



openEuler

20.03 LTS

升级指导书

发布日期

2020-03-23

目 录

法律声明.....	iii
前言.....	iv
1 升级前准备.....	5
1.1 升级路径	5
1.2 升级影响	5
1.3 升级注意事项	5
1.4 配置 repo 源	6
1.4.1 获取 ISO 镜像.....	6
1.4.2 挂载 ISO 并配置为 repo 源	7
2 升级操作.....	9
2.1 升级前检查	9
2.2 升级	9
2.3 升级后验证	10
3 常见异常问题处理.....	11
3.1 如何处理系统升级中断	11
3.2 系统一直停留在某一软件包安装阶段.....	12
3.3 升级软件包时出现冲突或缺少软件包.....	12

法律声明

版权所有 © 2020 华为技术有限公司。

您对“本文档”的复制、使用、修改及分发受知识共享(Creative Commons)署名-相同方式共享 4.0 国际公共许可协议(以下简称“CC BY-SA 4.0”)的约束。为了方便用户理解，您可以通过访问 <https://creativecommons.org/licenses/by-sa/4.0/> 了解 CC BY-SA 4.0 的概要 (但不是替代)。CC BY-SA 4.0 的完整协议内容您可以访问如下网址获取：
<https://creativecommons.org/licenses/by-sa/4.0/legalcode>。

商标声明

openEuler 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。

免责声明

本文档仅作为使用指导，除非适用法强制规定或者双方有明确书面约定，华为技术有限公司对本文档中的所有陈述、信息和建议不做任何明示或默示的声明或保证，包括但不限于不侵权，时效性或满足特定目的的担保。

前言

概述

本文档详细地描述了 openEuler 操作系统的升级操作步骤，以指导用户顺利完成 openEuler 操作系统的升级。



读者对象

本文档主要适用于升级的操作人员。操作人员必须具备以下经验和技能：

- 具备 openEuler 操作系统管理知识。
- 有 Linux 系统的维护经验，熟悉 Linux 系统的操作维护方式。

符号约定

在本文中可能出现下列标志，它们所代表的含义如下。

符号	说明
 须知	用于传递设备或环境安全警示信息。如不可避免则可能会导致设备损坏、数据丢失、设备性能降低或其它不可预知的结果。 “须知”不涉及人身伤害。
 说明	对正文中重点信息的补充说明。 “说明”不是安全警示信息，不涉及人身、设备及环境伤害信息。

1 升级前准备

- 1.1 升级路径
- 1.2 升级影响
- 1.3 升级注意事项
- 1.4 配置 repo 源

1.1 升级路径

表1-1 升级前的版本要求

升级前的版本	升级后的版本
openEuler 1.0 Base	openEuler 20.03 LTS

1.2 升级影响

如果需要从低版本升级到 openEuler 20.03 LTS 版本，您需要认真阅读本章，了解升级可能对现有系统造成的影响。

对业务的影响

- CPU 占用率
升级过程中，升级程序会占用一定的 CPU，为了保证升级程序正常运行，请确保系统有空闲的 CPU 资源用于升级。

1.3 升级注意事项

1. 在更新中，突然掉电或更新进程被强制中止，会有系统无法重新启动的风险。

2. 请保证根分区有足够的存储空间，一般要求大于 8G，否则升级可能会失败。
3. 升级后不支持进入原内核启动项，因为在新内核中路由信息接口有变更，涉及到 `monitor_netdev` 模块，在原来内核启动中会出现 `oops`。
4. 升级成功后需要重启系统。

1.4 配置 repo 源

升级 openEuler 前需要配置 repo 源。

1.4.1 获取 ISO 镜像

下载 ISO 镜像

- 通过跨平台文件传输工具下载 ISO 镜像
 - a. 登录 openEuler 社区，网址为：<https://openeuler.org>。
 - b. 单击“下载”，进入下载页面。
 - c. 单击“获取 ISO：”后面的“Link”，显示下载列表。
 - d. 单击“openEuler-20.03-LTS-everything-aarch64-dvd.iso”将 openEuler 发布包下载到本地。
 - e. 单击“openEuler-20.03-LTS-everything-aarch64-dvd.iso.sha256sum”将 openEuler 校验文件下载到本地。
 - f. 登录待升级的 openEuler 操作系统，新建用于存放发布包和检验文件的目录，如“/home/iso”。
- ```
mkdir /home/iso
```
- g. 使用跨平台文件传输工具（如 WinSCP）将本地的 openEuler 发布包和校验文件上传到待升级 openEuler 操作系统。

- 通过 `wget` 命令下载 ISO 镜像
  - a. 登录 openEuler 社区，网址为：<https://openeuler.org>。
  - b. 单击“下载”，进入下载页面。
  - c. 单击“获取 ISO：”后面的“Link”，显示下载列表。
  - d. 右键单击“openEuler-20.03-LTS-everything-aarch64-dvd.iso”，单击“复制链接地址”，将 openEuler 发布包地址记录好。
  - e. 右键单击“openEuler-20.03-LTS-everything-aarch64-dvd.iso.sha256sum”，单击“复制链接地址”，将 openEuler 校验文件地址记录好。
  - f. 登录待升级的 openEuler 操作系统，新建用于存放发布包和检验文件的目录，如“/home/iso”，并切换到该目录。

```
mkdir /home/iso
cd /home/iso
```

- g. 使用 `wget` 命令远程下载发布包和检验文件，命令中的 `ipaddriso` 和 `ipaddrisosum` 分别为 **d** 和 **e** 中记录的地址。

```
wget ipaddriso
wget ipaddrisosum
```

## 发布包完整性校验

1. 获取校验文件中的校验值。

```
cat openEuler-20.03-LTS-everything-aarch64-dvd.iso.sha256sum
```

2. 计算 openEuler 发布包的 sha256 校验值。

```
sha256sum openEuler-20.03-LTS-everything-aarch64-dvd.iso
```

命令执行完成后，输出校验值。

3. 对比步骤 1 和步骤 2 计算的校验值是否一致。

如果校验值一致说明 iso 文件完整性没有破坏，如果校验值不一致则可以确认文件完整性已被破坏，需要重新获取。

## 1.4.2 挂载 ISO 并配置为 repo 源

使用 mount 命令挂载镜像文件。

示例如下：

```
mount /home/iso/openEuler-20.03-LTS-everything-aarch64-dvd.iso /mnt/
```

挂载好的 mnt 目录如下：

```
.
├── boot.catalog
├── docs
├── EFI
├── images
├── Packages
├── repodata
├── TRANS.TBL
└── RPM-GPG-KEY-openEuler
```

其中，Packages 为 rpm 包所在的目录，repodata 为 repo 源元数据所在的目录，RPM-GPG-KEY-openEuler 为 openEuler 的签名公钥。。

挂载后的目录可以配置为 yum 源使用，在/etc/yum.repos.d/目录下创建\*\*\*.repo 的配置文件（必须以.repo 为扩展名）。

示例如下：

在/etc/yum.repos.d 目录下创建 openEuler.repo 文件，使用本地镜像挂载目录作为 yum 源，openEuler.repo 的内容如下：

```
[base]
name=base
baseurl=file:///mnt
enabled=1
gpgcheck=1
gpgkey=file:///mnt/RPM-GPG-KEY-openEuler
```

### 说明

- gpgcheck 可设置为 1 或 0，1 表示进行 gpg（GNU Private Guard）校验，0 表示不进行 gpg 校验，gpgcheck 可以确定 rpm 包的来源是有效和安全的。

- `gpgkey` 为签名公钥的存放路径。



# 2 升级操作

## 2.1 升级前检查

## 2.2 升级

## 2.3 升级后验证

## 2.1 升级前检查

执行如下命令检查升级前的版本，确保升级前的版本为 openEuler 1.0 Base。

```
uname -a
```

查看打印信息，打印信息中包括“4.19.90-vhulk2001.1.0.0026.aarch64 #1 SMP Fri Feb 7 04:09:58 UTC 2020”时表示为 openEuler 1.0 Base 版本。

## 2.2 升级

### 📖 说明

版本升级后不支持回退，请谨慎操作。

### 前提条件

- 已经完成本地 repo 源的构建。
- 已经完成 yum 配置。

### 操作步骤

步骤 1 备份已修改的系统配置文件。

步骤 2 执行 **yum update -y** 命令升级 openEuler 操作系统。

```
yum update -y
```

### 说明

执行 `yum update` 后，如果是跨版本升级，可能会报有冲突，可按文档 3.2 系统一直停留在某一软件包安装阶段章节处理。

步骤 3 恢复备份的系统配置文件。

步骤 4 执行 **reboot** 命令重启系统。

```
reboot
```

----结束

## 2.3 升级后验证

升级成功系统重启后，执行以下命令，确保系统已升级为 openEuler 20.03 LTS。

```
cat /etc/openEuler-release
```

查看打印信息，打印信息中包括“openEuler release 20.03 (LTS)”表示已升级为 openEuler 20.03 LTS。

# 3 常见异常问题处理

- 3.1 如何处理系统升级中断
- 3.2 系统一直停留在某一软件包安装阶段
- 3.3 升级软件包时出现冲突或缺少软件包

## 3.1 如何处理系统升级中断

### 问题现象

升级过程中 yum 命令执行中断。

### 问题原因

用户执行 CTRL+C 退出执行或进程异常退出。

### 处理步骤

在系统后台执行如下命令，进行系统恢复。

1. 重构 rpmdb，执行命令：

```
rpmdb --rebuilddb
```

2. 继续未完成的任务，执行命令：

```
yum history redo last -y
```

3. 擦除升级前的包，执行命令：

```
package-cleanup --cleandupes
```

4. 重新安装 kernel，执行命令：

```
yum reinstall kernel -y
```

5. 进行系统手动重启，执行命令：

```
reboot
```

## 3.2 系统一直停留在某一软件包安装阶段

### 问题现象

安装进程一直停留在安装某一 rpm 软件包阶段。

### 问题原因

安装过程中 systemd 会多次 reload 服务，可能会引起 systemd 关于 automount 的 bug。

### 处理步骤

- 方法一：注销 proc-sys-fs-binfmt\_misc.automount 服务。

- a. 在升级前注销服务，执行命令：

```
systemctl mask proc-sys-fs-binfmt_misc.automount
```

- b. 在升级后取消注销服务，执行命令：

```
systemctl unmask proc-sys-fs-binfmt_misc.automount
```

- 方法二：重启 proc-sys-fs-binfmt\_misc.automount 服务。

切换到系统的其他终端中，执行如下命令：

```
systemctl restart proc-sys-fs-binfmt_misc.automount
```

## 3.3 升级软件包时出现冲突或缺少软件包

### 问题现象

升级过程中，可能出现文件冲突或缺少软件包，从而导致升级被中断，最终升级失败。文件冲突和缺少软件包的报错信息分别如下所示。

文件冲突报错信息示例（以/usr/bin/containerd 文件冲突为例）：

```
Error: Transaction test error:
 file /usr/bin/containerd from install of containerd-1.2.0-101.oe1.aarch64
 conflicts with file from package docker-engine-18.09.0-100.aarch64
 file /usr/bin/containerd-shim from install of containerd-1.2.0-101.oe1.aarch64
 conflicts with file from package docker-engine-18.09.0-100.aarch64
```

缺少软件包的报错信息示例（以缺失 blivet-data 软件包为例）：

```
Error:
 Problem: cannot install both blivet-data-1:3.1.1-6.oe1.noarch and blivet-data-
 1:3.1.1-5.noarch
 - package python2-blivet-1:3.1.1-5.noarch requires blivet-data = 1:3.1.1-5, but
 none of the providers can be installed
 - cannot install the best update candidate for package blivet-data-1:3.1.1-
 5.noarch
 - problem with installed package python2-blivet-1:3.1.1-5.noarch(try to add '--
 allowerase' to command line to replace conflicting packages or '--skip-broken' to
 skip uninstallable packages or '--nobest' to use not only best candidate packages)
```

## 问题原因

- openEuler 提供的软件包中，有些软件包虽然名称不同，但功能相同，导致安装时安装后的文件相同，从而产生了文件冲突。
- 有些软件包，因在升级前被其他软件包所依赖，一旦该软件包升级后，可能导致依赖它的软件包因缺少软件包而不能安装。

## 处理步骤

若为文件冲突，则按如下步骤进行处理（以问题现象中的/usr/bin/containerd 文件冲突为例）：

1. 根据升级过程中的文件冲突报错信息，确定导致文件冲突的软件包名称为 containerd-1.2.0-101.oe1.aarch64 和 docker-engine-18.09.0-100.aarch64。
2. 将不需要安装的软件包名称记录下来，以不需要安装 docker-engine-18.09.0-100.aarch64 为例。
3. 执行 **dnf remove** 命令将不需要安装的软件包，单独卸载。

```
dnf remove docker-engine-18.09.0-100.aarch64
```

4. 重新进行升级操作。

若为缺少软件包，则按如下步骤进行处理（以问题现象中的缺少 blivet-data-1:3.1.1-5.noarch 软件包为例）：

1. 根据升级过程中的缺少软件包报错信息，确定待升级的软件包名称 blivet-data-1:3.1.1-5.noarch 及依赖它的软件包名称 python2-blivet-1:3.1.1-5.noarch。
2. 执行 **dnf remove** 命令将依赖待升级包才能安装的软件包单独卸载或在升级软件包时加上--allowerasing 参数。

- 执行 **dnf remove** 命令将依赖 blivet-data-1:3.1.1-5.noarch 软件包才能安装的软件包单独卸载。

```
dnf remove python2-blivet-1:3.1.1-5.noarch
```

- 升级软件包时加上--allowerasing 参数。

```
yum update blivet-data-1:3.1.1-5.noarch -y --allowerasing
```

3. 重新进行升级操作。