概述

oec-hardware工具是HCE提供的一款硬件兼容性测试工具，oec-hardware提供服务器整机、板卡与HCE的兼容性验证测试，验证仅限于基本功能验证，不包括性能测试等其它测试。

兼容性结论继承说明

- **整机兼容性结论继承策略**

  如果验证适配的服务器共主板、CPU代次相同，可以继承兼容性结论。

- **板卡兼容性结论继承策略**

  板卡型号一般通过四元组来进行确认，四元组信息：

  - vendorID：芯片厂商ID
  - deviceID: 芯片型号ID
  - svID：板卡厂商ID
  - ssID: 板卡型号ID

  板卡兼容性结论继承有以下几点:

  - vendorID和deviceID不同。

    无法继承兼容性结论。
  - vendorID和deviceID相同，svID不同。

    芯片型号相同但是板卡厂商不同，无法继承兼容性结论。
  - vendorID、deviceID、svID相同。

    代表同一个板卡厂商，使用同一种芯片做成的不同板卡，可以继承兼容性结论。
  - vendorID、deviceID、svID、ssID相同。

    代表同一个板卡厂商，使用同一种芯片做成的同一系列板卡，四元组信息相同，可以继承兼容性结论。厂商自行评估此系列板卡，可以写代表性的板卡名称。

运行环境

- **整机测试环境要求**

  表 9-10 整机测试环境要求

项目	要求
整机数量	需要两台整机，业务网口互通
硬件	至少有一张RAID卡和一张网卡（包括集成主板硬件)
内存	建议满配

- **板卡测试环境要求**

  表 9-11 板卡测试环境要求

项目	要求
服务器型号	Taishan200(Model 2280)、2288H V5或同等类型的服务器，对于x86_64服务器，icelake/cooperlake/cascade可任选一种，优选icelake。
RAID卡	需要组raid，至少组raid0。
NIC/IB卡	服务端和测试端需要分别插入一张同类型板卡，配置同网段IP，保证直连互通。
FC卡	需要连接磁阵，至少组两个lun。

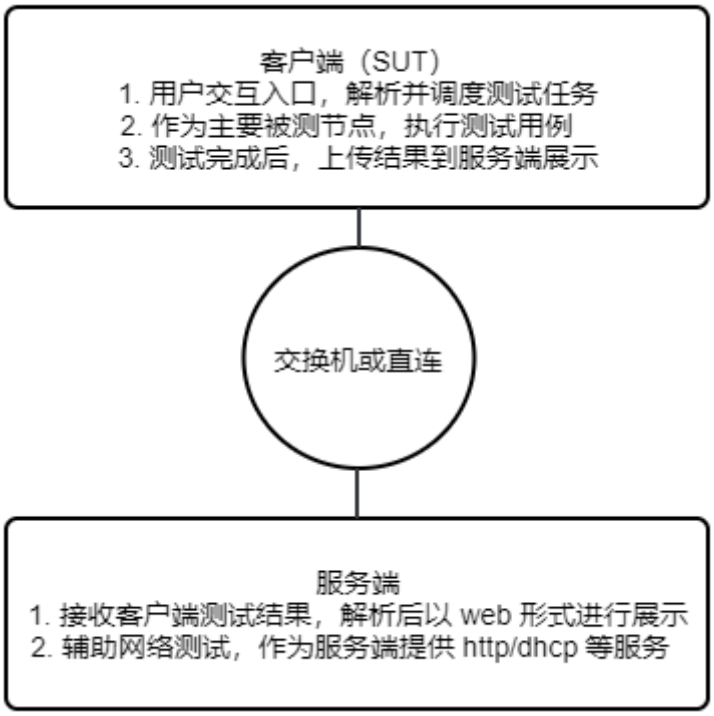
  说明

  如果要测试外部驱动，请提前安装驱动，配置测试环境。

  GPU、keycard等测试项需要提前安装外部驱动，保证环境部署完成，然后使用本工具进行测试。

- **运行环境组网**

  图 9-18 运行环境组网



工具安装

本工具支持在HCE 2.0或更高版本上运行，详细支持操作系统版本信息请查看/usr/share/oech/kernelrelease.json文件。

步骤1 获取安装包

- **在线安装**

配置HCE官方repo中对应版本的repo源，使用dnf获取软件包进行安装。

- **离线安装**

1. 本地挂载HCE镜像，配置repo源，用于依赖获取。
2. 从HCE官方repo的updates目录获取最新软件包进行安装。

步骤2 安装过程

- **客户端**

a. 使用dnf安装客户端oec-hardware。

```
dnf install oec-hardware
```

b. 输入oech命令，可正常运行，则表示安装成功。

- **服务端**

1. 使用dnf安装服务端oec-hardware-server。

```
dnf install oec-hardware-server
```

2. 启动服务。本服务通过搭配nginx服务提供web服务，默认使用80端口，可以通过nginx服务配置文件修改对外端口，启动前请保证这些端口未被占用。

```
systemctl start oech-server.service
systemctl start nginx.service
```

设置开机自启动。

```
systemctl enable oech-server.service
systemctl enable nginx.service
```

3. 关闭防火墙和SElinux。

```
systemctl stop firewalld
iptables -F
setenforce 0
```

----结束

测试项

- **测试项介绍**
  - 执行kdump和watchdog测试项时会自动重启，建议与其他测试项分开测试，单独测试这两项。
  - keycard测试项依赖开源的openssl指定版本，请在环境可以访问公网的情况下执行，或者提前将以下内容下载到/opt目录下：

    openssl.tar.gz
  - gpu测试项依赖一些开源工具，请在环境可以访问公网的情况下执行，或者提前将以下内容下载到/opt目录下：

    https://github.com/NVIDIA/cuda-samples/archive/refs/heads/master.zip

    radeontop

    glmark2

    clpeak

gpu-burn

VulkanSamples

- 涉及到两个节点交互的测试项时需要注意关闭两个节点上的防火墙，以免测试数据被过滤掉，如ethernet、dpdk等测试项：
  ```
  systemctl stop firewalld
  ```
- usb测试项当前只涉及对usb设备是否能正常识别的测试，需要在测试过程中根据提示在不同阶段手动插入或拔除usb设备。
- /usr/share/oech/lib/config/test_config.yaml 是硬件测试项配置文件模板，fc、raid、disk、ethernet、infiniband硬件测试前需先根据实际环境进行配置，指定待测的硬件，其它硬件测试不需要配置。

- **测试项策略**

表 9-12 测试项策略

测试项	整机必测项	板卡测试项
system	√	√
acpi	√	-
clock	√	-
cpufreq	√	-
cdrom	-	-
disk	√	-
dpdk	-	-
ethernet	√	√
fc	-	√
gpu	-	√
ipmi	√	-
infiniband	-	√
kabi	√	√
kdump	√	-
keycard	-	√
memory	√	-
nvme	-	√
perf	√	-
raid	√	√
usb	-	-
watchdog	√	-

使用指导

前提条件

- /usr/share/oech/kernelrelease.json文件中列出了当前支持的所有系统版本，使用uname -a命令确认当前系统内核版本是否属于框架支持的版本。
- 框架默认会扫描所有网卡，对网卡进行测试前，请自行筛选被测网卡；要求测试端口连通，状态为up。建议不要使用业务网口进行网卡测试。
- /usr/share/oech/lib/config/test_config.yaml是硬件测试项配置文件模板，fc、raid、disk、ethernet、infiniband硬件测试前需先根据实际环境进行配置，其它硬件测试不需要配置。对于网卡测试，如果是工具自动添加的IP地址，测试完成后，为了安全，服务端的IP需手动删除。

使用步骤

步骤1 在客户端启动测试框架。客户端启动oech。

```
# oech
```

步骤2 填写ID、URL、Server配置项。

ID是测试的序号，可自己定义（注意：ID中不能带特殊字符）；URL建议填写产品链接；Server 必须填写为客户端可以直接访问的服务器域名或 ip，用于展示测试报告和网络测试的服务端。服务端nginx默认端口号是80，如果服务端安装完成后没有修改该端口，Compatibility Test Server 的值只需要输入服务端的业务IP地址；否则需要带上端口号，比如：172.167.145.2:90。

```
The HCE Hardware Compatibility Test Suite
Please provide your Compatibility Test ID:
Please provide your Product URL:
Please provide the Compatibility Test Server (Hostname or Ipaddr):
```

步骤3 进入测试套选择界面。在用例选择界面，框架将自动扫描硬件并选取当前环境可供测试的测试套，输入edit可以进入测试套选择界面。

```
These tests are recommended to complete the compatibility test:
No. Run-Now? status    Class        Device        driverName     driverVersion     chipModel      boardModel
1     yes     NotRun    acpi
2     yes     NotRun    clock
3     yes     NotRun    cpufreq
4     yes     NotRun    disk
5     yes     NotRun    ethernet     enp3s0        hinic          2.3.2.17          Hi1822             SP580
6     yes     NotRun    ethernet     enp4s0        hinic          2.3.2.17          Hi1822             SP580
7     yes     NotRun    ethernet     enp125s0f0     hns3                        HNS GE/10GE/25GE     TM210/
TM280
8     yes     NotRun    ethernet     enp125s0f1     hns3                        HNS GE/10GE/25GE     TM210/
TM280
9     yes     NotRun    raid         0000:04:00.0   megaraid_sas   07.714.04.00-rc1   SAS3408            SR150-M
10    yes     NotRun    gpu          0000:03:00.0   amdgpu                       Navi               Radeon PRO
W6800
11    yes     NotRun    ipmi
12    yes     NotRun    kabi
13    yes     NotRun    kdump
14    yes     NotRun    memory
15    yes     NotRun    perf
16    yes     NotRun    system
17    yes     NotRun    usb
18    yes     NotRun    watchdog
Ready to begin testing? (run|edit|quit)
```

步骤4 选择测试套。all|none分别用于 全选|全取消（必测项system不可取消）；数字编号可选择测试套，每次只能选择一个数字，按回车符之后no变为yes，表示已选择该测试套。

```
Select tests to run:
No. Run-Now? status    Class        Device        driverName     driverVersion     chipModel
```



```
boardModel
1    no    NotRun    acpi
2    no    NotRun    clock
3    no    NotRun    cpufreq
4    no    NotRun    disk
5    yes   NotRun    ethernet    enp3s0        hinic         2.3.2.17        Hi1822            SP580
6    no    NotRun    ethernet    enp4s0        hinic         2.3.2.17        Hi1822            SP580
7    no    NotRun    ethernet    enp125s0f0    hns3                          HNS GE/10GE/25GE  TM210/
TM280
8    no    NotRun    ethernet    enp125s0f1    hns3                          HNS GE/10GE/25GE  TM210/
TM280
9    yes   NotRun    raid        0000:04:00.0  megaraid_sas  07.714.04.00-rc1  SAS3408           SR150-M
10   yes   NotRun    gpu         0000:03:00.0  amdgpu                        Navi              Radeon PRO
W6800
11   yes   NotRun    ipmi
12   yes   NotRun    kabi
13   yes   NotRun    kdump
14   yes   NotRun    memory
15   yes   NotRun    perf
16   yes   NotRun    system
17   yes   NotRun    usb
18   yes   NotRun    watchdog
Selection (<number>|all|none|quit|run):
```

步骤5 开始测试。选择完成后输入run开始测试。

步骤6 上传测试结果。测试完成后可以上传测试结果到服务器，便于结果展示和日志分析。如果上传失败，请检查网络配置，然后重新上传测试结果。

```
...
------------  Summary  ------------
ethernet-enp3s0                PASS
system                         PASS
Log saved to /usr/share/oech/logs/oech-20240928210118-TnvUJxFb50.tar succ.
Do you want to submit last result? (y|n) y
Uploading...
Successfully uploaded result to server X.X.X.X.
```

----结束

获取结果

- **测试日志查看**

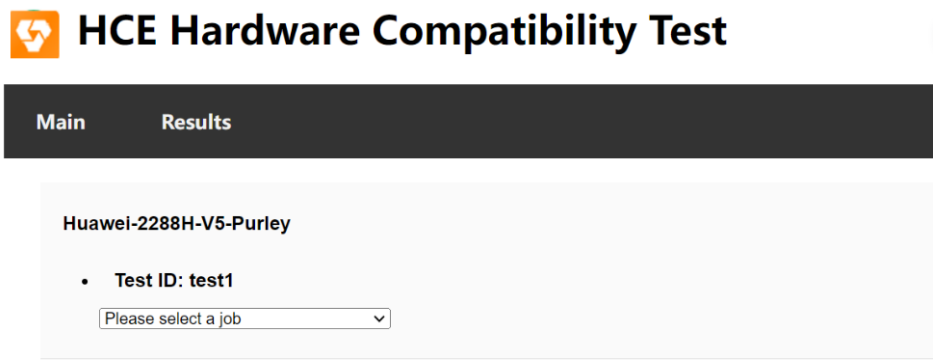
  测试完成后，测试日志会保存在/usr/share/oech/logs/目录下，将测试日志导出并解压即可查看测试日志。

- **浏览器查看测试报告**

  浏览器查看测试报告，前提需要配置server端接收测试结果。

  a. 浏览器打开服务端IP地址，单击导航栏Results界面，找到对应的测试id进入。

图 9-19 浏览器查看测试报告



b. 进入单个任务页可以看到具体的测试结果展示，包括环境信息和执行结果等。

- Summary查看所有测试结果。
- Devices查看所有硬件设备信息。
- Runtime查看测试运行时间和总任务执行日志。
- Attachment下载测试日志附件。

9.5 A-Tune 工具

9.5.1 认识 A-Tune

操作系统作为衔接应用和硬件的基础软件，如何调整系统和应用配置，充分发挥软硬件能力，从而使业务性能达到优化，对用户至关重要。然而，运行在操作系统上的业务类型成百上千，应用形态千差万别，对资源的要求各不相同。当前硬件和基础软件组成的应用环境涉及高达7000多个配置对象，随着业务复杂度和调优对象的增加，调优所需的时间成本呈指数级增长，导致调优效率急剧下降，调优成为了一项极其复杂的工程，给用户带来巨大挑战。

其次，操作系统作为基础设施软件，提供了大量的软硬件管理能力，每种能力适用场景不尽相同，并非对所有的应用场景都通用有益，因此，不同的场景需要开启或关闭不同的能力，组合使用系统提供的各种能力，才能发挥应用程序的最佳性能。

另外，实际业务场景成千上万，计算、网络、存储等硬件配置也层出不穷，实验室无法遍历穷举所有的应用和业务场景，以及不同的硬件组合。

为了应对上述挑战，HCE集成openEuler推出的A-Tune工具。

A-Tune是一款基于AI开发的系统性能优化引擎，它利用人工智能技术，对业务场景建立精准的系统画像，感知并推理出业务特征，进而做出智能决策，匹配并推荐最佳的系统参数配置组合，使业务处于最佳运行状态。

9.5.2 安装与部署

安装 A-Tune

本节介绍A-Tune的安装模式和安装方法。

安装模式介绍

A-Tune支持单机模式、分布式模式和集群模式安装：

- 单机模式
  client和server安装到同一台机器上。
- 分布式模式
  client和server分别安装在不同的机器上。
- 集群模式
  由一台client机器和大于一台server机器组成。

安装操作

安装A-Tune的操作步骤如下：

1. 安装A-Tune服务端。
   ```
   # yum install atune -y
   # yum install atune-engine -y
   ```
2. 若为分布式部署，请安装A-Tune客户端。
   ```
   # yum install atune-client -y
   ```
3. 验证是否安装成功。命令和回显如下表示安装成功。
   ```
   # rpm -qa | grep atune
   atune-client-xxx
   atune-db-xxx
   atune-xxx
   atune-engine-xxx
   ```

部署 A-Tune

本节介绍A-Tune的配置部署。

配置介绍

A-Tune配置文件/etc/atuned/atuned.cnf的配置项说明如下：

- A-Tune服务启动配置（可根据需要进行修改）。
  - protocol：系统gRPC服务使用的协议，取值为unix或tcp，unix为本地socket通信方式，tcp为socket监听端口方式。默认为unix。
  - address：系统gRPC服务的侦听地址，默认为unix socket，若为分布式部署，需修改为侦听的ip地址。
  - port：系统gRPC服务的侦听端口，范围为0~65535未使用的端口。如果protocol配置是unix，则不需要配置。
  - connect：若为集群部署时，A-Tune所在节点的ip列表，ip地址以英文逗号分隔。
  - rest_host：系统rest service的侦听地址，默认为localhost。
  - rest_port：系统rest service的侦听端口，范围为0~65535未使用的端口，默认为8383。
  - engine_host：与系统atune engine service链接的地址。
  - engine_port：与系统atune engine service链接的端口。
  - sample_num：系统执行analysis流程时采集样本的数量，默认为20。
  - interval：系统执行analysis流程时采集样本的间隔时间，默认为5s。

  - grpc_tls：系统gRPC的SSL/TLS证书校验开关，默认不开启。开启grpc_tls后，atune-adm命令在使用前需要设置以下环境变量方可与服务端进行通讯：
    - export ATUNE_TLS=yes
    - export ATUNED_CACERT=<客户端CA证书路径>
    - export ATUNED_CLIENTCERT=<客户端证书路径>
    - export ATUNED_CLIENTKEY=<客户端密钥路径>
    - export ATUNED_SERVERCN=server
  - tlsservercafile：gRPC服务端CA证书路径。
  - tlsservercertfile：gRPC服务端证书路径。
  - tlsserverkeyfile：gRPC服务端密钥路径。
  - rest_tls：系统rest service的SSL/TLS证书校验开关，默认开启。
  - tlsrestcacertfile：系统rest service的服务端CA证书路径。
  - tlsrestservercertfile：系统rest service的服务端证书路径
  - tlsrestserverkeyfile：系统rest service的服务端密钥路径。
  - engine_tls：系统atune engine service的SSL/TLS证书校验开关，默认开启。
  - tlsenginecacertfile：系统atune engine service的客户端CA证书路径。
  - tlsengineclientcertfile：系统atune engine service的客户端证书路径
  - tlsengineclientkeyfile：系统atune engine service的客户端密钥路径
- system信息

  system为系统执行相关的优化需要用到的参数信息，必须根据系统实际情况进行修改。
  - disk：执行analysis流程时需要采集的对应磁盘的信息或执行磁盘相关优化时需要指定的磁盘。
  - network：执行analysis时需要采集的对应的网卡的信息或执行网卡相关优化时需要指定的网卡。
  - user：执行ulimit相关优化时用到的用户名。目前只支持root用户。
- 日志信息

  根据情况修改日志的级别，默认为info级别，日志信息打印在/var/log/messages中。
- monitor信息

  为系统启动时默认采集的系统硬件信息。
- tuning信息

  tuning为系统进行离线调优时需要用到的参数信息。
  - noise：高斯噪声的评估值。
  - sel_feature：控制离线调优参数重要性排名输出的开关，默认关闭。

配置示例

```
################################### server ##############################
# atuned config
[server]
# the protocol grpc server running on
```



```
# ranges: unix or tcp
protocol = unix

# the address that the grpc server to bind to
# default is unix socket /var/run/atuned/atuned.sock
# ranges: /var/run/atuned/atuned.sock or ip address
address = /var/run/atuned/atuned.sock

# the atune nodes in cluster mode, separated by commas
# it is valid when protocol is tcp
# connect = ip01,ip02,ip03

# the atuned grpc listening port
# the port can be set between 0 to 65535 which not be used
# port = 60001

# the rest service listening port, default is 8383
# the port can be set between 0 to 65535 which not be used
rest_host = localhost
rest_port = 8383

# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
# the port can be set between 0 to 65535 which not be used
engine_host = localhost
engine_port = 3838

# when run analysis command, the numbers of collected data.
# default is 20
sample_num = 20

# interval for collecting data, default is 5s
interval = 5

# enable gRPC authentication SSL/TLS
# default is false
# grpc_tls = false
# tlsservercafile = /etc/atuned/grpc_certs/ca.crt
# tlsservercertfile = /etc/atuned/grpc_certs/server.crt
# tlsserverkeyfile = /etc/atuned/grpc_certs/server.key

# enable rest server authentication SSL/TLS
# default is true
rest_tls = true
tlsrestcacertfile = /etc/atuned/rest_certs/ca.crt
tlsrestservercertfile = /etc/atuned/rest_certs/server.crt
tlsrestserverkeyfile = /etc/atuned/rest_certs/server.key

# enable engine server authentication SSL/TLS
# default is true  engine_tls = true
tlsenginecacertfile = /etc/atuned/engine_certs/ca.crt
tlsengineclientcertfile = /etc/atuned/engine_certs/client.crt
tlsengineclientkeyfile = /etc/atuned/engine_certs/client.key    #

################################### log ##################################
[log]
# either "debug", "info", "warn", "error", "critical", default is "info"
level = info

################################### monitor ##################################
[monitor]
# with the module and format of the MPI, the format is {module}_{purpose}
# the module is Either "mem", "net", "cpu", "storage"
# the purpose is "topo"
module = mem_topo, cpu_topo

################################### system ##################################
# you can add arbitrary key-value here, just like key = value
# you can use the key in the profile
```



```
[system]
# the disk to be analysis
disk = sda

# the network to be analysis
network = enp189s0f0
user = root

##################################### tuning #################################
# tuning configs
[tuning]
noise = 0.000000001
sel_feature = false
```

A-Tune engine配置文件/etc/atuned/engine.cnf的配置项说明如下：

- A-Tune engine服务启动配置（可根据需要进行修改）。
  - engine_host：系统atune engine service的侦听地址，默认为localhost。
  - engine_port：系统atune engine service的侦听端口，范围为0~65535未使用的端口，默认为3838。
  - engine_tls：系统atune engine service的SSL/TLS证书校验开关，默认开启。
  - tlsenginecacertfile：系统atune engine service的服务端CA证书路径。
  - tlsengineservercertfile：系统atune engine service的服务端证书路径
  - tlsengineserverkeyfile：系统atune engine service的服务端密钥路径。
- 日志信息

  根据情况修改日志的级别，默认为info级别，日志信息打印在/var/log/messages中。

配置示例

```
 ##################################### engine #################################
[server]
# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
# the port can be set between 0 to 65535 which not be used
engine_host = localhost
engine_port = 3838

# enable engine server authentication SSL/TLS
# default is true
engine_tls = true
tlsenginecacertfile = /etc/atuned/engine_certs/ca.crt
tlsengineservercertfile = /etc/atuned/engine_certs/server.crt
tlsengineserverkeyfile = /etc/atuned/engine_certs/server.key

##################################### log ################################
[log]
# either "debug", "info", "warn", "error", "critical", default is "info"
level = info
```

启动 A-Tune

A-Tune安装完成后，需要配置A-Tune服务，然后启动A-Tune服务。

- 配置A-Tune服务： 修改atuned.cnf配置文件中网卡和磁盘的信息

  通过以下命令可以查找当前需要采集或者执行网卡相关优化时需要指定的网卡，并修改/etc/atuned/atuned.cnf中的network配置选项为对应的指定网卡。

  ```
  ip addr
  ```

  通过以下命令可以查找当前需要采集或者执行磁盘相关优化时需要指定的磁盘，并修改/etc/atuned/atuned.cnf中的disk配置选项为对应的指定磁盘。


```
fdisk -l | grep dev
```

- 关于证书： 因为A-Tune的引擎和客户端使用了grpc通信协议，所以为了系统安全，需要配置证书。因为信息安全的原因，A-Tune不会提供证书生成方法，请用户自行配置系统证书。 如果不考虑安全问题，可以将/etc/atuned/atuned.cnf中的rest_tls 和 engine_tls配置选项设置为false，并且将/etc/atuned/engine.cnf中的engine_tls配置选项设为false。 如果不配置安全证书导致的一切后果与A-Tune无关。
- 启动atuned服务：
  ```
  # systemctl start atuned
  ```
- 查询atuned服务状态：
  ```
  # systemctl status atuned
  ```
  若回显为如下，则服务启动成功。

```
● atuned.service - A-Tune Daemon
   Loaded: loaded (/usr/lib/systemd/system/atuned.service; disabled; vendor preset: disabled)
   Active: active (running) since Sat 2019-12-21 19:28:47 CST; 7s ago
 Main PID: 94790 (atuned)
    Tasks: 19 (limit: 104856)
   Memory: 132.2M
   CGroup: /system.slice/atuned.service
           ├─94790 /usr/bin/atuned
           ├─94797 python3 /usr/libexec/atuned/analysis/app.py /etc/atuned/atuned.cnf
           └─94801 /usr/bin/python3 -c from multiprocessing.semaphore_tracker import main;main(3)
```

启动 A-Tune engine

若需要使用AI相关的功能，需要启动A-Tune engine服务才能使用。

- 启动atune-engine服务：
  ```
  # systemctl start atune-engine
  ```
- 查询atune-engine服务状态：
  ```
  # systemctl status atune-engine
  ```
  若回显为如下，则服务启动成功。

```
[root@localhost misc]# systemctl status atune-engine
● atune-engine.service - A-Tune AI service
   Loaded: loaded (/usr/lib/systemd/system/atune-engine.service; disabled; vendor preset: disabled)
   Active: active (running) since Sat 2020-09-05 04:24:12 EDT; 33min ago
 Main PID: 476292 (python3)
    Tasks: 3
   Memory: 102.5M
   CGroup: /system.slice/atune-engine.service
           ├─476292 /usr/bin/python3 /usr/libexec/atuned/analysis/app_engine.py /etc/atuned/atuned.cnf
           └─476312 /usr/bin/python3 -c from multiprocessing.semaphore_tracker import main;main(3)
```

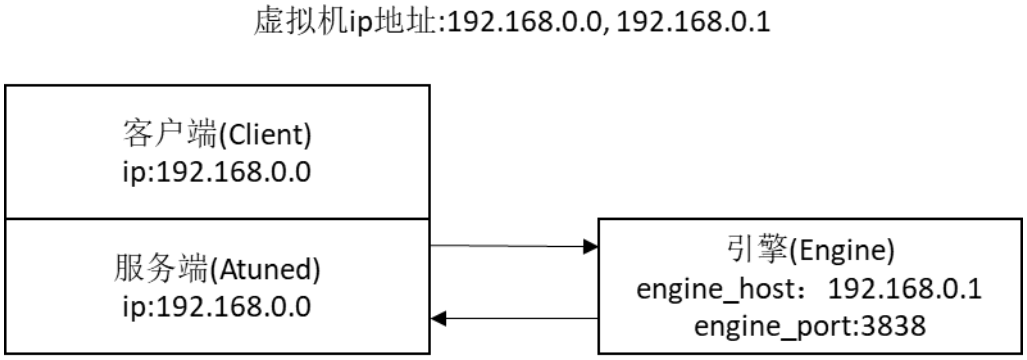
分布式部署

分布式部署目的

为了实现分布式架构和按需部署的目标，A-Tune支持分布式部署。可以将三个组件分开部署，轻量化组件部署对业务影响小，也避免安装过多依赖软件，减轻系统负担。

部署方式：本文档只介绍常用的一种部署方式：在同一节点部署客户端和服务端，在另一个节点上部署引擎模块。其他的部署方式请联系技术人员。

部署关系图：



配置文件

分布式部署需要修改配置文件，将引擎的ip地址和端口号写入配置文件中，别的组件才能访问该ip地址上的引擎组件。

1. 修改服务端节点上的/etc/atuned/atuned.cnf文件：
   - 34行的engine_host和engine_port修改为引擎节点的ip地址和端口号。如上图，应该修改为engine_host = 192.168.0.1 engine_port = 3838。
   - 将49行和55行的 rest_tls 和engine_tls 改为false，否则需要申请和配置证书。在测试环境中可以不用配置ssl证书，但是正式的现网环境需要配置证书，否则会有安全隐患。
2. 修改引擎节点/etc/atuned/engine.cnf文件：
   - 17行和18行的engine_host和engine_port修改为引擎节点的ip地址和端口号。如上图，应该修改为engine_host = 192.168.0.1 engine_port = 3838。
   - 第22行的engine_tls的值改成false。
3. 修改完配置文件后需要重启服务，配置才会生效：
   - 服务端节点输入命令：systemctl restart atuned。
   - 引擎端节点输入命令：systemctl restart atune-engine。
4. （可选步骤）在A-Tune/examples/tuning/compress文件夹下运行tuning命令，检验分布式部署是否成功：
   a. 请先参考A-Tune/examples/tuning/compress/README的指导进行预处理。
   b. 执行atune-adm tuning --project compress --detail compress_client.yaml。

注意事项

1. 本文档不对认证证书配置方法作详细说明，如有需要也可以将atuned.cnf和engine.cnf中的rest_tls/engine_tls设成false。
2. 修改完配置文件后需要重启服务，否则修改不会生效。
3. 注意使用atune服务时不要同时打开代理。
4. atuned.cnf 文件中的[system]模块的disk和network项需要修改为实际使用的磁盘和网卡。

举例

atuned.cnf

```
# ...前略...

# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
```

```
# the port can be set between 0 to 65535 which not be used
engine_host = 192.168.0.1
engine_port = 3838

# ...后略...
```

engine.cnf

```
[server]
# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
# the port can be set between 0 to 65535 which not be used
engine_host = 192.168.0.1
engine_port = 3838
```

集群部署

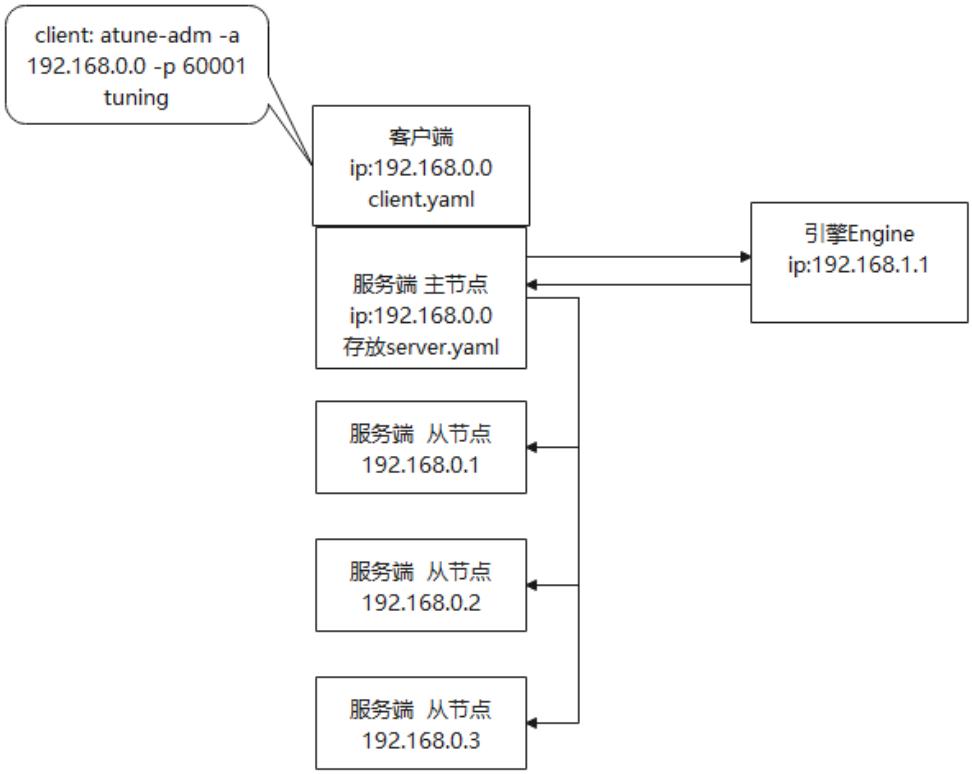
集群部署的目的

为了支持多节点场景快速调优，A-Tune支持对多个节点里的参数配置同时进行动态调优，避免用户单独多次对每个节点进行调优，从而提升调优效率。

集群部署的方式

分为一个主节点和若干个从节点。在主节点部署客户端和服务端，负责接受命令和引擎交互。其他节点接受主节点的指令，对当前节点的参数进行配置。

部署关系图：



上图中客户端和服务端部署在ip为192.168.0.0的节点上，项目文件存放在该节点上，其他节点不用放置项目文件。主节点和从节点之间通过tcp协议通信，所以需要修改配置文件。

atuned.cnf配置文件修改：

1. protocol 值设置为tcp。
2. address设置为当前节点的ip地址。
3. connect设置为所有节点的ip地址，第一个为主节点，其余为从节点ip，中间用逗号隔开。
4. 在调试时，可以设置rest_tls 和engine_tls 为false。
5. 所有的主从节点的atuned.cnf都按照上方描述修改。

注意事项

1. 将engine.cnf中的engine_host和engine_port设置为服务端atuned.cnf中engine_host和engine_port一样的ip和端口号。
2. 本文档不对认证证书配置方法作详细说明，如有需要也可以将atuned.cnf和engine.cnf中的rest_tls和engine_tls设置为false。
3. 修改完配置文件后需要重启服务，否则修改不会生效。
4. 注意使用atune服务时不要同时打开代理。

举例

atuned.cnf

```
# ...前略...

[server]
# the protocol grpc server running on
# ranges: unix or tcp
protocol = tcp

# the address that the grpc server to bind to
# default is unix socket /var/run/atuned/atuned.sock
# ranges: /var/run/atuned/atuned.sock or ip address
address = 192.168.0.0

# the atune nodes in cluster mode, separated by commas
# it is valid when protocol is tcp
connect = 192.168.0.0,192.168.0.1,192.168.0.2,192.168.0.3

# the atuned grpc listening port
# the port can be set between 0 to 65535 which not be used
port = 60001

# the rest service listening port, default is 8383
# the port can be set between 0 to 65535 which not be used
rest_host = localhost
rest_port = 8383

# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
# the port can be set between 0 to 65535 which not be used
engine_host = 192.168.1.1
engine_port = 3838

# ...后略...
```

engine.cnf

```
[server]
# the tuning optimizer host and port, start by engine.service
# if engine_host is same as rest_host, two ports cannot be same
# the port can be set between 0 to 65535 which not be used
engine_host = 192.168.1.1
engine_port = 3838
```


备注： engine.cnf参考分布式部署的配置文件。

9.5.3 使用方法

用户可以通过命令行客户端atune-adm使用A-Tune提供的功能。本章介绍A-Tune客户端包含的功能和使用方法。

总体说明

- 使用A-Tune需要使用root权限。
- atune-adm支持的命令可以通过 atune-adm help/--help/-h 查询。
- define、update、undefine、collection、train、upgrade不支持远程执行。
- 命令格式中，[ ] 表示参数可选，<> 表示参数必选，具体参数由实际情况确定。

查询负载类型

list

功能描述

查询系统当前支持的profile，以及当前处于active状态的profile。

命令格式

atune-adm list

使用示例

```
# atune-adm list
Support profiles:
+-----------------------------------------------+-----------+
| ProfileName                                   | Active    |
+===============================================+===========+
| arm-native-android-container-robox            | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-euleros-baseline-fio         | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-euleros-baseline-lmbench     | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-euleros-baseline-netperf     | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-euleros-baseline-stream      | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-euleros-baseline-unixbench   | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-speccpu-speccpu2006          | false     |
+-----------------------------------------------+-----------+
| basic-test-suite-specjbb-specjbb2015          | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-hdfs-dfsio-hdd                | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-hdfs-dfsio-ssd                | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-spark-bayesian                | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-spark-kmeans                  | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-spark-sql1                    | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-spark-sql10                   | false     |
+-----------------------------------------------+-----------+
| big-data-hadoop-spark-sql2                    | false     |
+-----------------------------------------------+-----------+
```



```
| big-data-hadoop-spark-sql3                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql4                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql5                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql6                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql7                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql8                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-sql9                    | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-tersort                 | false    |
+-----------------------------------------------+----------+
| big-data-hadoop-spark-wordcount               | false    |
+-----------------------------------------------+----------+
| cloud-compute-kvm-host                        | false    |
+-----------------------------------------------+----------+
| database-mariadb-2p-tpcc-c3                   | false    |
+-----------------------------------------------+----------+
| database-mariadb-4p-tpcc-c3                   | false    |
+-----------------------------------------------+----------+
| database-mongodb-2p-sysbench                  | false    |
+-----------------------------------------------+----------+
| database-mysql-2p-sysbench-hdd                | false    |
+-----------------------------------------------+----------+
| database-mysql-2p-sysbench-ssd                | false    |
+-----------------------------------------------+----------+
| database-postgresql-2p-sysbench-hdd           | false    |
+-----------------------------------------------+----------+
| database-postgresql-2p-sysbench-ssd           | false    |
+-----------------------------------------------+----------+
| default-default                               | false    |
+-----------------------------------------------+----------+
| docker-mariadb-2p-tpcc-c3                     | false    |
+-----------------------------------------------+----------+
| docker-mariadb-4p-tpcc-c3                     | false    |
+-----------------------------------------------+----------+
| hpc-gatk4-human-genome                        | false    |
+-----------------------------------------------+----------+
| in-memory-database-redis-redis-benchmark      | false    |
+-----------------------------------------------+----------+
| middleware-dubbo-dubbo-benchmark              | false    |
+-----------------------------------------------+----------+
| storage-ceph-vdbench-hdd                      | false    |
+-----------------------------------------------+----------+
| storage-ceph-vdbench-ssd                      | false    |
+-----------------------------------------------+----------+
| virtualization-consumer-cloud-olc             | false    |
+-----------------------------------------------+----------+
| virtualization-mariadb-2p-tpcc-c3             | false    |
+-----------------------------------------------+----------+
| virtualization-mariadb-4p-tpcc-c3             | false    |
+-----------------------------------------------+----------+
| web-apache-traffic-server-spirent-pingpo      | false    |
+-----------------------------------------------+----------+
| web-nginx-http-long-connection                | true     |
+-----------------------------------------------+----------+
| web-nginx-https-short-connection              | false    |
+-----------------------------------------------+----------+
```

Active为true表示当前激活的profile，示例表示当前激活的profile是web-nginx-http-long-connection。

分析负载类型并自优化

analysis

功能描述

采集系统的实时统计数据进行负载类型识别，并进行自动优化。

命令格式

atune-adm analysis [OPTIONS]

参数说明

- OPTIONS

参数	描述
--model, -m	用户自训练产生的新模型
--characterization, -c	使用默认的模型进行应用识别，不进行自动优化
--times value, -t value	指定收集数据的时长
--script value, -s value	指定需要运行的文件

使用示例

- 使用默认的模型进行应用识别
  ```
  # atune-adm analysis --characterization
  ```
- 使用默认的模型进行应用识别，并进行自动优化
  ```
  # atune-adm analysis
  ```
- 使用自训练的模型进行应用识别
  ```
  # atune-adm analysis --model /usr/libexec/atuned/analysis/models/new-model.m
  ```

自定义模型

A-Tune支持用户定义并学习新模型。定义新模型的操作流程如下：

1. 用define命令定义一个新应用的profile
2. 用collection命令收集应用对应的系统数据
3. 用train命令训练得到模型

define

功能描述

添加用户自定义的应用场景，及对应的profile优化项。

命令格式

atune-adm define <service_type> <application_name> <scenario_name> <profile_path>

使用示例

新增一个profile，service_type的名称为test_service，application_name的名称为test_app，scenario_name的名称为test_scenario，优化项的配置文件为example.conf。

```
# atune-adm define test_service test_app test_scenario ./example.conf
```

example.conf 可以参考如下方式书写（以下各优化项非必填，仅供参考），也可通过atune-adm info查看已有的profile是如何书写的。

```
 [main]
# list its parent profile
[kernel_config]
# to change the kernel config
[bios]
# to change the bios config
[bootloader.grub2]
# to change the grub2 config
[sysfs]
# to change the /sys/* config
[systemctl]
# to change the system service status
[sysctl]
# to change the /proc/sys/* config
[script]
# the script extension of cpi
[ulimit]
# to change the resources limit of user
[schedule_policy]
# to change the schedule policy
[check]
# check the environment
[tip]
# the recommended optimization, which should be performed manunaly
```

collection

功能描述

采集业务运行时系统的全局资源使用情况以及OS的各项状态信息，并将收集的结果保存到csv格式的输出文件中，作为模型训练的输入数据集。

说明：

- 本命令依赖采样工具perf，mpstat，vmstat，iostat，sar。
- CPU型号目前仅支持鲲鹏920，可通过dmidecode -t processor检查CPU型号。

命令格式

atune-adm collection <OPTIONS>

参数说明

- OPTIONS

参数	描述
--filename, -f	生成的用于训练的csv文件名：*名称-时间戳*.csv
--output_path, -o	生成的csv文件的存放路径，需提供绝对路径
--disk, -b	业务运行时实际使用的磁盘，如/dev/sda

参数	描述
--network, -n	业务运行时使用的网络接口，如eth0
--app_type, -t	标记业务的应用类型，作为训练时使用的标签
--duration, -d	业务运行时采集数据的时间，单位秒，默认采集时间1200秒
--interval，-i	采集数据的时间间隔，单位秒，默认采集间隔5秒

使用示例

```
# atune-adm collection --filename name --interval 5 --duration 1200 --output_path /home/data --disk sda --network eth0 --app_type test_service-test_app-test_scenario
```

说明：

实例中定义了每隔5秒收集一次数据，一共收集1200秒；采集后的数据存放在/home/data目录下名称为name的文件中，业务的应用类型是通过atune-adm define指定的业务类型，这里为test_service-test_app-test_scenario 采集间隔和采集时间都可以通过上述选项指定时长。

train

功能描述

使用采集的数据进行模型的训练。训练时至少采集两种应用类型的数据，否则训练会出错。

命令格式

atune-adm train <OPTIONS>

参数说明

- OPTIONS

参数	描述
--data_path, -d	存放模型训练所需的csv文件的目录
--output_file, -o	训练生成的新模型

使用示例

使用data目录下的csv文件作为训练输入，生成的新模型new-model.m存放在model目录下。

```
# atune-adm train --data_path /home/data --output_file /usr/libexec/atuned/analysis/models/new-model.m
```

undefine

功能描述

删除用户自定义的profile。

命令格式

atune-adm undefine <profile>

使用示例

删除自定义的profile。

```
# atune-adm undefine test_service-test_app-test_scenario
```

info

功能描述

查看对应的profile内容。

命令格式

atune-adm info <profile>

使用示例

查看web-nginx-http-long-connection的profile内容：

```
# atune-adm info web-nginx-http-long-connection
*** web-nginx-http-long-connection:
#
# nginx http long connection A-Tune configuration
#
[main]
include = default-default

[kernel_config]
#TODO CONFIG

[bios]
#TODO CONFIG

[bootloader.grub2]
iommu.passthrough = 1

[sysfs]
#TODO CONFIG

[systemctl]
sysmonitor = stop
irqbalance = stop

[sysctl]
fs.file-max = 6553600
fs.suid_dumpable = 1
fs.aio-max-nr = 1048576
kernel.shmmax = 68719476736
kernel.shmall = 4294967296
kernel.shmmni = 4096
kernel.sem = 250 32000 100 128
net.ipv4.tcp_tw_reuse = 1
net.ipv4.tcp_syncookies = 1
net.ipv4.ip_local_port_range = 1024     65500
net.ipv4.tcp_max_tw_buckets = 5000
net.core.somaxconn = 65535
net.core.netdev_max_backlog = 262144
net.ipv4.tcp_max_orphans = 262144
net.ipv4.tcp_max_syn_backlog = 262144
net.ipv4.tcp_timestamps = 0
net.ipv4.tcp_synack_retries = 1
net.ipv4.tcp_syn_retries = 1
net.ipv4.tcp_fin_timeout = 1
net.ipv4.tcp_keepalive_time = 60
net.ipv4.tcp_mem =  362619      483495    725238
net.ipv4.tcp_rmem = 4096        87380    6291456
```



```
net.ipv4.tcp_wmem = 4096        16384   4194304
net.core.wmem_default = 8388608
net.core.rmem_default = 8388608
net.core.rmem_max = 16777216
net.core.wmem_max = 16777216

[script]
prefetch = off ethtool =  -X

{network} hfunc toeplitz

[ulimit]
{user}.hard.nofile = 102400
{user}.soft.nofile = 102400

[schedule_policy]
#TODO CONFIG

[check]
#TODO CONFIG

[tip] SELinux provides extra control and security features to linux kernel. Disabling SELinux will improve the performance but may cause security risks. = kernel disable the nginx log = application
```

更新 profile

用户根据需要更新已有profile。

update

功能描述

将已有profile中原来的优化项更新为new.conf中的内容。

命令格式

atune-adm update <profile> <profile_path>

使用示例

更新名为test_service-test_app-test_scenario的profile优化项为new.conf。

```
# atune-adm update test_service-test_app-test_scenario ./new.conf
```

激活 profile

profile

功能描述

手动激活profile，使其处于active状态。

命令格式

atune-adm profile <profile>

参数说明

profile名参考list命令查询结果。

使用示例

激活web-nginx-http-long-connection对应的profile配置。

```
# atune-adm profile web-nginx-http-long-connection
```


回滚 profile

rollback

功能描述

回退当前的配置到系统的初始配置。

命令格式

atune-adm rollback

使用示例

```
# atune-adm rollback
```

更新数据库

upgrade

功能描述

更新系统的数据库。

命令格式

atune-adm upgrade <DB_FILE>

参数说明

- DB_FILE
  新的数据库文件路径

使用示例

数据库更新为new_sqlite.db。

```
# atune-adm upgrade ./new_sqlite.db
```

系统信息查询

check

功能描述

检查系统当前的cpu、bios、os、网卡等信息。

命令格式

atune-adm check

使用示例

```
# atune-adm check
cpu information:
   cpu:0   version: Kunpeng 920-6426  speed: 2600000000 HZ   cores: 64
   cpu:1   version: Kunpeng 920-6426  speed: 2600000000 HZ   cores: 64
system information:
   DMIBIOSVersion: 20.47
   OSRelease: 5.10.0-182.0.0.95.r2055_140.hce2.aarch64
network information:
   name: eth0          product: HNS GE/10GE/25GE RDMA Network Controller
   name: eth1          product: HNS GE/10GE/25GE Network Controller
   name: eth2          product: HNS GE/10GE/25GE RDMA Network Controller
```



```
name: eth3      product: HNS GE/10GE/25GE Network Controller
name: eth4      product: HNS GE/10GE/25GE RDMA Network Controller
name: eth5      product: HNS GE/10GE/25GE Network Controller
name: eth6      product: HNS GE/10GE/25GE RDMA Network Controller
name: eth7      product: HNS GE/10GE/25GE Network Controller
```

参数自调优

A-Tune提供了最佳配置的自动搜索能力，免去人工反复做参数调整、性能评价的调优过程，极大地提升配置的搜寻效率。

tuning

功能描述

使用指定的项目文件对参数进行动态空间的搜索，找到当前环境配置下的最优解。

命令格式

说明： 在运行命令前，需要满足如下条件：

1. 服务端的yaml配置文件已经编辑完成并放置于 atuned服务下的**/etc/atuned/tuning/**目录中。
2. 客户端的yaml配置文件已经编辑完成并放置于atuned客户端任意目录下。

atune-adm tuning [OPTIONS] <PROJECT_YAML>

参数说明

- OPTIONS

参数	描述
--restore, -r	恢复tuning优化前的初始配置
--project, -p	指定需要恢复的yaml文件中的项目名称
--restart, -c	基于历史调优结果进行调优
--detail, -d	打印tuning过程的详细信息

  说明： 当使用参数时，-p参数后需要跟具体的项目名称且必须指定该项目yaml文件。

- PROJECT_YAML：客户端yaml配置文件。

配置说明

表 9-13 服务端 yaml 文件

配置名称	配置说明	参数类型	取值范围
project	项目名称。	字符串	-
startworkload	待调优服务的启动脚本。	字符串	-

配置名称	配置说明	参数类型	取值范围
stopworkload	待调优服务的停止脚本。	字符串	-
maxiterations	最大调优迭代次数，用于限制客户端的迭代次数。一般来说，调优迭代次数越多，优化效果越好，但所需时间越长。用户必须根据实际的业务场景进行配置。	整型	>10
object	需要调节的参数项及信息。 object 配置项请参见表9-14。	-	-

表 9-14 object 项配置说明

配置名称	配置说明	参数类型	取值范围
name	待调参数名称	字符串	-
desc	待调参数描述	字符串	-
get	查询参数值的脚本	-	-
set	设置参数值的脚本	-	-
needrestart	参数生效是否需要重启业务	枚举	"true", "false"
type	参数的类型，目前支持discrete, continuous两种类型，对应离散型、连续型参数	枚举	"discrete", "continuous"
dtype	该参数仅在type为discrete类型时配置，目前支持int, float, string类型	枚举	"int","float","string"
scope	参数设置范围，仅在type为discrete且dtype为int或float时或者type为continuous时生效	整型/浮点型	用户自定义，取值在该参数的合法范围

配置名称	配置说明	参数类型	取值范围
step	参数值步长，dtype为int或float时使用	整型/浮点型	用户自定义
items	参数值在scope定义范围之外的枚举值，dtype为int或float时使用	整型/浮点型	用户自定义，取值在该参数的合法范围
options	参数值的枚举范围，dtype为string时使用	字符串	用户自定义，取值在该参数的合法范围

表 9-15 客户端 yaml 文件配置说明

配置名称	配置说明	参数类型	取值范围
project	项目名称，需要与服务端对应配置文件中的project匹配	字符串	-
engine	调优算法	字符串	"random", "forest", "gbrt", "bayes", "extraTrees"
iterations	调优迭代次数	整型	>=10
random_starts	随机迭代次数	整型	<iterations
feature_filter_engine	参数搜索算法，用于重要参数选择，该参数可选	字符串	"lhs"
feature_filter_cycle	参数搜索轮数，用于重要参数选择，该参数配合feature_filter_engine使用	整型	-
feature_filter_iters	每轮参数搜索的迭代次数，用于重要参数选择，该参数配合feature_filter_engine使用	整型	-

配置名称	配置说明	参数类型	取值范围
split_count	调优参数取值范围中均匀选取的参数个数，用于重要参数选择，该参数配合feature_filter_engine使用	整型	-
benchmark	性能测试脚本	-	-
evaluations	性能测试评估指标 evaluations 配置项请参见**表9-16**	-	-

表 9-16 evaluations 项配置说明

配置名称	配置说明	参数类型	取值范围
name	评价指标名称	字符串	-
get	获取性能评估结果的脚本	-	-
type	评估结果的正负类型，positive代表最小化性能值，negative代表最大化对应性能值	枚举	"positive","negative"
weight	该指标的权重百分比，0-100	整型	0-100
threshold	该指标的最低性能要求	整型	用户指定

配置示例

服务端yaml文件配置示例：

```
project: "compress"
maxiterations: 500
startworkload: ""
stopworkload: ""
object :
    -
      name : "compressLevel"
      info :
          desc : "The compresslevel parameter is an integer from 1 to 9 controlling the level of compression"
          get : "cat /root/A-Tune/examples/tuning/compress/compress.py | grep 'compressLevel=' | awk -F '='
'{print $2}'"
          set : "sed -i 's/compressLevel=\\s*[0-9]*/compressLevel=$value/g' /root/A-Tune/examples/tuning/
compress/compress.py"
          needrestart : "false"
```

```
        type : "continuous"
        scope :
          - 1
          - 9
        dtype : "int"
  -
    name : "compressMethod"
    info :
        desc : "The compressMethod parameter is a string controlling the compression method"
        get : "cat /root/A-Tune/examples/tuning/compress/compress.py | grep 'compressMethod=' | awk -F '='
'{print $2}' | sed 's/\"//g'"
        set : "sed -i 's/compressMethod=\\s*[0-9,a-z,\"]*/compressMethod=\"$value\"/g' /root/A-Tune/
examples/tuning/compress/compress.py"
        needrestart : "false"
        type : "discrete"
        options :
          - "bz2"
          - "zlib"
          - "gzip"
        dtype : "string"
```

客户端yaml文件配置示例：

```
project: "compress"
engine : "gbrt"
iterations : 20
random_starts : 10

benchmark : "python3 /root/A-Tune/examples/tuning/compress/compress.py"
evaluations :
 -
   name: "time"
   info:
       get: "echo '$out' | grep 'time' | awk '{print $3}'"
       type: "positive"
       weight: 20
 -
   name: "compress_ratio"
   info:
       get: "echo '$out' | grep 'compress_ratio' | awk '{print $3}'"
       type: "negative"
       weight: 80
```

使用示例

- 下载测试数据

  ```
  wget http://cs.fit.edu/~mmahoney/compression/enwik8.zip
  ```

- 准备调优环境 prepare.sh文件示例：

  ```
  #!/usr/bin/bash
  if [ "$#" -ne 1 ]; then
    echo "USAGE: $0 the path of enwik8.zip"
    exit 1
  fi
  path=$(
    cd "$(dirname "$0")"
    pwd
  )
  echo "unzip enwik8.zip"
  unzip "$path"/enwik8.zip

  echo "set FILE_PATH to the path of enwik8 in compress.py"
  sed -i "s#compress/enwik8#$path/enwik8#g" "$path"/compress.py

  echo "update the client and server yaml files"
  sed -i "s#python3 .*compress.py#python3 $path/compress.py#g" "$path"/compress_client.yaml
  sed -i "s# compress/compress.py# $path/compress.py#g" "$path"/compress_server.yaml

  echo "copy the server yaml file to /etc/atuned/tuning/"
  cp "$path"/compress_server.yaml /etc/atuned/tuning/
  ```


运行脚本：

```
sh prepare.sh enwik8.zip
```

- 进行tuning调优

```
atune-adm tuning --project compress --detail compress_client.yaml
```

- 恢复tuning调优前的初始配置，compress为yaml文件中的项目名称

```
atune-adm tuning --restore --project compress
```


10 内核功能与接口

10.1 内核 memory 的 OOM 进程控制策略

背景信息

现有操作系统中，支持配置离线业务和在线业务。当内存发生OOM时，会优先选择离线业务控制组中的消耗内存最多的进程，结束进程回收内存，但是对于某些离线业务也有核心业务，因此会造成很大的影响。

针对这个问题，HCE调整了OOM时回收内存的策略，增加了配置cgroup优先级的功能。 内存紧张情况下内核会遍历cgroup，对低优先级的cgroup结束进程，并回收内存，使离线业务中重要的业务可以存活下来。

前提条件

vm.panic_on_oom默认关闭配置为0，系统OOM时不会发生panic。在使用memcg OOM优先级配置时（即memcg_qos_enable配置为1或2），如果vm.panic_on_oom接口被打开，需先执行**sysctl -w vm.panic_on_oom=0**命令，关闭系统参数vm.panic_on_oom。

memcg OOM 优先级接口功能说明

表 10-1 接口功能说明

接口	说明	取值
memcg_qos_enable	memcg OOM优先级策略开关。 ● 0：不开启优先级配置。当OOM时，按照系统原有的OOM操作结束进程，结束内存消耗最大的进程，回收内存。 ● 1：开启优先级配置并以cgroup为粒度。当OOM时，结束优先级低的cgroup所有进程，并回收内存。 ● 2：开启优先级配置并以单个进程个为粒度。当OOM时，结束优先级低的cgroup中的最大的一个进程，并回收内存。	整数形式，取值范围为0~2，默认值为0。
memory.qos_level	配置cgroup组优先级。值越小cgroup组优先级越低。 ● 当memcg OOM时，会以当前cgroup组为父节点，查找子节点优先级最低的cgroup组中内存使用最大的进程，结束该进程，回收内存。 ● 当OOM时，对于优先级相等的cgroup组，会根据组的内存使用量进行二次排序，选择内存使用最大的进行OOM操作。 **说明** ● 使用memory.qos_level的前提条件为memcg_qos_enable取值须为1或2。 ● 新创建的cgroup组的memory.qos_level值默认会继承父节点的memory.qos_level的值，但是子节点的优先级不受父节点的限制。 ● 如果修改cgroup组父节点的优先级，子节点的优先级会自动调整，和父节点保持一致。	整数形式，取值范围为-1024~1023，默认值为0。

接口配置示例

按如下所示创建6个cgroup子节点A、B、C、D、E、F，配置memcg_qos_enable接口，并通过memory.qos_level接口设置OOM的优先级，优先级取值如图所示。

图 10-1 优先级取值

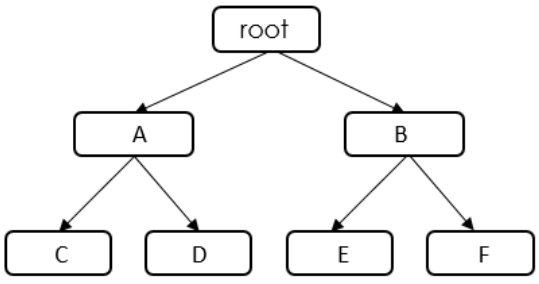


表 10-2 数据规划

cgroup组	memory.qos_level取值	说明
A	**-8**	当在root中进行OOM操作时，内核遍历root所有cgroup组，最终选择优先级最低的A、E。由于A和E优先级相同，内核继续对A和E使用的内存进行比较。 ● 如果设置memcg_qos_enable=1，优先对A/E中内存使用量大的一个cgroup组回收内存，结束cgroup组中的所有进程，并回收内存。 ● 如果设置memcg_qos_enable=2，优先对A/E中内存使用量最大的一个进程回收内存。
B	10	
C	1	
D	2	
E	**-8**	
F	3	

1. 关闭系统参数vm.panic_on_oom。
```
sysctl -w vm.panic_on_oom=0
```
2. 开启memcg OOM优先级策略功能。
```
echo 1 > /proc/sys/vm/memcg_qos_enable
```
3. 运行以下命令创建两个cgroup节点 A、B，并分别设置A、B节点的memcg OOM优先级值为-8、10。
```
mkdir /sys/fs/cgroup/memory/A
mkdir /sys/fs/cgroup/memory/B
cd /sys/fs/cgroup/memory/A
echo -8 > memory.qos_level
cd /sys/fs/cgroup/memory/B
echo 10 > memory.qos_level
```
4. 运行以下命令分别在A节点下创建C、D子节点，在B节点下创建E、F子节点， 并分别设置C、D、E、F子节点的memcg OOM优先级值为1、2、-8、3。
```
mkdir /sys/fs/cgroup/memory/A/C
mkdir /sys/fs/cgroup/memory/A/D
mkdir /sys/fs/cgroup/memory/B/E
mkdir /sys/fs/cgroup/memory/B/F
cd /sys/fs/cgroup/memory/A/C
echo 1 > memory.qos_level
cd /sys/fs/cgroup/memory/A/D
echo 2 > memory.qos_level
cd /sys/fs/cgroup/memory/B/E
echo -8 > memory.qos_level
cd /sys/fs/cgroup/memory/B/F
echo 3 > memory.qos_level
```

10.2 内核 memory 的多级内存回收策略

需求背景

在容器高密度混合部署场景中，IO读写较多的离线业务消耗大量page cache，导致系统空闲内存降低，达到全局空闲内存水位线后触发全局内存回收，使得在线任务申请内存时进入内存回收的慢路径，引发时延抖动。

为解决此问题，HCE 2.0新增支持多级内存回收策略。申请内存时，设置内存警示值，可触发内存回收任务，保证可用内存空间；回收内存时，设置多级内存保护水位线，保护任务可用性。

约束与限制

memory.min和memory.low只在叶子节点生效。创建memory cgroup时，会将父节点的memory.min和memory.low清零。

多级内存回收接口说明

memory.min、memory.low和memory.high接口在非根的memory cgroup下面默认存在，可以向文件内写值配置，也可以读取当前配置。合理的取值大小顺序为memory.min≤memory.low<memory.high，三者可独立使用，也可联合使用。

内存回收机制如下图。

图 10-2 内存回收机制

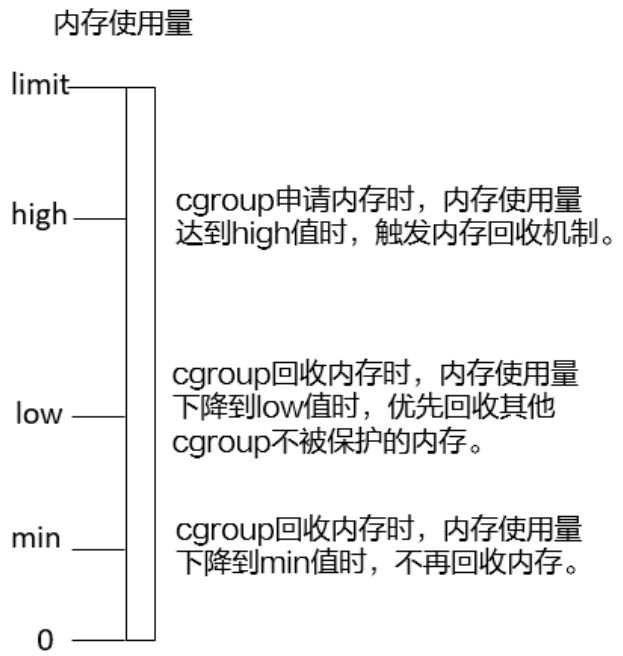


表 10-3 接口参数说明

接口	说明
memory.min	硬保护内存保护值，默认值为0。系统没有可回收内存的时候，也不会回收在该值边界及以下的内存。读写说明如下： ● 读该接口可以查看硬保护内存大小，单位为byte。 ● 写该接口可以设置硬保护内存大小，单位不做限制。 ● 配置范围：0-memory.limit_in_bytes。

接口	说明
memory.low	尽力而为的内存保护值，默认值为0。 系统优先回收未被保护的cgroup组的内存。如果内存还不足，再回收memory.min到memory.low之间的内存。 读写说明如下： ● 读该接口可以查看Best-effort内存保护值，单位为byte。 ● 写该接口可以设置Best-effort内存保护值，单位不做限制。 ● 配置范围：0-memory.limit_in_bytes。
memory.high	内存回收警示值，默认为max。当cgroup组内存使用量达到high值，会触发对该cgroup及子节点的同步内存回收任务，将内存尽量限制在high之下，避免触发memory.limit限制导致OOM。读写说明如下： ● 读该接口可以查看Throttle limit内存保护值，单位为byte。 ● 写该接口可以设置Throttle limit内存保护值，单位不做限制。 ● 配置范围：0-memory.limit_in_bytes。

接口示例

按如下所示创建6个cgroup节点A、B、C、D、E、F，配置memory.min接口，示例说明多级内存回收策略。

图 10-3 多级内存回收策略

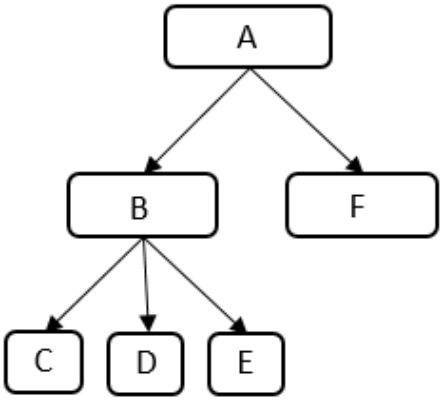


表 10-4 数据规划

cgroup组	memory.limit_in_bytes	memory.min	memory.usage_in_bytes
A	200M	0	-
B	-	0	-
C	-	75M	50M

cgroup组	memory.limit_in_bytes	memory.min	memory.usage_in_bytes
D	-	25M	50M
E	-	0	50M
F	-	125M	-

1. 创建cgroup节点A，并配置memory.limit_in_bytes=200M。
   ```
   mkdir /sys/fs/cgroup/memory/A
   echo 200M > /sys/fs/cgroup/memory/A/memory.limit_in_bytes
   ```
2. 创建cgroup节点B。
   ```
   mkdir /sys/fs/cgroup/memory/A/B
   ```
3. 创建cgroup节点C，并配置memory.min=75M，在当前节点创建申请50M cache的进程。
   ```
   mkdir /sys/fs/cgroup/memory/A/B/C
   echo 75M > /sys/fs/cgroup/memory/A/B/C/memory.min
   ```
4. 创建cgroup节点D，并配置memory.min=25M，在当前节点创建申请50M cache的进程。
   ```
   mkdir /sys/fs/cgroup/memory/A/B/D
   echo 25M > /sys/fs/cgroup/memory/A/B/D/memory.min
   ```
5. 创建cgroup节点E，并配置memory.min=0，在当前节点创建申请50M cache的进程。
   ```
   mkdir /sys/fs/cgroup/memory/A/B/E
   ```
6. 创建cgroup节点F，并配置memory.min=125M，申请125M保护内存，触发内存回收。
   ```
   mkdir /sys/fs/cgroup/memory/A/F
   echo 125M > /sys/fs/cgroup/memory/A/F/memory.min
   ```

返回结果：

C节点memory.min=75M，memory.usage_in_bytes=50M。

D节点memory.min=25M，memory.usage_in_bytes=25M。

E节点memory.min=0，memory.usage_in_bytes=0。

B节点memory.usage_in_bytes=75M。

10.3 内核 cpu cgroup 的多级混部调度

需求背景

在业务混部场景中，Linux内核调度器需要为高优先级任务赋予更多的调度机会，并需要把低优先级任务对内核调度带来的影响降到最低。原有的在线、离线两级混部调度无法满足业务需求。

为解决此问题，HCE 2.0内核cpu cgroup支持多级混部调度，提供cgroup接口/sys/fs/cgroup/cpu/cpu.qos_level将任务调度级别扩展到5个级别，支持用户对每个cgroup组单独设置优先级。

约束与限制

当前cpu.qos_level仅支持cgroup-v1， 不支持cgroup-v2。

多级混部调度接口说明

cpu.qos_level的生效规则：

- CFS调度器自上而下逐层选择task_group，同一个父节点内的子节点之间cpu.qos_level生效。
- 子cgroup创建时默认继承父cgroup的cpu.qos_level，支持重新配置cpu.qos_level值。
- 同优先级的qos_level之间的资源竞争服从CFS调度器的策略。
- 同一个cpu上，qos_level < 0 的任务始终会被qos_level >= 0的任务无条件抢占，不受层级约束。

在调度高优先级任务时：

- 在线任务可无条件抢占离线任务，在多核调度时，在线任务可优先抢占其他核上的离线任务。超线程（ Hyper Thread ）场景，优先级为2的在线任务可驱逐SMT上的离线任务。
- 高优先级的任务被唤醒时获得一定的时间片加速，可立刻抢占低优先级的任务（ 忽略CFS的最小运行时间片），获得更低的时延响应。

表 10-5 cpu.qos_level 接口说明

接口	说明
cpu.qos_level	配置cgroup的cpu优先级。取值类型为整数形式，取值范围 为[-2, 2]，默认值为0。 ● cpu.qos_level >= 0 标识cgroup组内任务为在线任务，在线任务可无条件抢占离线任务。 优先级 0 < 1 < 2，同为在线业务的任务，高优先级的在线相比低优先级的在线可获取更多的CPU资源抢占机会。 ● cpu.qos_level < 0 标识cgroup组内的任务为离线任务，-1优先级高于-2。 -1级别的任务比-2级别的任务可获得更多的CPU资源抢占机会。 父节点运行离线任务时，子节点只能继承父节点的优先级，不支持修改为其他优先级。

接口示例

按如下所示创建3个cgroup节点A、B、C，配置并查看qos_level接口。

图 10-4 cgroup 节点 A、B、C

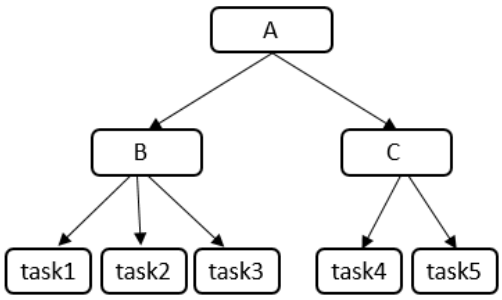


表 10-6 数据规划

cgroup组	cpu.qos_level
A	1
B	-2
C	2

1. 创建cgroup A及子节点B、C，依次设置A、B、C的cpu调度优先级为1、-2、2。

   cgroup A和cgroup C中的任务可无条件抢占cgroup B任务的CPU资源，cgroup C优先级大于cgroup A。

   ```
   mkdir -p /sys/fs/cgroup/cpu/A
   echo 1 > /sys/fs/cgroup/cpu/A/cpu.qos_level
   mkdir -p /sys/fs/cgroup/cpu/A/B
   echo -2 > /sys/fs/cgroup/cpu/A/B/cpu.qos_level
   mkdir -p /sys/fs/cgroup/cpu/A/C
   echo 2 > /sys/fs/cgroup/cpu/A/C/cpu.qos_level
   ```

2. 将task1、task2、task3进程加入cgroup B。

   task1、task2、task3进程加入cgroup B后，task1、task2、task3进程的cpu调度优先级为-2。

   ```
   echo $PID1 > /sys/fs/cgroup/cpu/A/B/tasks
   echo $PID2 > /sys/fs/cgroup/cpu/A/B/tasks
   echo $PID3 > /sys/fs/cgroup/cpu/A/B/tasks
   ```

3. 将task4、task5进程加入cgroup C。

   task4、task5进程加入cgroup C后，task4、task5进程的cpu调度优先级为2。

   ```
   echo $PID4 > /sys/fs/cgroup/cpu/A/C/tasks
   echo $PID5 > /sys/fs/cgroup/cpu/A/C/tasks
   ```

4. 查看cgroup B的cpu调度优先级及进程。

   ```
   [root@localhost cpu_qos]# cat /sys/fs/cgroup/cpu/A/B/cpu.qos_level
   -2
   [root@localhost boot]# cat /sys/fs/cgroup/cpu/A/B/tasks
   1879
   1880
   1881
   ```

5. 查看cgroup C的cpu调度优先级及进程。

   ```
   [root@localhost cpu_qos]# cat /sys/fs/cgroup/cpu/A/C/cpu.qos_level
   2
   [root@localhost boot]# cat /sys/fs/cgroup/cpu/A/C/tasks
   1882
   1883
   ```


10.4 内核异常事件分析指南

背景说明

HCE运行时，不可避免地会出现一些内核事件，例如**soft lockup**、**RCU(Read-Copy Update) stall**、**hung task**、**global OOM**、**cgroup OOM**、**page allocation failure**、**list corruption**、**Bad mm_struct**、**I/O error**、**EXT4-fs error**、**MCE (Machine Check Exception)**、**fatal signal**、**warning**、**panic**、**oops**。本节为您介绍这些内核事件的原理及触发方法。

soft lockup

soft lockup是内核检测CPU在一定时间内（默认20秒）没有发生调度切换时，上报的异常。

- 原理

  soft lockup利用Linux内核watchdog机制触发。内核会为每一个CPU启动一个优先级最高的FIFO实时内核线程watchdog，名称为watchdog/0、watchdog/1以此类推。这个线程会定期调用watchdog函数，默认每4秒执行一次。同时调用过后会重置一个hrtimer定时器在2倍的watchdog_thresh时间后到期。watchdog_thresh是内核参数，对应默认超时时间为20秒。

  在超时时间内，如果内核线程watchdog没被调度，hrtimer定时器到期，即触发内核打印类似如下的soft lockup异常。

  ```
  BUG: soft lockup - CPU#3 stuck for 23s! [kworker/3:0:32]
  ```

- 触发方法

  关闭中断或关闭抢占，软件执行死循环。

RCU(Read-Copy Update) stall

RCU stall是一种rcu宽限期内rcu相关内核线程没有得到调度的异常。

- 原理

  在RCU机制中，reader不用等待，可以任意读取数据，RCU记录reader的信息；writer更新数据时，先复制一份副本，在副本上完成修改，等待所有reader退出后，再一次性地替换旧数据。

  writer需要等所有reader都停止引用“旧数据”才能替换旧数据。这相当于给了这些reader一个优雅退出的宽限期，这个等待的时间被称为grace-period，简称GP。

  当reader长时间没有退出，writer等待的时间超过宽限期时，即上报RCU Stall。

- 触发方法

  内核在Documentation/RCU/stallwarn.txt文档列出了可能触发RCU stall的场景：cpu在rcu reader临界区一直循环，cpu在关闭中断或关闭抢占场景中一直循环等。

hung task

当内核检测到进程处于D状态超过设定的时间时，上报hung task异常。

- 原理

  进程其中一个状态是TASK_UNINTERRUPTIBLE，也叫D状态，处于D状态的进程只能被wake_up唤醒。内核引入D状态时，是为了让进程等待IO完成。正常情况下，IO正常处理，进程不应该长期处于D状态。

  hung task检测进程长期处于D状态的原理，内核会创建一个线程khungtaskd，用来定期遍历系统中的所有进程，检查是否存在处于D状态超过设置时长（默认120秒）的进程。如果存在这样的进程，则打印并上报相关警告和进程堆栈。如果配置了hung_task_panic（通过proc或内核启动参数配置），则直接发起panic。

- 触发方法

  创建内核线程，设成D状态，scheduler释放时间片。

global OOM

Linux的OOM killer特性是一种内存管理机制，在系统可用内存较少的情况下，内核为保证系统还能够继续运行下去，会选择结束一些进程释放掉一些内存。

- 原理

  通常oom_killer的触发流程是：内核为某个进程分配内存，当发现当前物理内存不够时，触发OOM。OOM killer遍历当前所有进程，根据进程的内存使用情况进行打分，然后从中选择一个分数最高的进程，终止进程释放内存。

  OOM killer的处理主要集中在mm/oom_kill.c，核心函数为out_of_memory，函数处理流程为：

  a. 通知系统中注册了oom_notify_list的模块释放一些内存，如果从这些模块中释放出了一些内存，直接结束oom killer流程；如果回收失败，进入下一步。
  b. 触发oom killer通常是由当前进程进行内存分配所引起。如果当前进程已经挂起了一个SIG_KILL信号或者正在退出，直接选中当前进程，终止进程释放内存；否则进入下一步。
  c. 检查panic_on_oom系统管理员的设置，决定OOM时是进行oom killer还是panic。如果选择panic，则系统崩溃并重启；如果选择oom killer，进入下一步。
  d. 进入oom killer，检查系统设置，系统管理员可设置终止当前尝试分配内存、引起OOM的进程或其它进程。如果选择终止当前进程，oom killer结束；否则进入下一步。
  e. 调用select_bad_process选中合适进程，然后调用oom_kill_process终止选中的进程。如果select_bad_process没有选出任何进程，内核进入panic。

- 触发方法

  执行占用大内存的程序，直到内存不足。

cgroup OOM

- 与global OOM的区别

  cgroup OOM与global OOM的内存范围不同。当FS cgroup组内进程使用内存超过了设置的上限，cgroup通过KILL相应的进程来释放内存。

- 触发方法

  指定cgroup下某个进程持续占用大内存直至内存不足。

page allocation failure

page allocation failure是申请空闲页失败时，系统上报的错误。当程序申请某个阶数（order）的内存，但系统内存中，没有比申请阶数高的空闲页，即触发内核报错。

- 原理

  Linux使用伙伴系统（buddy system）内存分配算法。将所有的空闲页表（一个页表的大小为4K）分别链接到包含了11个元素的数组中，数组中的每个元素将大小相同的连续页表组成一个链表，页表的数量为1、2、4、8、16、32、64、128、256、512、1024，所以一次性可以分配的最大连续内存为1024个连续的4k页表，即4MB的内存。

  假设申请一个包括256个页表的内存，指定阶数order为6，系统会依次查找数组中的第9、10、11个链表，上一个为空，表示没有此阶数的空闲内存，查找下一个，直到最后一个链表。

  如果所有链表均为空，则申请失败，内核上报page allocation failure错误。输出报错信息，描述申请阶数为6的内存页失败：

  ```
  page allocation failure:order:6
  ```

- 触发方法

  用alloc_pages连续申请高阶数内存页（例如order=10），不释放，直到申请失败。

list corruption

list corruption是内核检查链表有效性失败的报错，报错类型分为list_add corruption和list_del corruption。

- 原理

  内核提供list_add和list_del，对传入的链表先检查链表的有效性（valid），检查通过后，修改链表增加或删除节点。如果检查链表失败，则上报corruption错误。检查和报错代码在内核lib/list_debug.c。

图 10-5 报错类型 list_add corruption 和 list_del corruption

```
bool __list_add_valid(struct list_head *new, struct list_head *prev,
              struct list_head *next)
{
    if (CHECK_DATA_CORRUPTION(next->prev != prev,
            "list_add corruption. next->prev should be prev (%px), but was %px. (next=%px).\n",
            prev, next->prev, next) ||
        CHECK_DATA_CORRUPTION(prev->next != next,
            "list_add corruption. prev->next should be next (%px), but was %px. (prev=%px).\n",
            next, prev->next, prev) ||
        CHECK_DATA_CORRUPTION(new == prev || new == next,
            "list_add double add: new=%px, prev=%px, next=%px.\n",
            new, prev, next))
        return false;

    return true;
}
EXPORT_SYMBOL(__list_add_valid);

bool __list_del_entry_valid(struct list_head *entry)
{
    struct list_head *prev, *next;

    prev = entry->prev;
    next = entry->next;

    if (CHECK_DATA_CORRUPTION(next == LIST_POISON1,
            "list_del corruption, %px->next is LIST_POISON1 (%px)\n",
            entry, LIST_POISON1) ||
        CHECK_DATA_CORRUPTION(prev == LIST_POISON2,
            "list_del corruption, %px->prev is LIST_POISON2 (%px)\n",
            entry, LIST_POISON2) ||
        CHECK_DATA_CORRUPTION(prev->next != entry,
            "list_del corruption. prev->next should be %px, but was %px\n",
            entry, prev->next) ||
        CHECK_DATA_CORRUPTION(next->prev != entry,
            "list_del corruption. next->prev should be %px, but was %px\n",
            entry, next->prev))
        return false;

    return true;

}
EXPORT_SYMBOL(__list_del_entry_valid);
```

这种错误通常为内存异常操作导致，例如内存踩踏、内存损坏等。

- 触发方法

  用list.h的内核标准接口创建链表，非法修改链表节点的prev或next指针，再调用内核list_add/list_del接口。

Bad mm_struct

Bad mm_struct错误通常是由于内核中的一个或多个mm_struct数据结构被破坏或损坏所导致。

- 原理

  mm_struct是Linux内核中的一个重要数据结构，用于跟踪进程的虚拟内存区域。如果该数据结构被破坏，可能会导致进程崩溃或系统崩溃。这种错误通常由内存异常导致，例如mm_struct区域的内存被踩踏、内存越界等。

- 触发方法

  无人为触发方法，当硬件错误，或者Linux系统内核代码错误时会触发Bad mm_struct。

I/O error

Linux I/O error报错通常表示输入/输出操作失败，在网卡、磁盘等IO设备驱动异常，或文件系统异常都可能打印这个错误。

- 原理

  错误原因取决于代码执行失败的条件。常见的触发异常的原因是硬件故障、磁盘损坏、文件系统错误、驱动程序问题、权限问题等。例如当系统尝试读取或写入磁盘上的数据时，如果发生错误，就会出现I/O错误。

- 触发方法

  系统读写磁盘过程，拔出磁盘，导致磁盘数据损坏。

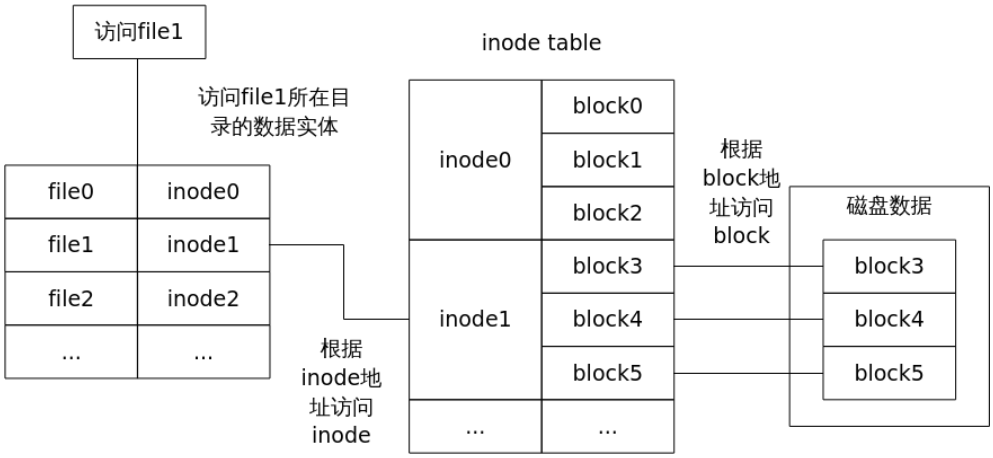
EXT4-fs error

EXT4-fs error是由于ext4格式的文件系统中，文件节点的错误导致。

- 原理

  文件储存的最小存储单位叫做“扇区”（sector），连续多个扇区组成“块”（block）。inode节点储存文件的元信息，包括文件的创建者、创建日期、大小、属性、实际存储的数据块（block number）。EXT4格式的inode信息校验失败会触发EXT4-fs error。

  图 10-6 EXT4 文件系统结构



  内核ext4校验使用checksum校验inode信息，当出现分区表错误、磁盘硬件损坏时，内核返回-EIO错误码，系统上报EXT4-fs error checksum invalid错误。

- 触发方法

  使用磁盘过程中强行拔盘，重新接入读盘。

MCE (Machine Check Exception)

MCE是CPU发现硬件错误时触发的异常（exception），上报中断号是18，异常的类型是abort。

- 原理

  导致MCE的原因主要有：总线故障、内存ECC校验错、cache错误、TLB错误、内部时钟错误等。不仅硬件故障会引起MCE，不恰当的BIOS配置、firmware bug、软件bug也有可能引起MCE。

  MCE中断上报，操作系统检查一组寄存器称为Machine-Check MSR，根据寄存器的错误码执行对应的处理函数（函数实现依赖不同的芯片架构实现）。

- 触发方法

  无人为触发方法，当总线故障、内存ECC校验错、cache错误、TLB错误、内部时钟错误等时会触发MCE。

fatal signal

fatal signal指信号处理方式不能被设置为忽略或执行自定义处理函数的信号类型，包括SIGKILL、SIGSTOP、SIGILL等。

- 原理

  Linux信号（signal）机制，用于系统中进程间通讯，是一种异步的通知机制。当一个信号发送给一个进程，而操作系统中断了进程正常的控制流程时，任何非原子操作都将被中断。

  如果SIG符合条件，即为fatal信号：

  图 10-7 fatal 信号示例

  ```
  #define sig_fatal(t, signr) \
      (!siginmask(signr, SIG_KERNEL_IGNORE_MASK|SIG_KERNEL_STOP_MASK) && \
  ```

- 触发方法

  用户态程序执行非法指令、kill -9杀进程。

warning

warning是指操作系统在运行时检测到需要立即注意的内核问题（issue），而采取的上报动作，记录并打印发生时的调用栈信息。上报后，系统继续运行。

- 原理

  warning是通过调用WARN、WARN_ON、WARN_ON_ONCE等宏来触发的。

  导致warning的原因有多种，需要根据调用栈回溯，找到调用warning宏的具体原因。warning宏并不会导致系统运行状态发生改变，也不提供处理warning的指导。

- 触发方法

  根据系统调用构造warning条件。

panic

Kernel panic是指操作系统在监测到内部的致命错误，并无法安全处理此错误时采取的动作。内核触发到某种异常情况，运行kernel_panic函数，并尽可能把异常发生时获取的全部信息打印出来。

- 原理

  导致异常的原因多种多样，通过异常打印的调用信息，找到调用kernel_panic的原因。常见的原因包括内核堆栈溢出、内核空间的除0异常、内存访问越界、内核陷入死锁等。

- 触发方法

  内核态读0地址。

oops

oops 是内核在检测到严重错误时生成的错误报告。这些错误通常包括内核代码中的非法内存访问、除零错误、无效的指令等。

- 原理

当内核检测到这些错误时，会生成一个 oops 报告，记录错误发生时的上下文信息，包括寄存器状态、调用栈、内存状态等。oops 报告通常会打印到系统日志中，并且可以通过 dmesg 命令查看。内核会尝试恢复执行，但如果错误严重到无法恢复，可能会导致系统崩溃或重启。

- 触发方法

  a. 非法内存访问：访问未分配或已释放的内存。

  b. 除零错误：在内核代码中进行除零操作。

  c. 无效指令：执行非法或未定义的指令。

  d. 硬件故障：如内存故障、CPU故障等。

10.5 内核代码大页

背景说明

对于一些代码段较大，访问频次较高的应用，会导致较高的Tlb miss，HCE支持代码大页特性(提供配置参数，默认关闭可手动开启)，支持将应用程序的可执行部分放入到大页中, 以降低程序的TLB miss，提升数据库(MySQL)、大型应用软件等大代码段应用性能。

使用说明

- 检查内核启动参数是否包含exec_hugepages，若无则要添加上才能使用该功能（需要重启）。
- 为单个程序开启代码大页

```
HUGEPAGE_ELF=1 ./app    // 该环境变量不会被复制到子进程。HUGEPAGE_ELF=0时应用程序不使用大页。
```

- 为被hugepageedit标记的程序开启代码大页

```
export HUGEPAGE_PROBE=1    // 导出环境变量HUGEPAGE_PROBE。该环境变量可被复制到子进程。
hugepageedit ./app         // 使用hugepageedit工具标记欲使能程序的二进制文件，glibc-devel包提供hugepageedit工具。
./app                      // 启动使能代码大页的程序。
```

- 全局开启代码大页

```
echo 1 > /sys/kernel/mm/exec_hugepages/enabled    // echo 0时关闭，且默认为0。开启后启动符合条件的程序会自动使用代码大页
```

- 查看是否启用

```
cat /proc/进程号/smaps | grep eh    // 若有被标记eh的则说明成功启用
```

约束限制

- 程序未按照2MB对齐编译时，每段中不足2MB的部分无法使用大页，建议对齐使用。
- mprotect系统调用要求传入的地址按2M对齐、大小为2M的倍数，程序代码需要进行适配修改。

  2MB地址对齐需要重新编译程序，并在链接器选项中加入"-Wl,-zcommon-page-size=2097152 -Wl,-zmax-page-size=2097152"。
- 大页预留不足时会尝试申请2MB页，如果由于内存碎片等原因申请失败时，程序会回退至小页。程序中多个线程进入回退小页流程会触发SIGBUS。

10.6 定制 TCP 重传策略

背景说明

TCP报文重传遵循指数退避原则，在弱网场景下包到达率较低，时延较高。因此新增通过编程接口定制TCP重传策略的能力，包括定制线性退避次数、最大重传次数、最大重传间隔时间，以优化弱网场景的包到达率和时延。

接口说明

- 通过sysctl设置TCP重传策略功能

  net.ipv4.tcp_sock_retrans_policy_custom，用于控制定制TCP重传策略功能是否开启。

  0：关闭定制TCP重传策略功能。

  1：开启定制TCP重传策略功能。

  ```
  sysctl -w net.ipv4.tcp_sock_retrans_policy_custom=1
  ```

- 调用setsockopt针对指定socket定制TCP重传策略
  - 接口：

    ```
    int setsockopt(int sockfd, int level, int optname, const void *optval, socklen_t optlen);
    int getsockopt(int sockfd, int level, int optname, const void *optval, socklen_t optlen);
    ```

    optname传递枚举，optval传递结构体起始地址。
  - 枚举：

    ```
    TCP_SOCK_RETRANS_POLICY_CUSTOM 100
    ```

    结构体：

    ```
    struct tcp_sock_retrans_policy {
        uint8_t tcp_linear_timeouts_times; /* number of times linear backoff */
        uint8_t tcp_retries_max; /* maximum retransmission times */
        uint8_t tcp_rto_max; /* maximum RTO time, unit:second */
        uint8_t unused;
    };
    ```

    其中：

    i. tcp_linear_timeouts_times表示线性退避次数，范围：取值是在0到32之间。

    0：使用默认策略（指数退避）。

    非0：定制线性退避次数。

    说明

    - 指数退避：发生报文重传时，RTO值直接翻倍
    - 线性退避：发生报文重传时，RTO值保持不变

    ii. tcp_retries_max表示最大重传次数，范围：取值是在0到64之间。

    0：使用系统配置的最大重传次数（默认15次）。

    非0：定制最大重传次数。

    最大重传次数需要大于sysctl选项net.ipv4.tcp_retries1（最小重传次数，默认3次）。

    iii. tcp_rto_max表示最大重传间隔时间，单位为秒，不允许设置小数，范围：取值是0或者在20到120之间。

    0：使用默认最大重传间隔时间（120秒）。

非0：定制最大重传间隔时间。

使用说明

- 开启TCP重传策略定制能力。通过如下命令，将net.ipv4.tcp_sock_retrans_policy_custom配置为1：
  ```
  [root@localhost ~]# sysctl -w net.ipv4.tcp_sock_retrans_policy_custom=1
  ```
  设置线性退避4次，最大重传10次，最大重传间隔时间20秒：
  ```
  tcp_sock_retrans_policy policy = {0};
  policy.tcp_linear_timeouts_times = 4;
  policy.tcp_retries_max = 10;
  policy.tcp_rto_max = 20;
  setsockopt(sockfd, SOL_TCP, TCP_SOCK_RETRANS_POLICY_CUSTOM, (const void*)&policy,
  sizeof(struct tcp_sock_retrans_policy));
  ```
- 恢复为默认策略：
  ```
  tcp_sock_retrans_policy policy = {0};
  setsockopt(sockfd, SOL_TCP, TCP_SOCK_RETRANS_POLICY_CUSTOM, (const void*)&policy,
  sizeof(struct tcp_sock_retrans_policy));
  ```
- 关闭TCP重传策略定制能力：
  ```
  [root@localhost ~]# sysctl -w net.ipv4.tcp_sock_retrans_policy_custom=0
  ```

⚠ 注意

定制TCP重传策略功能在HCE2.0版本内核进行修改，在HCE2.0版本glibc中使用该功能，需要在用户态代码中添加TCP_SOCK_RETRANS_POLICY_CUSTOM 宏以及对应的结构体tcp_sock_retrans_policy。

约束限制

1. 只针对ESTABLISHED状态的TCP链接生效。
2. 支持IPv4和IPv6，由于历史原因，TCP相关sysctl选项都挂在net.ipv4下，实际以net.ipv4.tcp打头的sysctl选项，对IPv4和IPv6均生效。
3. 应用级TCP重传策略优先级高于其他全局级TCP重传配置。
4. 定制TCP重传策略可能会增加系统负载，需要根据系统资源合理定制，建议定制的线性退避次数不超过6次。
5. 如果定制的最大重传次数小于net.ipv4.tcp_retries1（默认值3），则将无法触发网络探测，可能导致部分网络变更场景报文不通。
6. 如果定制的最大重传次数非0，则其必须大于等于线性退避次数，否则返回错误。
7. 如果只定制最大重传间隔时间，不定制最大重传次数，那么实际最大重传次数可能会大于net.ipv4.tcp_retries2（默认值15，描述最大重传次数或时间，下简称R2）。RFC6069规定，如果R2使用次数描述，则必须换算成时间。为避免使能本功能后TCP过早断连，该换算过程使用系统默认TCP_RTO_MAX（120s）而不是定制的最大重传时间，因此对外表现为实际重传次数多于R2。
8. 由于TCP-TLP算法，未收到ACK的尾包可能会被Loss Probe定时器选中做一次重传，用于后续触发快速重传。该次重传不计入本功能定制的重传次数内。

10.7 CPU 软绑定

背景说明

做NUMA亲和性绑定后，假如两个VM负载很高，但其他VM空闲，空闲的CPU无法被利用。如果不做绑定，则无法有效利用NUMA亲和性，尤其整机繁忙情况下，性能严重下降。无论哪种方式都有缺陷，所以提供了软绑定方案：

- 优先使用preferred CPU：当preferred CPU利用率低于阈值时，优先使用preferred CPU。
- 允许使用allowed CPU：当preferred CPU利用率超过阈值时，在所有allowed CPU内选核。
- 该软绑定方案同样适用于容器场景。

说明

- preferred CPU为软绑定的优先调度CPU。
- allowed CPU为通过sched_setaffinity或cgroup等设置的绑定的CPU列表。

为了能更直观地观测到软绑定的执行，增加了软绑定调度计数值；为了在不关机的情况下实现开启/关闭软绑定功能，增加了软绑定调度开关。

接口说明

- 软绑定功能开启时，cgroup组增加/sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_select_allowed_cpus计数值，统计cgroup组内的任务在唤醒选核时，选择共享核的次数。
- 软绑定功能开启时，cgroup组增加/sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_select_prefer_cpus计数值，统计cgroup组内的任务在唤醒选核时，选择优先核的次数。
- 软绑定功能开启时，cgroup组增加/sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_smoothed_prefer_cpus计数值，统计cgroup组内的任务在唤醒选核时，由于平滑算法没有从优先核飘到共享核的次数。
- 提供/proc/sys/kernel/sched_dynamic_affinity_disable接口，用于禁用软绑定功能。参数值为0时表示软绑定开启，参数值为1时表示禁用软绑定。

使用说明

软绑定的使用设置：

- 使用步骤
  a. 使用/proc/$PID/task/$TID/preferred_cpuset或者/sys/fs/cgroup/cpuset/下的cpuset.preferred_cpus为进程或cgroup组配置软绑定CPU列表。
  b. 通过/proc/sys/kernel/sched_util_low_pct设置preferred CPU利用率阈值，当preferred CPU利用率低于该阈值时，限制业务在preferred CPU中选核，否则在allowed CPU中选核。其中preferred CPU利用率有两种计算方式可选，详见步骤3。
  c. 通过/sys/kernel/debug/sched_features中开启或关闭DA_UTIL_TASKGROUP控制是基于taskgroup的preferred CPU利用率还是基于preferred CPU总利用率进行选核范围决策。

- 线程preferred_cpuset设置接口，用于配置软绑定CPUlist。cpulist是cpu逻辑编号的一个列表，以逗号隔开。比如CPU{1,3,5,6,7}的cpulist为“1,3,5,6,7”，其中连续的编号支持以范围的形式简写，比如“5,6,7”可简写为“5-7"。

  a. preferred_cpuset必须为allowed cpuset子集。

  b. 当preferred_cpuset未配置、配置为空或配置与allowed cpuset相同时，软绑定不生效。

  ```
  查看接口：
  cat /proc/$PID/task/$TID/preferred_cpuset
  给接口赋值示例：
  echo 5-7 > /proc/$PID/task/$PID/preferred_cpuset
  ```

- cgroup cpuset控制器下的每个目录，都会有cpuset.preferred_cpus设置接口，用于cgroup场景下软绑定设置。接口参数形式和线程preferred_cpuset设置一致。

  a. 当前cgroup cpuset.preferred_cpus 必须为 allowed cpu（即cpuset.cpus）子集。

  b. 当cpuset.preferred_cpus未配置、配置为空或配置与cpuset.cpus相同时，软绑定不生效。

  c. cpuset.preferred_cpus与父子cpuset.preferred_cpus都无约束关系。

  ```
  查看接口：
  cat /sys/fs/cgroup/cpuset/cpuset.preferred_cpus
  给接口赋值示例：
  echo 5-7 > /sys/fs/cgroup/cpuset/子cgroup/cpuset.preferred_cpus
  ```

- sched_util_low_pct设置接口，取值范围0-100（单位%），默认为85。参数0表示，不再受阈值限制，但选核时依然会优先考虑空闲的preferred CPU。参数100表示利用率超过preferred_cpus capacity时才会从cpuset.cpus中选核。

  ```
  查看接口：
  cat /proc/sys/kernel/sched_util_low_pct
  给接口赋值示例：
  echo 90 > /proc/sys/kernel/sched_util_low_pct
  ```

- DA_UTIL_TASKGROUP开关， /sys/kernel/debug/sched_features中新增DA_UTIL_TASKGROUP开关，默认开启。开启时，通过检测taskgroup在preferred_cpus中的利用率进行选核范围决策。关闭时，则通过检测preferred_cpus的总利用率（即包括非taskgroup进程在preferred_cpus的使用量）进行选核范围决策。

  ```
  开启：echo DA_UTIL_TASKGROUP > /sys/kernel/debug/sched_features
  关闭：echo NO_DA_UTIL_TASKGROUP > /sys/kernel/debug/sched_features
  ```

软绑定调度可视化和禁用开关的使用设置：

- 软绑定功能开启时，查看和清零接口：

  ```
  查看接口：
  cat /sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_select_allowed_cpus //选择共享核的次数
  cat /sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_select_prefer_cpus //选择优先核的次数
  cat /sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_smoothed_prefer_cpus //由于平滑算法没有从优先核飘到共享核的次数
  清零接口：
  echo 0 > /sys/fs/cgroup/cpuacct/子cgroup/cpuacct.nr_smoothed_prefer_cpus
  ```

- 对/proc/sys/kernel/sched_dynamic_affinity_disable接口，可以用cat的方式进行查看，也可以通过echo的方式修改其值。

  ```
  查看接口：cat /proc/sys/kernel/sched_dynamic_affinity_disable  // 接口值为0或1，为0时表示软绑定开启，为1时表示禁用软绑定
  接口赋值：echo 0 > /proc/sys/kernel/sched_dynamic_affinity_disable
  ```

- 在每个进程下新增了/proc/$pid/task/$pid/selected_cpuset接口，可以查看进程选定要运行的CPU范围。

说明

如果cgroup v1未配置cpu子组或目标进程不存在于cpu子组中，则无论开关是否开启，都通过检测preferred_cpus总利用率进行选核范围决策。

约束限制

1. 需要通过root用户权限使用，root用户具有系统最高权限，在使用root用户进行操作时，请严格按照操作指导进行操作，避免其他操作造成系统管理及安全风险。
2. 设置preferred_cpuset后，只有进程在唤醒或周期负载均衡时才会重新选核。
3. 软绑定在选择preferred CPU或allowed CPU时依赖pelt负载计算，1ms更新一次，所以如果负载频繁变化，会导致频繁选preferred CPU或allowed CPU。

11 XGPU 共享技术

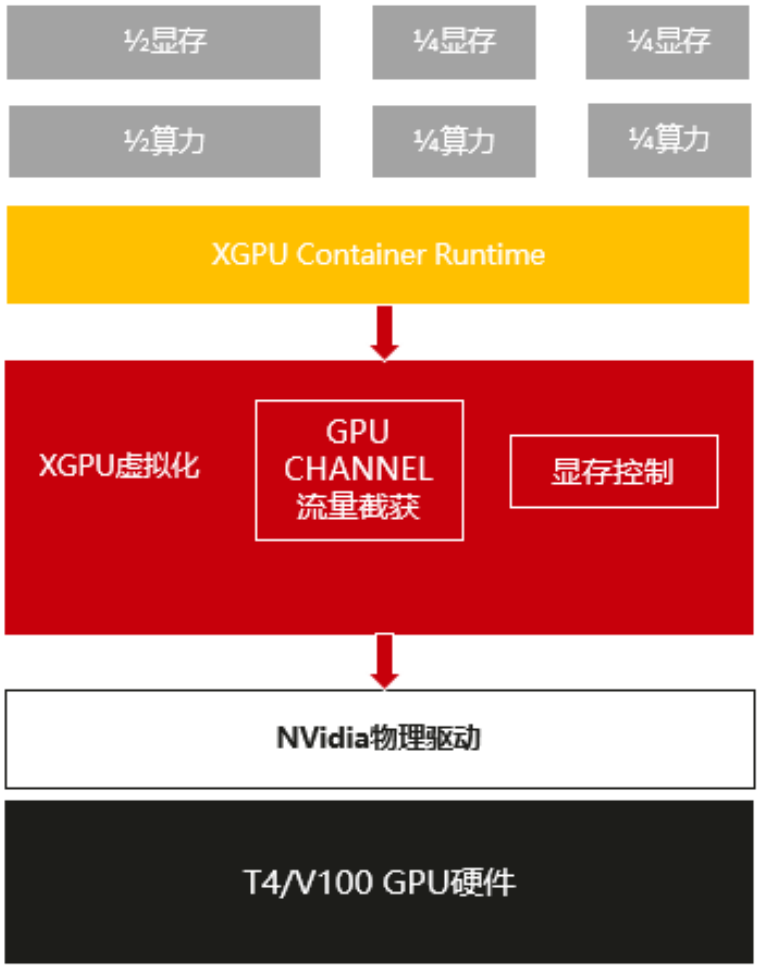
11.1 XGPU 共享技术概述

XGPU共享技术是华为云基于内核虚拟GPU开发的共享技术。XGPU服务可以隔离GPU资源，实现多个容器共用一张显卡，从而实现业务的安全隔离，提高GPU硬件资源的利用率并降低使用成本。

XGPU 共享技术架构

XGPU通过自研的内核驱动为容器提供虚拟的GPU设备，在保证性能的前提下隔离显存和算力，为充分利用GPU硬件资源进行训练和推理提供有效保障。您可以通过命令方便地配置容器内的虚拟GPU设备。

图 11-1 XGPU 共享技术架构图



产品优势

- 节约成本

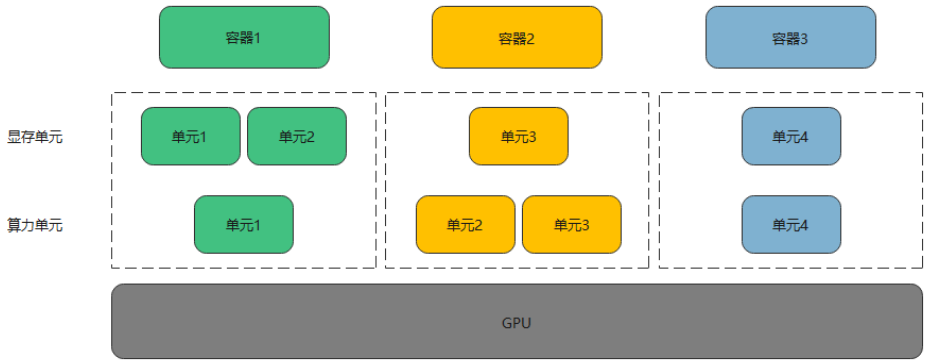
  随着显卡技术的不断发展，单张GPU卡的算力越来越强，同时价格也越来越高。但在很多的业务场景下，一个AI应用并不需要一整张的GPU卡。XGPU的出现让多个容器共享一张GPU卡，从而实现业务的安全隔离，提升GPU利用率，节约用户成本。

- 可灵活分配资源

  XGPU实现了物理GPU的资源任意划分，您可以按照不同比例灵活配置。

  - 支持按照显存和算力两个维度划分，您可以根据需要灵活分配。

图 11-2 GPU 资源分配



  - XGPU支持只隔离显存而不隔离算力的策略，同时也支持基于权重的算力分配策略。算力支持最小1%粒度的划分，推荐最小算力不低于4%。
- 兼容性好

  不仅适配标准的Docker和Containerd工作方式，而且兼容Kubernetes工作方式。
- 操作简单

  无需重编译AI应用，运行时无需替换CUDA库。

11.2 安装并使用 XGPU

本章节介绍如何安装和使用XGPU服务。

约束限制

- XGPU功能仅在Nvidia Tesla T4、V100、L2上支持。
- HCE内核版本为5.10及以上版本。
- XGPU功能支持cuda 12.2.0至12.8.0版本。
- GPU实例已安装535.54.03、535.216.03或570.86.15版本的NVIDIA驱动。
- GPU实例已安装18.09.0-300或更高版本的docker。
- 受GPU虚拟化技术的限制，XGPU的算力隔离功能在渲染场景无法使用，渲染场景请使用原生调度策略。
- 受GPU虚拟化技术的限制，容器内应用程序初始化时，通过nvidia-smi监测工具监测到的实时算力可能超过容器可用的算力上限。
- 当CUDA应用程序创建时，会在GPU卡上申请一小部分UVM显存（在Nvidia Tesla T4上大约为3 MiB），这部分显存属于管理开销，不受XGPU服务管控。
- 暂不支持同时在裸机环境以及该环境直通卡的虚拟机中同时使用。
- XGPU服务的隔离功能不支持以UVM的方式申请显存，即调用CUDA API cudaMallocManaged()，更多信息，请参见**NVIDIA官方文档**。请使用其他方式申请显存，例如调用cudaMalloc()等。

  说明

  - XGPU允许用户动态禁用UVM的方式申请显存，禁用方法参考uvm_disable接口说明。

安装 XGPU 服务

安装XGPU服务请联系技术支持。

推荐您通过云容器引擎服务使用XGPU虚拟化服务，相关操作请参见**GPU虚拟化**。

XGPU 服务使用示例

影响XGPU服务的环境变量如下表所示，您可以在创建容器时指定环境变量的值。容器引擎可以通过XGPU服务获得算力和显存。

表 11-1 影响 XGPU 服务的环境变量

环境变量名称	取值类型	说明	示例
GPU_IDX	Integer	指定容器可使用的GPU显卡。	为容器分第一张显卡： GPU_IDX=0
GPU_CONTAINER_MEM	Integer	设置容器内可使用的显存大小，单位 MiB。	为容器分配的显存大小为5120MiB： GPU_CONTAINER_MEM=5120
GPU_CONTAINER_QUOTA_PERCENT	Integer	指定显卡算力分配百分比。 算力支持最小1%粒度的划分，推荐最小算力不低于4%。	为容器分配50%的算力比例： GPU_CONTAINER_QUOTA_PERCEN=50
GPU_POLICY	Integer	指定GPU使用的算力隔离的策略。 • 0：不隔离算力，即原生调度。 • 1：固定算力调度。 • 2：平均调度。 • 3：抢占调度。 • 4：权重抢占调度。 • 5：混合调度。 • 6：权重弱调度。 算力隔离策略示例详见**XGPU算力调度示例**。	设置算力隔离策略为固定算力调度： GPU_POLICY=1
GPU_CONTAINER_PRIORITY	Integer	指定容器的优先级。 • 0：低优先级 • 1：高优先级	创建高优先级容器： GPU_CONTAINER_PRIORITY=1

NVIDIA Docker是一个可以使用GPU的docker，用于在容器中支持 NVIDIA GPU，下面以使用NVIDIA Docker创建两个容器为例，介绍XGPU服务的使用方法，数据规划如下。

表 11-2 数据规划

参数	容器1	容器2	说明
GPU_IDX	0	0	指定两个容器使用第一张显卡。
GPU_CONTAINER_QUOTA_PERCENT	50	30	为容器1分配50%算力，为容器2分配30%算力。
GPU_CONTAINER_MEM	5120	1024	为容器1分配5120MiB显存，为容器2分配1024MiB显存。
GPU_POLICY	1	1	设置第一张显卡使用固定算力调度策略。
GPU_CONTAINER_PRIORITY	1	0	指定容器1为高优先级容器，容器2为低优先级容器。

配置示例：

```
docker run --rm -it --runtime=nvidia -e GPU_CONTAINER_QUOTA_PERCENT=50 -e GPU_CONTAINER_MEM=5120 -e GPU_IDX=0 -e GPU_POLICY=1 -e GPU_CONTAINER_PRIORITY=1 --shm-size 16g -v /mnt/:/mnt nvcr.io/nvidia/tensorrt:19.07-py3 bash
docker run --rm -it --runtime=nvidia -e GPU_CONTAINER_QUOTA_PERCENT=30 -e GPU_CONTAINER_MEM=1024 -e GPU_IDX=0 -e GPU_POLICY=1 -e GPU_CONTAINER_PRIORITY=0 --shm-size 16g -v /mnt/:/mnt nvcr.io/nvidia/tensorrt:19.07-py3 bash
```

查看 procfs 节点

XGPU服务运行时会在/proc/xgpu下生成并自动管理多个procfs节点，您可以通过procfs节点查看和配置XGPU服务相关的信息。下面介绍各procfs节点的用途。

1. 执行以下命令，查看节点信息。

   ```
   ls /proc/xgpu/
   0 container version uvm_disable
   ```

   目录内容说明如下表所示：

   表 11-3 目录内容说明

目录	读写类型	说明
0	读写	XGPU服务会针对GPU实例中的每张显卡生成一个的目录，并使用数字作为目录名称，例如0、1、2。本示例中只有一张显卡，对应的目录ID为0。
container	读写	XGPU服务会针对运行在GPU实例中的每个容器生成一个的目录。
version	只读	XGPU的版本。

目录	读写类型	说明
uvm_disable	读写	是否禁用UVM的方式申请显存，全局粒度，默认值为0。 ● 0：不禁用 ● 1：禁用

2. 执行以下命令，查看第一张显卡对应的目录内容。

```
ls /proc/xgpu/0/
max_inst meminfo policy quota utilization_line utilization_rate xgpu1 xgpu2
```

目录内容说明如下表所示：

表 11-4 目录内容说明

目录	读写类型	说明
max_inst	读写	用于设置容器的最大数量，取值范围为1~25。仅在没有容器运行的情况下可修改。
meminfo	只读	此显卡总共可用的显存大小。
policy	读写	指定GPU使用的算力隔离的策略，默认值为1。 ● 0：不隔离算力，即原生调度。 ● 1：固定算力调度。 ● 2：平均调度。 ● 3：抢占调度。 ● 4：权重抢占调度。 ● 5：混合调度。 ● 6：权重弱调度。 算力隔离策略示例详见**XGPU算力调度示例**。
quota	只读	算力总权重。
utilization_line	读写	在离线混部的算力压制水位线。 当GPU整卡利用率超过该值时，在线容器完全压制离线容器，否则在线容器部分压制离线容器。
utilization_rate	只读	GPU整卡利用率。
xgpuIndex	读写	属于此显卡的xgpu子目录。 本示例中，属于第1张显卡的xgpu为xgpu1和xgpu2

3. 执行以下命令，查看container目录内容。

```
ls /proc/xgpu/container/
9418 9582
```

目录内容说明如下表所示：

表 11-5 目录内容说明

目录	读写类型	说明
containerID	读写	容器的ID。 使用XGPU创建容器的时候分配的ID，每个容器对应一个ID。

4. 执行以下命令，查看containerID目录内容。

```
ls /proc/xgpu/container/9418/
xgpu1 uvm_disable
ls /proc/xgpu/container/9582/
xgpu2 uvm_disable
```

目录内容说明如下表所示：

表 11-6 目录内容说明

目录	读写类型	说明
xgpuIndex	读写	属于此容器的xgpu子目录。 本示例中，属于容器9418的xgpu为xgpu1，属于容器9582的xgpu为xgpu2。
uvm_disable	读写	是否禁用UVM的方式申请显存，容器粒度，默认值为0。 ● 0：不禁用 ● 1：禁用

5. 执行以下命令，查看xgpuIndex目录内容。

```
ls /proc/xgpu/container/9418/xgpu1/
meminfo priority quota
```

目录内容说明如下表所示：

表 11-7 目录内容说明

目录	读写类型	说明
meminfo	只读	此XGPU分配的可见显存大小和当前剩余可用显存大小。 如3308MiB/5120MiB, 64% free，指分配了5120MiB，剩余64%可使用。
priority	读写	用于设置容器的优先级，默认值为0。 ● 0：低优先级 ● 1：高优先级 该功能用于在线离线混合使用场景，高优先级容器可以抢占低优先级容器的算力。

目录	读写类型	说明
quota	只读	此XGPU分配的算力百分比。 如50，指此XGPU分配了显卡50%的算力。

了解procfs节点的用途后，您可以在GPU实例中执行命令进行切换调度策略、查看权重等操作，示例命令如下表所示。

表 11-8 示例命令

命令	效果
echo 1 > /proc/xgpu/0/policy	修改第一张显卡的调度策略为权重调度。
cat /proc/xgpu/container/$containerID/$xgpuIndex/meminfo	查看指定容器里xgpu分配的显存大小。
cat /proc/xgpu/container/$containerID/$xgpuIndex/quota	查看指定容器里xgpu分配的算力权重。
cat /proc/xgpu/0/quota	查看第一张显卡上剩余可用的算力权重。
echo 20 > /proc/xgpu/0/max_inst	设置第一张显卡最多可以创建20个容器。
echo 1 > /proc/xgpu/container/$containerID/$xgpuIndex/priority	设置指定容器里xgpu的优先级为高优先级。
echo 40 > /proc/xgpu/0/utilization_line	设置第一张显卡的在离线混部算力压制水位线为40%。
echo 1 > /proc/xgpu/container/$containerID/uvm_disable	设置指定容器里xgpu禁用UVM的方式申请显存

升级 XGPU 服务

XGPU服务采用冷升级的方式。

1. 关闭所有运行中的容器。
   docker ps -q | xargs -I {} docker stop {}
2. 升级xgpu的RPM包。
   rpm -U hce_xgpu

卸载 XGPU 服务

1. 关闭所有运行中的容器。
   docker ps -q | xargs -I {} docker stop {}

2. 卸载xgpu的RPM包。

   rpm -e hce_xgpu

通过 xgpu-smi 工具监控容器

您可以通过xgpu-smi工具查看XGPU容器的相关信息，包括容器ID、算力使用、分配情况、显存使用及分配情况等。

xgpu-smi的监控展示信息如下所示：

图 11-3 xgpu-smi 的监控展示信息

```
+=============================================================================+
|  HUAWEI CLOUD XGPU-SMI                          XGPU Version: 1.0           |
|=============================================================================|
|   Container-Id   |  GPU   |   GPU-Util/Limit   |   GPU-Memory-Usage/Limit   |
+------------------+--------+--------------------+----------------------------+
|     800a09ae74bc |      1 |           0% / 30% |                 0M / 2000M |
|     b8402c9316e4 |      0 |           0% / 10% |                 0M / 6000M |
+-----------------------------------------------------------------------------+
```

您可使用**xgpu-smi -h**命令查看xgpu-smi工具的帮助信息。

图 11-4 xgpu-smi 工具的帮助信息

```
[root@localhost ~]#
[root@localhost ~]# xgpu-smi -h
xgpu-smi provides monitoring information about xgpu containers.
The data is presented in either a plain text or a json format, via stdout or a file.

Usage:
  xgpu-smi [flags]
  xgpu-smi [command]

Available Commands:
  clean       Release XGPU
  list        List XGPU containers and used GPUs
  query       Show XGPU containers' information and GPUs' available resource information in JSON format

Flags:
  -c, --container string                filter the specific containers' information
  -f, --filename string                 log to a specified file, rather than to stdout
  -g, --gpu string                      filter the specific GPUs' information
  -h, --help                            help for xgpu-smi
  -r, --remote-runtime-endpoint string  endpoint of remote runtime endpoint (default "unix:///var/run/containerd
/containerd.sock")

Use "xgpu-smi [command] --help" for more information about a command.
[root@localhost ~]#
[root@localhost ~]#
```

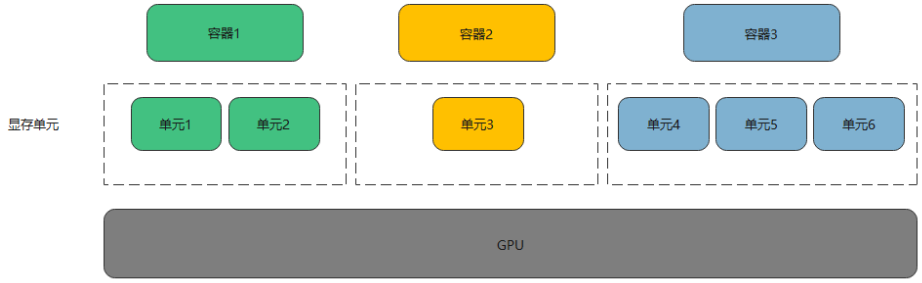
11.3 XGPU 算力调度示例

当使用XGPU服务创建XGPU时，XGPU服务会按照最大容器数量（**max_inst**）为每张显卡设置时间片（X ms）用于为容器分配GPU算力，以单元1、单元2…单元N表示。本节max_inst以20为例，介绍使用不同调度策略时对算力的调度示例。

原生调度（policy=0）

原生调度表示使用NVIDIA GPU本身的算力调度方式。在原生调度策略下XGPU只用来做显存的隔离。

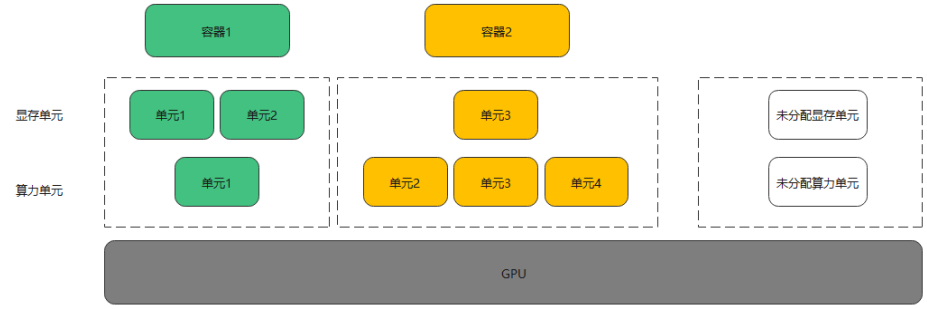
图 11-5 原生调度



固定算力调度（policy=1）

固定算力调度表示以固定的算力百分比为容器分配算力。例如为容器1和容器2分别分配5%和15%的算力，如下图所示。

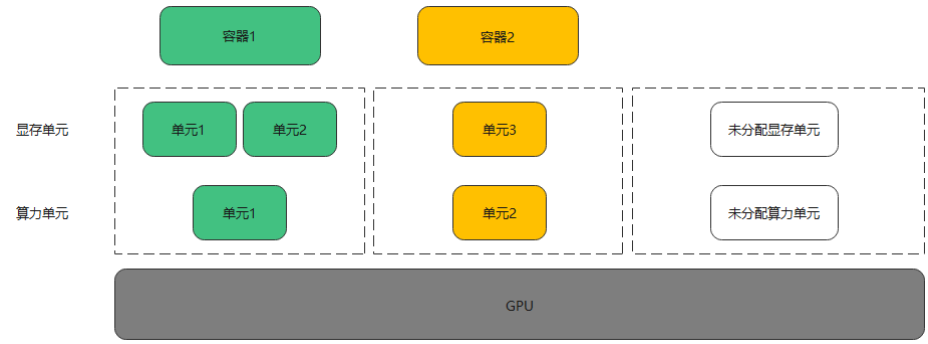
图 11-6 固定算力调度



平均调度（policy=2）

平均调度表示每个容器固定获得1/max_inst的算力。以max_inst=20为例，每个容器固定获得1/max_inst，即5%的算力，如下图所示。

图 11-7 平均调度

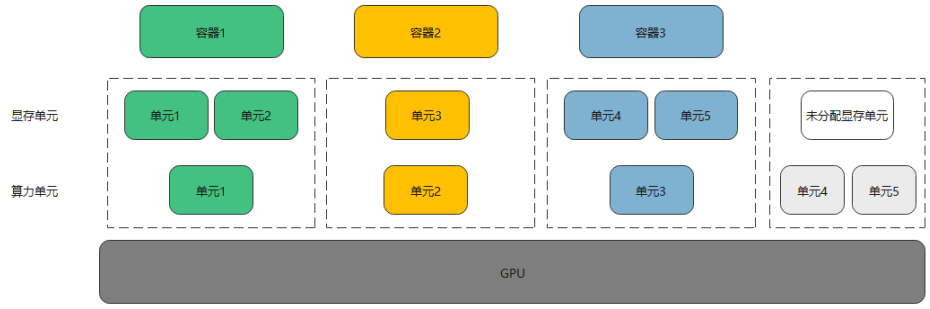


抢占调度（policy=3）

抢占调度表示每个容器固定获得1个时间片，XGPU服务会从算力单元1开始调度。但如果某个算力单元没有分配给某个容器，或者容器内没有进程打开GPU设备，则跳过调度切换到下一个时间片。图中灰色部分的算力单元表示被跳过不参与调度。

本例中容器1、2、3占用的实际算力百分比均为33.33%。

图 11-8 抢占调度

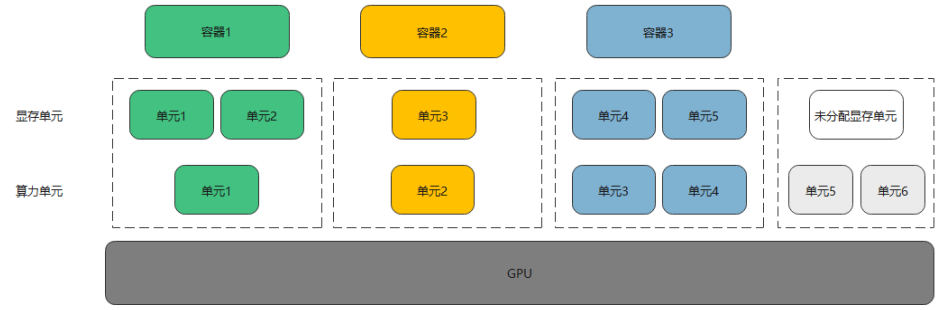


权重抢占调度（policy=4）

权重抢占调度表示按照每个容器的算力比例为容器分配时间片。XGPU服务会从算力单元1开始调度，但如果某个算力单元没有分配给某个容器，则跳过调度切换到下一个时间片。例如为容器1、2、3分别分配5%、5%、10%的算力，则容器1、2、3分别占用1、1、2个算力单元。图中灰色部分的算力单元表示被跳过不参与调度。

本例中容器1、2、3占用的实际算力百分比为25%、25%、50%。

图 11-9 权重抢占调度

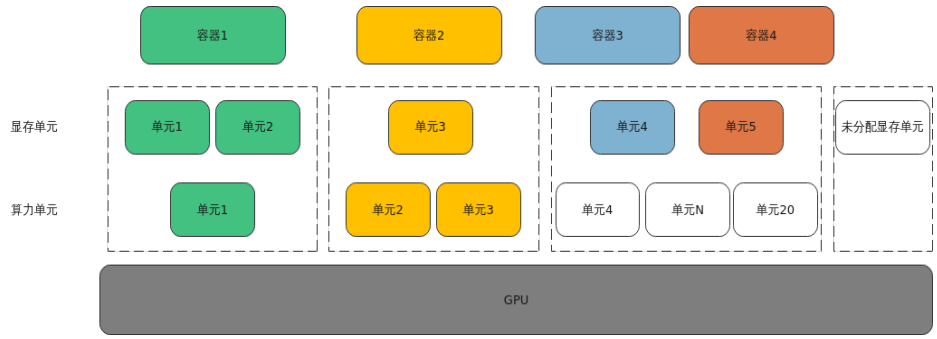


混合调度（policy=5）

混合调度表示单张GPU卡支持单显存隔离和算力显存隔离类型。其中算力显存隔离的容器其隔离效果同固定算力（policy=1）完全一致，单显存隔离的容器共享算力显存隔离的容器分配后剩余的GPU算力。以max_inst=20为例，容器1、2为算力显存隔离容器，其分配的算力分别为5%、10%，容器3、4为单显存隔离的容器，则容器1、2分别占用1、2个算力单元，容器3、4共享剩余17个算力单元。此外，当容器2中没有进程打开GPU设备时，则容器1、2分别占用1、0个算力单元，容器3、4共享剩余19个算力单元。

在混合调度下，根据GPU_CONTAINER_QUOTA_PERCENT是否为0来区分容器是否开启算力隔离，GPU_CONTAINER_QUOTA_PERCENT为0的所有容器共享GPU的空闲算力。

图 11-10 混合调度



说明

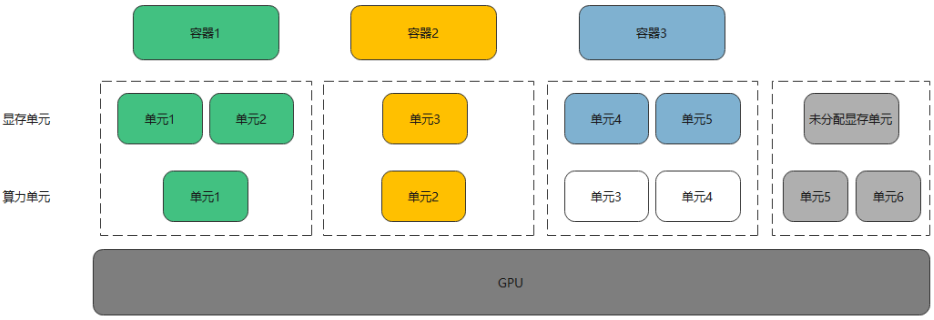
混合调度策略不支持高优先级容器。

权重弱调度（policy=6）

权重弱调度表示按照每个容器的算力比例为容器分配时间片，隔离性弱于权重抢占调度。XGPU服务会从算力单元1开始调度，但如果某个算力单元没有分配给某个容器，或者容器内没有进程打开GPU设备，则跳过调度切换到下一个时间片。例如为容器1、2、3分别分配5%、5%、10%的算力，则容器1、2、3分别占用1、1、2个算力单元。图中白色部分的算力单元表示容器3的空闲算力，图中白色部分和灰色部分的算力单元表示被跳过不参与调度。

本例中容器1、2、3占用的实际算力百分比为50%、50%、0%。

图 11-11 权重弱调度



说明

权重弱调度涉及空闲算力的抢占和抢回，因此容器在空闲和忙碌之间切换时会影响其他容器的算力，该算力波动属于正常情况。当某个容器从空闲切换到忙碌时，其抢回算力的时延不超过100ms。

12 HCE 的 REPO 源配置

HCE采用RPM包形式管理软件，并且提供了与系统配套的官方REPO源来发布软件包及其更新。您可通过dnf/yum命令实现常见的软件管理功能，包括安装、升级、卸载等。

官方 repo 源

HCE镜像，在**/etc/yum.repos.d/hce.repo**文件中会默认配置官方repo源。以**HCE 2.0版本**为例，其内容如下：

```
[base]
name=HCE $releasever base
baseurl=https://repo.huaweicloud.com/hce/$releasever/os/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/os/RPM-GPG-KEY-HCE-2

[updates]
name=HCE $releasever updates
baseurl=https://repo.huaweicloud.com/hce/$releasever/updates/$basearch/
enabled=1
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/updates/RPM-GPG-KEY-HCE-2

[debuginfo]
name=HCE $releasever debuginfo
baseurl=https://repo.huaweicloud.com/hce/$releasever/debuginfo/$basearch/
enabled=0
gpgcheck=1
gpgkey=https://repo.huaweicloud.com/hce/$releasever/debuginfo/RPM-GPG-KEY-HCE-2
```

其中各字段含义如下：

- name：对repo源的描述。
- baseurl：仓库所在的服务器地址，支持http://、ftp://、file://三种格式。
- enabled：是否启用该软件仓库，1表示启用，0表示禁用。
- gpgcheck：是否进行gpg校验，1表示启用校验，0表示禁用校验。
- gpgkey：公钥保存的地址，用于gpg校验。

⚠ 注意

修改该文件可能会对系统的软件安装、升级产生影响，不建议修改该文件。

第三方 repo 源配置

HCE 2.0如果要新增第三方repo源，可按下述过程进行配置（以openEuler社区的镜像源为例）：

1. 在**/etc/yum.repos.d/**目录新增**openEuler.repo**文件（名称可以自定义，文件后缀需以.repo结尾）。使用**vim /etc/yum.repos.d/openEuler.repo**命令进行编辑。
2. 配置仓库名字，如openEuler-everything，仓库名必须唯一，可根据实际情况进行调整。
3. 配置name选项，如openEuler everything repository，表示仓库的具体描述，可根据实际情况进行调整。
4. 配置baseurl选项，此处为：**https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64**，表示软件包从该链接获取，具体可参考openEuler或者对应repo提供者的官方说明。
5. 配置gpgcheck选项，为1表示对安装的软件包进行gpg校验。
6. 配置enabled选项，为1表示启用该repo源。
7. 配置gpgkey选项，此处为：**https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64/RPM-GPG-KEY-openEuler**，表示gpg校验使用的公钥来源于该链接。

按照上述方法，再添加上openEuler update repo源，最终**openEuler.repo**文件效果如下：

```
[openEuler-everything]
name=openEuler everything repository
baseurl=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64
gpgcheck=1
enabled=1
priority=3
gpgkey=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64/RPM-GPG-KEY-openEuler
[openEuler-update]
name=openEuler update repository
baseurl=https://repo.openeuler.org/openEuler-22.03-LTS/update/x86_64/
gpgcheck=1
enabled=1
priority=3
gpgkey=https://repo.openeuler.org/openEuler-22.03-LTS/everything/x86_64/RPM-GPG-KEY-openEuler
```

须知

可以通过配置中的priority字段控制repo源的优先级。如果优先使用HCE默认源，可在hce.repo配置中都加上priority=1（数值越小优先级越高），然后在第三方源配置中加上priority=2，数值根据实际情况进行调整。

说明

如果要升级软件包，可参考用户指南**更新HCE系统和RPM包**。

yum/dnf 常见使用方式

HCE 1.1仅支持通过yum命令进行软件管理相关操作，HCE 2.0同时支持yum与dnf命令。常用的软件管理相关的命令如下：

功能	yum命令	dnf命令	示例
安装软件包	yum install <软件包>	dnf install <软件包>	安装gcc：yum install gcc
卸载软件包	yum remove <软件包>	dnf remove <软件包>	卸载gcc：yum remove gcc
列出已安装的软件包	yum list installed	dnf list installed	列出系统所有的包：yum list installed
搜索软件包	yum search <软件包>	dnf search <软件包>	在repo源中搜索gcc包：yum search gcc
查询软件包信息	yum info <软件包>	dnf info <软件包>	查询gcc软件包信息：yum info gcc

13 HCE 定制内核参数说明

HCE 2.0相较CentOS 8存在部分定制内核参数，具体说明及用途如下。

kernel 参数

以下是所属文件在/proc/sys/kernel和/sys/kernel目录下的参数：

扫描任务

自动NUMA平衡会扫描任务的地址空间并取消页面映射，以检测页面放置是否正确，或者数据是否应迁移到靠近任务运行的内存节点。每次“扫描延迟”，任务会扫描其地址空间中的下一个“扫描大小”数量的页面。当达到地址空间的末尾时，扫描器会从头开始。

“扫描延迟”和“扫描大小”共同决定了扫描速率。当“扫描延迟”减少时，扫描速率增加。因此“扫描延迟”和每个任务的扫描速率是自适应的，取决于历史行为。如果页面放置正确，则扫描延迟增加；否则，扫描延迟减少。“扫描大小”不是自适应的，但“扫描大小”越大，扫描速率越高。

更高的扫描速率会带来更高的系统开销，因为必须捕获页面错误，并迁移数据。然而，扫描速率越高，任务的内存迁移到本地节点的速度越快，如果工作负载模式发生变化，可以最小化由于远程内存访问带来的性能影响。表13-1中的参数控制扫描延迟和扫描页面数量的阈值。

表 13-1 扫描参数说明

参数	说明	取值
kernel.numa_balancing_scan_delay_ms	任务初次启动时的扫描延迟。	默认值1000（单位：毫秒）
kernel.numa_balancing_scan_period_max_ms	控制扫描任务的最长时间间隔，从而控制最小扫描频率。	默认值60000（单位：毫秒）
kernel.numa_balancing_scan_period_min_ms	控制扫描任务的最短时间间隔，从而控制最大扫描频率。	默认值1000（单位：毫秒）

参数	说明	取值
kernel.numa_balancing_scan_size_mb	定义每次扫描的页面大小。	默认值256（单位：MB）

CFS调度器

CFS（完全公平调度器）采用纳秒级的精度进行会计，不依赖于任何jiffies或其他HZ相关的细节。因此，CFS调度器不再像之前的调度器那样具有“时间片”概念，也不包含任何启发式算法。唯一的可调参数（需要启用 CONFIG_SCHED_DEBUG）是：

```
/proc/sys/kernel/sched_min_granularity_ns
```

由于其设计，CFS调度器不易受到针对传统调度器启发式算法的现有“攻击”，如fiftyp.c、thud.c、chew.c、ring-test.c和massive_intr.c等测试均能正常执行，不会影响交互性。

CFS调度器在处理优先级（nice levels）和SCHED_BATCH上的表现显著优于之前的传统调度器：这两类工作负载的隔离性得到了更为积极的增强。对称多处理（SMP）负载均衡的实现经过了重构和清理：负载均衡代码中不再包含运行队列遍历的假设，现采用调度模块的迭代器。由此，负载均衡代码变得更加简洁。

表 13-2 CFS sched_min_granularity_ns 参数说明

参数	描述	取值
kernel.sched_min_granularity_ns	该参数用于将调度器调整为“桌面”模式（即低延迟）或“服务器”模式（即良好的批处理）工作负载。默认设置适合桌面工作负载。SCHED_BATCH也由CFS调度模块处理。	默认值3000000（单位：纳秒）

压力场景定位开关

表 13-3 net_res_debug_enhance 参数说明

参数	描述	取值
kernel.net_res_debug_enhance	在海量收发报文等压力场景下，内核网络协议栈可能会发生资源不足或超限，从而导致用户态socket、send接口返回失败或者网络丢包等故障，**net_res_debug_enhance**开启的情况下会在该类异常场景下打印定位信息至系统日志中。 1表示启用，0表示不启用。	默认值0（不启用）

oom事件故障定位

产品/平台业务软件或操作系统本身可能因为某种特殊原因导致内存空间不足，触发oom事件。kbox能够将oom事件发生的时间、发生oom进程信息、系统进程信息等异常信息记录到存储设备中，方便维护人员定位。

表 13-4 信息控制参数说明

参数	描述	取值
kernel.oom_enhance_enable	控制是否启用oom信息打印功能，1表示启用，0表示不启用。	默认值1（启用）
kernel.oom_print_file_info	控制是否打印文件系统信息，1表示启用，0表示不启用。	默认值1（启用）
kernel.oom_show_file_num_in_dir	控制打印的文件系统信息中文件的数量。	默认值10

SMT驱离特性

表 13-5 SMT 驱离特性参数说明

参数	描述	取值
kernel.qos_offline_wait_interval_ms	离线任务在超负载情况下返回用户态时，每次睡眠的时间。	取值范围[100-1000]，默认值100（单位：毫秒）

参数	描述	取值
kernel.qos_overload_detect_period_ms	非离线任务持续占有CPU超过此时间会触发优先级反转解决流程。	取值范围[100-100000]，默认值5000（单位：毫秒）

openEuler新增内核参数，详情请参见**openEuler技术白皮书第五章节——SMT 驱离防止优先级反转特性**。

算力统计

同一节点内睿频、调频、SMT、大小核等因素导致从cpuacct子系统统计的CPU利用率不能真实反映使用了多少算力，节点间CPU利用率所代表的算力没有可比性（差异达到30%+）。算力统计的目标是基于CPU真实算力利用率进行统计，解决当前CPU利用率（占空比）不能真实反映算力利用的问题，基于真实算力推导的cgroup级别的CPU利用率，相对占空比的利用率更能表征业务性能指标。现阶段算力统计主要考虑SMT影响。

表 13-6 算力统计参数说明

参数	描述	取值
kernel.normalize_capacity.sched_normalize_util	动态开启CPU利用率归一化功能，0关闭，1开启。	默认值0
kernel.normalize_capacity.sched_single_task_factor	在超线程的场景，设置逻辑核单独运行算力系数。取值1-100，数值越大，说明逻辑核的算力越大。	默认值100
kernel.normalize_capacity.sched_multi_task_factor	在超线程的场景，设置逻辑核并行运行算力系数。取值1-100，数值越大说明逻辑核的算力越大。	默认值60
kernel.normalize_capacity.sched_normalize_adjust	可读可写，动态开启算力补偿总开关，0关闭，1开启。	默认值0

preferred CPU利用率

表 13-7 sched_util_low_pct 参数说明

参数	描述	取值
kernel.sched_util_low_pct	preferred CPU利用率阈值，当preferred CPU利用率低于该阈值时，限制业务在preferred CPU中选核，否则在allowed CPU中选核。	默认值85

软件狗检测周期

HCE在内核watchdog基础上，增加LOCKUP情况下错误重置软件狗导致watchdog失效的场景检测功能，记录或回显有效信息，协助开发运维快速定位问题。软件狗检测周期增加了sysctl参数设置，可根据产品诉求，动态调整检测和告警日志打印周期。

表 13-8 软件狗检测周期参数说明

参数	描述	取值
kernel.watchdog_enhance_enable	控制watchdog增强的所有功能，0表示关闭，其他值表示开启。建议配置0或者1。	默认值1（开启）
kernel.watchdog_softlockup_divide	软件狗检测周期调整参数，**kernel.watchdog_thresh** * 2 / **kernel.watchdog_softlockup_divide**后的值为软件狗检测周期。	取值范围[1, 60]，默认值5（单位：秒）
kernel.watchdog_print_period	设置软件狗检测到进程不调度后，打印进程信息的时间间隔。	取值范围[1, 60]，默认值10（单位：秒）

cpu qos接口兼容开关

表 13-9 sched_qos_level_0_has_smt_expell 参数说明

参数	描述	取值
kernel.sched_qos_level_0_has_smt_expell	HCE+CCE混部2级扩展到5级后的cpu qos接口兼容开关，0表示关闭，即启用5级cpu qos状态。 当cce遇到需要保持兼容的场景时，可开启kernel.sched_qos_level_0_has_smt_expell开关，将开关置1，使0优先级保持原来在、离线混部时的语义。	默认值0

通过内核参数动态调整hz

表 13-10 actual_hz 参数说明

参数	描述	取值
kernel.actual_hz	系统启动后可通过内核参数动态调整hz的能力，为0时表示关闭hz调整特性，此时将使用内核默认的 1000 HZ。	取值范围[0, 1000]，默认值0

内核调度策略

表 13-11 内核调度策略参数说明

参数	描述	取值
kernel.sched_latency_ns	**sched_latency_ns**是一个用于定义目标抢占延迟时间的变量。它指定了在这个时间周期内，调度器会至少一次调度所有运行队列中的任务。	默认值24000000（单位：纳秒）
kernel.sched_migration_cost_ns	**sched_migration_cost_ns**参数指定以纳秒为单位的时间间隔。任务最后一次执行后，CPU 缓存被视为具有有用内容，直到此间隔过期为止。增加这个间隔会导致任务迁移减少。	默认值500000（单位：纳秒）

参数	描述	取值
kernel.sched_nr_migrate	如果一个SCHED_OTHER任务产生了大量其他任务，它们将全部在同一CPU上运行。迁移任务或softirq将尝试平衡这些任务，以便它们可以在空闲的CPU上运行。**sched_nr_migrate**选项可以设置为指定一次移动的任务数。	默认值32
kernel.sched_tunable_scaling	控制调度程序是否可以根据在线CPU数量自动调整sched_min_granularity_ns/sched_latency_ns/sched_wakeup_granularity_ns的值。	0：不调整 1：对数调整（以 2 为底对数进行调整） 2：线性调整（线性比例调整） 默认值1
kernel.sched_wakeup_granularity_ns	调度程序唤醒间隔时间。	默认值4000000（单位：纳秒）
kernel.sched_autogroup_enabled	是否开启**autogroup**调度特性，默认关闭。 当开启**autogroup**调度特性时，一个**autogroup**中的所有成员都属于同一个内核调度器"任务组"。CFS调度器使用了在任务组间均衡分配CPU时钟周期的算法。	0：关闭 1：开启 默认值0

潮汐调度功能

表 13-12 禁用潮汐调度功能开关

参数	描述	取值
kernel.sched_dynamic_affinity_disable	禁用潮汐调度功能。参数值为0时表示潮汐调度开启，参数值为1时表示禁用禁用潮汐调度。	默认值0

CPU QoS 优先级负载均衡特性

表 13-13 sched_prio_load_balance_enabled 参数说明

参数	描述	取值
kernel.sched_prio_load_balance_enabled	是否开启CPU QoS优先级负载均衡，0表示关闭，1表示开启。	取值范围{0,1}，默认值0

openEuler新增内核参数，请参考**openEuler技术白皮书第五章节——CPU QoS 优先级负载均衡特性**。

支持核挂死检测特性

CPU核挂死问题比较特殊，它出现时，CPU核无法执行任何指令，也不响应中断，故内核检测无法覆盖此场景，需要芯片使用仿真器来定位根因。为提升此类问题定位的效率，内核开发了核挂死检测特性，此特性可快速定性是否出现核挂死问题。

表 13-14 核挂死检测阈值

参数	描述	取值
kernel.corelockup_thresh	设置此阈值为x，当某个CPU连续x次收不到hrtimer和NMI中断时，就判定该CPU出现了核挂死。	默认值5

控制idle polling执行参数

表 13-15 kernel.halt_poll_threshold 参数说明

参数	描述	取值
kernel.halt_poll_threshold	开启此特性时，GuestOS kernel在进入idle时，会在先进行idle polling一段时间，而不发生VM-exit。此参数控制idle polling执行的时间，在idle polling过程中发生任务调度可以避免IPI中断开销。	默认值：0 取值范围：0-max(uint64)

用户态穿越内核UCE不复位

表 13-16 kernel.machine_check_safe 参数说明

参数	描述	取值
kernel.machine_check_safe	确保内核开启config开关CONFIG_ARCH_HAS_COPY_MC，/proc/sys/kernel/machine_check_safe值为1时代表全场景使能，改为0代表不使能，其他值均为非法。	默认值1 取值范围{0,1}

支持网络数据包校验错误时打印源 IP 地址

当发生硬件错误或者网络遭受攻击时，内核会收到检验错误的网络数据包并直接抛弃，无法定位数据包来源。本特性提供如上场景下的可定位能力：校验失败后，在系统日志中打印该数据包的源IP地址。

表 13-17 kernel.net_csum_debug 参数说明

参数	描述	取值
kernel.net_csum_debug	配置该参数值为1时开启此特性，为0时关闭此特性 该参数仅存在于arm环境。	默认值0

集群调度

表 13-18 集群调度参数说明

参数	描述	取值
kernel.sched_cluster	值为1时开启集群调度，为0时关闭集群调度。	默认值1

net 参数

以下是所属文件在/proc/sys/net目录下的参数：

TCP套接字缓存控制

表 13-19 net.ipv4.tcp_rx_skb_cache 参数说明

参数	描述	取值
net.ipv4.tcp_rx_skb_cache	控制一个skb的每个TCP套接字缓存，这可能有助于改善某些工作负载的性能。这在具有大量TCP套接字的系统上可能是危险的，因为它会增加内存使用。	默认值0（不启用）

网络命名空间控制

表 13-20 netfilter.nf_namespace_change_enable 参数说明

参数	描述	取值
net.netfilter.nf_namespace_change_enable	如果此选项设置为0，则在非初始化网络命名空间中，命名空间为只读状态。	默认值0

查询VF信息显示广播地址信息

表 13-21 查询 VF 信息显示广播地址信息开关

参数	描述	取值
net.core.vf_attr_mask	用于控制netlink消息查询VF信息时（如“ip linkshow”命令）是否显示广播地址信息。 默认值1，显示广播地址信息，与社区保持一致。	默认值1

定制 TCP 重传策略

HCE的TCP报文重传遵循指数退避原则，在弱网场景下包到达率较低，时延较高。因此新增通过编程接口定制TCP重传策略的能力，包括定制线性退避次数、最大重传次数、最大重传间隔时间，以优化弱网场景的包到达率和时延。

表 13-22 定制 TCP 重传策略开关

参数	描述	取值
net.ipv4.tcp_sock_retrans_policy_custom	用于控制定制TCP重传策略功能是否开启。0：关闭定制TCP重传策略功能。1：开启定制TCP重传策略功能。	默认值0

IPv6 repath 拥塞换路

云DCN网络存在丰富的多路径， 现有云DCN网络采用ECMP方式进行负荷分担来减少冲突。在动态突发和流大小不一致的情况下，依旧会出现负载不均和拥塞热点。

IPv6 Re-path方案通过端侧拥塞检测与切换路径来实现网络负载均衡，提升业务流吞吐，降低传输时延。发送端动态感知网络拥塞状态，评估是否需要进行路径切换，如果需要，则执行Re-path换路，调整到轻载路径，从而避免网络拥塞“热点”。

表 13-23 IPv6 repath 拥塞换路内核参数说明

参数	描述	取值
net.ipv6.tcp_repath_cong_thresh	空闲时指定的拥塞轮次数目，若连续拥塞轮次超过该值则执行换路	取值范围：[1-8192] 默认值：10
net.ipv6.tcp_repath_enabled	特性开关（值为0表示特性不使能，值为1表示特性使能）	取值范围：{0,1} 默认值：0
net.ipv6.tcp_repath_idle_rehash_rounds	非空闲时指定的拥塞轮次数目，若连续拥塞轮次超过该值则执行换路	取值范围：[3-31] 默认值：3
net.ipv6.tcp_repath_rehash_rounds	控制调整一个轮次内丢失报文所占的比例的阈值，超过阈值则判断本轮次拥塞	取值范围：[3-31] 默认值：3
net.ipv6.tcp_repath_times_limit	每秒钟允许的最大换路次数	取值范围：[1-10] 默认值：2

网络PPS性能调优参数

当大规格虚拟机（例如192U的虚机）采用HCE 5.10内核时，在高并发（并发数大于8）场景下，可以使用net.core.high_order_alloc_disable参数提升接收网络PPS性能。

表 13-24 网络 PPS 性能调优参数

参数名称	参数描述	临时开启方法	永久开启方法
net.core.high_order_alloc_disable	默认情况下网络内存尝试使用高阶页面，该参数开启后选择order0页面分配。 默认值：0	执行命令**sysctl -w net.core.high_order_alloc_disable=1** 重启后该参数将变回默认值。	1. 将**net.core.high_order_alloc_disable=1**配置到/etc/sysctl.conf文件。 2. 执行sysctl -p命令立即生效。 重启后该参数值不变。

说明

该参数开启选择order0页面分配，建议tcp、udp业务高并发业务场景下(并发数大于8)开启此选项，对性能可能有较大提升，低并发场景不建议开启此参数，可能对性能产生劣化。

vm 参数

以下是所属文件在/proc/sys/vm目录下的参数：

周期性回收

表 13-25 周期性回收参数说明

参数	描述	取值
vm.cache_reclaim_s	**cache_reclaim_s**用于设置周期性内存回收的回收间隔。当启用周期性内存回收时，它会每**cache_reclaim_s**秒回收一次内存。	默认值0

参数	描述	取值
vm.cache_reclaim_weight	**cache_reclaim_weight**是每次周期性回收中的回收因子。当启用周期性内存回收时，每次回收的回收量可以通过以下公式计算： `reclaim_amount = cache_reclaim_weight * SWAP_CLUSTER_MAX * nr_cpus_node(nid)` ● SWAP_CLUSTER_MAX在 include/linux/swap.h中定义。 ● nr_cpus_node用于获取节点nid上的CPU数量。 内存回收使用工作队列机制，如果内存回收任务耗时较长，会阻塞后续工作的执行，从而可能影响时间敏感的工作。	默认值1
vm.cache_reclaim_enable	**cache_reclaim_enable**用于启用或禁用周期性内存回收功能。	默认值1

页面缓存上限

表 13-26 cache_limit_mbytes 参数说明

参数	描述	取值
vm.cache_limit_mbytes	**cache_limit_mbytes**用于设置页面缓存的上限（以兆字节为单位）。如果页面缓存超过此限制，将会定期回收页面缓存。	默认值0

最大批处理大小和高水位线

表 13-27 percpu_max_batchsize 参数说明

参数	描述	取值
vm.percpu_max_batchsize	此参数用于设置每个区域中每个CPU的最大批处理大小和高水位线。	• 默认值设置为 (64 * 1024) / PAGE_SIZE。 • 最大值限制为 (512 * 1024) / PAGE_SIZE。 • 最小值限制为 (64 * 1024) / PAGE_SIZE。

页面最大比例

表 13-28 percpu_pagelist_fraction 参数说明

参数	描述	取值
vm.percpu_pagelist_fraction	此参数定义了每个区域中为每个CPU页面列表分配的页面的最大比例（高水位标记为pcp->high）。 此值的最小限制为8，这意味着在任何单个per_cpu_pagelist中，不允许分配超过每个区域1/8的页面。此条目仅会影响热的每个CPU 页面列表。用户可以指定一个数字，例如100，以便为每个CPU页面列表分配每个区域的1/100。 每个CPU页面列表的批处理值也会因此更新，设置为 pcp->high/4 。批处理的上限为 (PAGE_SHIFT * 8)。 初始值为零。内核在启动时不会使用此值来设置每个CPU页面列表的高水位线。如果用户将0写入此sysctl，它将恢复到默认行为。	默认值0

内存优先级分级特性

表 13-29 memcg_qos_enable 参数说明

参数	描述	取值
vm.memcg_qos_enable	动态的开启内存优先级分级特性，0代表关闭特性，1代表开启特性。	默认值0

mmap加载的虚拟地址周期

表 13-30 mmap_rnd_mask 参数说明

参数	描述	取值
vm.mmap_rnd_mask	该参数可以将mmap加载的虚拟地址任意几个bit置成0，以此控制mmap加载的虚拟地址周期。	默认值null

大页管理

表 13-31 大页管理参数说明

参数	描述	取值
vm.hugepage_mig_noalloc	move_pages将大页从一个NUMA NODE迁移到另一个NUMA NODE，当目标NUMA NODE上的可用大页数量不足时，根据参数**hugepage_mig_noalloc**的值决定是否申请新的大页。值为1时不会申请新的大页，而是直接迁移失败。值为0时会在目标NUMA NODE上申请新的大页，并释放被迁移的大页。	默认值0
vm.hugepage_nocache_copy	X86架构下，move_pages迁移大页到AEP对应的NUMA NODE时，根据参数**hugepage_nocache_copy**的值决定内存拷贝方式，值为1时会使用NT优化指令进行内存拷贝，值为0时会使用原生的mov指令进行内存拷贝。	默认值0

参数	描述	取值
vm.hugepage_pmem_allocall	在AEP对应的NUMA NODE上申请大页时，根据参数**hugepage_pmem_allocall**的值决定是否对大页进行数量限制。值为0时允许转换为的大页数量受到内核水线的控制。值为1时允许将所有的可用内存都申请为大页。	默认值0

vmemmap内存来源

表 13-32 vmemmap_block_from_dram 参数说明

参数	描述	取值
vm.vmemmap_block_from_dram	在AEP的内存热插到系统NUMA NODE中时，根据参数**vmemmap_block_from_dram**的值决定从AEP上申请的vmemmap的内存的来源。值为1时会从DRAM中申请。值为0时正常从对应的AEP中申请。	默认值0

内存复用

表 13-33 swap_madvised_only 参数说明

参数	描述	取值
vm.swap_madvised_only	内存复用开关。参数**swap_madvised_only**值为1表示开启内存复用。值为0表示关闭内存复用。	默认值0

Qemu热替换

表 13-34 enable_hotreplace 参数说明

参数	描述	取值
vm.enable_hotreplace	使能Qemu热替换能力，该功能支持在业务不中断的情况下快速升级Qemu版本，仅支持在HostOS系统热补丁升级场景使用，GuestOS场景不支持开启。 支持参数：0/1。	默认值0（不使能）

释放主机缓存

提供主动释放主机缓存的方法。slab allocation是一种内存管理机制， 专门用于缓存内核的数据对象，可以减少内存碎片提高系统性能。但随着进程的进行，slab可能会占用大量内存。启动drop_slabs释放缓存以达到增加主机可用内存的目的。

表 13-35 释放主机缓存参数说明

参数	描述	取值
vm.drop_slabs	若值为0表示未清理；若值为1表示启动清理流程。	取值范围{0,1}，默认值为1
vm.drop_slabs_limit	表示优先级，值越大则表示更少比例的slabs缓存被清理。这是为了避免清理过多slabs而长时间占用CPU的情况。	取值范围[0, 12]内的整数，默认值为7

zram 内存压缩支持 cgroup 隔离

本特性实现memcg与zram设备之间的绑定，支持指定memcg使用指定zram设备，并且zram设备使用的内存从该组的容器的内存中获取。

表 13-36 特性开关

参数	描述	取值
vm.memcg_swap_qos_enable	可读可写接口，默认为0关闭特性，向该接口输入1时将所有memcg的memory.swapfile设置为all，向该接口输入2时会将所有memcg的memory.swapfile设置为none。 当接口值为1或2时，若要修改接口的值，需要先将接口置0后再操作。	取值范围：{0，1，2} 默认值：0

支持关闭全局 swap，强制匿名页不换出

社区内核的全局swappiness配置为0时，并不代表一定不会换出匿名页。现提供swap_extension开关，用于关闭全局swap，强制匿名页不换出。

表 13-37 vm.swap_extension 参数说明

参数	描述	取值
vm.swap_extension	用于关闭全局swap，强制匿名页不换出。 swap_extension设置为1，关闭全局匿名页回收。全局匿名页默认不会被换出，除非进程主动调用madvise使用swap空间。cgroup内的进程匿名页交换不受影响。 swap_extension设置为2，会在swap空间用完时清空swapcache，此功能与本特性无关。 swap_extension设置为3，会同时使能功能1和功能2。	取值范围：{0，1，2，3} 默认值：0

内核水线参数

表 13-38 vm.lowmem_reserve_dma_ratio 参数说明

参数	描述	取值
vm.lowmem_reserve_dma_ratio	如果没有通过GFP标志位做出内存分配的强制匹配限定（如GFP_DMA等），那么当ZONE_HIGHMEM中内存不足时，可从更低位的ZONE_NORMAL中分配，ZONE_NORMAL中内存不足时，可从更低位的ZONE_DMA32中分配。这里的“低位”是指zone的物理内存的地址更小。这里所谓的“不足”就是当前zone的空余内存低于了本次内存分配的请求大小。除了最高位的ZONE_HIGHMEM，其他zones都会为比它更高位的zone单独划分出一片内存作为预留，这部分内存被称作“lowmem reserve”。当前默认值为“DMA/normal/HighMem: 256 320”，单位是page。	默认值0

oom事件管理

在某些情况下，我们希望oom在发生的时候不要杀死进程，能够通知黑匣子上报事件，能够触发crash来定位问题。

表 13-39 vm.enable_oom_killer 参数说明

参数	描述	取值
vm.enable_oom_killer	值为1时开启该特性，值为0时关闭此特性。	默认值0

内存 UCE 故障采集上报

使能内存UCE故障采集上报服务后，在内存UCE故障发生时，系统将会采集相关故障信息，将其作为告警事件发送给告警转发服务，之后订阅内存UCE故障事件的程序可接收到此告警进行自定义处理，从而提供内存UCE故障信息实时感知能力。

表 13-40 内核内存 UCE 故障上报能力开关

参数	描述	取值
vm.uce_handle_event_enable	是否使能内核内存UCE故障上报，0表示不使能，非0表示使能。 该特性仅存在arm环境。	默认值0

mce 参数

以下是所属文件在/proc/sys/mce目录下的参数：

UCE机制增强

表 13-41 mce_kernel_recover 参数说明

参数	描述	取值
mce.mce_kernel_recover	内核UCE机制增强功能开关，值为1表示UCE机制增强功能处于使能状态。 可通过 `echo 0 > /proc/sys/mce/mce_kernel_recover` 关闭特性。	默认值为1

debug 参数

以下是所属文件在/proc/sys/debug目录下的参数：

系统异常通知

当Linux内核发生Oops时，会先后进入到die流程及panic流程中，此时会调用其他模块注册到die通知链及panic通知链的回调函数，当回调函数中存在使得内核发生Oops的异常时，内核会再次进入Oops流程， 而后在Oops流程中再次调用该回调函数时，又产生Oops，这样便形成了嵌套的Oops，无法走出Oops流程，系统挂死。

为了增强系统可定位性，提高可靠性，引入系统异常通知链特性，在用户注册的panic/die通知链中出现嵌套Oops异常时，打印错误日志并执行crash流程复位系统。

表 13-42 系统异常参数说明

参数	描述	取值
debug.nest_oops_enhance	panic通知链的嵌套Oops增强接口。	默认值为1（开启）
debug.nest_panic_enhance	die通知链的嵌套Oops增强接口。	默认值为1（开启）

14 HCE 定制系统启动参数说明

HCE相较CentOS 8存在部分定制系统启动参数，具体说明及用途如下。

nohz

在Linux内核2.6.17版本之前，Linux内核为每个CPU设置一个周期性的时钟中断，Linux内核利用这个中断处理一些定时任务，如线程调度等。这样导致就算CPU不需要定时器的时候，也会有很多时钟中断，导致资源的浪费。Linux内核2.6.17版本引入了nohz机制，实际就是让时钟中断的时间可编程，在CPU空闲的情况下减少不必要的时钟中断。

nohz功能对CPU的能耗有积极的作用，但对负载均衡而言却不友好，在部分业务场景下开启nohz会导致性能下降，因此建议默认关闭nohz功能，可以通过在系统启动参数中添加**nohz=off**并重启系统从而关闭该功能。

⚠ 注意

nohz功能在部分特殊场景下对性能有积极作用，例如多个线程同时并发读取/proc/cpuinfo的场景。

mitigations

2018年01月，谷歌Project Zero公布现代处理器存在安全漏洞Spectre与Meltdown，这两组漏洞几乎涉及当今大部分主流处理器（包括Intel、AMD、ARM等多种架构）。随后，包括Linux在内的主流操作系统都对漏洞进行了相应的软件修复，mitigations即是控制这些CPU漏洞修复是否开启的开关。

由于漏洞利用处理器硬件的推测执行（Speculative Execution）以及乱序执行（Out-of-order Execution）特性，而这些特性对于现代处理器的性能提升具有不可或缺的作用。因此开启CPU漏洞修复会有一定的性能下降，在某些极端场景下性能下降甚至超过50%，并且软件修复的方法只能缓解不能根治漏洞问题，因此建议默认关闭mitigations功能，可以通过在系统启动参数中添加**mitigations=off**并重启系统从而关闭该功能。

cstate

HCE2.0从2412版本开始支持intel GNR服务器c状态能力，开启c状态会进入节能模式，相应的内核调度性能会下降。该功能为BIOS默认设置，用户可通过**cpupower idle-info** 查看是否开启c状态，返回信息如下图：

```
[                                                    ]# cpupower idle-info
CPUidle driver: intel_idle
CPUidle governor: menu
analyzing CPU 0:

Number of idle states: 3
Available idle states: POLL C1 C1E
POLL:
Flags/Description: CPUIDLE CORE POLL IDLE
Latency: 0
Usage: 27
Duration: 163
C1:
Flags/Description: MWAIT 0x00
Latency: 1
Usage: 66
Duration: 11541
C1E:
Flags/Description: MWAIT 0x01
Latency: 4
Usage: 132338
Duration: 63065904
```

其中Number of idle states表示支持的c状态数量，Available idle states表示支持的具体c状态。若回显为No idle states则表示未开启c状态。若在性能要求较高场景下，可通过**cpupower idle-set --disable level**(level为c状态等级，0到Number of idle states - 1，此关闭方法无需重启机器)或者启动参数中添加**intel_idle.max_cstate=0**(需重启生效)来关闭c状态。

15 网卡重命名

前言

在早期版本的操作系统（如Fedora 13、Ubuntu 15、CentOS 6及更早版本）中，网络接口的命名方式为eth0、eth1、eth2等。Linux内核通过组合固定前缀和索引为网络接口分配名称。在系统初始化时，内核会根据索引顺序为网络设备分配名称，例如eth0代表系统启动时检测到的第一个以太网接口。如果添加了额外的网络接口，设备名称的分配顺序则不再固定，因为系统重启后设备可能会按不同的顺序初始化。

为了解决这一问题并提高网络设备的可识别性和一致性，现代Linux发行版采用了一致的网络设备命名规范，并通过udev设备管理器来实现这一规范。udev支持多种命名方案，默认情况下，它根据固件信息、拓扑结构及设备物理位置分配网络接口的名称。系统中的网卡设备重命名服务由systemd-udevd提供支持，以确保网络设备名称的一致性和稳定性。

一致的网络命名规则

命名规范为：设备类型 + 设备位置。

设备类型：

- en：以太网
- wl：无线局域网
- ww：无线广域网

设备位置

表 15-1 一致的网络命名规则设备位置含义表

格式	含义
o<on-board_index_number>	主板bios内置的网卡
s<hot_plug_slot_index_number>[f<function>][d<device_id>]	主板bios内置的PCI-E网卡
x<MAC>	MAC地址
p<bus>s<slot>[f<function>][d<device_id>]	PCI-E独立网卡

格式	含义
P<domain_number>]p<bus>s<slot>[f<function>][u<usb_port>][.][c<config>][i<interface>]	USB网卡

网卡重命名步骤

本小节详细介绍网卡重命名方法。以将ens5修改为eth0为例。

步骤1 修改网卡配置文件。

- 如果网卡的配置文件/etc/sysconfig/network-scripts/ifcfg-ens5存在，将原配置文件ifcfg-ens5重命名为ifcfg-eth0：
  ```
  mv /etc/sysconfig/network-scripts/ifcfg-ens5 /etc/sysconfig/network-scripts/ifcfg-eth0
  ```
  编辑ifcfg-eth0配置文件，将NAME和DEVICE参数的值修改为新网卡名eth0。
- 如果网卡的配置文件/etc/sysconfig/network-scripts/ifcfg-ens5不存在，请直接创建配置文件/etc/sysconfig/network-scripts/ifcfg-eth0，并添加以下内容：
  ```
  DEFROUTE=yes
  BOOTPROTO=dhcp
  NAME=eth0
  DEVICE=eth0
  ONBOOT=yes
  ```

步骤2 修改GRUB配置。

编辑/etc/default/grub，添加net.ifnames=0 biosdevname=0到GRUB_CMDLINE_LINUX项中。修改完成后，执行以下命令重新加载GRUB配置：

```
grub2-mkconfig -o /boot/grub2/grub.cfg
```

步骤3 创建持久化规则文件。

在/etc/udev/rules.d/中创建70-persistent-net.rules文件，并添加以下规则内容：

```
SUBSYSTEM=="net", ACTION=="add", DRIVERS=="?*", ATTR{address}=="xx:xx:xx:xx:xx:xx", NAME="eth0"
```

步骤4 重启节点。

----结束

此外，用户也可以通过执行以下脚本来修改网卡名称。执行脚本后，按照提示输入修改前的网卡名称和修改后的网卡名称。完成操作后，重新启动系统，网卡将被重命名为新的名称。

```
#!/bin/bash

function check_networkcard() {
    while true; do
        echo "请输入要修改的网卡名称（例如：ens33）："
        read interface_name
        if [[ -z "$interface_name" ]]; then
            echo "输入不能为空，请重新输入"
            continue
        fi
        ifconfig -a | grep -q "$interface_name"
        if [[ $? -eq 0 ]]; then
            break
        else
            echo "网卡 $interface_name 不存在，请重新输入"
        fi
```



```
    done
}
function modify_grub() {
    sed -i 's/resume/net.ifnames=0 biosdevname=0 &/' /etc/sysconfig/grub
    grub2-mkconfig -o /boot/grub2/grub.cf
}
function modify_adaptername() {
    while true; do
        echo "请输入修改后的网卡名称（例如：eth0）："
        read new_interface_name
        if [[ -z "$new_interface_name" ]]; then
            echo "输入不能为空，请重新输入"
            continue
        else
            break
        fi
    done
    cd /etc/sysconfig/network-scripts/
    if [[ -e ifcfg-$interface_name ]]; then
        mv ifcfg-$interface_name ifcfg-$new_interface_name
        sed -i "s/$interface_name/$new_interface_name/g" ifcfg-$new_interface_name
    else
       sudo cat <<EOF >/etc/sysconfig/network-scripts/ifcfg-$new_interface_name
DEFROUTE=yes
BOOTPROTO=dhcp
NAME="$new_interface_name"
DEVICE="$new_interface_name"
ONBOOT=yes
EOF
    fi
}
function reload_network() {
    mac=$(ip link show $interface_name | awk '/ether/{print $2}')
    echo "SUBSYSTEM==\"net\", ACTION==\"add\", DRIVERS==\"?*\", ATTR{address}==\"$mac\", NAME=
\"$new_interface_name\"" >> /etc/udev/rules.d/70-persistent-net.rules
}
function clear_variable() {
    unset interface_name
    unset interface_cardname
    unset mac
}
function main() {
    check_networkcard
    modify_adaptername
    modify_grub
    reload_network
    clear_variable
    echo "网卡名称修改完成，请使用reboot命令重启节点后生效"
}
main
```

16 透明大页相关调优方法

16.1 概述

透明大页（Transparent Huge Pages，简称THP）是Linux内核提供的一项优化内存管理性能的机制。

它通过将默认的4 KB小页，在满足条件的情况下自动合并为较大的页（通常为2 MB），以减少页表项数量、提升TLB（Translation Lookaside Buffer，快表）命中率，从而降低内存访问开销，改善系统整体性能。

16.2 相关配置

功能开关

THP的功能开关控制系统是否启用透明大页功能。

配置路径：

```
/sys/kernel/mm/transparent_hugepage/enabled
```

参数选项说明：

- always：所有合适的内存分配都尽可能使用大页。
- madvise：仅当程序显式通过madvise()系统调用请求时才使用大页。
- never：彻底禁用透明大页。

碎片整理

由于大页分配需要连续的物理内存区域，长时间运行的系统往往因频繁分配和释放导致内存碎片化。碎片整理（defragmentation）是指内核通过页面迁移等手段合并零散的物理页，从而形成足够大的连续区域用于大页分配。碎片整理可能会导致系统延迟上升，特别是在前台执行时。

配置路径：

```
/sys/kernel/mm/transparent_hugepage/defrag
```

参数选项说明：

- always：当前线程始终等待内核整理碎片后再分配大页，可能严重影响响应时间。
- defer：当前线程不会等待，碎片整理在后台异步执行。
- madvise：仅针对显式调用madvise()请求的内存区域进行整理。
- defer+madvise：当前线程不等待，碎片整理交给后台，仅针对明确请求使用大页的内存。
- never：完全不整理碎片，若无法直接分配大页，则回退为小页。

后台合并线程

khugepaged是内核中的后台线程，定期扫描匿名内存区域，将合适的4KB页自动合并为2MB的大页。

相关配置文件目录如下，

```
/sys/kernel/mm/transparent_hugepage/khugepaged/
```

关键配置文件：

- defrag：功能开关，可选的配置项如下：
  - 1：允许khugepaged主动整理内存碎片。
  - 0：关闭khugepaged主动整理内存碎片功能。
- alloc_sleep_millisecs：重试间隔（单位：毫秒），当分配大页失败时，khugepaged下一次尝试分配前的等待时间，默认值为60000。
- pages_to_scan：扫描页数，khugepaged每次唤醒后扫描的页数，默认值为4096。
- scan_sleep_millisecs：扫描间隔（单位：毫秒），用于调节后台扫描频率，默认值为10000。

说明

如业务对延迟敏感，可适当加大扫描间隔，或关闭khugepaged整理功能。

16.3 常见场景与调优建议

推荐配置：通用业务场景

开启madvise模式，仅在需要时才使用大页：

```
echo madvise > /sys/kernel/mm/transparent_hugepage/enabled
```

内存整理使用defer+madvise，后台合并且仅针对显式请求：

```
echo defer+madvise > /sys/kernel/mm/transparent_hugepage/defrag
```

数据库或延迟敏感服务

数据库如MySQL、PostgreSQL、Redis对性能抖动敏感，建议关闭THP：

```
echo never > /sys/kernel/mm/transparent_hugepage/enabled
```

如需永久关闭，可在启动参数中加入：


```
sudo grubby --args="transparent_hugepage=never" --update-kernel="/boot/vmlinuz-$(uname -r)" sudo reboot
```

控制 khugepaged 活跃度

如果系统中khugepaged守护进程的CPU使用率较高时，可以考虑以下配置降低khugepaged守护进程活跃度。

降低khugepaged扫描频率：

```
echo 30000 > /sys/kernel/mm/transparent_hugepage/khugepaged/scan_sleep_millisecs
```

增加分配失败后的重试间隔：

```
echo 120000 > /sys/kernel/mm/transparent_hugepage/khugepaged/alloc_sleep_millisecs
```

减少每次扫描页数：

```
echo 2048 > /sys/kernel/mm/transparent_hugepage/khugepaged/pages_to_scan
```

禁用khugepaged自动整理功能（如确实需要）：

```
echo 0 > /sys/kernel/mm/transparent_hugepage/khugepaged/defrag
```

16.4 查看 THP 的使用情况

查看THP全局使用情况：

```
cat /proc/meminfo | grep AnonHugePages
```

返回值为非零表示系统当前存在透明大页使用。

查看特定进程是否使用THP：

```
cat /proc/<PID>/smaps | grep AnonHugePages
```

将<PID>替换为实际进程号，可查看该进程哪些内存段正在使用大页。

17 NetworkManager 选型及使用指导

NetworkManager 选型介绍

NetworkManger作为network的替代品，功能更强大，在新版os中基本都是以NetworkManager作为主流。综合来说，network服务更适合那些熟悉命令行和手动配置的用户，而NetworkManager则更适合普通用户或需要自动管理网络连接的情况。在大多数Linux发行版中，NetworkManager已成为默认的网络管理工具。

⚠ 注意

NetworkManager和network不可同时使用。

NetworkManager 使用指导

- 查看NetworkManager服务状态：
  ```
  systemctl status NetworkManager
  ```
- 关闭NetworkManager服务命令：
  ```
  systemctl stop NetworkManager
  ```
- 重启NetworkManager服务命令：
  ```
  systemctl restart NetworkManager
  ```

18 XFS 文件系统

XFS是一种高性能的日志文件系统，HCE继承社区的开源特性受限商用交付部分特性。

约束限制

1. 不支持格式化选项inobtcount=1 / rmapbt=1 / reflink=1。

2. 不支持挂载选项discard / logdev=device / noattr2 / ikeep。

3. 仅支持磁盘扇区大小512B，其余不支持。

4. 挂载XFS文件系统，remount操作仅支持inode32 / inode64以及ro / rw两组参数切换，其余不支持。

说明

对于未提供支持的参数，在使用时可能会出现预期外的结果：如logdev=device，在某些情况下可能会出现io超时卡住的情况。

使用方法

XFS特性为社区开源特性，与开源保持功能用法一致，参考标准文件系统用法使用。

注意

使用XFS文件系统前请确认HCE内核版本不低于kernel-5.10.0-182.0.0.95.r1941_123.hce2，通过uname -r命令查看。