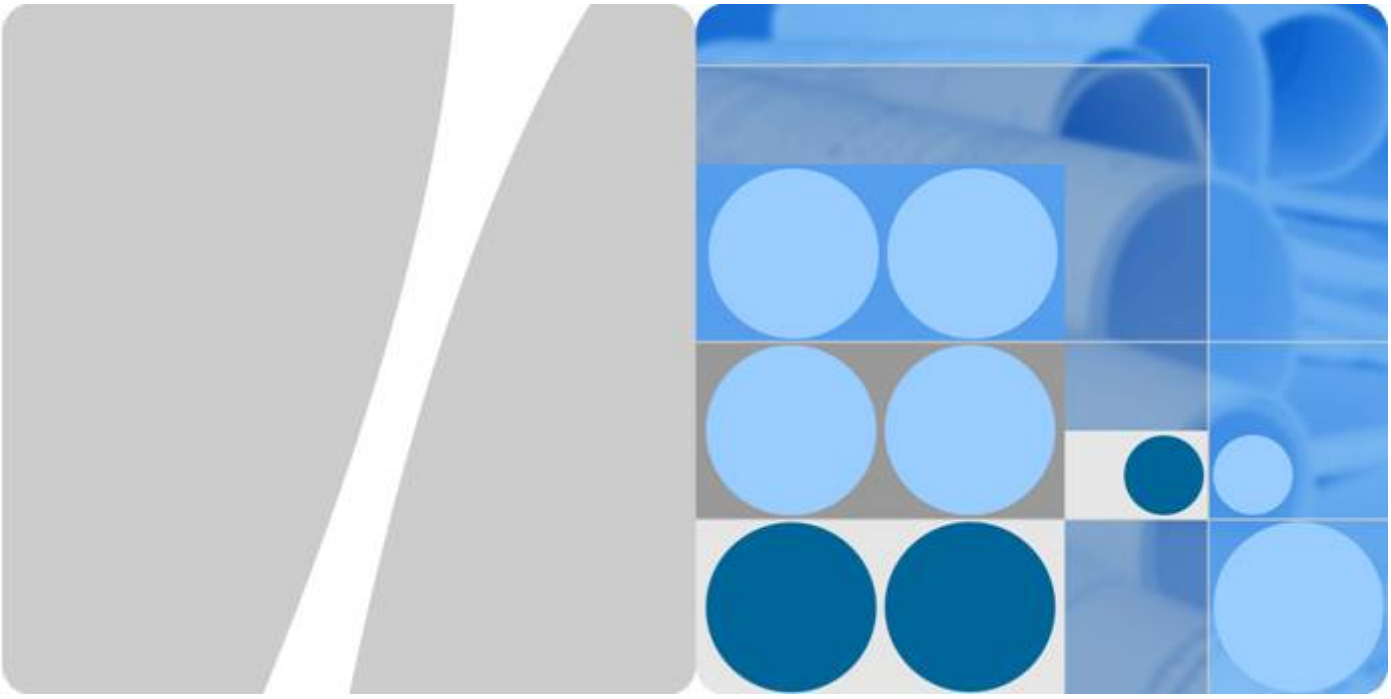


华为欧拉服务器操作系统软件 V2.0 管理员指南

文档版本　01
发布日期　2019-08-12

华为技术有限公司

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，华为公司对本文档内容不做任何明示或默示的声明或保证。

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。

华为技术有限公司

地址： 深圳市龙岗区坂田华为总部办公楼 邮编：518129

网址： http://www.huawei.com

客户服务邮箱： support@huawei.com

客户服务电话： 4008302118

目 录

1 前言

概述

华为欧拉服务器操作系统软件V2.0是华为公司研发的企业级linux服务器操作系统，具有较高的性能、可靠性、易维护以及安全性，与业界软硬件良好兼容，能够满足您日常业务运维和管理的需求。

本手册为初次使用华为欧拉服务器操作系统软件V2.0产品的用户提供日常使用和配置维护的指引，文档包括基础配置、软件包管理、用户管理等相关说明，用户可以通过本文档对华为欧拉服务器操作系统软件V2.0进行日常的维护和相关配置。

华为欧拉服务器操作系统软件V2.0的软件简称为EulerOS V2.0，本手册及软件产品相关界面均使用了软件简称EulerOS V2.0。

读者对象

本手册适用于使用EulerOS V2.0 产品的用户，特别是初次使用或想了解EulerOS V2.0的用户，包括系统工程师、管理员及维护人员等。

使用本手册的用户需要具备基础的linux系统管理知识。

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
 危险	用于警示紧急的危险情形，若不避免，将会导致人员死亡或严重的人身伤害。
 警告	用于警示潜在的危险情形，若不避免，可能会导致人员死亡或严重的人身伤害。
 小心	用于警示潜在的危险情形，若不避免，可能会导致中度或轻微的人身伤害。

符号	说明
 注意	用于传递设备或环境安全警示信息，若不避免，可能会导致设备损坏、数据丢失、设备性能降低或其他不可预知的结果。“注意”不涉及人身伤害。
 说明	用于突出重要/关键信息、最佳实践和小窍门等。“说明”不是安全警示信息，不涉及人身、设备及环境伤害。

2 基础配置

2.1 通过命令设置

2.1.1 设置语言环境

您可以通过localectl修改系统的语言环境，对应的参数设置保存在/etc/locale.conf文件中。这些参数，会在系统启动的早期被systemd的守护进程读取。

显示当前语言环境状态

要显示当前语言环境，使用命令如下：

```
localectl status
```

例如显示系统当前的设置，命令如下：

```
$ localectl status
   System Locale: LANG=zh_CN.UTF-8
       VC Keymap: cn
      X11 Layout: cn
```

列出可用的语言环境

要显示当前可用的语言环境，使用命令如下：

```
localectl list-locales
```

例如显示当前系统中所有可用的中文环境，命令如下：

```
$ localectl list-locales | grep zh
zh_CN
zh_CN.gb18030
zh_CN.gb2312
zh_CN.gbk
zh_CN.utf8
zh_HK
zh_HK.big5hkscs
zh_HK.utf8
```



```
zh_SG
zh_SG.gb2312
zh_SG.gbk
zh_SG.utf8
zh_TW
zh_TW.big5
zh_TW.euctw
zh_TW.utf8
```

设置语言环境

要设置语言环境，在root权限下使用命令如下：

```
localectl set-locale LANG=locale
```

例如设置为简体中文语言环境，在root权限下执行如下命令：

```
# localectl set-locale LANG=zh_CN.utf8
```

2.1.2 设置键盘

您可以通过localectl修改系统的键盘设置，对应的参数设置保存在/etc/locale.conf文件中。这些参数，会在系统启动的早期被systemd的守护进程读取。

显示当前设置

要显示当前键盘设置，使用命令如下：

```
localectl status
```

例如显示系统当前的设置，命令如下：

```
$ localectl status
   System Locale: LANG=zh_CN.UTF-8
       VC Keymap: cn
      X11 Layout: cn
```

列出可用的键盘布局

要显示当前可用的键盘布局，使用命令如下：

```
localectl list-keymaps
```

例如显示系统当前的中文键盘布局，命令如下：

```
$ localectl list-keymaps | grep cn
cn
```

设置键盘布局

要设置键盘布局，在root权限下使用命令如下：

```
localectl set-keymap map
```

此时设置的键盘布局同样也会应用到图形界面中。

设置完成后，查看当前状态：

```
$ localectl status
   System Locale: LANG=zh_CN.UTF-8
       VC Keymap: cn
      X11 Layout: us
```


2.1.3 设置日期和时间

本节介绍了如何通过timedatectl、date、hwclock命令来设置系统的日期、时间和时区等。

2.1.3.1 使用 timedatectl 命令设置

显示日期和时间

要显示当前的日期和时间，使用命令如下：

```
timedatectl
```

例如显示系统当前的日期和时间，如下：

```
$ timedatectl
      Local time: 五2015-08-14 15:57:24 CST
  Universal time: 五2015-08-14 07:57:24 UTC
        RTC time: 五2015-08-14 07:57:24
        Timezone: Asia/Shanghai (CST, +0800)
     NTP enabled: yes
NTP synchronized: no
 RTC in local TZ: no
      DST active: n/a
```

修改时间

要修改当前的时间，在root权限下使用命令如下：

```
timedatectl set-time HH:MM:SS
```

例如修改当前的时间，例如修改当前的时间为 15:57:24，如下：

```
# timedatectl set-time 15:57:24
```

修改日期

要修改当前的日期，在root权限下使用命令如下：

```
timedatectl set-time YYYY-MM-DD
```

例如修改当前的日期，例如修改当前的日期为2015年8月14号，如下：

```
# timedatectl set-time '2015-08-14'
```

修改时区

显示当前可用时区，使用命令如下：

```
timedatectl list-timezones
```

要修改当前的时区，在root权限下使用命令如下：

```
timedatectl set-timezone time_zone
```

例如修改当前的时区，首先查询所在地域的可用时区（以Asia为例）：

```
# timedatectl list-timezones | grep Asia
Asia/Aden
Asia/Almaty
Asia/Amman
Asia/Anadyr
Asia/Aqtau
```



```
Asia/Aqtobe
Asia/Ashgabat
Asia/Baghdad
Asia/Bahrain
……

Asia/Seoul
Asia/Shanghai
Asia/Singapore
Asia/Srednekolymsk
Asia/Taipei
Asia/Tashkent
Asia/Tbilisi
Asia/Tehran
Asia/Thimphu
Asia/Tokyo
```

修改当前的时区为“shanghai”，如下：

```
# timedatectl set-timezone Asia/Shanghai
```

通过远程服务器进行时间同步

您可以启用NTP远程服务器进行系统时钟的自动同步。是否启用NTP，在root权限下使用命令如下：

```
timedatectl set-ntp boolean
```

例如启动自动远程时间同步，如下：

```
# timedatectl set-ntp yes
```

2.1.3.2 使用 date 命令设置

显示当前的日期和时间

要显示当前的日期和时间，使用命令如下：

```
date
```

默认情况下， date命令显示本地时间。要显示UTC时间，添加--utc或-u参数：

```
date --utc
```

要自定义对应的输出信息格式，添加 +"format" 参数：

```
date +"format"
```

表 2-1 参数说明

格式参数	说明
%H	小时以HH格式（例如 17）。
%M	分钟以MM格式（例如 37）。
%S	秒以SS格式（例如 25）。
%d	日期以DD格式（例如 15）。
%m	月份以MM格式（例如 07）。
%Y	年以YYYY格式（例如 2015）。

格式参数	说明
%Z	时区缩写（例如CEST）。
%F	日期整体格式为YYYY-MM-DD（例如 2015-7-15），和%Y-%m-%d等同。
%T	时间整体格式为HH:MM:SS（例如 18:30:25），和%H:%M:%S等同。

实际使用示例如下：

- 显示当前的日期和本地时间。
```
$ date
2015年 08月 17日 星期一 17:26:34 CST
```
- 显示当前的日期和UTC时间。
```
$ date --utc
2015年 08月 17日 星期一 09:26:18 UTC
```
- 自定义date命令的输出。
```
$ date +"%Y-%m-%d %H:%M"
2015-08-17 17:24
```

修改时间

要修改当前的时间，添加--set或者-s参数。在root权限下使用命令如下：

```
date --set HH:MM:SS
```

默认情况下， date命令设置本地时间。要设置UTC时间，添加--utc或-u参数：

```
date --set HH:MM:SS --utc
```

例如修改当前的时间，在root权限下使用命令如下：

```
# date --set 23:26:00
```

修改日期

要修改当前的日期，添加--set或者-s参数。在root权限下使用命令如下：

```
date --set YYYY-MM-DD
```

例如修改当前的日期为2015年11月2日，如下：

```
# date --set 2015-11-02
```

2.1.3.3 使用 hwclock 命令设置

hwclock用来进行硬件的时钟设置（RTC，Real Time Clock）。

硬件时钟和系统时钟

Linux将时钟分为系统时钟(System Clock)和硬件时钟(Real Time Clock，RTC)两种。系统时间是指当前Linux Kernel中的时钟，而硬件时钟则是主板上由电池供电的主板硬件时钟，这个时钟可以在BIOS的"Standard BIOS Feture"项中进行设置。

当Linux启动时，硬件时钟会去读取系统时钟的设置，然后系统时钟就会独立于硬件运作。

从Linux启动过程来看，系统时钟和硬件时钟不会发生冲突，系统中的所有命令(包括函数)都是采用的系统时钟。不仅如此，系统时钟和硬件时钟还可以采用异步方式，即系统时间和硬件时间可以不同。这样做的好处对于普通用户意义不大，但对于Linux网络管理员却有很大的用处。例如，要将一个很大的网络中(跨越若干时区)的服务器同步，其中一台服务器无须改变硬件时钟而只需临时设置一个系统时间，比如将北京服务器上的时间设置为纽约时间，两台服务器完成文件的同步后，再与原来的时钟同步一下即可。

显示日期和时间

要显示当前硬件的日期和时间，在root权限下使用命令如下：

```
hwclock
```

例如显示当前硬件的日期和时间，如下：

```
# hwclock
2015年08月17日  星期一 14时34分42秒. -0.094973 秒
```

设置日期和时间

要修改当前硬件的日期和时间，在root权限下使用命令如下：

```
hwclock --set --date "dd mmm yyyy HH:MM"
```

例如修改当前的时间，如下：

```
# hwclock --set --date "21 Oct 2015 21:17" --utc
```

2.2 通过图形界面设置

本节主要介绍在图形界面环境下针对一些系统基本设置选项进行说明，指导用户操作。

2.2.1 进入设置界面

在桌面环境中，单击右上角的“用户名”，在弹出的菜单中，单击“设置”，如图2-1所示。

图 2-1 选择设置

弹出用户设置界面，如图2-2所示。用户可以对背景、语言、网络等一系列配置项进行设置。

图 2-2 设置界面

2.2.2 设置语言

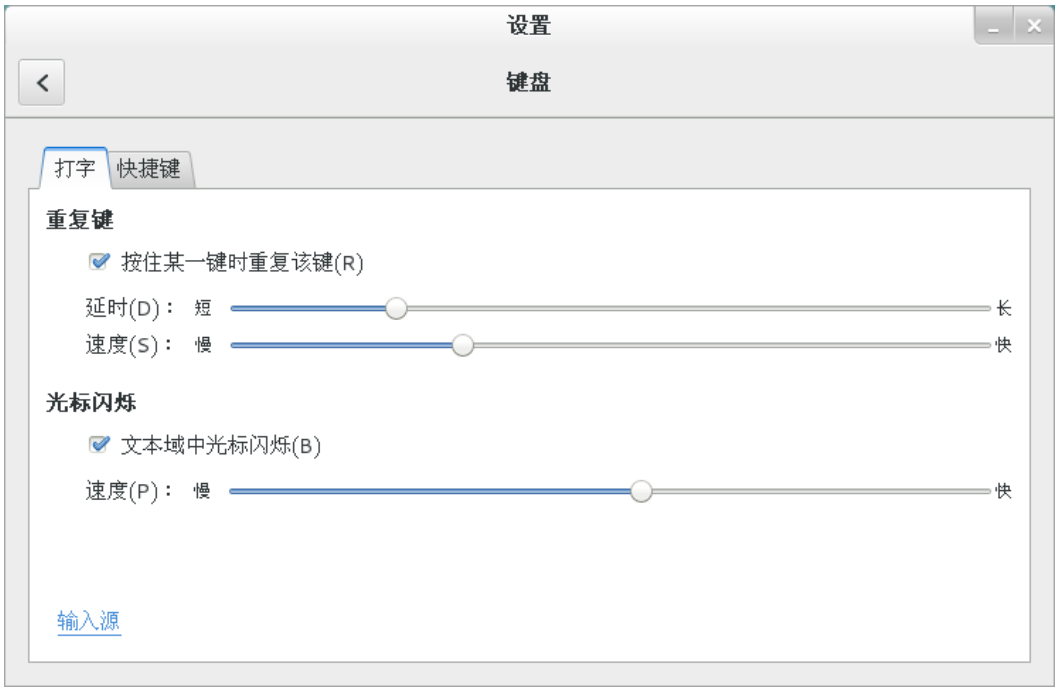
在所示的选项中，单击“区域和语言”。在弹出的设置界面中，如图2-3所示，用户可以根据需要，分别设置语言、格式和输入源。

图 2-3 语言

2.2.3 设置键盘

在设置界面所示的选项中，单击“键盘”。在弹出的设置界面中，如图2-4所示，用户可以根据需要，分别设置重复键、光标闪烁和快捷键等功能。

图 2-4 键盘



2.2.4 设置日期和时间

在设置界面所示的选项中，单击“日期和时间”，弹出修改日期和时间的主界面。

设置时，非root用户需要单击界面右上角“解锁”，输入管理员密码进行解锁，然后进行设置修改，如图2-5所示。

说明

如果安装过程中未创建管理员用户，只创建了普通用户，则此时需要输入root账户密码。

图 2-5 时间和日期

完成认证后，用户可以对地区、城市以及时间等进行修改，如图2-6所示。

图 2-6 时间和日期

3 管理用户

在linux中，每个普通用户都有一个账户，包括用户名、密码和主目录等信息。除此之外，还有一些系统本身创建的特殊用户，它们具有特殊的意义。其中最重要的是管理员账户，它默认用户名是root。同时linux也提供了用户组，使每一个用户至少属于一个组，这样便于进行权限的管理。

用户和组管理是系统安全管理的重要组成部分，本章主要介绍EulerOS提供的用户管理和组管理命令，如何建立多个管理员账户以及为普通用户分配特权的方法。

3.1 增加用户

useradd 命令

通过useradd命令可以为系统添加新用户信息。

useradd [options] LOGIN

useradd可使用的常见选项说明如**表3-1**所示。

表 3-1 参数说明

选项	说明
-c comment	新账号password文件的说明。
-d home dir	新账号每次登入时所使用的主目录(home dir)，默认值为/home/账户名称，并当成登入时的目录名称。
-e expire_date	账号过期日期。日期的指定格式为MM/DD/YY。
-f inactive_days	账号过期几日后永久停用。当值为0时账号则立刻被停用；而当值为-1时，则关闭此功能，默认值为-1。

选项	说明
-g initial_group	group名称或一个数字作为用户的起始群组(group)。
-G	新账户的附加组列表。
-M	如果用户目录不存在，则自动创建。
-n	默认情况下，用户组与用户名称相同，此选项将取消此默认设值。
-r	此参数用来建立系统账号。在EulerOS中系统账号的UID小于500。 **说明** useradd此用法所建立的账号不会建立用户主目录。如果要建立用户主目录需使用-m选项。
-s shell	用户登入后使用的shell名称。
-u uid	用户的ID值，必须唯一，除非用-o选项。数字不可为负值。默认的最小值不得小于99，而且逐次增加。0～99保留给系统账号使用。
-D	当-D选项出现时，useradd显示现在的默认值，或者通过命令行方式更改这些默认值。

用户信息文件

与用户账号信息有关的文件如下：

- /etc/passwd——用户账号信息。
- /etc/shadow——用户账号信息加密文件。
- /etc/group——组信息文件。
- /etc/defaut/useradd——定义默认设置文件。
- /etc/login.defs——系统广义设置文件。
- /etc/skel——默认的初始配置文件目录。

创建用户实例

例如新建一个用户XXX。命令如下：

```
[root@localhost ~]# useradd XXX
```

说明

没有任何提示，表明用户建立成功。这时并没有设置用户的口令，必须使用passwd命令修改用户的密码，没有设置密码的新账号将不能使用。

使用id命令查看新建的用户信息，命令如下：

```
[root@localhost ~]#  id  user_example
uid=502(user_example)    gid=502(user_example)
```

修改用户user_example的密码：

```
[root@localhost ~]# passwd  user_example
```


根据提示两次输入新用户的密码，完成口令更改。过程如下：

```
Changing password for user user_example.
New  password:
BAD PASSWORD: it is based on a dictionary word
Retype  new  password:
passwd: all authentication tokens updated successfully.
```

3.2 修改用户账号

usermod 命令

usermod命令根据命令选项修改系统账号文件里的相应信息。usermod的大多数选项与useradd命令相同，在此用于修改对应的属性，usermod特有的选项如表3-2所示。

表 3-2 参数说明

选项	说明
-a	将用户追加至 -G 中提到的附加组中，并不从其它组中删除此用户。
-L	锁定用户账号。

修改密码

普通用户可以用passwd修改自己的密码，只有管理员才能用passwd usename为其他用户修改密码。

修改用户 shell 设置

使用chsh命令可以修改自己的shell，只有管理员才能用chsh usename为其他用户修改shell设置。

用户也可以使用usermod命令修改shell信息，命令如下：

```
usermod  -s  [new_shell_path] usename
```

其中new_shell_path和usename要取相应的值。

例：将用户user_example的shell改为csh。命令如下：

```
[root@localhost ~]# usermod -s /bin/csh user_example
```

修改主目录

```
usermod  -d  [new_home_directory] usename
```

将用户user_example的主目录更改为/home/user_example，命令如下。

```
[root@localhost ~]#usermod -d /home/user_example user_example
```

如果想将现有主目录的内容转移到新的目录，应该使用-m选项，如下所示：

```
usermod  -d /new/home  -m  usename
```


修改 UID

```
usermod  -u  UID  usename
```

该用户主目录中所拥有的文件和目录都将自动修改UID设置。但是，对于主目录外所拥有的文件，只能手工用chown命令修改所有权设置。

修改账号的有效期

如果使用了影子口令，则可以使用如下命令来修改一个账号的有效期：

```
usermod  -e  MM/DD/YY  usename
```

3.3 删除用户

使用userdel命令可删除现有用户。

例：下面的命令将删除用户Test。命令如下：

```
[root@localhost ~]# userdel Test
```

如果想同时删除该用户的主目录以及其中所有内容，要使用-r参数来递归删除。

说明

无法删除已经进入系统的用户，如果想强行完成，需要先杀死有关的进程，然后再使用userdel命令。

3.4 通过图形界面设置

在图2-2所示的选项中，单击“用户”，弹出用户设置的主界面。

设置时，非root用户需要单击界面右上角“解锁”，输入管理员密码进行解锁，然后进行设置修改，如图3-1所示。

说明

如果安装过程中未创建管理员用户，只创建了普通用户，则此时需要输入root账户密码。

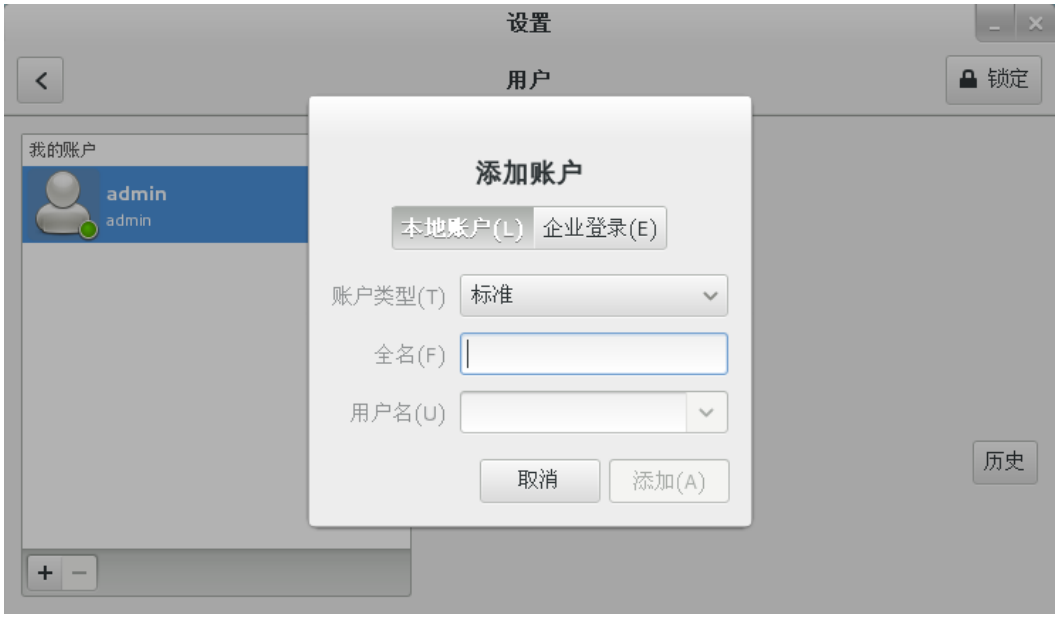
图 3-1 认证

完成认证后，用户单左下角的“+”或者“-”，进行增加/删除用户操作。

3.4.1 添加用户

用户单左下角的“+”（非root用户需要完成认证），添加用户，弹出如图3-2所示。

图 3-2 添加用户



用户需要设置账户类型（有“标准”和“管理员”两种），输入全名并设置用户名。设置完成后，单击“添加”即可，如图3-3所示。

图 3-3 账户

用户创建后，默认是被禁用的，需要设置登录选项。单击“账户已禁用”，弹出密码设置对话框，用户可以根据实际情况选择是否设置密码，如图3-4所示。

图 3-4 密码

3.4.2 删除用户

用户选择需要删除的用户，单左下角的“-”（非root用户需要完成认证）删除用户，弹出如图3-5所示。删除用户时，系统会提示是否删除相关的用户数据，用户需要根据实际情况进行选择。

说明

当前正在登录的用户无法删除。另外，如果当前登录的是普通用户，要删除的是唯一的管理员用户，则操作无法执行。

图 3-5 删除用户

4 使用 yum 管理软件包

yum是一个Shell前端软件包管理器。基于RPM包管理，能够从指定的服务器自动下载RPM包并且安装，可以自动处理依赖性关系，并且一次安装所有依赖的软件包。

4.1 配置 yum

4.1.1 修改配置文件

yum的主要配置文件是/etc/yum.conf，这个文件里面包含“main”部分，保存着yum的全局设置；也可以包含一个或者多个repository部分，用来设置需要安装的软件源位置。在/etc/yum.repos.d目录中有一些repo源相关文件，它们定义了各个仓库。

yum的配置一般有两种方式，一种是直接配置/etc目录下的yum.conf文件，另外一种是在/etc/yum.repos.d目录下增加.repo文件。

修改 main 部分

/etc/yum.conf文件包含一个“main”部分，配置文件示例如下：

```
[main]
cachedir=/var/cache/yum
keepcache=0
debuglevel=2
logfile=/var/log/yum.log
pkgpolicy=newest
tolerant=1
exactarch=1
obsoletes=1
gpgcheck=1
plugins=1
installonly_limit=3

[comments abridged]
```



```
# PUT YOUR REPOS HERE OR IN separate files named file.repo
# in /etc/yum.repos.d
```

说明

关于配置文件的完整说明，请参见man帮助信息的yum.conf(5) 。

常用选项说明：

表 4-1 main 参数说明

参数	说明
cachedir=/var/cache/yum	yum缓存的目录，该目录用于存储下载的RPM软件包和RPM数据库，例如：当您安装一个软件包时，在这个目录下对应的子目录（base/packages）中先下载这个软件包，然后再进行安装软件包，但是安装完后，它不会自动删除这个下载的软件包，所以占空间，但可以手动删除
keepcache=1	值可以是1和0，表示是否要缓存已安装成功的那些RPM包及头文件，默认值为1，表示缓存这些RPM包和头文件，作用：下一次安装相同的RPM包的时，yum就不用下载了，直接可以从缓存（/var/cache/yum…）安装些RPM包
reposdir	设置.repo配置文件所存放的目录，默认是：/etc/yum.repos.d、/etc/yum/repos.d
assumeyes=0	值可以是1和0，表示是否安装RPM包时直接自动确认应答，而不用手动确认，默认值为0：表示要手动确认
alwaysprompt=1	值可以是1和0，表示是否安装RPM包时总是要手动确认，默认值为1：表示要手动确认
retries=2	网络连接发生错误后的重试次数，如果设为0，则会无限重试
debuglevel=2	除错级别：0-10
logfile=/var/log/yum.log	yum的日志文件
pkgpolicy=newest	包的策略：newest和last，作用：如果您设置了多个repository，而同一软件在不同的repository中同时存在，yum应该安装哪一个，如果是newest，则yum会安装最新的那个版本。如果是last，则yum会将服务器id以字母表排序，并选择最后的那个服务器上的软件安装。一般都是选newest
distroverpkg=redhat-release	指定一个软件包，yum会根据这个包判断您的发行版本，默认是redhat-release（例如：centos-release、rpmforge-release等），也可以是安装的任何针对自己发行版的rpm包

参数	说明
tolerant=1	也有1和0两个选项，表示yum是否容忍命令行发生与软件包有关的错误，比如您要安装：1.i386.rpm、2.i386.rpm、3.i386.rpm这三个包，而其中3.i386.rpm在此之前已经安装了，如果现在您将tolerant的值设为1，则yum不会出现错误信息。默认是0
exactarch=1	可选值有两个1和0，表示是否只升级和您安装软件包cpu体系一致的包，默认值为1：表示如果您已经安装了一个i386的包，那么就不会再安装相同的i686的包
obsoletes=1	该参数主要与升级有关
gpgcheck=1	是否进行gpg校验，默认是进行检验的
plugins=1	当要使用plugin（例如：python）时，plugins的值要设置成1
metadata_expire=1800 exclude=…	Metadata的过期时间，单位为秒 排除某些软件在升级名单之外，可以用通配符（例如：*与？），列表中各个项目要用空格隔开，这个对于安装了诸如美化包，中文补丁。

修改 repository 部分

repository部分允许您定义定制化的yum软件源仓库，各个仓库的名称不能相同，否则会引起冲突。下面是[repository]部分的一个最小配置示例：

```
[repository]
name=repository_name
baseurl=repository_url
```

选项说明：

表 4-2 repository 参数说明

参数	说明
name=repository_name	软件仓库（repository ）描述的字符串。
baseurl=repository_url	软件仓库（repository ）的地址。 ● 使用http协议的网络位置：http://path/to/repo ● 使用ftp协议的网络位置：ftp://path/to/repo ● 本地位置：file:///path/to/local/repo

显示当前配置

要显示当前yum参数的值，执行如下命令：

```
yum-config-manager
```

要显示配置文件某一部分参数的值，执行如下命令：


```
yum-config-manager section…
```

您也可以使用一个全局正则表达式，来显示所有匹配部分的配置：

```
yum-config-manager glob_expression…
```

例如要列出的“main”部分的所有配置选项及其对应的值，命令如下：

```
$ yum-config-manager main \*
Loaded plugins: langpacks, product-id, subscription-manager
================================== main ==================================
[main]
alwaysprompt = True
assumeyes = False
bandwith = 0
bugtracker_url = https://bugzilla.redhat.com/enter_bug.cgi?product=Red%20Hat%20Enterprise%20Linux
%206&component=yum
cache = 0
[output truncated]
```

4.1.2 创建 yum 源

要建立一个yum源，请按照下列步骤操作。

1. 安装createrepo软件包。在root权限下执行如下命令：
   ```
   yum install createrepo
   ```
2. 将需要的软件包复制到一个目录下，如/mnt/local_repo/ 。
3. 切换到该目录并运行以下命令：
   ```
   createrepo --database /mnt/local_repo
   ```

4.1.3 添加、启用和禁用 yum 源

本节将介绍如何通过“yum-config-manager”命令添加、启用和禁用yum源。

添加 yum 源

要定义一个新的yum源，您可以在/etc/yum.conf文件中添加“repository”部分，或者在/etc/yum.repos.d/目录下添加“.repo文件”进行说明。建议您通过添加“.repo”的方式来定义yum源，每个yum源都有自己对应的“.repo文件”。

要在您的系统中添加一个这样的源，请在root权限下执行如下命令：

```
yum-config-manager --add-repo repository_url
```

例如要添加位于http://www.example.com/example.repo的源，命令如下：

```
# yum-config-manager --add-repo http://www.example.com/example.repo
Loaded plugins: langpacks, product-id, subscription-manager
adding repo from: http://www.example.com/example.repo
grabbing file http://www.example.com/example.repo to /etc/yum.repos.d/example.repo
example.repo                                             |  413 B     00:00
repo saved to /etc/yum.repos.d/example.repo
```

启用 yum 源

要启用yum源，请在root权限下执行如下命令：

```
yum-config-manager --enable repository…
```

您也可以使用一个全局正则表达式，来启用所有匹配的yum源：

```
yum-config-manager --enable glob_expression…
```

例如要启用已定义的example、example-debuginfo和example-source源，命令如下：

```
# yum-config-manager --enable example\*
Loaded plugins: langpacks, product-id, subscription-manager
============================== repo: example ==============================
[example]
bandwidth = 0
base_persistdir = /var/lib/yum/repos/x86_64/6Server
baseurl = http://www.example.com/repo/6Server/x86_64/
cache = 0
cachedir = /var/cache/yum/x86_64/6Server/example
[output truncated]
```

禁用 yum 源

要禁用yum源，请在root权限下执行如下命令：

```
yum-config-manager --disable repository…
```

同样的，您也可以使用一个全局正则表达式，来禁用所有匹配的yum源：

```
yum-config-manager --disable glob_expression…
```

4.2 管理软件包

使用yum能够让您方便的进行查询、安装、删除软件包等操作。

搜索软件包

您可以使用RPM包名称、缩写或者描述搜索需要的RPM包，使用命令如下：

```
yum search term…
```

例如搜索，命令如下：

```
$ yum search meld kompare
Loaded plugins: langpacks, langpacks, product-id, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
=========================== N/S matched: kompare ============================
kompare.x86_64 : Diff tool
...

  Name and summary matches mostly, use "search all" for everything.
Warning: No matches found for: meld
```

列出软件包清单

要列出系统中所有已安装的以及可用的RPM包信息，使用命令如下：

```
yum list all
```

要列出系统中特定的RPM包信息，使用命令如下：

```
yum list glob_expression…
```

例如列出ABRT相关的RPM包，命令如下：

```
$ yum list abrt-addon\* abrt-plugin\*
Loaded plugins: langpacks, product-id, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
Installed Packages
abrt-addon-ccpp.x86_64                          1.0.7-5.el6             @rhel
```

```
abrt-addon-kerneloops.x86_64                1.0.7-5.el6             @rhel
abrt-addon-python.x86_64                    1.0.7-5.el6             @rhel
abrt-plugin-bugzilla.x86_64                 1.0.7-5.el6             @rhel
abrt-plugin-logger.x86_64                   1.0.7-5.el6             @rhel
abrt-plugin-sosreport.x86_64                1.0.7-5.el6             @rhel
abrt-plugin-ticketuploader.x86_64           1.0.7-5.el6             @rhel
```

显示 RPM 包信息

要显示一个或者多个RPM包信息，使用命令如下：

```
yum info package_name…
```

例如搜索，命令如下：

```
$ yum info httpd
Available Packages
Name        : httpd
Arch        : x86_64
Version     : 2.4.6
Release     : 40.4.h1
Size        : 1.2 M
Repo        : EulerOS-base
Summary     : Apache HTTP Server
URL         : http://httpd.apache.org/
License     : ASL 2.0
Description : The Apache HTTP Server is a powerful, efficient, and extensible
            : web server.
```

安装 RPM 包

要安装一个软件包及其所有未安装的依赖，请在root权限下执行如下命令：

```
yum install package_name
```

您也可以通过添加软件包名字同时安装多个软件包。请在root权限下执行如下命令：

```
yum install package_name package_name…
```

例如在i686体系结构上安装的源码包，命令如下：

```
# yum install sqlite.i686
```

下载软件包

在使用yum安装过程中，可能会有提示信息需要您确认，如下：

```
...
Total size: 1.2 M
Is this ok [y/d/N]:
...
```

输入d，yum会下载相应的软件包，但是并不会安装。您可以根据自己需要在离线状态下安装这些软件包。下载的软件包默认保存在“/var/cache/yum/$basearch/$releasever/packages/”目录。

删除软件包

要卸载软件包以及相关的依赖软件包，请在root权限下执行如下命令：

```
yum remove package_name…
```

例如删除totem包，命令如下：

```
# yum remove totem
```


4.3 管理软件包组

软件包集合是服务于一个共同的目的一组软件包，例如系统工具集等。使用yum可以对软件包组进行安装/删除等操作，使相关操作更高效。

列出软件包组清单

使用summary参数，可以列出系统中所有已安装软件包组、可用的组，可用的环境组的数量，命令如下：

```
yum groups summary
```

使用实例如下：

```
$ yum groups summary
Loaded plugins: langpacks, product-id, subscription-manager
Available Environment Groups: 12
Installed Groups: 10
Available Groups: 12
```

要列出所有软件包组和它们的组ID ，命令如下：

```
yum group list ids
```

要列出yum库中特定的软件包组，可添加list选项，命令如下：

```
$ yum group list ids kde\*
Available environment groups:
   KDE Plasma Workspaces (kde-desktop-environment)
Done
```

显示软件包组信息

要列出包含在一个软件包组中必须安装的包和可选包，使用命令如下：

```
yum group info glob_expression…
```

例如显示LibreOffice软件包组信息，示例如下：

```
$ yum group info LibreOffice
Loaded plugins: langpacks, product-id, subscription-manager

Group: LibreOffice
 Group-Id: libreoffice
 Description: LibreOffice Productivity Suite
 Mandatory Packages:
  =libreoffice-calc
   libreoffice-draw
  -libreoffice-emailmerge
   libreoffice-graphicfilter
  =libreoffice-impress
  =libreoffice-math
  =libreoffice-writer
  +libreoffice-xsltfilter
 Optional Packages:
   libreoffice-base
   libreoffice-pyuno
```

安装软件包组

每一个软件包组都有自己的名称以及相应的ID（groupid），您可以使用软件包组名称或它的ID进行安装。

要安装一个软件包组，请在root权限下执行如下命令：

```
yum group install "group name"
yum group install groupid
```

例如安装KDE桌面相应的软件包组，命令如下：

```
# yum group install "KDE Desktop"
# yum group install kde-desktop
```

删除软件包组

要卸载软件包组，您可以使用软件包组名称或它的ID，在root权限下执行如下命令：

```
yum group remove group_name
yum group remove groupid
```

例如删除KDE桌面相应的软件包组，命令如下：

```
# yum group remove "KDE Desktop"
# yum group remove kde-desktop
```

4.4 检查并更新

yum可以检查您的系统中是否有软件包需要更新。您可以通过yum列出需要更新的软件包，一次性全部更新或者选择对应的包单独更新。

检查更新

如果您需要显示当前系统可用的更新，使用命令如下：

```
yum check-update
```

使用实例如下：

```
# yum check-update
Loaded plugins: langpacks, product-id, subscription-manager
Updating Red Hat repositories.
INFO:rhsm-app.repolib:repos updated: 0
PackageKit.x86_64                   0.5.8-2.el6                rhel
PackageKit-glib.x86_64              0.5.8-2.el6                rhel
PackageKit-yum.x86_64               0.5.8-2.el6                rhel
PackageKit-yum-plugin.x86_64        0.5.8-2.el6                rhel
glibc.x86_64                        2.11.90-20.el6             rhel
glibc-common.x86_64                 2.10.90-22                 rhel
kernel.x86_64                       2.6.31-14.el6              rhel
rpm.x86_64                          4.7.1-5.el6                rhel
rpm-libs.x86_64                     4.7.1-5.el6                rhel
rpm-python.x86_64                   4.7.1-5.el6                rhel
yum.noarch                          3.2.24-4.el6               rhel
```

升级

如果您需要升级单个软件包，在root权限下使用命令如下：

```
yum update package_name
```

例如升级rpm包，示例如下：

```
# yum update rpm
Loaded plugins: langpacks, product-id, subscription-manager
Updating EulerOS repositories.
INFO:rhsm-app.repolib:repos updated: 0
Setting up Update Process
```



```
Resolving Dependencies
--> Running transaction check
---> Package rpm.x86_64 0:4.11.1-3.el7 will be updated
--> Processing Dependency: rpm = 4.11.1-3.el7 for package: rpm-libs-4.11.1-3.el7.x86_64
--> Processing Dependency: rpm = 4.11.1-3.el7 for package: rpm-python-4.11.1-3.el7.x86_64
--> Processing Dependency: rpm = 4.11.1-3.el7 for package: rpm-build-4.11.1-3.el7.x86_64
---> Package rpm.x86_64 0:4.11.2-2.el7 will be an update
--> Running transaction check
...
--> Finished Dependency Resolution

Dependencies Resolved
================================================================================
 Package                    Arch        Version          Repository       Size
================================================================================
Updating:
 rpm                        x86_64      4.11.2-2.el7     rhel            1.1 M
Updating for dependencies:
 rpm-build                  x86_64      4.11.2-2.el7     rhel            139 k
 rpm-build-libs             x86_64      4.11.2-2.el7     rhel             98 k
 rpm-libs                   x86_64      4.11.2-2.el7     rhel            261 k
 rpm-python                 x86_64      4.11.2-2.el7     rhel             74 k

Transaction Summary
================================================================================
Upgrade  1 Package (+4 Dependent packages)

Total size: 1.7 M
Is this ok [y/d/N]:
```

类似的，如果您需要升级软件包组，在root权限下使用命令如下：

```
yum group update group_name
```

更新所有的包和它们的依赖

要更新所有的包和它们的依赖，在root权限下使用命令如下：

```
yum update
```


5 管理服务

本章介绍如何使用systemd进行系统和服务管理。

5.1 简介

systemd是Linux下一个与SysV和LSB初始化脚本兼容的系统和服务管理器。systemd使用socket和D-Bus来开启服务，提供基于守护进程的按需启动策略，支持快照和系统状态恢复，维护挂载和自挂载点，实现了各服务间基于从属关系的一个更为精细的逻辑控制，拥有更高的并行性能。

概念介绍

systemd开启和监督整个系统是基于unit的概念。unit是由一个与配置文件对应的名字和类型组成的（例如：avahi.service unit有一个具有相同名字的配置文件，是守护进程Avahi的一个封装单元）。unit有多重类型，如**表5-1**所示。

表 5-1 units 说明

unit名称	后缀名	描述
Service unit	.service	系统服务。
Target unit	.target	一组systemd units。
Automount unit	.automount	文件系统挂载点。
Device unit	.device	内核识别的设备文件。
Mount unit	.mount	文件系统挂载点。

unit名称	后缀名	描述
Path unit	.path	在一个文件系统中的文件或目录。
Scope unit	.scope	外部创建的进程。
Slice unit	.slice	一组用于管理系统进程分层组织的units。
Snapshot unit	.snapshot	systemd manager的保存状态。
Socket unit	.socket	一个进程间通信的Socket。
Swap unit	.swap	swap设备或者swap文件。
Timer unit	.timer	systemd计时器。

所有的可用systemd unit类型，可在如表5-2所示的路径下查看。

表 5-2 可用 systemd unit 类型

路径	描述
/usr/lib/systemd/system/	随安装的RPM产生的systemd units。
/run/systemd/system/	在运行时创建systemd units。
/etc/systemd/system/	由系统管理员创建和管理的systemd units。

5.2 特性说明

更快的启动速度

Systemd提供了比UpStart更激进的并行启动能力，采用了socket/D-Bus activation等技术启动服务，带来了更快的启动速度。

为了减少系统启动时间，systemd的目标是：

- 尽可能启动更少的进程
- 尽可能将更多进程并行启动

同样地，UpStart也试图实现这两个目标。UpStart采用事件驱动机制，服务可以暂不启动，当需要的时候才通过事件触发其启动，这符合第一个设计目标；此外，不相干的服务可以并行启动，这也实现了第二个目标。

提供按需启动能力

当sysvinit系统初始化的时候，它会将所有可能用到的后台服务进程全部启动运行。并且系统必须等待所有的服务都启动就绪之后，才允许用户登录。这种做法有两个缺点：首先是启动时间过长；其次是系统资源浪费。

某些服务很可能在很长一段时间内，甚至整个服务器运行期间都没有被使用过。比如CUPS，打印服务在多数服务器上很少被真正使用到。您可能没有想到，在很多服务器

上SSHD也是很少被真正访问到的。花费在启动这些服务上的时间是不必要的；同样，花费在这些服务上的系统资源也是一种浪费。

Systemd可以提供按需启动的能力，只有在某个服务被真正请求的时候才启动它。当该服务结束，systemd可以关闭它，等待下次需要时再次启动它。

采用 Cgroup 特性跟踪和管理进程的生命周期

init系统的一个重要职责就是负责跟踪和管理服务进程的生命周期。它不仅可以启动一个服务，也能够停止服务。这看上去没有什么特别的，然而在真正用代码实现的时候，您或许会发现停止服务比一开始想的要困难。

服务进程一般都会作为精灵进程（daemon）在后台运行，为此服务程序有时候会派生(fork)两次。在UpStart中，需要在配置文件中正确地配置expect小节。这样UpStart通过对fork系统调用进行计数，从而获知真正的精灵进程的PID号。

CGroup已经出现了很久，它主要用来实现系统资源配额管理。CGroup提供了类似文件系统的接口，使用方便。当进程创建子进程时，子进程会继承父进程的CGroup。因此无论服务如何启动新的子进程，所有的这些相关进程都会属于同一个CGroup，systemd只需要简单地遍历指定的CGroup即可正确地找到所有的相关进程，将它们逐一停止即可。

启动挂载点和自动挂载的管理

传统的Linux系统中，用户可以用/etc/fstab文件来维护固定的文件系统挂载点。这些挂载点在系统启动过程中被自动挂载，一旦启动过程结束，这些挂载点就会确保存在。这些挂载点都是对系统运行至关重要的文件系统，比如HOME目录。和sysvinit一样，Systemd管理这些挂载点，以便能够在系统启动时自动挂载它们。Systemd还兼容/etc/fstab文件，您可以继续使用该文件管理挂载点。

有时候用户还需要动态挂载点，比如打算访问DVD内容时，才临时执行挂载以便访问其中的内容，而不访问光盘时该挂载点被取消(umount)，以便节约资源。传统地，人们依赖autofs服务来实现这种功能。

Systemd内建了自动挂载服务，无需另外安装autofs服务，可以直接使用systemd提供的自动挂载管理能力来实现autofs的功能。

实现事务性依赖关系管理

系统启动过程是由很多的独立工作共同组成的，这些工作之间可能存在依赖关系，比如挂载一个NFS文件系统必须依赖网络能够正常工作。Systemd虽然能够最大限度地并发执行很多有依赖关系的工作，但是类似“挂载NFS”和“启动网络”这样的工作还是存在天生的先后依赖关系，无法并发执行。对于这些任务，systemd维护一个“事务一致性”的概念，保证所有相关的服务都可以正常启动而不会出现互相依赖，以至于死锁的情况。

与 SysV 初始化脚本兼容

和UpStart一样，systemd引入了新的配置方式，对应用程序的开发也有一些新的要求。如果systemd想替代目前正在运行的初始化系统，就必须和现有程序兼容。任何一个Linux发行版都很难为了采用systemd而在短时间内将所有的服务代码都修改一遍。

Systemd提供了和Sysvinit以及LSB initscripts兼容的特性。系统中已经存在的服务和进程无需修改。这降低了系统向systemd迁移的成本，使得systemd替换现有初始化系统成为可能。

能够对系统进行快照和恢复

systemd支持按需启动，因此系统的运行状态是动态变化的，人们无法准确地知道系统当前运行了哪些服务。Systemd快照提供了一种将当前系统运行状态保存并恢复的能力。

比如系统当前正运行服务A和B，可以用systemd命令行对当前系统运行状况创建快照。然后将进程A停止，或者做其他的任意的对系统的改变，比如启动新的进程C。在这些改变之后，运行systemd的快照恢复命令，就可立即将系统恢复到快照时刻的状态，即只有服务A，B在运行。一个可能的应用场景是调试：比如服务器出现一些异常，为了调试用户将当前状态保存为快照，然后可以进行任意的操作，比如停止服务等等。等调试结束，恢复快照即可。

5.3 管理系统服务

systemd提供systemctl命令来运行、关闭、重启、显示、启用/禁用系统服务。

sysvinit 命令和 Systemd 命令

systemd提供systemctl命令与sysvinit命令的功能类似。当前版本中依然兼容service和chkconfig命令，相关说明如表 1，但建议用户使用systemd进行系统服务管理。

表 5-3 sysvinit 命令和 Systemd 命令的对照表

Sysvinit命令	Systemd命令	备注
service foo start	systemctl start foo.service	用来启动一个服务 (并不会重启现有的)。
service foo stop	systemctl stop foo.service	用来停止一个服务 (并不会重启现有的)。
service foo restart	systemctl restart foo.service	用来停止并启动一个服务。
service foo reload	systemctl reload foo.service	当支持时，重新装载配置文件而不中断等待操作。
service foo condrestart	systemctl condrestart foo.service	如果服务正在运行那么重启它。
service foo status	systemctl status foo.service	汇报服务是否正在运行。
chkconfig foo on	systemctl enable foo.service	在下次启动时或满足其他触发条件时设置服务为启用。
chkconfig foo off	systemctl disable foo.service	在下次启动时或满足其他触发条件时设置服务为禁用。
chkconfig foo	systemctl is-enabled foo.service	用来检查一个服务在当前环境下被配置为启用还是禁用。

Sysvinit命令	Systemd命令	备注
chkconfig –list	systemctl list-unit-files --type=service	输出在各个运行级别下服务的启用和禁用情况。
chkconfig foo –list	ls /etc/systemd/system/*.wants/foo.service	用来列出该服务在哪些运行级别下启用和禁用。
chkconfig foo –add	systemctl daemon-reload	当您创建新服务文件或者变更设置时使用。

显示所有当前服务

如果您需要显示当前正在运行的服务，使用命令如下：

```
systemctl list-units --type service
```

如果您需要显示所有的服务（包括未运行的服务），需要添加-all参数，使用命令如下：

```
systemctl list-units --type service --all
```

例如显示当前正在运行的服务，命令如下：

```
$ systemctl list-units --type service
UNIT                            LOAD   ACTIVE SUB     DESCRIPTION
abrt-ccpp.service               loaded active exited  Install ABRT coredump hook
abrt-oops.service               loaded active running ABRT kernel log watcher
abrt-vmcore.service             loaded active exited  Harvest vmcores for ABRT
abrt-xorg.service               loaded active running ABRT Xorg log watcher
abrtd.service                   loaded active running ABRT Automated Bug Reporting Tool
...
systemd-vconsole-setup.service loaded active exited  Setup Virtual Console
tog-pegasus.service             loaded active running OpenPegasus CIM Server

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

46 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'
```

显示服务状态

如果您需要显示某个服务的状态，可执行如下命令：

```
systemctl status name.service
```

相关状态显示参数说明如表5-4所示。

表 5-4 状态参数说明

参数	描述
Loaded	说明服务是否被加载，并显示服务对应的绝度路径以及是否启用。
Active	说明服务是否正在运行，并显示时间节点。
Main PID	相应的系统服务的PID值。

参数	描述
Status	相关系统服务的其他信息
Process	相关进程的其他信息。
CGroup	相关控制组（CGroup）的其他信息。

如果您需要鉴别某个服务是否运行，可执行如下命令：

```
systemctl is-active name.service
```

同样，如果您需要判断某个服务是否被启用，可执行如下命令：

```
systemctl is-enabled name.service
```

例如查看gdm.service服务状态，命令如下：

```
# systemctl status gdm.service
gdm.service - GNOME Display Manager
   Loaded: loaded (/usr/lib/systemd/system/gdm.service; enabled)
   Active: active (running) since Thu 2013-10-17 17:31:23 CEST; 5min ago
 Main PID: 1029 (gdm)
   CGroup: /system.slice/gdm.service
           ├─1029 /usr/sbin/gdm
           ├─1037 /usr/libexec/gdm-simple-slave --display-id /org/gno...
           └─1047 /usr/bin/Xorg :0 -background none -verbose -auth /r...

Oct 17 17:31:23 localhost systemd[1]: Started GNOME Display Manager.
```

运行服务

如果您需要运行某个服务，请在root权限下执行如下命令：

```
systemctl start name.service
```

例如运行httpd服务，命令如下：

```
# systemctl start httpd.service
```

关闭服务

如果您需要关闭某个服务，请在root权限下执行如下命令：

```
systemctl stop name.service
```

例如关闭蓝牙服务，命令如下：

```
# systemctl stop bluetooth.service
```

重启服务

如果您需要重启某个服务，请在root权限下执行如下命令：

```
systemctl restart name.service
```

执行命令后，当前服务会被关闭，但马上重新启动。如果您指定的服务，当前处于关闭状态，执行命令后，服务也会被启动。

例如重启蓝牙服务，命令如下：

```
# systemctl restart bluetooth.service
```


启用服务

如果您需要在开机时启用某个服务，请在root权限下执行如下命令：

```
systemctl enable name.service
```

例如设置httpd服务开机时启动，命令如下：

```
# systemctl enable httpd.service
ln -s '/usr/lib/systemd/system/httpd.service' '/etc/systemd/system/multi-user.target.wants/
httpd.service'
```

禁用服务

如果您需要在开机时禁用某个服务，请在root权限下执行如下命令：

```
systemctl disable name.service
```

例如在开机时禁用蓝牙服务启动，命令如下：

```
# systemctl disable bluetooth.service
rm '/etc/systemd/system/dbus-org.bluez.service'
rm '/etc/systemd/system/bluetooth.target.wants/bluetooth.service'
```

5.4 改变运行级别

Target 和运行级别

systemd用目标（target）替代了运行级别的概念，提供了更大的灵活性，如您可以继承一个已有的目标，并添加其他服务，来创建自己的目标。下表列举了systemd下的目标和常见runlevel的对应关系：

表 5-5 运行级别和 systemd 目标

运行级别	systemd目标（target）	描述
0	runlevel0.target，poweroff.target	关闭系统。
1	runlevel1.target，rescue.target	单用户模式。
2	runlevel2.target，multi-user.target	用户定义/域特定运行级别。默认等同于3。
3	runlevel3.target，multi-user.target	多用户，非图形化。用户可以通过多个控制台或网络登录。
4	runlevel4.target，multi-user.target	用户定义/域特定运行级别。默认等同于3。
5	runlevel5.target，graphical.target	多用户，图形化。通常为所有运行级别3的服务外加图形化登录。
6	runlevel6.target，reboot.target	重启系统。

查看系统默认目标

查看当前系统默认的启动级别，使用如下命令：

```
systemctl get-default
```

示例如下：

```
$ systemctl get-default
graphical.target
```

查看当前系统的目标

查看当前系统默认的启动级别，使用如下命令：

```
systemctl list-units --type target
```

示例如下：

```
$ systemctl list-units --type target
UNIT                   LOAD   ACTIVE SUB    DESCRIPTION
basic.target           loaded active active Basic System
cryptsetup.target      loaded active active Encrypted Volumes
getty.target           loaded active active Login Prompts
graphical.target       loaded active active Graphical Interface
local-fs-pre.target    loaded active active Local File Systems (Pre)
local-fs.target        loaded active active Local File Systems
multi-user.target      loaded active active Multi-User System
network.target         loaded active active Network
paths.target           loaded active active Paths
remote-fs.target       loaded active active Remote File Systems
sockets.target         loaded active active Sockets
sound.target           loaded active active Sound Card
spice-vdagentd.target loaded active active Agent daemon for Spice guests
swap.target            loaded active active Swap
sysinit.target         loaded active active System Initialization
time-sync.target       loaded active active System Time Synchronized
timers.target          loaded active active Timers

LOAD   = Reflects whether the unit definition was properly loaded.
ACTIVE = The high-level unit activation state, i.e. generalization of SUB.
SUB    = The low-level unit activation state, values depend on unit type.

17 loaded units listed. Pass --all to see loaded but inactive units, too.
To show all installed unit files use 'systemctl list-unit-files'.
```

改变默认目标

改变系统默认的目标，在root权限下使用如下命令：

```
systemctl set-default name.target
```

示例如下：

```
# systemctl set-default multi-user.target
rm '/etc/systemd/system/default.target'
ln -s '/usr/lib/systemd/system/multi-user.target' '/etc/systemd/system/default.target'
```

改变当前目标

改变当前系统的目标，在root权限下使用如下命令：

```
ssystemctl isolate name.target
```

示例如下：


```
# systemctl isolate multi-user.target
```

切换到救援模式

改变当前系统为救援模式，在root权限下使用如下命令：

```
systemctl rescue
```

这条命令和“systemctl isolate rescue.target”类似，但它会给当前所有的登录用户发送一条提示消息。如果想避免systemd发送这个消息，您可以添加“--no-wall”参数。具体命令如下：

```
systemctl --no-wall rescue
```

示例如下：

```
# systemctl rescue
Broadcast message from root@localhost on pts/0 (Fri 2013-10-25 18:23:15 CEST):
The system is going down to rescue mode NOW!
```

切换到紧急模式

改变当前系统为紧急模式，在root权限下使用如下命令：

```
systemctl emergency
```

这条命令和“systemctl isolate emergency.target”类似，但它会给当前所有的登录用户发送一条提示消息。如果想避免systemd发送这个消息，您可以添加“--no-wall”参数。具体命令如下：

```
systemctl --no-wall emergency
```

示例如下：

```
# systemctl --no-wall emergency
```

5.5 关闭、暂停和休眠系统

systemctl 命令

systemd通过systemctl命令可以对系统进行关机、重启、休眠等已系列操作。当前仍兼容linux常用管理命令，对应关系如下表。建议用户使用systemctl命令进行操作。

表 5-6 命令对应关系

之前命令	systemctl命令	描述
halt	systemctl halt	关闭系统
poweroff	systemctl poweroff	关闭电源
reboot	systemctl reboot	重启
pm-suspend	systemctl suspend	待机
pm-hibernate	systemctl hibernate	休眠
pm-suspend-hybrid	systemctl hybrid-sleep	缓和休眠模式

关闭系统

要关闭系统并下电，在root权限下执行如下命令：

```
systemctl poweroff
```

要关闭系统但不下电机器，在root权限下执行如下命令：

```
systemctl halt
```

执行上述会给当前所有的登录用户发送一条提示消息。如果想避免systemd发送这个消息，您可以添加“--no-wall”参数。具体命令如下：

```
systemctl --no-wall poweroff
```

重启系统

要重启系统，在root权限下执行如下命令：

```
systemctl reboot
```

执行上述会给当前所有的登录用户发送一条提示消息。如果想避免systemd发送这个消息，您可以添加“--no-wall”参数。具体命令如下：

```
systemctl --no-wall reboot
```

使系统待机

要使系统待机，在root权限下执行如下命令：

```
systemctl suspend
```

使系统休眠

要使系统休眠，在root权限下执行如下命令：

```
systemctl hibernate
```

要使系统待机且处于休眠状态，在root权限下执行如下命令：

```
systemctl hybrid-sleep
```


6 管理进程

本章介绍了Linux内核的进程管理方式，然后以实例的方式讲解了Linux提供的常用的进程控制命令、at和cron服务，以及进程查看命令。

6.1 管理系统进程

操作系统管理多个用户的请求和多个任务。大多数系统都只有一个CPU和一个主要存储，但一个系统可能有多个二级存储磁盘和多个输入/输出设备。操作系统管理这些资源并在多个用户间共享资源，当用户提出一个请求时，造成好像系统被用户独占的假象。实际上操作系统监控着一个等待执行的任务队列，这些任务包括用户任务、操作系统任务、邮件和打印任务等。本节将从用户的角度讲述如何控制进程。

6.1.1 手工启动进程

在Linux系统中每个进程都具有一个进程号，用于进程调度。

启动一个进程有两个主要途径：手工启动和调度启动，后者是事先进行设置，根据用户要求自行启动的。

由用户在shell提示符后输八命令，手工启动一个进程。手工启动进程又可以分为前台启动和后台启动。

- 前台启动是手工启动进程最常用的方式。例如用户键入一个命令df -h，就启动了一个进程，而且是一个前台进程，这时候系统已经处于一个多进程状态。

  说明

  当执行df -h命令以后，紧接着使用ps -x查看，却没有看到df进程，原因是df进程执行太快，执行ps时谊进程已经执行结束了。

- 直接从后台手工启动一个进程不太常用，除非是该进程执行时间很长，且用户也不急于查看执行输出。假设用户要启动一个需要长时间运行的格式化文本文件的进程。为了在整个shell在格式化过程中，还能使用该shell，从后台启动这个进程是明智的选择。

说明

对于手工后台启动一个进程，需要命令之后加一个“&”字符。例如手工后台启动slocate程序，更新搜索库的命令如下所示。

slocate -u&

6.1.2 调度启动进程

有时候需要对系统进行一些比较费时而且占用资源的维护工作，这些工作适合在深夜进行，这时候用户就珂以事先进行调度安排，指定任务运行的时间或者场合，到时候系统会自动完成这些任务。要使用自动启动进程的功能，就需要掌握以下几个启动命令。

6.1.2.1 定时运行一批程序（at）

at 命令

用户使用at命令在指定时刻执行指定的命令序列。该命令至少需要指定一个命令和一个执行时间。at命令可以只指定时间，也可以时间和日期一起指定。

at命令的语法格式如下：

```
at [-V] [-q 队列] [-f 文件名] [-mldbv] 时间
at -c作业[作业…]
```

设置时间

at允许使用一套相当复杂的时间指定方法，比如：

- 接受在当天的hh:mm（小时：分钟）式的时间指定。如果该时间已经过去，那么就放存第二天执行。
- 使用midnight（深夜）、noon（中午）、teatime（饮茶时间，一般是下午4点）等比较模糊的词语来指定时间。
- 采用12小时计时制，即在时间后面加上AM（上午）或者PM（下午）来说明是上午还是下午。
- 指定命令执行的具体日期，指定格式为month day（月日）或者mm/dd/yy（月/日/年）或者dd.mm.yy（日.月.年）。指定的日期必须跟在指定时间的后面。

上面介绍的都是绝对计时法，其实还可以使用相对计时法，这对于安排不久就要执行的命令是很有好处的。指定格式为now+count time-units，now就是当前时间，time-units是时间单位，这里可以是minutes（分钟）、hours（小时）、days（天）、weeks（星期）。count是时间的数量，究竟是几天，还是几小时等。还有一种计时方法就是直接使用today（今天）、tomorrow（明天）来指定完成命令的时间。下面通过一些例子来说明具体用法。

例如指定在今天下午4:30执行某个命令。假设现在时间是中午12:30，2015年6月7日，可用命令格式如下：

```
at 4:30pm
at 16:30
at 16:30 today
at now+4 hours
at now+ 240 minutes
at 16:30 7.6.15
at 16:30 6/7/15
at 16:30 Jun 7
```


以上这些命令表达的意义是完全一样的，所以在安排时间的时候完全可以根据个人喜好和具体情况自由选择。一般采用绝对时间的24小时计时法可以避免由于用户自己的疏忽造成计时错误，例如上例可以写成：at 16:30 6/7/15。

执行权限

对于at命令来说，需要定时执行的命令是从标准输入或者使用-f选项指定的文件中读取并执行的。如果at命令是从一个使用su命令切换到用户shell中执行的，那么当前用户被认为是执行用户，所有的错误和输出结果都会送给这个用户。但是如果有邮件送出的话，收到邮件的将是原来的用户，也就是登录时shell的所有者。

例如在6月8日上午10点执行slocate -u命令。命令如下：

```
# at  10:00  6/8/15
at> slocate -u
at>
[1]+   Stopped    at  10:00  6/8/15
```

上面的结果中，输入at命令之后，会出现提示符at>，提示用户输入命令，在此输入了slocate -u，然后按回车键。还可以输入多条命令，当所有要执行的命令输入结束后，按Ctrl+d键结束at命令。

在任何情况下，管理员账户都可以使用这个命令。对于其他用户来说，是否可以使用就取决于/etc/at.allow和/etc/at.deny文件。

6.1.2.2 周期性运行一批程序（cron）

前面介绍at命令都会在一定时间内完成一定任务，但是它只能执一次。也就是说，当指定了运行命令后，系统在指定时间完成任务，以后就不再执行了。但是在很多情况下需要周期性重复执行一些命令，这时候就需要使用cron命令来完成任务。

运行机制

首先cron命令会搜索/var/spool/cron目录，寻找以/etc/passwd文件中的用户名命名的crontab文件，被找到的这种文件将装入内存。比如一个用户名为globus的用户，对应的crontab立件应该是/var/spool/cron/globus，即以该用户命名的crontab文件存放在/var/spool/cron目录下面。

cron命令还将搜索/etc/crontab文件，这个文件是用不同的格式写成的。cron启动以后，它将首先检查是否有用户设置了crontab文件，如果没有就转入睡眠状态，释放系统资源。所以该后台进程占用资源极少，它每分钟被换醒一次，查看当前是否有需要运行的命令。

命令执行结束后，任何输出都将作为邮件发送给crontab的所有者，或者是/etc/crontab文件中MAILTO环境变量中指定的用户。这是cron的工作原理，但是cron命令的执行不需要用户干涉，用户只需要修改crontab中要执行的命令。

crontab 命令

crontab命令用于安装、删除或者显示用于驱动cron后台进程的表格。用户把需要执行的命令序列放到crontab文件中以获得执行，而且每个用户都可以有自己的crontab文件。

crontab命令的常用方法如下：

- crontab -u //设置某个用户的cron服务，root用户在执行crontab时需要此参数。
- crontab -l //列出某个用户cron服务的详细内容。

- crontab -r //删除某个用户的cron服务。
- crontab -e //编辑某个用户的cron服务。

例如root查看自己的cron设置。命令如下：

```
crontab -u root -l
```

crontab 文件

在crontab文件中输入需要执行的命令和时间。该文件中每行都包括6个域，其中前5个域是指定命令被执行的时间，最后一个域是要被执行的命令。每个域之间使用空格或者制表符分隔。格式如下：

```
minute hour day-of-month month-of-year day-of-week commands
```

对于每一项的说明如所示。

表 6-1 参数说明

参数	描述
minute	分钟（0~59）。
hour	小时（0~23）。
day-of-month	一个月（）的第几天（1~31）。
month-of-year	一年的第几个月（1~12）。
day-of-week	一周的星期几（0~6），0代表星期天。
commands	需要执行的命令。

这些项都不能为空，必须指定值。除了数字还有几个特殊的符号“*”、“/”和“-”、“，”。其中，*代表所有的取值范围内的数字，/代表每的意思，“*/5”表示每5个单位，“-”代表从某个数字到某个数字，“，”分开几个离散时数字。对于要执行的命令，调用的时候需要写出命令的完整路径。

例如晚上11点到早上8点之间每两个小时，在/tmp/test.txt文件中加入sleepy文本。在crontab文件中对应的行如下：

```
* 23-8/2 * * * echo"sleepy" >> /tmp/test.txt
```

每次编辑完某个用户的cron设置后，cron自动在/var/spool/cron下生成一个与此用户同名的文件。此用户的cron信息都记录在这个文件中，这个文件是不可以直接编辑的，只可以用crontab -e来编辑。用户也可以另外建立一个文件，使用“cron文件名”命令导入cron设置。

假设有个用户名为globus，它需要为自己创建的一个crontab文件。步骤如下：

1. 首先可以使用任何文本编辑器建立一个新文件，并将向该文件加入需要运行的命令和要定期执行的时间，假发该文件为 ~/globus.cron。
2. 然后使用crontab命令安装这个文件，使用crontab命令使之成为该用户的crontab文件。命令如下：
   ```
   crontab  globus. -/globus.cron
   ```


这样crontab文件就建立好了，可以转到/var/spool/cron目录下面查看，发现多了一个globus文件。这个文件就是所需的crontab文件。

说明

cron启动后，每过一分钟读一次crontab文件，检查是否要执行里面的命令。因此该文件被修改后不需要重新启动cron服务。

编辑配置文件

cron服务每分钟不仅要读一次/var/spool/cron内的所有文件，还需要读一次/etc/crontab，因此通过配置这个文件也能得到cron的服务。用crontab配置是针对某个用户的，而编辑/etc/crontab是针对系统的任务。此文件的文件格式如下：

```
SHELL=/bin/sh
PATH=/usr/bin:/usr/sbin:/sbin:/bin:/usr/lib/news/bin
MAILTO=root  //如果出现错误，或者有数据输出，数据作为邮件发给这个账号
HOME=/
#  run-parts
01  * * * *   root run-parts /etc/cron.hourly     //每个小时执行一次/etc/cron.hourly里的脚本
02 4 * * *   root run-parts /etc/cron.daily    //每天执行一次/etc/cron.daily里的脚本
22 4 * * 0  root run-parts /etc/cron.weekly     //每周执行一次/etc/cron.weekly里的脚本
42 4 1  * *  root run-parts /etc/cron.monthly     //每月执行一次/etc/cron.monthly里的脚本
```

说明

如果去掉run-parts参数，其后面就是运行的某个脚本名，而不是目录名。

6.1.3 挂起/恢复进程

作业控制允许进程挂起并可以在需要时恢复进程的运行，被挂起的作业恢复后将从中止处开始继续运行。只要在键盘上按Ctrl+Z键，即可挂起当前的前台作业。在键盘上按Ctrl+Z键后，将挂起当前执行的命令cat。使用jobs命令可以显示shell的作业清单，包括具体的作业、作业号以及作业当前所处的状态。

恢复进程执行时，有两种选择：用fg命令将挂起的作业放回到前台执行；用bg命令将挂起的作业放到后台执行。灵活使用上述命令，将给自己带来很大的方便。

6.2 查看进程

Linux是一个多任务系统，经常需要对这些进程进行一些调配和管理。要进行管理，首先就要知道现在的进程情况：有哪些进程、进程的状态如何等。Linux提供了多种命令来了解进程的状况。

who 命令

who命令主要用于查看当前系统中的用户情况。如果用户想和其他用户建立即时通讯，比如使用talk命令，那么首先要确定的就是该用户确实在线上，不然talk进程就无法建立起来。又如，系统管理员希望监视每个登录的用户此时此刻的所作所为，也要使用who命令。who命令应用起来非常简单，可以比较准确地掌握用户的情况，所以使用非常广泛。

例如查看系统中的用户及其状态。使用如下：

```
# who
admin     tty1         Jul 28 15:55
admin     pts/0        Aug  5 15:46 (9.1.0.110)
admin     pts/2        Jul 29 19:52 (9.1.0.110)
root     pts/3        Jul 30 12:07 (9.1.0.110)
```



```
root     pts/4        Jul 31 10:29 (9.1.0.144)
root     pts/5        Jul 31 14:52 (9.1.0.11)
root     pts/6        Aug  6 10:12 (9.1.0.234)
root     pts/8        Aug  6 11:34 (9.1.0.234)
```

ps 命令

ps命令是最基本又非常强大的进程查看命令。使用该命令可以确定有哪些进程正在运行和运行的状态、进程是否结束、进程有没有僵尸、哪些进程占用了过多的资源等，大部分进程信息都是可以通过执行该命令得到的。

ps命令最常用的还是用来监控后台进程的工作情况，因为后台进程是不与屏幕、键盘这些标准输入/输出设备进行通信的，所以如果需要检测其状况，就可使用ps命令。ps命令的常见选项如表6-2所示。

表 6-2 选项说明

选项	描述
-e	显示所有进程。
-f	全格式。
-h	不显示标题。
-l	使用长格式。
-w	宽行输出。
-a	显示终端上的所有进程，包括其他用户的进程。
-r	只显示正在运行的进程。
-x	显示没有控制终端的进程。

例如显示系统中终端上的所有进行进程。命令如下：

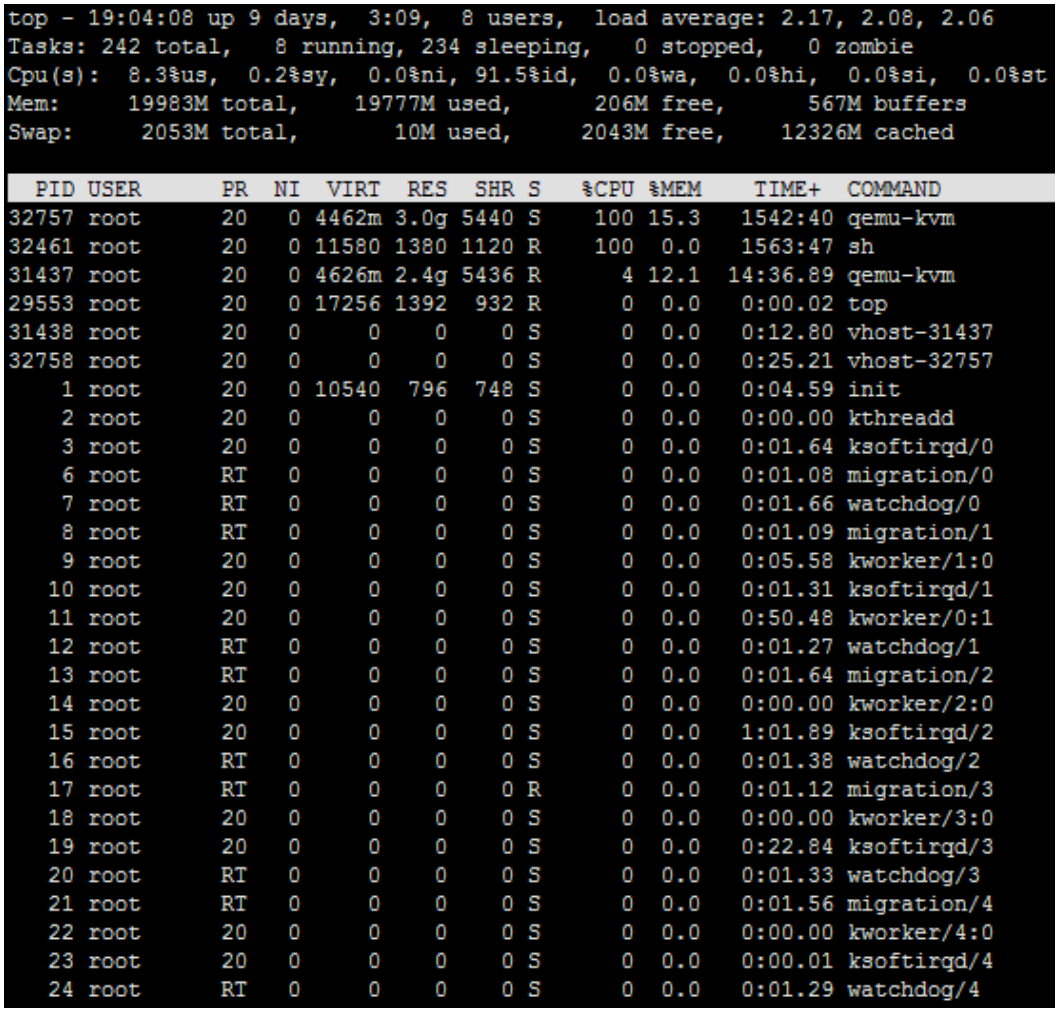
```
# ps -a
  PID TTY          TIME CMD
12175 pts/6    00:00:00 bash
24526 pts/0    00:00:00 vsftpd
29478 pts/5    00:00:00 ps
32461 pts/0    1-01:58:33 sh
```

top 命令

top命令和ps命令的基本作用是相同的，显示系统当前的进程和其他状况，但是top是一个动态显示过程，即可以通过用户按键来不断刷新进程的当前状态，如果在前台执行该命令，它将独占前台，直到用户终止该程序为止。其实top命令提供了实时的对系统处理器的状态监视。它将显示系统中CPU的任务列表。该命令可以按CPU使用、内存使用和执行时间对任务进行排序，而且该命令的很多特性都可以通过交互式命令或者在定制文件中进行设定。

top命令输出的实例如图6-1所示：

图 6-1 top 显示



kill 命令

当需要中断一个前台进程的时候，通常足使用“Ctrl+c”组合键，而对于后台进程不能用组合健来终止，这时就可以使用kill命令。该命令可以终止前台和后台进程。终止后台进程的原因包括：该进程占用CPU的时间过多、该进程已经死锁等。

kill命令是通过向进程发送指定的信号来结束进程的。如果没有指定发送的信号，那么默认值为TERM信号。TERM信号将终止所有不能捕获该信号的进程。至于那些可以捕获该信号的进程可能就需要使用KILL信号（它的编号为9），而该信号不能被捕捉。

kill命令的语法格式有以下两种方式：

```
kill [-s 信号 | -p] [-a] 进程号…
kill -l [信号]
```

其中进程号可以通过ps命令的输出得到。-s选项是给程序发送指定的信号，详细的信号可以用“kill -l”命令查看；-p选项只显示指定进程的ID号，而发送信号。

杀死pid为1409的进程，示例如下：

```
# kill -9 1409
```

显示所有的信号及其编号对应关系，示例如下：

```
# kill -l
 1) SIGHUP? 2) SIGINT? 3) SIGQUIT? 4) SIGILL
 5) SIGTRAP? 6) SIGABRT? 7) SIGBUS? 8) SIGFPE
 9) SIGKILL?10) SIGUSR1?11) SIGSEGV?12) SIGUSR2
13) SIGPIPE?14) SIGALRM?15) SIGTERM?16) SIGSTKFLT
17) SIGCHLD?18) SIGCONT?19) SIGSTOP?20) SIGTSTP
21) SIGTTIN?22) SIGTTOU?23) SIGURG?24) SIGXCPU
25) SIGXFSZ?26) SIGVTALRM?27) SIGPROF?28) SIGWINCH
29) SIGIO?30) SIGPWR?31) SIGSYS?34) SIGRTMIN
35) SIGRTMIN+1?36) SIGRTMIN+2?37) SIGRTMIN+3?38) SIGRTMIN+4
39) SIGRTMIN+5?40) SIGRTMIN+6?41) SIGRTMIN+7?42) SIGRTMIN+8
43) SIGRTMIN+9?44) SIGRTMIN+10?45) SIGRTMIN+11?46) SIGRTMIN+12
47) SIGRTMIN+13?48) SIGRTMIN+14?49) SIGRTMIN+15?50) SIGRTMAX-14
51) SIGRTMAX-13?52) SIGRTMAX-12?53) SIGRTMAX-11?54) SIGRTMAX-10
55) SIGRTMAX-9?56) SIGRTMAX-8?57) SIGRTMAX-7?58) SIGRTMAX-6
59) SIGRTMAX-5?60) SIGRTMAX-4?61) SIGRTMAX-3?62) SIGRTMAX-2
```


7 监控告警使用指南

7.1 监控配置

7.1.1 简介

简介

监控框架负责监控OS系统运行过程中的异常，并将监控到的异常上报告警模块，并通过告警模块上报告警到产品的告警平台。监控框架以服务的形式提供，可以通过systemctl start/stop/restart/reload sysmonitor启动、关闭、重启、重载服务。

注意事项

- 监控框架不支持并发执行。
- 各配置文件须合法配置，否则可能造成监控框架异常。

配置总览

监控框架有一个主配置文件（/etc/sysconfig/sysmonitor），用于配置各监控项的监控周期、是否需要监控及异常时是否需要上报告警等。

配置示例如下：

说明

配置项的=和"之间不能有空格。

```
# 关键进程监控开关
PROCESS_MONITOR=″on″

# 关键进程监控周期（秒）
PROCESS_MONITOR_PERIOD=″3″

# 关键进程恢复失败后再次尝试拉起周期（分）
PROCESS_RECALL_PERIOD=″1″
```



```
# 关键进程服务异常恢复过程中超时时间（秒）
PROCESS_RESTART_TIMEOUT="90"

# 关键进程监控告警
PROCESS_ALARM="off"

# 文件系统故障监控开关
FILESYSTEM_MONITOR="on"

# 文件系统故障告警开关
FILESYSTEM_ALARM="off"

# 信号监控开关
SIGNAL_MONITOR="on"

# 信号告警开关
SIGNAL_ALARM="off"

# 磁盘空间监控开关
DISK_MONITOR="on"

# 磁盘空间告警开关
DISK_ALARM="off"

# 磁盘监控周期（秒）
DISK_MONITOR_PERIOD="60"

# 网卡监控开关
NETCARD_MONITOR="on"

# 网卡告警开关
NETCARD_ALARM="off"

# 文件监控开关
FILE_MONITOR="on"

# 文件告警开关
FILE_ALARM="off"

# CPU监控开关
CPU_MONITOR="on"

# CPU告警开关
CPU_ALARM="off"

# 内存监控开关
MEM_MONITOR="on"

# 内存告警开关
MEM_ALARM="off"

# 进程（线程）数监控开关
PSCNT_MONITOR="on"

# 进程（线程）数告警开关
PSCNT_ALARM="off"

# 系统fd总数监控开关
FDCNT_MONITOR="on"

# 系统fd总数告警开关
FDCNT_ALARM="off"
```

各配置项说明见**表7-1**。

表 7-1 配置说明

配置项	配置项说明	是否必配	默认值
PROCESS_MONITOR	设定是否开启关键进程监控，on为开启，off为关闭	否	开启
PROCESS_MONITOR_PERIOD	设定关键进程监控的周期，单位秒	否	3s
PROCESS_RECALL_PERIOD	关键进程恢复失败后再次尝试拉起周期，单位分，取值范围为1至1440之间的整数	否	1min
PROCESS_RESTART_TIMEOUT	关键进程服务异常恢复过程中超时时间，单位秒，取值范围为30至300之间的整数 **说明** 例如因DNS配置的ip不通等原因导致rsyslog进程恢复时间较长时，需调整该配置项的值	否	90s
PROCESS_ALARM	设定是否开启关键进程异常告警，on为开启，off为关闭	否	关闭
FILESYSTEM_MONITOR	设定是否开启ext3文件系统监控，on为开启，off为关闭	否	开启
FILESYSTEM_ALARM	设定是否开启ext3文件系统异常告警，on为开启，off为关闭	否	关闭
SIGNAL_MONITOR	设定是否开启信号监控，on为开启，off为关闭	否	开启
SIGNAL_ALARM	设定是否开启信号监控异常告警，on为开启，off为关闭	否	关闭
DISK_MONITOR	设定是否开启磁盘空间监控，on为开启，off为关闭	否	开启
DISK_ALARM	设定是否开启磁盘空间告警，on为开启，off为关闭	否	关闭

配置项	配置项说明	是否必配	默认值
DISK_MONITOR_PERIOD	设定磁盘监控周期，单位秒	否	60s
NETCARD_MONITOR	设定是否开启网卡监控，on为开启，off为关闭	否	开启
NETCARD_ALARM	设定是否开启网卡监控异常告警，on为开启，off为关闭	否	关闭
FILE_MONITOR	设定是否开启文件监控，on为开启，off为关闭	否	开启
FILE_ALARM	设定是否开启文件监控异常告警，on为开启，off为关闭	否	关闭
CPU_MONITOR	设定是否cpu监控，on为开启，off为关闭	否	开启
CPU_ALARM	设定是否cpu告警，on为开启，off为关闭	否	关闭
MEM_MONITOR	设定是否内存监控，on为开启，off为关闭	否	开启
MEM_ALARM	设定是否内存告警，on为开启，off为关闭	否	关闭
PSCNT_MONITOR	设定是否进程数监控，on为开启，off为关闭	否	开启
PSCNT_ALARM	设定是否进程数告警，on为开启，off为关闭	否	关闭
FDCNT_MONITOR	设定是否fd总数监控，on为开启，off为关闭	否	开启
FDCNT_ALARM	设定是否fd总数告警，on为开启，off为关闭	否	关闭

注意

1. 修改/etc/sysconfig/sysmonitor配置文件后，需要重启sysmonitor服务生效。
2. 由于ext3文件系统故障监控只支持ext3文件系统，故非ext3文件系统的版本上即使监控开关打开也不会监控。

目前版本中支持ext3文件系统监控、关键进程模块监控、磁盘空间监控、文件监控、信号监控、网卡监控、cpu占用率监控、内存监控、进程数监控以及fd总数监控。

表 7-2 监控项列表

监控项	是否可配
ext3文件系统故障	否
关键进程/模块异常	是
磁盘空间	是
文件	是
信号	是
网卡	是
cpu占用率	是
内存	是
进程数	是
fd总数	是

命令参考

- 启动监控服务：
  ```
  systemctl start sysmonitor
  ```
- 关闭监控服务：
  ```
  systemctl stop sysmonitor
  ```
- 重启监控服务：
  ```
  systemctl restart sysmonitor
  ```

  说明

  修改/etc/sysconfig/sysmonitor后，须重启服务修改后的配置才能生效。
- 修改监控项的配置文件后，重载监控服务可使修改后的配置动态生效：
  ```
  systemctl reload sysmonitor
  ```

  说明

  重载操作对网卡监控不生效，修改完网卡监控的配置文件须重启服务才能生效。

7.1.2 ext3 文件系统监控

简介

当文件系统出现故障时会导致IO操作异常从而引发操作系统一系列问题。通过文件系统故障检测及时发现文件系统故障，并上报告警给产品，以便系统管理员或用户及时恢复故障。

配置文件说明

无。

异常日志

文件系统只读场景下，如果监控到ext3文件系统故障，/var/log/sysmonitor.log中打印异常信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[5026]: sdc1 ext3-fs error. Remount filesystem read-only.
```

其他异常场景下，如果监控到ext3文件系统故障，/var/log/sysmonitor.log中打印异常信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[6172]: sda6 ext3-fs error.
```

7.1.3 关键进程监控

简介

定期监控系统中的关键进程，当系统内关键进程异常退出时，自动尝试恢复关键进程。如果恢复失败并需要告警，可上报告警。系统管理员能及时感知进程异常退出事件，以及进程是否被恢复拉起。问题定位人员能从日志中定位进程异常退出的时间。

配置文件说明

配置目录为/etc/sysmonitor/process，每个进程或模块一个配置文件。

配置文件示例如下：

```
NAME=crond
RECOVER_COMMAND=systemctl restart crond
MONITOR_COMMAND=systemctl status crond
STOP_COMMAND=systemctl stop crond
```

各配置项说明见表7-3。

表 7-3 配置项说明

配置项	配置项说明	是否必配	默认值
NAME	进程或模块名	是	无
RECOVER_COMMAND	恢复命令	是	无

配置项	配置项说明	是否必配	默认值
MONITOR_COMMAND	监控命令 **说明** 命令返回值为0视为进程正常，命令返回大于0视为进程异常。	否	pgrep -f $(which xxx) **说明** “xxx”为NAME字段中配置的进程名。
STOP_COMMAND	停止命令	否	无

注意

1. 修改关键进程监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. 恢复命令和监控命令不能阻塞，否则造成关键进程监控线程异常。
3. 当恢复命令执行超时90s时，会调用停止命令终止进程。
4. 当关键进程异常后，并且尝试拉起三次都不成功，最终会按照全局配置文件中配置的PROCESS_RECALL_PERIOD周期进行拉起。

异常日志

如果监控到进程或模块异常，/var/log/sysmonitor.log中打印异常信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[5018]: crond is abornmal, use "systemctl restart crond" to recover
[2015-03-16:18:41:38]sysmonitor[5018]: crond is abnormal
```

7.1.4 文件监控

简介

系统关键文件被意外删除后，会导致系统运行异常甚至崩溃。通过文件监控可以及时获知系统中关键文件被删除或者有恶意文件被添加，并且上报告警给产品，以便管理员和用户及时获知并处理故障。

配置文件说明

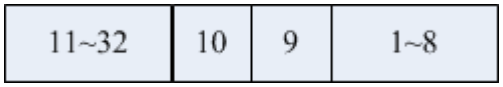
配置文件为/etc/sysmonitor/file。每个监控配置项为一行，监控配置项包含两个内容：监控文件（目录）和监控事件。监控文件（目录）是绝对路径，监控文件（目录）和监控事件中间由一个或多个空格隔开。

注意

1. 由于告警信息和日志长度限制，建议监控的文件或目录绝对路径长度小于或等于144。如果配置的监控对象绝对路径长度大于144，可能会有日志打印不完整的现象出现。
2. 请用户自行确保监控文件路径正确，如果配置文件不存或错误在则无法监控到该文件。
3. 由于系统路径长度限制，监控的文件或目录绝对路径长度必须小于4096。
4. 支持监控目录和常规文件，/proc和/proc/*、/dev和/dev/*、/sys和/sys/*、管道文件、socket文件等均不支持监控。
5. /var/log和/var/log/*均只支持删除事件。
6. 当配置文件中存在多条相同路径的时候，以第一条合法配置为准，其他相同配置均不生效。在日志文件中可以查看到其他相同配置被忽略的提示。
7. 不支持对软链接配置监控；当配置硬链接文件的删除事件时，需删除该文件和它的全部硬链接才会打印文件删除事件。
8. 当文件添加监控成功及监控的事件发生时，监控日志打印的是配置文件中路径的绝对路径。

监控文件（目录）采用了位图的方式配置要监控的事件，对文件或目录进行监控的事件位图如图7-1所示。

图 7-1 监控事件位图



事件位图中每一位代表一个事件。第n位如果置1，则表示监控第n位对应的事件；第n位如果置0，则表示不监控第n位对应的事件。监控位图对应的16进制数，即是写到配置文件中的监控事件项。目前每一位对应的事件见表7-4。

表 7-4 文件事件位图中的位与事件对应关系表

配置项	配置项说明	是否必配
1～8	保留	否
9	文件、目录添加事件	是
10	文件、目录删除事件	是
11～32	保留	否

注意

1. 修改文件监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在最多60秒后生效。
2. 监控事件需要严格遵守上述规则，如果配置有误，则无法监控；如果配置项中监控事件填空，则默认只监控删除事件，即0x200。
3. 文件或目录删除后，只有当所有打开该文件的进程都停止后才会上报删除事件。
4. 监控的文件通过vi、sed等操作修改后会在监控日志中打印File "XXX" may have been changed。

文件监控目前实现了对添加和删除事件的监控，即第9和第10位有效，其他位为保留位，暂不生效。如果配置事件配置了1-8、11-32位为1，监控日志中会提示监控事件配置错误。

示例

配置对/home下子目录的增加和删除事件监控，低12位位图为：001100000000，则可以配置如下：

```
/home      0x300
```

配置对/etc/ssh/sshd_config文件的删除事件监控，低12位位图为：001000000000，则可以配置如下：

```
/etc/ssh/sshd_config  0x200
```

异常日志

如果监控文件有配置的事件发生，/var/log/sysmonitor.log中打印日志示例如下：

```
[2015-03-16:18:41:38]sysmonitor[3373]: 1 events queued
[2015-03-16:18:41:38]sysmonitor[3373]: 1th event handled
[2015-03-16:18:41:38]sysmonitor[3373]: Subdir "b" under "/home/a" was added.
[2015-03-16:18:41:38]sysmonitor[3373]: 1 events queued
[2015-03-16:18:41:38]sysmonitor[3373]: 1th event handled
[2015-03-16:18:41:38]sysmonitor[3373]: Subfile "c" under "/home/a" was deleted.
```

7.1.5 信号监控

简介

当进程被信号异常终止时，通过记录信号发送的进程以及发送的信号以便于问题定位。

配置文件说明

配置文件为/etc/sysmonitor/signal，示例如下：

```
SIGKILL="on"

SIGTERM="on"
```

各配置项说明见表7-5。

表 7-5 配置项说明

配置项	配置项说明	是否必配	默认值
SIGKILL	SIGKILL信号	否	on
SIGTERM	SIGTERM信号	否	on
SIGHUP	SIGHUP信号	否	off
SIGINT	SIGINT信号	否	off
SIGQUIT	SIGQUIT信号	否	off
SIGILL	SIGILL信号	否	off
SIGTRAP	SIGTRAP信号	否	off
SIGABRT	SIGABRT信号	否	off
SIGBUS	SIGBUS信号	否	off
SIGFPE	SIGFPE信号	否	off
SIGUSR1	SIGUSR1信号	否	off
SIGSEGV	SIGSEGV信号	否	off
SIGUSR2	SIGUSR2信号	否	off
SIGPIPE	SIGPIPE信号	否	off
SIGALRM	SIGALRM信号	否	off
SIGSTKFLT	SIGSTKFLT信号	否	off
SIGCHLD	SIGCHLD信号	否	off
SIGCONT	SIGCONT信号	否	off
SIGSTOP	SIGSTOP信号	否	off
SIGTSTP	SIGTSTP信号	否	off
SIGTTIN	SIGTTIN信号	否	off
SIGTTOU	SIGTTOU信号	否	off
SIGURG	SIGURG信号	否	off
SIGXCPU	SIGXCPU信号	否	off
SIGXFSZ	SIGXFSZ信号	否	off
SIGVTALRM	SIGVTALRM信号	否	off
SIGPROF	SIGPROF信号	否	off
SIGWINCH	SIGWINCH信号	否	off
SIGIO	SIGIO信号	否	off

配置项	配置项说明	是否必配	默认值
SIGPWR	SIGPWR信号	否	off
SIGSYS	SIGSYS信号	否	off

注意

修改关键进程监控的配置文件后，须执行systemctl reload sysmonitor后生效。

异常日志

如果监控到配置的信号事件，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[7965]: sshd[18883](parent:sshd[944]) send SIGKILL to sshd[18884]
```

7.1.6 磁盘分区监控

简介

定期监控系统中挂载的磁盘分区空间，当磁盘分区使用率大于或等于用户设置的告警阀值，上报磁盘空间告警。发生告警后，当磁盘分区使用率小于用户设置的告警恢复阀值，上报磁盘空间恢复告警。

配置文件说明

配置目录为/etc/sysmonitor/disk。

配置文件示例如下：

```
DISK="/var/log" ALARM="90" RESUME="80"
DISK="/" ALARM="95" RESUME="85"
```

各配置项说明见表7-6。

表 7-6 配置项说明

配置项	配置项说明	是否必配	默认值
DISK	磁盘挂载目录名 **说明** 必须为磁盘挂载点或被挂载的磁盘分区。 最大长度为64字节。	是	无

配置项	配置项说明	是否必配	默认值
ALARM	整数，磁盘空间告警阈值 **说明** 0-100的整数，不要有空格等其他额外字符。	否	90
RESUME	整数，磁盘空间恢复阈值 **说明** 0-100的整数，不要有空格等其他额外字符。	否	80

注意

1. 修改磁盘空间监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. 重复配置的挂载目录，最后一个配置项生效。
3. ALARM值应该大于RESUME值。
4. 只能针对挂载点或被挂载的磁盘分区做监控。
5. 在CPU和I/O高压场景下，df执行命令超时，会导致磁盘利用率获取不到。
6. 当多个挂载点对应同一个磁盘分区时，以挂载点为准来上报告警。

异常日志

如果监控到磁盘空间告警，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[5018]: report disk alarm, /var/log used:90% alarm:90%
[2015-03-16:18:41:38]sysmonitor[5018]: report disk recoverd, /var/log used:79% resume:80%
```

7.1.7 网卡状态监控

简介

系统运行过程中可能出现因人为原因或异常而导致网卡状态或IP发生改变，对网卡状态和IP变化进行监控，以便及时感知到异常并方便定位异常原因。

配置文件说明

配置文件为/etc/sysmonitor/network，示例如下：

```
#dev event
eth1 UP
```

各配置项说明见表7-7。

表 7-7 配置项说明

配置项	配置项说明	是否必配	默认值
dev	网卡名	是	无
event	监听事件，可取值为UP、DOWN、NEWADDR、DELADDR。 ● UP：网卡UP ● DOWN：网卡DOWN ● NEWADDR：增加IP地址 ● DELADDR：删除IP地址	否	监控所有事件

注意

1. 修改网卡监控的配置文件后，须执行systemctl restart sysmonitor，新的配置才可生效。
2. 如果配置文件为空，则不监控网卡事件。
3. 目前仅支持监控IPv4地址的添加和删除事件。
4. 不支持虚拟网卡UP和DOWN的状态监控。

异常日志

如果监控到配置的网卡事件，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-03-16:18:41:38]sysmonitor[5018]: eth1: dev is up
[2015-03-16:18:41:38]sysmonitor[5018]: eth1: ip[192.168.0.1] prefixlen[24] is added
```

7.1.8 cpu 监控

简介

监控系统cpu占用情况，当cpu使用率超出或低于阈值时，记录日志或上报告警。

配置文件说明

配置目录为/etc/sysmonitor/cpu。

配置文件示例如下：

```
# 告警产生百分比上限
ALARM="90"

# 告警恢复百分比下限
RESUME="80"
```



```
# 监控周期（秒）
MONITOR_PERIOD="60"

# 统计周期（秒）
STAT_PERIOD="300"
```

表 7-8 配置项说明

配置项	配置项说明	是否必配	默认值
ALARM	大于0的整数，cpu占用率告警阈值。	否	90
RESUME	大于等于0的整数，cpu占用率恢复阈值。	否	80
MONITOR_PERIOD	监控周期（秒），取值大于0。	否	60
STAT_PERIOD	统计周期（秒），取值大于0。	否	300

注意

1. 修改cpu监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. ALARM值应该大于RESUME值。

异常日志

如果监控到cpu告警，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-04-16:06:18:54]sysmonitor[27482]: CPU usage alarm:  91.3%
[2015-04-16:06:24:16]sysmonitor[31896]: CPU usage resume:  70.1%
```

7.1.9 内存监控

简介

监控系统内存占用情况，当内存使用率超出或低于阈值时，记录日志或上报告警。

配置文件说明

配置目录为/etc/sysmonitor/memory。

配置文件示例如下：

```
# 告警产生百分比上限
ALARM="90"

# 告警恢复百分比下限
RESUME="80"
```

```
# 监控周期（秒）
PERIOD="60"
```

表 7-9 配置项说明

配置项	配置项说明	是否必配	默认值
ALARM	大于0的整数，内存占用率告警阈值。	否	90
RESUME	大于等于0的整数，内存占用率恢复阈值。	否	80
PERIOD	监控周期（秒），取值大于0。	否	60

注意

1. 修改内存监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. ALARM值应该大于RESUME值。
3. 取三个监控周期的内存占用的平均值，来作为是否上报发生告警或者恢复告警的依据。

异常日志

如果监控到内存告警，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-04-16:07:24:23]sysmonitor[31896]: memory usage alarm:  90.1%
[2015-04-16:07:28:16]sysmonitor[31896]: memory usage resume:  60.4%
```

7.1.10 进程数监控

简介

监控系统进程数目，当进程总数超出或低于阈值时，记录日志或上报告警。

配置文件说明

配置目录为/etc/sysmonitor/pscnt。

配置文件示例如下（在4P及4P以下服务器场景下）：

```
# 进程（包括线程）数告警产生上限
ALARM="1600"

# 进程（包括线程）数告警恢复下限
RESUME="1500"

# 监控周期（秒）
PERIOD="60"
```

表 7-10 配置项说明

配置项	配置项说明	是否必配	默认值
ALARM	大于0的整数，进程总数告警阈值。	否	1600
RESUME	大于等于0的整数，进程总数恢复阈值。	否	1500
PERIOD	监控周期（秒），取值大于0。	否	60

注意

1. 修改进程数监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. ALARM值应该大于RESUME值。
3. 在16P服务器场景下，配置项默认值分别为：ALARM="10000" RESUME="8000" PERIOD="60"。在8P/32P机器上，请将告警值与恢复值在16P基础上相应的减半/翻倍。

异常日志

如果监控到进程数告警，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-04-16:07:44:54]sysmonitor[31896]: process count alarm: 1657
[2015-04-16:07:45:17]sysmonitor[31896]: process count resume: 1200
```

7.1.11 fd 总数监控

简介

监控系统文件句柄（fd）数目，当系统文件句柄总数超出或低于阈值时，记录日志或上报告警。

配置文件说明

配置目录为/etc/sysmonitor/fdcnt。

配置文件示例如下：

```
# 系统fd总数百分比上限
ALARM="80"

# 系统fd总数百分比下限
RESUME="70"

# 监控周期（秒）
PERIOD="600"
```


表 7-11 配置项说明

配置项	配置项说明	是否必配	默认值
ALARM	大于0的整数，fd总数与系统最大fd数百分比的告警阈值。	否	80%
RESUME	大于等于0的整数，fd总数与系统最大fd数百分比的恢复阈值。	否	70%
PERIOD	监控周期（秒），取值大于0。	否	600

注意

1. 修改fd总数监控的配置文件后，须执行systemctl reload sysmonitor，新的配置在一个监控周期后生效。
2. ALARM值应该大于RESUME值。

异常日志

如果监控到fd总数告警，/var/log/sysmonitor.log中打印信息示例如下：

```
[2015-04-16:07:44:54]sysmonitor[31896]: fd count alarm: 138400
[2015-04-16:07:45:17]sysmonitor[31896]: fd count resume: 121100
```

7.2 监控日志

简介

为防止sysmonitor.log文件过大，日志特性提供了切分转储日志的机制。在默认情况下，cron服务每小时调用一次logrotate机制，读取sysmonitor的配置文件/usr/libexec/sysmonitor/log_dump.conf检查，当/var/log/sysmonitor.log大于8M时就转储。转储时使用zip方式压缩。

配置文件说明

配置文件为：/usr/libexec/sysmonitor/log_dump.conf，配置文件示例如下：

```
/var/log/sysmonitor.log
{
create
compress
rotate 20
maxage 30
missingok
notifempty
size +8192k
```



```
sharedscripts
}
```

每个参数的意义如下：

- create：表示创建新的日志文件
- compress：表示压缩备份文件
- rotate 20：保留分割后的20份日志
- maxage 30：保留30天的日志
- missingok：如果日志不存在则忽略
- notifempty：如果log文件是空的，不进行rotate
- size +8192k：日志文件达到8192k时转储
- sharedscripts：表示为整个日志组运行一次的脚本

日志转储局限

logrotate当前配置文件rotate 为5，当产生5个转储文件后，再把rotate 修改为3，这时logrotate不再转储。

8 日志防爆特性

8.1 功能简介

日志防爆特性，是指系统会通过日志切割工具logrotate ，每隔一小时对/etc/logrotate.d目录下的日志配置文件中配置的日志文件进行切割，以防止磁盘空间被占满。

8.2 使用场景

系统使用过程中，日志持续增长会使存储日志的磁盘空间被占满，从而导致日志无法继续打印，或者影响到系统部分业务功能（进程异常或一些操作无法进行）；另外如果系统出现了问题，因为缺少日志也会影响问题定位效率，甚至导致问题无法定位。

日志防爆特性，主要是针对大小会持续增长的日志文件，对其添加切割配置，确保存储日志文件磁盘空间不被占满，从而增强系统的稳定性与可靠性。

8.3 使用方法

在/etc/logrotate.d目录下，对需要切割的日志文件进行配置。

文件配置方法

在/etc/logrotate.d目录下，添加配置文件（新增配置文件的权限建议不大于640），一个配置文件中可同时配置多个需要切割的日志。

如：/etc/logrotate.d/example文件中配置。

```
/var/log/logexample
/var/log/logexample1
{
maxage 365
rotate 30
notifempty
compress
```



```
copytruncate
missingok
size +4096k
}
```

该配置会对/var/log/logexample、/var/log/logexample1进行切割。

- maxage 365：只存储最近365天的切割出来的日志文件，超过365天则删除。
- rotate 30：指定日志文件删除之前切割的次数，此处保留30个备份。
- notifempty：表示日志为空则不处理。
- compress：通过gzip压缩转储以后的日志。
- copytruncate：用于还在打开中的日志文件，把当前日志备份并截断。
- missingok：如果日志文件丢失，不报错继续执行下一个。
- size +4096k：表示日志超过4096k大小才分割，size默认单位是KB，可使用k、M和G来指定KB、MB和GB。

配置项说明

配置项说明如**表1**。

表 8-1 配置项说明

配置项	功能
compress	通过gzip压缩转储以后的日志。
missingok	找不到日志时，跳过。
nomissingok	找不到日志时，报错。
nocompress	不需要压缩时，用这个参数。
copytruncate	用于还在打开中的日志文件，把当前日志备份并截断。
nocopytruncate	备份日志文件但是不截断。
create mode owner group	转储文件，使用指定的文件模式创建新的日志文件。
nocreate	不建立新的日志文件。
prerotate/endscript	在转储以前需要执行的命令可以放入这个对，这两个关键字必须单独成行。
postrotate/endscript	在转储以后需要执行的命令可以放入这个对，这两个关键字必须单独成行。
daily	指定转储周期为每天。
weekly	指定转储周期为每周。
monthly	指定转储周期为每月。
rotate count	指定日志文件删除之前转储的次数，0指没有备份，5指保留5个备份。

配置项	功能
size	当日志文件到达指定的大小时才转储，size可以指定bytes（缺省）以及KB、MB或者GB。

配置项中“转储”的含义：用来把旧的日志文件删除，并创建新的日志文件。

说明

1. nocreate与配置文件中的copytruncate是互斥的，不能同时配置，否则nocreate不生效。
2. create mode owner group（例如：create 0600 root root）与配置文件中的copytruncate是互斥的，否则create配置不生效。
3. 时间频度（daily，weekly，monthly，yearly）和日志大小（size）这两项参数同时配置的时候，会以日志大小为条件，达到一定大小就会切割。

9 kbox

9.1 概述

本章主要介绍Kbox（内核黑匣子）基础概念、结构，以及一些软硬件需求和约束限制等。

9.1.1 Kbox 简介

本节介绍Kbox的背景信息和一些基础概念，帮助用户了解Kbox。

概述

linux内核比较复杂，且各个模块之间联系紧密，缺少有效的维护工具，进一步增加了维护难度。虽然内核有klogd和syslogd等日志记录系统，但在一些异常情况下，如系统突然重启、内核崩溃、oom（内存溢出）等，无法（或者来不及）记录这些信息，进而无法对这些问题的根因进行定位。

针对上述情况，为了挽救丢失的内核日志，EulerOS提供了内核黑匣子（Kernel black box）特性。类似在飞行系统中常用的“黑匣子”，在系统异常触发的时候记录重要信

息，并通过某种特殊渠道（非易失性存储）把这些信息保存下来，供维护人员用来分析发生异常时的系统状态。

特性简介

Kbox提供一种记录内核信息的机制，在系统异常发生时，记录下内核的重要信息，并把这些信息保存到非易失存储介质中。维护人员可通过这些信息，得到发生异常时的具体情况，进而分析发生异常的原因，支撑问题定位。

例如：当panic事件发生时，Kbox将内核产生的异常信息收集并存储在临时存储区（region）。当信息全部收集完成后，Kbox会将临时存储区的信息转储到非易失性设备中（如NVRAM等）或通过kdump保存到指定内存中，供用户查看。

注意

kbox只收集内核打印信息及导致系统异常的函数调用关系，不会保存用户敏感数据。如需关闭kbox功能，请参见**命令参考**。

Kbox 组成

内核黑匣子模块主要分为以下三个部分，详细信息如**表9-1**所示。

表 9-1 黑匣子组成

组成	说明
内核黑匣子	管理非易失性设备，抓取并存储异常事件产生的信息。
非易失性设备驱动	为Kbox提供读写接口。Kbox启动时加载相应的驱动模块，将存储设备注册到Kbox中。
非易失性设备	用于存储异常事件产生的信息。异常事件发生时，Kbox会将异常信息转储到非易失性设备中 。若硬件上不存在非易失性设备，kbox的日志先保存在预留内存中，通过kdump来保存到磁盘中。

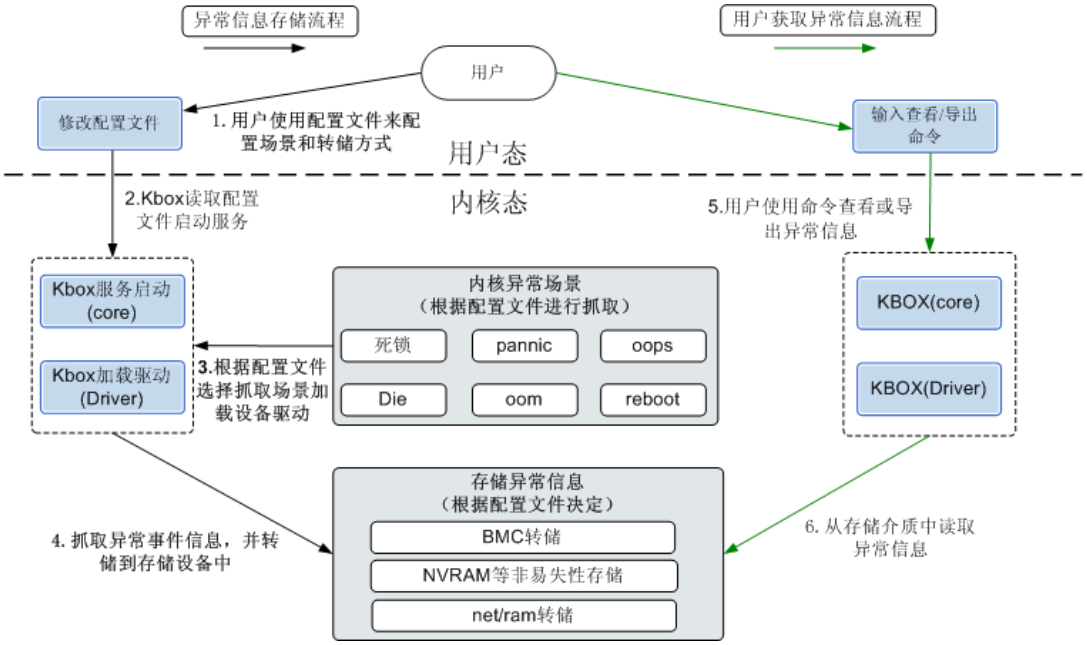
9.1.2 Kbox 结构

本节介绍Kbox的结构以及使用流程。

结构说明

Kbox整体结构和业务流程如下图所示。

图 9-1 Kbox 结构



说明

1、当前存储异常信息暂不支持BMC转储、net/ram转储。

2、当前版本仅支持OOM、OOPS（Die）、Panic、死锁这几种异常场景。

流程说明

1. 修改配置文件。产品根据自己的实际情况，通过配置文件（/etc/kbox/config）配置自己的产品信息、异常抓取场景和转储设备。
2. Kbox以服务的方式启动，读取配置文件（/etc/kbox/config）后，把产品信息和异常场景以模块参数的方式传递给Kbox内核模块。
3. 根据配置文件（/etc/kbox/config）中存储设备参数的配置，Kbox加载相应的存储设备驱动，将存储设备注册到Kbox内核模块。
4. 当某个异常事件发生并被Kbox内核模块抓取后，先记录异常情况相应的信息到对应的临时存储区域（region），然后刷新到存储设备。
5. 具有root权限的用户通过Kbox提供的日志导出命令，将存储设备中的异常信息导出到“/var/log/kbox/”目录下。
6. 用户读取异常信息文件，获取异常信息内容，进行问题定位。

9.1.3 软硬件需求

本节介绍Kbox的软硬件需求。

- 软件需求

  仅支持EulerOS操作系统。

- 硬件需求

  Kbox转储功能必须依赖产品提供的转储介质，当前支持NVRAM、hmem、pcie。

  说明

  EulerOS RAS版本在启动参数中添加“node0_reserve_kbox_mem=16M”，预留kbox内存；其他版本使用“reserve_kbox_mem=16M”预留kbox内存。系统启动时，默认已经设置。

9.2 功能说明

本章节介绍了内核黑匣子支持的异常事件场景和抓取的信息，以及对抓取内容的解析。

9.2.1 约束限制

本节介绍Kbox使用时会遇到的一些限制约束。

说明

禁止使用Kbox特性导出敏感信息。

- 当异常事件并发触发时，Kbox只记录最早发生的异常事件，并转储到存储设备。例如：当系统发生panic、oops或者oom等事件后，如果又有同类panic、oops等事件发生，Kbox不做记录，只打印提示信息到系统日志中 。
- Kbox在记录过程中，如果当前CPU上有中断到来，Kbox执行进程被打断，且在中断中又触发异常事件，此时Kbox只打印提示信息到系统日志中，不记录日志信息（包括之前和中断又触发的异常事件）到存储设备。
- 在使用命令行os_kbox_config导出Kbox记录的转储信息时，系统发生panic、oops或者oom等异常事件，由于并发对锁的获取，Kbox可能会概率性出现转储信息不完整的现象。
- 栈向下溢出场景下，Stack获取到信息全为0。栈刚好等于8k时，Stack获取到信息则为空。

9.2.2 提供系统 panic 信息

本节主要介绍当系统由于某种异常，直接或间接调用panic函数时，黑匣子所记录的异常信息。

功能概述

产品/平台业务软件或操作系统本身可能因为某种特殊异常，导致操作系统内核发生严重错误。Kbox能够将panic事件发生的时间、调用轨迹等异常信息记录到存储设备中，方便维护人员定位。

信息清单

记录的异常panic信息包含三个部分内容。

1. panic异常信息，最多记录信息容量为32K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - panic发生的原因。
   - 异常进程PID、异常进程TGID及进程名称。
   - 内核函数调用轨迹及栈信息（栈信息最多打印150行）。
   - 当前进程命令行信息。
   - 系统环境变量。
   - 寄存器信息。
2. 内核消息打印日志，最多记录信息容量为64K。该记录区主要内容包括：

- 内核异常发生的时间(UTC时间)。
- 异常发生时最近64K日志信息，即内核打印到循环缓冲区中的最后64K日志信息。

3. 控制台打印日志，最多记录信息容量为32K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 其他内核打印日志消息。

   说明

   如果当前其他进程没有向控制台打印日志，收集的控制台打印日志为空。

信息举例

在日志信息文件中，包含如下内容：

1. panic异常信息

```
//panic异常存储区，当前panic日志位于panic存储区索引为0
*****area type:panic - location in panic area:0*****
---------KBOX_START----------
//内核异常发生的时间，该时间为UTC时间，北京时间=UTC时间+8小时
panic time:20080120231731-a7634
//panic 原因
panic reason:Watchdog detected hard LOCKUP on cpu 1
panic stack:
//异常进程PID、异常进程TGID及进程名称
<pid:7344:7344:insmod>
<kernel_stack>
Stack:
 ffff880130467ea0 ffffffffa0015177 ffffffffa00e5740 ffffffffa00d7cb0
 00000000000005a0 ffffffffa00e5b98 ffff880130467ef0 ffffffffa00d3aa4
 ffff880130467ec0 ffffffffa00d7c60 ffff880130467ef0 ffffffffa00e5740
 ffffffffa00e5760 ffff880106768000 431bde82d7b634db ffff880106768000
 ffff880130467f40 ffffffffa00d3e1c ffff880c4ee54700 0000000000000000
 ffff8801064bd000 0000000000000000 0000000000000000 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 ffffffff8145a5d4
//函数调用轨迹
Call Trace:
 <NMI>  [<ffffffffa0002220>] kbox_show_task_kernel_info+0x200/0x300 [kbox]
 [<ffffffffa00054c7>] kbox_print_specified_tasks+0x17/0x50 [kbox]
 [<ffffffffa0000174>] kbox_panic_notifier_callback+0x174/0x210 [kbox]
 [<ffffffff811f7e4f>] ? __const_udelay+0x2f/0x40
 [<ffffffff8102156f>] ? native_safe_apic_wait_icr_idle+0x3f/0x60
 [<ffffffff8140b84f>] notifier_call_chain+0x3f/0x80
 [<ffffffff8140b8b5>] atomic_notifier_call_chain+0x15/0x20
 [<ffffffff814044d4>] panic+0xf2/0x20f
 [<ffffffff810af362>] watchdog_overflow_callback+0xd2/0xe0
 [<ffffffff810c63b8>] __perf_event_overflow+0xa8/0x220
 [<ffffffff8101204f>] ? x86_perf_event_set_period+0xdf/0x170
 [<ffffffff810c6884>] perf_event_overflow+0x14/0x20
 [<ffffffff81017991>] intel_pmu_handle_irq+0x1a1/0x350
 [<ffffffff81409639>] perf_event_nmi_handler+0x19/0x20
 [<ffffffff81408e28>] default_do_nmi+0x78/0x2d0
 [<ffffffffa0a9e020>] ? yh_exit+0x20/0x20 [hard_lockup]
 [<ffffffff81409128>] do_nmi+0xa8/0xe0
 [<ffffffff8140844c>] end_repeat_nmi+0x1a/0x1e
 [<ffffffffa0a9e020>] ? yh_exit+0x20/0x20 [hard_lockup]
 [<ffffffff811f7daf>] ? delay_tsc+0x4f/0x90
 [<ffffffff811f7daf>] ? delay_tsc+0x4f/0x90
 [<ffffffff811f7daf>] ? delay_tsc+0x4f/0x90
 <<EOE>>  [<ffffffffa0a9e020>] ? yh_exit+0x20/0x20 [hard_lockup]
 [<ffffffff811f7e4f>] __const_udelay+0x2f/0x40
 [<ffffffffa0a9e07a>] yh_init+0x5a/0xfe0 [hard_lockup]
 [<ffffffff810001cd>] do_one_initcall+0x3d/0x180
 [<ffffffff81092c95>] sys_init_module+0xc5/0x220
 [<ffffffff8140e4f9>] system_call_fastpath+0x16/0x1b
```



```
</kernel_stack>
</pid:7344>
<pid:7344:7344:insmod>
<basic>
insmod         R  running 1      0  7344   7344   5706 NA-e NA-y NA-o (NOTLB)
</basic>
//当前进程命令行信息
<cmdline>
insmod hard_lockup.ko
</cmdline>
//环境变量
<env>
LESSKEY=/etc/lesskey.bin NNTPSERVER=news INFODIR=/usr/local/info:/usr/share/info:/usr/info
MANPATH= HOSTNAME=Storage XKEYSYMDB=/usr/X11R6/lib/X11/XKeysymDB HOST= TERM=xterm SHELL=/bin/
bash PROFILEREAD=true HISTSIZE=1000 SSH_CLIENT=128.5.64.161 54825 22 MORE=-sl
SSH_TTY=/dev/pts/0 USER=admin
LS_COLORS=no=00:fi=00:di=01;34:ln=00;36:pi=40;33:so=01;35:do=01;35:bd=40;33;01:cd=40;33;01:or
=41;33;01:ex=00;32:*.cmd=00;32:*.exe=01;32:*.com=01;32:*.bat=01;32:*.btm=01;32:*.dll=01;32:*.
tar=00;31:*.tbz=00;31:*.tgz=00;31:*.rpm=00;31:*.deb=00;31:*.arj=00;31:*.taz=00;31:*.lzh=00;31
:*.lzma=00;31:*.zip=00;31:*.zoo=00;31:*.z=00;31:*.Z=00;31:*.gz=00;31:*.bz2=00;31:*.tb2=00;31:
*.tz2=00;31:*.tbz2=00;31:*.avi=01;35:*.bmp=01;35:*.fli=01;35:*.gif=01;35:*.jpg=01;35:*.jpeg=0
1;35:*.mng=01;35:*.mov=01;35:*.mpg=01;35:*.pcx=01;35:*.pbm=01;35:*.pgm=01;35:*.png=01;35:*.pp
m=01;35:*.tga=01;35:*.tif=01;35:*.xbm=01;35:*.xpm=01;35:*.dl=01;35:*.gl=01;35:*.wmv=01;35:*.a
iff=00;32:*.au=00;32:*.mid=00;32:*.mp3=00;32:*.ogg=00;32:*.voc=00;32:*.wav=00;32:
XNLSPATH=/usr/X11R6/lib/X11/nls ENV=/etc/bash.bashrc HOSTTYPE=x86_64 FROM_HEADER= PAGER=less
CSHEDIT=emacs XDG_CONFIG_DIRS=/etc/xdg MINICOM=-c on MAIL=/var/mail/admin PATH=/sbin:/usr/
sbin:/usr/local/sbin:/OSM/bin:/OSM/bin/script:/OSM/script:/home/permitdir/bin:/bin:/usr/
bin:/usr/local/bin:/usr/games:/usr/lib/mit/bin:/usr/lib/mit/sbin CPU=x86_64 INPUTRC=/etc/
inputrc PWD=/startup_disk/conf/euler-kbox-220 LANG=POSIX TEXINPUTS=:/home/
permitdir/.TeX:/usr/share/doc/.TeX:/usr/doc/.TeX SHLVL=1 HOME=/home/permitdir
LESS_ADVANCED_PREPROCESSOR=no OSTYPE=linux LS_OPTIONS=-A -N --color=tty -T 0 WINDOWMANAGER=
LESS=-M -I MACHTYPE=x86_64-suse-linux-gnu LOGNAME=admin XDG_DATA_DIRS=/usr/share
LC_CTYPE=en_US.UTF-8 SSH_CONNECTION=128.5.64.161 54825 128.5.110.220 22 LESSOPEN=lessopen.sh
%s INFOPATH=/usr/local/info:/usr/share/info:/usr/info LESSCLOSE=lessclose.sh %s %s
COLORTERM=1 OLDPWD=/startup_disk/conf _=/sbin/insmod
</env>
//寄存器信息
<pt_regs>
RIP: 0033:[<00007f7258c1e35a>] RSP: 002b:00007fff72520e58  EFLAGS: 00010202
RAX: 00000000000000af RBX: ffffffff8140e4f9 RCX: 00007f7258c10130
RDX: 0000000000603010 RSI: 0000000000000e9f RDI: 0000000000603030
RBP: 0000000000004000 R08: 00007fff72520f70 R09: 00007f7258c58c60
R10: 00007f7258c10130 R11: 0000000000000202 R12: 0000000000603010
R13: 0000000000004000 R14: 0000000000000e9f R15: 0000000000603030
FS:  00007f72590d2700(0000) GS:ffff880067820000(0000) knlGS:0000000000000000
CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
CR2: 00007f7258c1e350 CR3: 00000000368c8000 CR4: 00000000000407e0
</pt_regs>
```

2. 内核消息打印日志

```
message:
*****area type:message - location in message area:2*****
//内核异常发生的时间，该时间为UTC时间，北京时间=UTC时间+8小时
panic time:20080120231731-a7634
[_main]: create_bond4 bond0 ...
<6>[   24.448240] bonding: bond0: releasing active interface eth1
<1>[   24.534488] [os_mgtibc_netconfig.rc] config mgt ipv6 ip ...
<1>[   24.544442] [os_mgtibc_netconfig.rc] eth2 use default ipv6.
<5>[   24.561750] [6073][5400040600ce][INF][Startup mode value=40][BSP][COM_Star.eQuery,362]
[cat]
<6>[   24.579933] Extended CMOS year: 2000
<6>[   24.583514] Extended CMOS year: 2000
<5>[   24.616672] [6087][5400040607da][INF][Current bios channel is : [1].][BSP]
[SP5_GetB.ersion,1386][cat]
<5>[   24.635688] [6092][5400040607e3][INF][SPV2R1 BIOS Date:2013-07-12.][BSP]
[SPV2R1_G.osDate,1544][cat]
<4>[   24.654536] boot disk is /dev/sda
<1>[   24.694332] [os_netmgt_main]: create_ifcfg6 eth2 ...
<1>[   24.754426] [/etc/hotplug/os_mgtibc_netconfig.rc],OS_COPY_IP_FILE ifconfig-eth2/ip6g
not exist
```



```
<1>[   24.766967] [os_mgtibc_netconfig.rc] ------call product_set_firewall.sh .------
<1>[   24.792468] manage net is eth2.
<6>[   24.800258] ip_tables: (C) 2000-2006 Netfilter Core Team
<6>[   24.815956] nf_conntrack version 0.5.0 (16384 buckets, 65536 max)
<6>[   24.877504] ip6_tables: (C) 2000-2006 Netfilter Core Team
<6>[   32.728790] ADDRCONF(NETDEV_UP): eth2: link is not ready
<4>[   32.734510] ip used greatest stack depth: 3840 bytes left
<6>[   34.332129] e1000e: eth2 NIC Link is Up 100 Mbps Full Duplex, Flow Control: None
<6>[   34.339509] e1000e 0000:34:00.0: eth2: 10/100 speed: disabling TSO
<6>[   34.346105] ADDRCONF(NETDEV_CHANGE): eth2: link becomes ready
<6>[   37.306582] bonding: bond0: Adding slave eth1.
<6>[   37.389690] bonding: bond0: enslaving eth1 as an active interface with a down link.
<6>[   38.890611] NET: Registered protocol family 17
<1>[   38.930757] ----call maintain ip effect----
<1>[   38.969397] [os_mgtibc_netconfig.rc] ----Call maintain ip----
<1>[   38.979805] [os_mgtibc_netconfig.rc] config maintain ip ...
<1>[   39.009213] [os_netmgt_main]: get_ip4 eth2:1 ...
<1>[   39.071508] [os_netmgt_main]: get_pf4 eth2:1 ...
<1>[   39.133710] [os_netmgt_main]: get_gw4 eth2:1 ...
<1>[   39.196157] [os_netmgt_main]: get_bc4 eth2:1 ...
......
......
```

3. 控制台打印日志

```
console:
*****area type:console - location in console area:2*****
//内核异常发生的时间，该时间为UTC时间，北京时间=UTC时间+8小时
panic time:20080120231731-a7634
//内核栈信息
[  392.145269]  ffff880061163c58 ffffffff811f52a9 0000000000000000 ffff8800e1163e27
[  392.152675]  ffff880061163d18 ffffffff811f5611 0000000000000292 ffff880061163c98
[  392.160075]  ffff880061163c0a ffffffff0000ffff 0000000000000006 00ffffff819994a0
//内核函数调用轨迹
[  392.167477] Call Trace:
[  392.169912]  [<ffffffff811f52a9>] ? put_dec+0x59/0x60
[  392.174934]  [<ffffffff811f5611>] ? number+0x2f1/0x320
[  392.180044]  [<ffffffff811f5611>] ? number+0x2f1/0x320
[  392.185155]  [<ffffffff811f6a87>] ? vsnprintf+0x1d7/0x5b0
[  392.190524]  [<ffffffff81039166>] ? console_unlock+0x246/0x2a0
[  392.196325]  [<ffffffff8102e5a0>] ? __cpa_flush_all+0x60/0x60
[  392.202047]  [<ffffffffa0a9e020>] ? yh_exit+0x20/0x20 [hard_lockup]
[  392.208278]  [<ffffffff814046b4>] ? printk+0x3c/0x40
[  392.213213]  [<ffffffff811f7db4>] ? delay_tsc+0x54/0x90
[  392.218410]  [<ffffffffa0a9e020>] ? yh_exit+0x20/0x20 [hard_lockup]
[  392.224641]  [<ffffffff811f7e4f>] ? __const_udelay+0x2f/0x40
[  392.230270]  [<ffffffffa0a9e07a>] ? yh_init+0x5a/0xfe0 [hard_lockup]
[  392.236588]  [<ffffffff810001cd>] ? do_one_initcall+0x3d/0x180
[  392.242389]  [<ffffffff81092c95>] ? sys_init_module+0xc5/0x220
[  392.248188]  [<ffffffff8140e4f9>] ? system_call_fastpath+0x16/0x1b
[  392.255023] calling kbox_sync :begin
[  392.258576] sync kbox :begin
[  392.261438] open all redirect device :begin
[  392.265596] open all redirect device :end
[  392.269580] flush kbox regions :begin
[  392.273223] kbox region (panic) is writing into (hmem), action is 202
[  392.280057] test write len : 6590
[  392.283352] first start addr : ffff88007e586000
[  392.287854] second start addr : ffff88007e58e000
[  392.292444] cur_index : 2, offset : 32768, third start addr : ffff88007e58e008
[  392.299623] first length : 6590
[  392.302766] bios_write_file_data write data len *ptr_length : 6590
[  392.308908] cur_index : 3, record_number : 3, total_number : 16
[  392.314791] kbox region (panic) has been written into (hmem)
[  392.320847] dev hmem is dirty
//内核异常发生的时间
panic time:20080120231731-a7634
```

9.2.3 提供系统 oom 信息

本节主要介绍当内存不足发生时内核黑匣子所记录的异常信息。

功能概述

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余不足，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。

信息清单

记录的异常oom信息包含三个部分内容。

1. oom异常信息，最多记录信息容量为256K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 当前进程的pid、进程名称、内核版本、cpu号。
   - 异常进程的调用轨迹及栈信息（栈信息最多打印150行）。
   - 内存统计信息。
   - slab分配信息。
   - 当前系统中进程信息。
   - vmalloc信息。
   - rootfs/ramfs/tmpfs等内存文件系统中的文件信息。
   - rootfs/ramfs/tmpfs等内存文件系统中的文件占用内存的信息。
   - oom后的操作，0：不做任何操作 1：调用panic。
2. 内核消息打印日志，最多记录信息容量为64K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 异常发生时最近64K日志信息，即内核打印到循环缓冲区中的最后64K日志信息。
3. 控制台打印日志，最多记录信息容量为32K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 其他内核打印日志消息。

信息举例

在日志信息文件中，包含如下内容：

1. oom异常信息

```
// oom异常存储区，当前oom日志记录1个
area type: 0-panic 1-reboot 2-emerge 3-intermit 4-oom 5-oops/die 6-rlock 7-message 8-console
---------------area type:4, record num:1, unit_size:64 ------------


###num:0, record len:83407
---------KBOX_START----------
// 内核异常发生的时间，该时间为UTC时间，北京时间=UTC时间+8小时
oom time:20160218155821-e0387
//当前触发oom进程的pid，进程名及内核版本和cpu号
Current Pid: 44, comm: kworker/1:1 Tainted: G           O E ---- -------
3.10.0-229.20.1.22.hulk.x86_64 #1
//异常进程函数调用轨迹
Call Trace:
```

```
 [<ffffffffa037ca53>] kbox_dump_oom+0x13/0x30 [kbox]
 [<ffffffffa037bb17>] kbox_oom_callback+0x107/0x210 [kbox]
 [<ffffffff810a8c20>] ? wake_up_state+0x20/0x20
 [<ffffffff8109ffe8>] ? __wake_up_common+0x58/0x90
 [<ffffffff8160a2fc>] notifier_call_chain+0x4c/0x70
 [<ffffffff8109d07d>] __blocking_notifier_call_chain+0x4d/0x70
 [<ffffffff8109d0b6>] blocking_notifier_call_chain+0x16/0x20
 [<ffffffff81159e6e>] out_of_memory+0x4e/0x4f0
 [<ffffffff81390b1c>] ? tty_ldisc_deref+0x3c/0xa0
 [<ffffffff81394fad>] moom_callback+0x4d/0x50
 [<ffffffff8108f445>] process_one_work+0x175/0x430
 [<ffffffff8109028b>] worker_thread+0x11b/0x3a0
 [<ffffffff81090170>] ? manage_workers.isra.25+0x2a0/0x2a0
 [<ffffffff81096fbf>] kthread+0xcf/0xe0
 [<ffffffff81096ef0>] ? insert_kthread_work+0x40/0x40
 [<ffffffff8160e858>] ret_from_fork+0x58/0x90
 [<ffffffff81096ef0>] ? insert_kthread_work+0x40/0x40
oom stack:
 ffff8801385bfbf0 ffff8801385bfc48 ffff8801385bfc6c 0000000000000000
 ffff8801385bfdc8 0000000000000000 ffff8801385bfc48 ffffffffa037c04a
 0000000000000002 0000000045746160 0000000000000000 ffff8801385bfc58
 ffffffffa037ca53 ffff8801385bfce8 ffffffffa037bb17 36313032810a8c32
 3835353138313230 37383330652d3132 0000000000000000 ffff880035ec2280
 ffffffff810a8c20 0000000000000000 0000000000000000 0000000045746160
 ffff8801385bfcf8 ffffffff8109ffe8 0000000100000000 0000000000000000
 0000000045746160 00000000ffffffff 0000000000000000 ffff8801385bfd20
 ffffffff8160a2fc ffffffff81977f80 0000000000000000 0000000000000000
 ffff8801385bfdc8 00000000ffffffff ffff8801385bfd60 ffffffff8109d07d
 0000000000000000 0000000000000000 0000000000000001 ffff88013ec92f00
 ffff88013efd8300 00000000000000d0 ffff8801385bfd70 ffffffff8109d0b6
 ffff8801385bfe08 ffffffff81159e6e 00000000fffc4c8b 0000000000000000
 ffff8801385bfde0 0000000000000286 0000000000000286 0000000000000286
 ffff8801385bfdd0 ffffffff81390b1c 0000000000000282 0000000000000000
 ffff8801385bfe18 0000000045746160 ffffffff819bd9a0 ffff88013846a000
 ffff88013ec92f00 ffff88013ec96f00 0000000000000040 ffff8801385bfe18
 ffffffff81394fad ffff8801385bfe60 ffffffff8108f445 000000003ec92f00
 0000000000000000 ffff88013ec92f00 ffff88013ec92f18 ffff88013846a030
 ffff880138589700 ffff88013846a000 ffff8801385bfec0 ffffffff8109028b
 ffff8801385bffd8 ffff8801385bffd8 ffff880138589700 ffff880138589700
 ffff880138589700 ffff880138bafd38 ffff88013846a000 ffffffff81090170
 0000000000000000 0000000000000000 ffff8801385bff48 ffffffff81096fbf
 0000000000000000 0000000000000000 ffff88013846a000 0000000000000000
 0000000000000000 ffff8801385bfef8 ffff8801385bfef8 ffff880100000000
 ffff880100000000 ffff8801385bff18 ffff8801385bff18 0000000045746160
 ffffffff81096ef0 0000000000000000 0000000000000000 ffff880138bafd38
 ffffffff8160e858 0000000000000000 0000000000000000 0000000000000000
 0000000000000000 ffff880138bafd38 ffffffff81096ef0 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 0000000000000000
 ffffffffffffffff 0000000000000000 0000000000000010 0000000000000202
 ffff8801385bff58 0000000000000018
//内存统计信息
*******  meminfo  *******
MemTotal:     3768752 kB
MemFree:      3522112 kB
TotalHigh:          0 kB
FreeHigh:           0 kB
Buffers:          764 kB
SwapTotal:     4063228 kB
SwapFree:      4063228 kB
//slab分配信息
*******  slabinfo  *******
# name            <active_objs> <num_objs> <objsize> <objperslab> <pagesperslab>: tunables
<limit> <batchcount> <sharedfactor>: slabdata <active_slabs> <num_slabs> <sharedavail>
nf_conntrack_ffffffff819dcb40     50     50    320   25    2 : tunables    0    0    0 :
slabdata      2      2      0
xfs_icr                0      0    144   28    1 : tunables    0    0    0 :
slabdata      0      0      0
xfs_ili                104    104    152   26    1 : tunables    0    0    0 :
```

```
slabdata      4      4      0
xfs_inode                 1760   1760   1024   16    4 : tunables    0    0    0 :
slabdata    110    110      0
xfs_efd_item                80     80    400   20    2 : tunables    0    0    0 :
slabdata      4      4      0
xfs_da_state                64     64    488   16    2 : tunables    0    0    0 :
slabdata      4      4      0
xfs_btree_cur               76     76    208   19    1 : tunables    0    0    0 :
slabdata      4      4      0
xfs_log_ticket              88     88    184   22    1 : tunables    0    0    0 :
slabdata      4      4      0
scsi_cmd_cache              72     72    448   18    2 : tunables    0    0    0 :
slabdata      4      4      0
kcopyd_job                   0      0   3312    9    8 : tunables    0    0    0 :
slabdata      0      0      0
dm_uevent                    0      0   2608   12    8 : tunables    0    0    0 :
slabdata      0      0      0
dm_rq_target_io              0      0    424   19    2 : tunables    0    0    0 :
slabdata      0      0      0
UDPLITEv6                    0      0   1152   28    8 : tunables    0    0    0 :
slabdata      0      0      0
UDPv6                       84     84   1152   28    8 : tunables    0    0    0 :
slabdata      3      3      0
tw_sock_TCPv6                0      0    256   16    1 : tunables    0    0    0 :
slabdata      0      0      0
TCPv6                       16     16   2048   16    8 : tunables    0    0    0 :
slabdata      1      1      0
cfq_queue                  102    102    232   17    1 : tunables    0    0    0 :
slabdata      6      6      0
bsg_cmd                      0      0    312   26    2 : tunables    0    0    0 :
slabdata      0      0      0
mqueue_inode_cache          18     18    896   18    4 : tunables    0    0    0 :
slabdata      1      1      0
hugetlbfs_inode_cache       52     52    608   26    4 : tunables    0    0    0 :
slabdata      2      2      0
configfs_dir_cache           0      0     88   46    1 : tunables    0    0    0 :
slabdata      0      0      0
dquot                        0      0    256   16    1 : tunables    0    0    0 :
slabdata      0      0      0
kioctx                       0      0    576   28    4 : tunables    0    0    0 :
slabdata      0      0      0
pid_namespace                0      0   2176   15    8 : tunables    0    0    0 :
slabdata      0      0      0
posix_timers_cache           0      0    248   16    1 : tunables    0    0    0 :
slabdata      0      0      0
UDP-Lite                     0      0   1024   16    4 : tunables    0    0    0 :
slabdata      0      0      0
......

//当时系统的进程信息
*******  kbox show all tasks  *******
[ pid ]   uid  ppid  tgid total_vm      rss cpu stat name
[    1]     0     0     1    14373     1847   3    S                    systemd
[  482]     0     1   482    10753      663   1    R           systemd-journal
[  497]     0     1   497    12521     1073   1    S                   lvmetad
[  503]     0     1   503    10676      546   1    S             systemd-udevd
[  585]     0     1   585    29179      406   0    S                    auditd
[  607]     0     1   607    66139     5887   0    S                firewalld
[  612]     0     1   612    35618      693   3    S                  rsyslogd
[  614]     0     1   614    88477     4062   0    S                     tuned
[  615]     0     1   615     4787      289   3    S                irqbalance
[  616]     0     1   616     8677      419   0    S             systemd-logind
[  617]    81     1   617     8738      450   1    S                dbus-daemon
[  625]     0     1   625    31582      414   2    S                      crond
[  630]     0     1   630     1628       85   0    S                     mcelog
[  633]     0     1   633    27507      198   2    S                     agetty
[  776]     0     1   776    73003     1861   1    S             NetworkManager
[  873]   999     1   873    46674     2780   3    S                    polkitd
[  913]     0     1   913    13262      673   0    S             wpa_supplicant
```

```
[ 1227]     0     1  1227    16627      814   0    S                              sshd
[ 2436]     0  1227  2436    31363     1154   2    S                              sshd
[ 2440]     0  2436  2440    28844      473   0    S                              bash
[ 2472]     0  2440  2472    28844      196   2    R                              bash
total real task number: 21
*****************Start oom extend info.*****************
Vmallocinfo Start >>>>>>>>>>>>>>>>>>>>//通过内核接口申请的内存
0xffffc90000000000-0xffffc90000002000       8192 hpet_enable+0x2d/0x2e9 phys=0xfed00000
ioremap
0xffffc90000002000-0xffffc90000403000    4198400 alloc_large_system_hash+0x171/0x239
pages=1024 vmalloc vpages
0xffffc90000403000-0xffffc90000406000      12288 alloc_large_system_hash+0x171/0x239 pages=2
vmalloc
0xffffc90000406000-0xffffc90000607000    2101248 alloc_large_system_hash+0x171/0x239
pages=512 vmalloc
0xffffc90000608000-0xffffc9000060b000      12288 acpi_os_map_memory+0xea/0x142
phys=0xbffdf000 ioremap
0xffffc9000060c000-0xffffc9000060f000      12288 acpi_os_map_memory+0xea/0x142
phys=0xbffe0000 ioremap
0xffffc90000610000-0xffffc90000612000       8192 acpi_os_map_memory+0xea/0x142
phys=0xfed00000 ioremap
0xffffc90000612000-0xffffc90000653000     266240 alloc_large_system_hash+0x171/0x239
pages=64 vmalloc
0xffffc90000653000-0xffffc900006d4000     528384 alloc_large_system_hash+0x171/0x239
pages=128 vmalloc
0xffffc900006d4000-0xffffc900006e5000      69632 alloc_large_system_hash+0x171/0x239
pages=16 vmalloc
0xffffc900006e5000-0xffffc900006f6000      69632 alloc_large_system_hash+0x171/0x239
pages=16 vmalloc
0xffffc900006f6000-0xffffc90000707000      69632 alloc_large_system_hash+0x171/0x239
pages=16 vmalloc
0xffffc90000708000-0xffffc9000070a000       8192 cirrus_device_init+0x86/0x130 [cirrus]
phys=0xfebf4000 ioremap
0xffffc9000070a000-0xffffc9000070c000       8192 dm_table_create+0x9e/0x130 [dm_mod] pages=1
vmalloc
0xffffc9000070c000-0xffffc9000072d000     135168 raw_init+0x41/0x141 pages=32 vmalloc
0xffffc9000072d000-0xffffc90000736000      36864 drm_ht_create+0x55/0x80 [drm] pages=8
vmalloc
0xffffc90000736000-0xffffc90000738000       8192 dm_table_create+0x9e/0x130 [dm_mod] pages=1
vmalloc
0xffffc90000739000-0xffffc9000073c000      12288 cirrusfb_create+0xde/0x390 [cirrus] pages=2
vmalloc
0xffffc9000073c000-0xffffc9000073e000       8192 swap_cgroup_swapon+0x48/0x160 pages=1
vmalloc
0xffffc9000073e000-0xffffc90000740000       8192 ebt_register_table+0xa2/0x380 [ebtables]
pages=1 vmalloc
0xffffc90000740000-0xffffc90000761000     135168 pci_ioremap_bar+0x46/0x80 phys=0xfebc0000
ioremap
0xffffc90000761000-0xffffc900009a2000    2363392 cirrusfb_create+0xde/0x390 [cirrus]
pages=576 vmalloc vpages
......
Vmallocinfo End <<<<<<<<<<<<<<<<<<<<

//内存文件系统
Filesystem            1K-blocks    Used   Available Use(%)   Mounted on
tmpfs                  1884376        0  1884376        0%   /dev/shm
tmpfs                  1884376     8504  1875872        0%   /run
tmpfs                  1884376        0  1884376        0%   /sys/fs/cgroup
********    mem info     ********?Total:                3768752 kB
?Total free:           3521908 kB
?User space:            157772 kB
?Mlock:                      0 kB
?Kernel space:           89072 kB
?Bootmem reserved:      409164 kB
?HugePages_Total:            0 kB
?HugePages_Free:             0 kB
?HugePages_Rsvd:             0 kB
?HugePages_Surp:             0 kB
?Hugepagesize:            2048 kB
```



```
//占用内存的文件，从大到小
******     pagecache_info:     ******
  /run/log/journal/ad0c8c7dc64c4ecb860400450d3d0825/system.journal : nrpages = 1971.
  /run/udev/data/c116:33 : nrpages = 1.
  /run/udev/data/c116:3 : nrpages = 1.
  /run/udev/data/b8:2 : nrpages = 1.
  /run/systemd/system/session-1.scope.d/90-SendSIGHUP.conf : nrpages = 1.
  /run/udev/data/b253:1 : nrpages = 1.
  /run/mcelog.pid : nrpages = 1.
  /run/udev/data/n2 : nrpages = 1.
  /run/systemd/system/session-1.scope.d/90-Slice.conf : nrpages = 1.
  /run/udev/data/b8:1 : nrpages = 1.
  /run/syslogd.pid : nrpages = 1.
  /run/udev/data/b8:0 : nrpages = 1.
  /unknown : nrpages = 1.
  /run/udev/data/b11:0 : nrpages = 1.
  /run/crond.pid : nrpages = 1.
  /run/udev/data/c13:33 : nrpages = 1.
  /run/systemd/system/session-1.scope : nrpages = 1.
  /run/udev/data/c13:67 : nrpages = 1.
  /run/auditd.pid : nrpages = 1.
  ......
******************End oom extend info.*****************
//oom异常记录后的操作(0:返回调用链，1:调用panic)
action after oom is:1
---------KBOX_END------------
```

2. 内核消息打印日志

```
---------------area type:7, record num:1, unit_size:64 ------------


###num:0, record len:39434
oom time:20160218155821-e0387
[    0.     0] Initializing cgroup subsys cpuset
[    0.     0] Initializing cgroup subsys cpu
[    0.     0] Initializing cgroup subsys cpuacct
[    0.     0] Linux version 3.10.0-229.20.1.22.hulk.x86_64 (abuild@HGH1000006709) (gcc
version 4.8.3 20140911 (EulerOS 4.8.3-10) (GCC) ) #1 SMP Thu Feb 4 14:30:09 UTC 2016
[    0.     0] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.22.hulk.x86_64 root=/dev/
mapper/euleros-root ro reserve_kbox_mem=16M crash_kexec_post_notifiers panic=3
8250.nr_uarts=8 efi=old_map rd.lvm.lv=euleros/swap crashkernel=2G-32G:256M,32G-1T:512M,1T-8T:
1G,8T-:4G rd.lvm.lv=euleros/root rhgb quiet
[    0.     0] e820: BIOS-provided physical RAM map:
[    0.     0] BIOS-e820: [mem 0x0000000000000000-0x000000000009fbff] usable
[    0.     0] BIOS-e820: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[    0.     0] BIOS-e820: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[    0.     0] BIOS-e820: [mem 0x0000000000100000-0x00000000bffdefff] usable
[    0.     0] BIOS-e820: [mem 0x00000000bffdf000-0x00000000bfffffff] reserved
[    0.     0] BIOS-e820: [mem 0x00000000feffc000-0x00000000feffffff] reserved
[    0.     0] BIOS-e820: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[    0.     0] BIOS-e820: [mem 0x0000000100000000-0x000000013fffffff] usable
[    0.     0] e820: reserve kbox memory size: 16M
[    0.     0] e820: reserve kbox memory success. [mem 0x000000013f000000-0x000000013fffffff]
[    0.     0] NX (Execute Disable) protection: active
[    0.     0] e820: user-defined physical RAM map:
[    0.     0] user: [mem 0x0000000000000000-0x000000000009fbff] usable
[    0.     0] user: [mem 0x000000000009fc00-0x000000000009ffff] reserved
[    0.     0] user: [mem 0x00000000000f0000-0x00000000000fffff] reserved
[    0.     0] user: [mem 0x0000000000100000-0x00000000bffdefff] usable
[    0.     0] user: [mem 0x00000000bffdf000-0x00000000bfffffff] reserved
[    0.     0] user: [mem 0x00000000feffc000-0x00000000feffffff] reserved
[    0.     0] user: [mem 0x00000000fffc0000-0x00000000ffffffff] reserved
[    0.     0] user: [mem 0x0000000100000000-0x000000013effffff] usable
[    0.     0] user: [mem 0x000000013f000000-0x000000013fffffff] reserved
[    0.     0] SMBIOS 2.8 present.
[    0.     0] DMI: QEMU Standard PC (i440FX + PIIX, 1996), BIOS rel-1.7.5-0-
ge51488c-20140602_164612-nilsson.home.kraxel.org 04/01/2014
[    0.     0] Hypervisor detected: KVM
[    0.     0] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[    0.     0] e820: remove [mem 0x000a0000-0x000fffff] usable
```



```
[    0.      0] No AGP bridge found
[    0.      0] e820: last_pfn = 0x13f000 max_arch_pfn = 0x400000000
[    0.      0] MTRR default type: write-back
[    0.      0] MTRR fixed ranges enabled:
[    0.      0]   00000-9FFFF write-back
[    0.      0]   A0000-BFFFF uncachable
[    0.      0]   C0000-FFFFF write-protect
[    0.      0] MTRR variable ranges enabled:
[    0.      0]   0 base 000C0000000 mask 3FFC0000000 uncachable
[    0.      0]   1 disabled
[    0.      0]   2 disabled
[    0.      0]   3 disabled
[    0.      0]   4 disabled
[    0.      0]   5 disabled
[    0.      0]   6 disabled
[    0.      0]   7 disabled
[    0.      0] PAT not supported by CPU.
[    0.      0] e820: last_pfn = 0xbffdf max_arch_pfn = 0x400000000
[    0.      0] found SMP MP-table at [mem 0x000f0e50-0x000f0e5f] mapped at [ffff8800000f0e50]
[    0.      0] Base memory trampoline at [ffff880000099000] 99000 size 24576
[    0.      0] init_memory_mapping: [mem 0x00000000-0x000fffff]
[    0.      0]  [mem 0x00000000-0x000fffff] page 4k
[    0.      0] BRK [0x01ee3000, 0x01ee3fff] PGTABLE
[    0.      0] BRK [0x01ee4000, 0x01ee4fff] PGTABLE
[    0.      0] BRK [0x01ee5000, 0x01ee5fff] PGTABLE
[    0.      0] init_memory_mapping: [mem 0x13ee00000-0x13efffffff]
[    0.      0]  [mem 0x13ee00000-0x13efffffff] page 2M
[    0.      0] BRK [0x01ee6000, 0x01ee6fff] PGTABLE
[    0.      0] init_memory_mapping: [mem 0x13c000000-0x13edfffff]
[    0.      0]  [mem 0x13c000000-0x13edfffff] page 2M
[    0.      0] init_memory_mapping: [mem 0x100000000-0x13bfffffff]
[    0.      0]  [mem 0x100000000-0x13bfffffff] page 2M
[    0.      0] init_memory_mapping: [mem 0x00100000-0xbffdefff]
[    0.      0]  [mem 0x00100000-0x001fffff] page 4k
[    0.      0]  [mem 0x00200000-0xbfdfffff] page 2M
[    0.      0]  [mem 0xbfe00000-0xbffdefff] page 4k
[    0.      0] RAMDISK: [mem 0x35ea8000-0x36f4bfff]
[    0.      0] ACPI: RSDP 00000000000f0e00 00014 (v00 BOCHS )
[    0.      0] ACPI: RSDT 00000000bffe16cf 00034 (v01 BOCHS  BXPCRSDT 00000001 BXPC 00000001)
[    0.      0] ACPI: FACP 00000000bffe0877 00074 (v01 BOCHS  BXPCFACP 00000001 BXPC 00000001)
[    0.      0] ACPI: DSDT 00000000bffdfd80 00AF7 (v01 BOCHS  BXPCDSDT 00000001 BXPC 00000001)
[    0.      0] ACPI: FACS 00000000bffdfd40 00040
[    0.      0] ACPI: SSDT 00000000bffe08eb 00D1C (v01 BOCHS  BXPCSSDT 00000001 BXPC 00000001)
[    0.      0] ACPI: APIC 00000000bffe1607 00090 (v01 BOCHS  BXPCAPIC 00000001 BXPC 00000001)
[    0.      0] ACPI: HPET 00000000bffe1697 00038 (v01 BOCHS  BXPCHPET 00000001 BXPC 00000001)
[    0.      0] ACPI: Local APIC address 0xfee00000
[    0.      0] No NUMA configuration found
[    0.      0] Faking a node at [mem 0x0000000000000000-0x000000013efffffff]
[    0.      0] Initmem setup node 0 [mem 0x00000000-0x13efffffff]
[    0.      0]  NODE_DATA [mem 0x13efd6000-0x13effcfff]
[    0.      0] Reserving 256MB of memory at 592MB for crashkernel (System RAM: 4079MB)
[    0.      0] kvm-clock: Using msrs 4b564d01 and 4b564d00
[    0.      0] kvm-clock: cpu 0, msr 1:3ef86001, primary cpu clock
[    0.      0]  [ffffea0000000000-ffffea0004ffffff] PMD -> [ffff88013a600000-
ffff88013e5fffff] on node 0
[    0.      0] Zone ranges:
[    0.      0]   DMA      [mem 0x00001000-0x00ffffff]
[    0.      0]   DMA32    [mem 0x01000000-0xffffffff]
[    0.      0]   Normal   [mem 0x100000000-0x13efffffff]
[    0.      0] Movable zone start for each node
[    0.      0] Mirror info
[    0.      0] Early memory node ranges
[    0.      0]   node   0: [mem 0x00001000-0x0009efff]
[    0.      0]   node   0: [mem 0x00100000-0xbffdefff]
[    0.      0]   node   0: [mem 0x100000000-0x13efffffff]
[    0.      0] On node 0 totalpages: 1044349
[    0.      0]   DMA zone: 64 pages used for memmap
[    0.      0]   DMA zone: 21 pages reserved
[    0.      0]   DMA zone: 3998 pages, LIFO batch:0
```



```
[    0.      0]   DMA32 zone: 12224 pages used for memmap
[    0.      0]   DMA32 zone: 782303 pages, LIFO batch:31
[    0.      0]   Normal zone: 4032 pages used for memmap
[    0.      0]   Normal zone: 258048 pages, LIFO batch:31
[    0.      0] ACPI: PM-Timer IO Port: 0x608
[    0.      0] ACPI: Local APIC address 0xfee00000
[    0.      0] ACPI: LAPIC_NMI (acpi_id[0xff] dfl dfl lint[0x1])
[    0.      0] IOAPIC[0]: apic_id 0, version 17, address 0xfec00000, GSI 0-23
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 5 global_irq 5 high level)
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 high level)
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 10 global_irq 10 high level)
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 11 global_irq 11 high level)
[    0.      0] ACPI: IRQ0 used by override.
[    0.      0] ACPI: IRQ2 used by override.
[    0.      0] ACPI: IRQ5 used by override.
[    0.      0] ACPI: IRQ9 used by override.
[    0.      0] ACPI: IRQ10 used by override.
[    0.      0] ACPI: IRQ11 used by override.
[    0.      0] Using ACPI (MADT) for SMP configuration information
[    0.      0] ACPI: HPET id: 0x8086a201 base: 0xfed00000
[    0.      0] smpboot: Allowing 4 CPUs, 0 hotplug CPUs
[    0.      0] nr_irqs_gsi: 40
[    0.      0] PM: Registered nosave memory: [mem 0x0009f000-0x0009ffff]
[    0.      0] PM: Registered nosave memory: [mem 0x000a0000-0x000effff]
[    0.      0] PM: Registered nosave memory: [mem 0x000f0000-0x000fffff]
[    0.      0] PM: Registered nosave memory: [mem 0xbffdf000-0xbfffffff]
[    0.      0] PM: Registered nosave memory: [mem 0xc0000000-0xfeffbfff]
[    0.      0] PM: Registered nosave memory: [mem 0xfeffc000-0xfeffffff]
[    0.      0] PM: Registered nosave memory: [mem 0xff000000-0xfffbffff]
[    0.      0] PM: Registered nosave memory: [mem 0xfffc0000-0xffffffff]
[    0.      0] e820: [mem 0xc0000000-0xfeffbfff] available for PCI devices
[    0.      0] Booting paravirtualized kernel on KVM
[    0.      0] setup_percpu: NR_CPUS:5120 nr_cpumask_bits:4 nr_cpu_ids:4 nr_node_ids:1
[    0.      0] PERCPU: Embedded 28 pages/cpu @ffff88013ec00000 s82880 r8192 d23616 u524288
[    0.      0] pcpu-alloc: s82880 r8192 d23616 u524288 alloc=1*2097152
[    0.      0] pcpu-alloc: [0] 0 1 2 3
[    0.      0] KVM setup async PF for cpu 0
[    0.      0] kvm-stealtime: cpu 0, msr 13ec0d000
[    0.      0] Built 1 zonelists in Node order, mobility grouping on.  Total pages: 1028008
[    0.      0] Policy zone: Normal
[    0.      0] Kernel command line: BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.22.hulk.x86_64
root=/dev/mapper/euleros-root ro reserve_kbox_mem=16M crash_kexec_post_notifiers panic=3
8250.nr_uarts=8 efi=old_map rd.lvm.lv=euleros/swap crashkernel=2G-32G:256M,32G-1T:512M,1T-8T:
1G,8T-:4G rd.lvm.lv=euleros/root rhgb quiet
[    0.      0] PID hash table entries: 4096 (order: 3, 32768 bytes)
[    0.      0] Checking aperture...
[    0.      0] No AGP bridge found
[    0.      0] Memory: 3750084k/5226496k available (6220k kernel code, 1049100k absent,
427312k reserved, 4174k data, 1604k init)
[    0.      0] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=4, Nodes=1
[    0.      0] Hierarchical RCU implementation.
[    0.      0] ?RCU restricting CPUs from NR_CPUS=5120 to nr_cpu_ids=4.
[    0.      0] ?Experimental no-CBs for all CPUs
[    0.      0] ?Experimental no-CBs CPUs: 0-3.
[    0.      0] NR_IRQS:327936 nr_irqs:712 0
[    0.      0] Console: colour VGA+ 80x25
[    0.      0] console [tty0] enabled
[    0.      0] allocated 16777216 bytes of page_cgroup
[    0.      0] please try 'cgroup_disable=memory' option if you don't want memory cgroups
[    0.      0] hpet clockevent registered
[    0.      0] tsc: Detected 2099.998 MHz processor
[    0.   2000] Calibrating delay loop (skipped) preset value.. 4199.99 BogoMIPS (lpj=2099998)
[    0.   2000] pid_max: default: 32768 minimum: 301
[    0.   2000] Security Framework initialized
[    0.   2000] SELinux:  Initializing.
[    0.   2000] SELinux:  Starting in permissive mode
[    0.   2000] Dentry cache hash table entries: 524288 (order: 10, 4194304 bytes)
[    0.   4006] Inode-cache hash table entries: 262144 (order: 9, 2097152 bytes)
```



```
[    0.  4512] Mount-cache hash table entries: 4096
[    0.  4709] Initializing cgroup subsys memory
[    0.  4717] Initializing cgroup subsys devices
[    0.  4719] Initializing cgroup subsys freezer
[    0.  4720] Initializing cgroup subsys net_cls
[    0.  4722] Initializing cgroup subsys blkio
[    0.  4724] Initializing cgroup subsys perf_event
[    0.  4726] Initializing cgroup subsys hugetlb
[    0.  5052] mce: CPU supports 10 MCE banks
[    0.  5096] Last level iTLB entries: 4KB 0, 2MB 0, 4MB 0
Last level dTLB entries: 4KB 0, 2MB 0, 4MB 0
tlb_flushall_shift: 6
[    0.  5263] Freeing SMP alternatives: 24k freed
[    0.  9184] ACPI: Core revision 20130517
[    0. 10146] ACPI: All ACPI Tables successfully acquired
[    0. 10187] ftrace: allocating 24008 entries in 94 pages
[    0. 17279] Enabling x2apic
[    0. 17286] Enabled x2apic
[    0. 17486] Switched APIC routing to physical x2apic.
[    0. 18975] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[    0. 18977] smpboot: CPU0: Intel QEMU Virtual CPU version 2.1.0 (fam: 06, model: 06,
stepping: 03)
[    0. 19000] Performance Events: Broken PMU hardware detected, using software events only.
[    0. 19007] Failed to access perfctr msr (MSR c1 is 0)
[    0. 20737] NMI watchdog: disabled (cpu0): hardware events not enabled
[    0. 20729] kvm-clock: cpu 1, msr 1:3ef86041, secondary cpu clock
[    0. 33015] KVM setup async PF for cpu 1
[    0. 33021] kvm-stealtime: cpu 1, msr 13ec8d000
[    0. 33023] kvm-clock: cpu 2, msr 1:3ef86081, secondary cpu clock
[    0. 45016] KVM setup async PF for cpu 2
[    0. 45023] kvm-stealtime: cpu 2, msr 13ed0d000
[    0. 20814] smpboot: Booting Node   0, Processors  #1 #2 #3 OK
[    0. 45023] kvm-clock: cpu 3, msr 1:3ef860c1, secondary cpu clock
[    0. 58021] Brought up 4 CPUs
[    0. 58025] smpboot: Total of 4 processors activated (16799.98 BogoMIPS)
[    0. 58014] KVM setup async PF for cpu 3
[    0. 58018] kvm-stealtime: cpu 3, msr 13ed8d000
[    0. 59002] devtmpfs: initialized
[    0. 61851] EVM: security.selinux
[    0. 61853] EVM: security.ima
[    0. 61854] EVM: security.capability
[    0. 62993] atomic64 test passed for x86-64 platform with CX8 and with SSE
[    0. 63103] NET: Registered protocol family 16
[    0. 63377] ACPI: bus type PCI registered
[    0. 63379] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[    0. 63503] PCI: Using configuration type 1 for base access
[    0. 64399] ACPI: Added _OSI(Module Device)
[    0. 64399] ACPI: Added _OSI(Processor Device)
[    0. 64399] ACPI: Added _OSI(3.0 _SCP Extensions)
[    0. 64399] ACPI: Added _OSI(Processor Aggregator Device)
[    0. 64399] ACPI: Added _OSI(EulerOS RAS Extensions)
[    0. 65301] ACPI: EC: Look up EC in DSDT
[    0. 66195] ACPI: Interpreter enabled
[    0. 66201] ACPI Exception: AE_NOT_FOUND, While evaluating Sleep State [\_S1_] (20130517/
hwxface-571)
[    0. 66205] ACPI Exception: AE_NOT_FOUND, While evaluating Sleep State [\_S2_] (20130517/
hwxface-571)
[    0. 66216] ACPI: (supports S0 S3 S4 S5)
[    0. 66217] ACPI: Using IOAPIC for interrupt routing
[    0. 66230] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and
report a bug
[    0. 68961] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-ff])
[    0. 68967] acpi PNP0A03:00: _OSC: OS supports [ASPM ClockPM Segments MSI]
[    0. 68972] acpi PNP0A03:00: _OSC failed (AE_NOT_FOUND); disabling ASPM
[    0. 69063] acpi PNP0A03:00: fail to add MMCONFIG information, can't access extended PCI
configuration space under this bridge.
[    0. 69322] acpiphp: Slot [3] registered
[    0. 69346] acpiphp: Slot [4] registered
[    0. 69370] acpiphp: Slot [5] registered
```



```
[    0.69394] acpiphp: Slot [6] registered
[    0.69416] acpiphp: Slot [7] registered
[    0.69437] acpiphp: Slot [8] registered
[    0.69457] acpiphp: Slot [9] registered
[    0.69478] acpiphp: Slot [10] registered
[    0.69499] acpiphp: Slot [11] registered
[    0.69519] acpiphp: Slot [12] registered
[    0.69541] acpiphp: Slot [13] registered
[    0.69563] acpiphp: Slot [14] registered
[    0.69583] acpiphp: Slot [15] registered
[    0.69604] acpiphp: Slot [16] registered
[    0.69625] acpiphp: Slot [17] registered
[    0.69645] acpiphp: Slot [18] registered
[    0.69666] acpiphp: Slot [19] registered
[    0.69688] acpiphp: Slot [20] registered
[    0.69710] acpiphp: Slot [21] registered
[    0.69730] acpiphp: Slot [22] registered
[    0.69751] acpiphp: Slot [23] registered
[    0.69771] acpiphp: Slot [24] registered
[    0.69792] acpiphp: Slot [25] registered
[    0.69813] acpiphp: Slot [26] registered
[    0.69835] acpiphp: Slot [27] registered
[    0.69856] acpiphp: Slot [28] registered
[    0.69877] acpiphp: Slot [29] registered
[    0.69898] acpiphp: Slot [30] registered
[    0.69919] acpiphp: Slot [31] registered
[    0.69931] PCI host bridge to bus 0000:00
[    0.69934] pci_bus 0000:00: root bus resource [bus 00-ff]
[    0.69937] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7]
[    0.69939] pci_bus 0000:00: root bus resource [io  0x0d00-0xadff]
[    0.69941] pci_bus 0000:00: root bus resource [io  0xae0f-0xaeff]
[    0.69943] pci_bus 0000:00: root bus resource [io  0xaf20-0xafdf]
[    0.69945] pci_bus 0000:00: root bus resource [io  0xafe4-0xffff]
[    0.69947] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff]
[    0.69949] pci_bus 0000:00: root bus resource [mem 0xc0000000-0xfebfffff]
[    0.69991] pci 0000:00:00.0: [8086:1237] type 00 class 0x060000
[    0.70387] pci 0000:00:01.0: [8086:7000] type 00 class 0x060100
[    0.70934] pci 0000:00:01.1: [8086:7010] type 00 class 0x010180
[    0.75005] pci 0000:00:01.1: reg 0x20: [io  0xc080-0xc08f]
[    0.76432] pci 0000:00:01.1: legacy IDE quirk: reg 0x10: [io  0x01f0-0x01f7]
[    0.76435] pci 0000:00:01.1: legacy IDE quirk: reg 0x14: [io  0x03f6]
[    0.76437] pci 0000:00:01.1: legacy IDE quirk: reg 0x18: [io  0x0170-0x0177]
[    0.76439] pci 0000:00:01.1: legacy IDE quirk: reg 0x1c: [io  0x0376]
[    0.76594] pci 0000:00:01.2: [8086:7020] type 00 class 0x0c0300
[    0.80787] pci 0000:00:01.2: reg 0x20: [io  0xc040-0xc05f]
[    0.82574] pci 0000:00:01.3: [8086:7113] type 00 class 0x068000
[    0.83098] pci 0000:00:01.3: quirk: [io  0x0600-0x063f] claimed by PIIX4 ACPI
[    0.83111] pci 0000:00:01.3: quirk: [io  0x0700-0x070f] claimed by PIIX4 SMB
[    0.83336] pci 0000:00:02.0: [1013:00b8] type 00 class 0x030000
[    0.85006] pci 0000:00:02.0: reg 0x10: [mem 0xfc000000-0xfdffffff pref]
[    0.87006] pci 0000:00:02.0: reg 0x14: [mem 0xfebf4000-0xfebf4fff]
[    0.97005] pci 0000:00:02.0: reg 0x30: [mem 0xfebe0000-0xfebeffff pref]
[    0.97204] pci 0000:00:03.0: [8086:100e] type 00 class 0x020000
[    0.99005] pci 0000:00:03.0: reg 0x10: [mem 0xfebc0000-0xfebdffff]
[    0.100004] pci 0000:00:03.0: reg 0x14: [io  0xc000-0xc03f]
[    0.108005] pci 0000:00:03.0: reg 0x30: [mem 0xfeb80000-0xfebbffff pref]
[    0.108209] pci 0000:00:04.0: [8086:2668] type 00 class 0x040300
[    0.109004] pci 0000:00:04.0: reg 0x10: [mem 0xfebf0000-0xfebf3fff]
[    0.114288] pci 0000:00:06.0: [1af4:1002] type 00 class 0x00ff00
[    0.115005] pci 0000:00:06.0: reg 0x10: [io  0xc060-0xc07f]
[    0.120244] pci 0000:00:1e.0: [1af4:1110] type 00 class 0x050000
[    0.122005] pci 0000:00:1e.0: reg 0x10: [mem 0xfebf5000-0xfebf50ff]
[    0.126005] pci 0000:00:1e.0: reg 0x18: [mem 0xfe000000-0xfe7fffff 64bit pref]
[    0.132465] ACPI: PCI Interrupt Link [LNKA] (IRQs 5 *10 11)
[    0.132576] ACPI: PCI Interrupt Link [LNKB] (IRQs 5 *10 11)
[    0.132673] ACPI: PCI Interrupt Link [LNKC] (IRQs 5 10 *11)
[    0.132777] ACPI: PCI Interrupt Link [LNKD] (IRQs 5 10 *11)
[    0.132827] ACPI: PCI Interrupt Link [LNKS] (IRQs *9)
[    0.133399] ACPI: Enabled 16 GPEs in block 00 to 0F
```



```
[    0.133478] vgaarb: device added: PCI:0000:00:02.0,decodes=io+mem,owns=io+mem,locks=none
[    0.133478] vgaarb: loaded
[    0.133478] vgaarb: bridge control possible 0000:00:02.0
[    0.133478] SCSI subsystem initialized
[    0.133478] ACPI: bus type USB registered
[    0.133478] usbcore: registered new interface driver usbfs
[    0.133478] usbcore: registered new interface driver hub
[    0.133478] usbcore: registered new device driver usb
[    0.133478] PCI: Using ACPI for IRQ routing
[    0.133478] PCI: pci_cache_line_size set to 64 bytes
[    0.134030] e820: reserve RAM buffer [mem 0x0009fc00-0x0009ffff]
[    0.134032] e820: reserve RAM buffer [mem 0xbffdf000-0xbfffffff]
[    0.134034] e820: reserve RAM buffer [mem 0x13f000000-0x13fffffff]
[    0.134136] NetLabel: Initializing
[    0.134138] NetLabel:  domain hash size = 128
[    0.134139] NetLabel:  protocols = UNLABELED CIPSOv4
[    0.134155] NetLabel:  unlabeled traffic allowed by default
[    0.134225] HPET: 3 timers in total, 0 timers will be used for per-cpu timer
[    0.134244] hpet0: at MMIO 0xfed00000, IRQs 2, 8, 0
[    0.134248] hpet0: 3 comparators, 64-bit 100.000000 MHz counter
[    0.138016] Switching to clocksource kvm-clock
[    0.143436] pnp: PnP ACPI init
[    0.143448] ACPI: bus type PNP registered
[    0.143515] pnp 00:00: Plug and Play ACPI device, IDs PNP0b00 (active)
[    0.143565] pnp 00:01: Plug and Play ACPI device, IDs PNP0303 (active)
[    0.143605] pnp 00:02: Plug and Play ACPI device, IDs PNP0f13 (active)
[    0.143646] pnp 00:03: [dma 2]
[    0.143660] pnp 00:03: Plug and Play ACPI device, IDs PNP0700 (active)
[    0.143757] pnp 00:04: Plug and Play ACPI device, IDs PNP0501 (active)
[    0.143905] pnp 00:05: Plug and Play ACPI device, IDs PNP0103 (active)
[    0.144030] pnp: PnP ACPI: found 6 devices
[    0.144032] ACPI: bus type PNP unregistered
[    0.151861] pci_bus 0000:00: resource 4 [io  0x0000-0x0cf7]
[    0.151864] pci_bus 0000:00: resource 5 [io  0x0d00-0xadff]
[    0.151866] pci_bus 0000:00: resource 6 [io  0xae0f-0xaeff]
[    0.151868] pci_bus 0000:00: resource 7 [io  0xaf20-0xafdf]
[    0.151870] pci_bus 0000:00: resource 8 [io  0xafe4-0xffff]
[    0.151872] pci_bus 0000:00: resource 9 [mem 0x000a0000-0x000bffff]
[    0.151874] pci_bus 0000:00: resource 10 [mem 0xc0000000-0xfebfffff]
[    0.151901] NET: Registered protocol family 2
[    0.152048] TCP established hash table entries: 32768 (order: 6, 262144 bytes)
[    0.152139] TCP bind hash table entries: 32768 (order: 7, 524288 bytes)
[    0.152193] TCP: Hash tables configured (established 32768 bind 32768)
[    0.152209] TCP: reno registered
[    0.152217] UDP hash table entries: 2048 (order: 4, 65536 bytes)
[    0.152235] UDP-Lite hash table entries: 2048 (order: 4, 65536 bytes)
[    0.152296] NET: Registered protocol family 1
[    0.152307] pci 0000:00:00.0: Limiting direct PCI/PCI transfers
[    0.152328] pci 0000:00:01.0: PIIX3: Enabling Passive Release
[    0.152348] pci 0000:00:01.0: Activating ISA DMA hang workarounds
[    0.182906] ACPI: PCI Interrupt Link [LNKD] enabled at IRQ 11
[    0.213623] pci 0000:00:02.0: Boot video device
[    0.213672] PCI: CLS 0 bytes, default 64
[    0.213724] Unpacking initramfs...
[    0.523944] Freeing initrd memory: 17040k freed
[    0.528748] PCI-DMA: Using software bounce buffering for IO (SWIOTLB)
[    0.528753] software IO TLB [mem 0xbbfdf000-0xbffdf000] (64MB) mapped at
[ffff8800bbfdf000-ffff8800bffdefff]
[    0.529517] microcode: CPU0 sig=0x663, pf=0x1, revision=0x1
[    0.529534] microcode: CPU1 sig=0x663, pf=0x1, revision=0x1
[    0.529552] microcode: CPU2 sig=0x663, pf=0x1, revision=0x1
[    0.529569] microcode: CPU3 sig=0x663, pf=0x1, revision=0x1
[    0.529616] microcode: Microcode Update Driver: v2.00 <tigran@aivazian.fsnet.co.uk>,
Peter Oruba
[    0.529912] futex hash table entries: 1024 (order: 4, 65536 bytes)
[    0.529933] Initialise system trusted keyring
[    0.529988] audit: initializing netlink socket (disabled)
[    0.529999] type=2000 audit(1455811046.013:1): initialized
[    0.555547] HugeTLB registered 2 MB page size, pre-allocated 0 pages
```

```
[    0.556760] zbud: loaded
[    0.556971] VFS: Disk quotas dquot_6.5.2
[    0.557018] Dquot-cache hash table entries: 512 (order 0, 4096 bytes)
[    0.557237] msgmni has been set to 7357
[    0.557294] Key type big_key registered
[    0.557298] SELinux:  Registering netfilter hooks
[    0.558946] NET: Registered protocol family 38
[    0.558956] Key type asymmetric registered
[    0.558959] Asymmetric key parser 'x509' registered
[    0.558994] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 252)
[    0.559078] io scheduler noop registered
[    0.559083] io scheduler deadline registered (default)
[    0.559115] io scheduler cfq registered
[    0.559204] pci_hotplug: PCI Hot Plug PCI Core version: 0.5
[    0.559218] pciehp: PCI Express Hot Plug Controller Driver version: 0.4
[    0.559277] intel_idle: does not run on family 6 model 6
[    0.559346] input: Power Button as /devices/LNXSYSTM:00/LNXPWRBN:00/input/input0
[    0.559350] ACPI: Power Button [PWRF]
[    0.559565] GHES: HEST is not enabled!
[    0.559652] Serial: 8250/16550 driver, 8 ports, IRQ sharing enabled
[    0.584666] 00:04: ttyS0 at I/O 0x3f8 (irq = 4) is a 16550A
[    0.585463] Non-volatile memory driver v1.3
[    0.585469] Linux agpgart interface v0.103
[    0.585612] crash memory driver: version 1.1
[    0.585645] rdac: device handler registered
[    0.585699] hp_sw: device handler registered
[    0.585701] emc: device handler registered
[    0.585704] alua: device handler registered
[    0.585736] libphy: Fixed MDIO Bus: probed
[    0.585798] ehci_hcd: USB 2.0 'Enhanced' Host Controller (EHCI) Driver
[    0.585803] ehci-pci: EHCI PCI platform driver
[    0.585813] ohci_hcd: USB 1.1 'Open' Host Controller (OHCI) Driver
[    0.585816] ohci-pci: OHCI PCI platform driver
[    0.585824] uhci_hcd: USB Universal Host Controller Interface driver
[    0.617200] uhci_hcd 0000:00:01.2: UHCI Host Controller
[    0.617270] uhci_hcd 0000:00:01.2: new USB bus registered, assigned bus number 1
[    0.617296] uhci_hcd 0000:00:01.2: detected 2 ports
[    0.617395] uhci_hcd 0000:00:01.2: irq 11, io base 0x0000c040
[    0.617488] usb usb1: New USB device found, idVendor=1d6b, idProduct=0001
[    0.617491] usb usb1: New USB device strings: Mfr=3, Product=2, SerialNumber=1
[    0.617493] usb usb1: Product: UHCI Host Controller
[    0.617495] usb usb1: Manufacturer: Linux 3.10.0-229.20.1.22.hulk.x86_64 uhci_hcd
[    0.617496] usb usb1: SerialNumber: 0000:00:01.2
[    0.617613] hub 1-0:1.0: USB hub found
[    0.617620] hub 1-0:1.0: 2 ports detected
[    0.617762] usbcore: registered new interface driver usbserial
[    0.617768] usbcore: registered new interface driver usbserial_generic
[    0.617774] usbserial: USB Serial support registered for generic
[    0.617800] i8042: PNP: PS/2 Controller [PNP0303:KBD,PNP0f13:MOU] at 0x60,0x64 irq 1,12
[    0.618540] serio: i8042 KBD port at 0x60,0x64 irq 1
[    0.618545] serio: i8042 AUX port at 0x60,0x64 irq 12
[    0.618647] mousedev: PS/2 mouse device common for all mice
[    0.618954] input: AT Translated Set 2 keyboard as /devices/platform/i8042/serio0/input/
input1
[    0.620143] rtc_cmos 00:00: RTC can wake from S4
[    0.620527] rtc_cmos 00:00: rtc core: registered rtc_cmos as rtc0
[    0.620742] rtc_cmos 00:00: alarms up to one day, 114 bytes nvram, hpet irqs
[    0.620805] cpuidle: using governor menu
[    0.620873] hidraw: raw HID events driver (C) Jiri Kosina
[    0.620984] usbcore: registered new interface driver usbhid
[    0.620986] usbhid: USB HID core driver
[    0.621049] drop_monitor: Initializing network drop monitor service
[    0.621131] TCP: cubic registered
[    0.621136] Initializing XFRM netlink socket
[    0.621226] NET: Registered protocol family 10
[    0.621378] NET: Registered protocol family 17
[    0.621721] Loading compiled-in X.509 certificates
[    0.622345] Loaded X.509 cert 'Red Hat Enterprise Linux Driver Update Program (key 3):
bf57f3e87362bc7229d9f465321773dfd1f77a80'
```



```
[    0.622928] Loaded X.509 cert 'Red Hat Enterprise Linux kpatch signing key:
4d38fd864ebe18c5f0b72e3852e2014c3a676fc8'
[    0.623888] Loaded X.509 cert 'Magrathea: Glacier signing key:
6e93856ba079c51674540346c06d7c031d25d30d'
[    0.623906] registered taskstats version 1
[    0.626078] Key type trusted registered
[    0.627915] Key type encrypted registered
[    0.629712] IMA: No TPM chip found, activating TPM-bypass!
[    0.630129] rtc_cmos 00:00: setting system clock to 2016-02-18 15:57:25 UTC (1455811045)
[    0.630912] Freeing unused kernel memory: 1604k freed
[    0.634541] systemd[1]: systemd 208 running in system mode. (+PAM +LIBWRAP +AUDIT
+SELINUX +IMA +SYSVINIT +LIBCRYPTSETUP +GCRYPT +ACL +XZ)
[    0.634589] systemd[1]: Detected virtualization 'kvm'.
[    0.634592] systemd[1]: Running in initial RAM disk.
[    0.634651] systemd[1]: Set hostname to <localhost.localdomain>.
[    0.668185] systemd[1]: Expecting device dev-mapper-euleros\x2droot.device...
[    0.668201] systemd[1]: Starting -.slice.
[    0.668336] systemd[1]: Created slice -.slice.
[    0.668382] systemd[1]: Starting System Slice.
[    0.668458] systemd[1]: Created slice System Slice.
[    0.668500] systemd[1]: Starting Slices.
[    0.668512] systemd[1]: Reached target Slices.
[    0.668546] systemd[1]: Starting Timers.
[    0.668558] systemd[1]: Reached target Timers.
[    0.668592] systemd[1]: Starting Journal Socket.
[    0.668660] systemd[1]: Listening on Journal Socket.
[    0.668754] systemd[1]: Started dracut ask for additional cmdline parameters.
[    0.668925] systemd[1]: Starting dracut cmdline hook...
[    0.669394] systemd[1]: Started Load Kernel Modules.
[    0.669411] systemd[1]: Starting Setup Virtual Console...
[    0.669714] systemd[1]: Starting Journal Service...
[    0.670110] systemd[1]: Started Journal Service.
[    0.676193] NMI watchdog: disabled (cpu0): hardware events not enabled
[    0.805130] device-mapper: uevent: version 1.0.3
[    0.805248] device-mapper: ioctl: 4.29.0-ioctl (2014-10-28) initialised: dm-
devel@redhat.com
[    0.840316] systemd-udevd[257]: starting version 208
[    0.870632] e1000: Intel(R) PRO/1000 Network Driver - version 7.3.21-k8-NAPI
[    0.870635] e1000: Copyright (c) 1999-2006 Intel Corporation.
[    0.875066] [drm] Initialized drm 1.1.0 20060810
[    0.875462] FDC 0 is a S82078B
[    0.883437] libata version 3.00 loaded.
[    0.911357] ACPI: PCI Interrupt Link [LNKB] enabled at IRQ 10
[    0.920031] usb 1-1: new full-speed USB device number 2 using uhci_hcd
[    0.943102] ACPI: PCI Interrupt Link [LNKC] enabled at IRQ 11
[    1.267681] e1000 0000:00:03.0 eth0: (PCI:33MHz:32-bit) 90:e2:ba:20:0d:21
[    1.267689] e1000 0000:00:03.0 eth0: Intel(R) PRO/1000 Network Connection
[    1.267744] ata_piix 0000:00:01.1: version 2.13
[    1.269451] scsi host0: ata_piix
[    1.269706] scsi host1: ata_piix
[    1.269745] ata1: PATA max MWDMA2 cmd 0x1f0 ctl 0x3f6 bmdma 0xc080 irq 14
[    1.269748] ata2: PATA max MWDMA2 cmd 0x170 ctl 0x376 bmdma 0xc088 irq 15
[    1.270098] [TTM] Zone  kernel: Available graphics memory: 1884376 kiB
[    1.270100] [TTM] Initializing pool allocator
[    1.270105] [TTM] Initializing DMA pool allocator
[    1.270581] [drm] fb mappable at 0xFC000000
[    1.270583] [drm] vram aper at 0xFC000000
[    1.270584] [drm] size 4194304
[    1.270585] [drm] fb depth is 24
[    1.270586] [drm]    pitch is 3072
[    1.271204] fbcon: cirrusdrmfb (fb0) is primary device
[    1.290116] Console: switching to colour frame buffer device 128x48
[    1.303472] cirrus 0000:00:02.0: fb0: cirrusdrmfb frame buffer device
[    1.303475] cirrus 0000:00:02.0: registered panic notifier
[    1.303481] [drm] Initialized cirrus 1.0.0 20110418 for 0000:00:02.0 on minor 0
[    1.310106] systemd-udevd[259]: renamed network interface eth0 to ens3
[    1.421760] ata1.00: ATAPI: QEMU DVD-ROM, 2.1.0, max UDMA/100
[    1.421767] ata1.01: ATA-7: QEMU HARDDISK, 2.1.0, max UDMA/100
[    1.421769] ata1.01: 83886080 sectors, multi 16: LBA48
```



```
[    1.422428] ata1.00: configured for MWDMA2
[    1.422878] ata1.01: configured for MWDMA2
[    1.423499] scsi 0:0:0:0: CD-ROM            QEMU     QEMU DVD-ROM     2.1. PQ: 0 ANSI: 5
[    1.424024] scsi 0:0:1:0: Direct-Access     ATA      QEMU HARDDISK    2.1. PQ: 0 ANSI: 5
[    1.432649] sr 0:0:0:0: [sr0] scsi3-mmc drive: 4x/4x cd/rw xa/form2 tray
[    1.432653] cdrom: Uniform CD-ROM driver Revision: 3.20
[    1.432760] sr 0:0:0:0: Attached scsi CD-ROM sr0
[    1.435856] sd 0:0:1:0: [sda] 83886080 512-byte logical blocks: (42.9 GB/40.0 GiB)
[    1.446975] sd 0:0:1:0: [sda] Write Protect is off
[    1.446979] sd 0:0:1:0: [sda] Mode Sense: 00 3a 00 00
[    1.447138] sd 0:0:1:0: [sda] Write cache: enabled, read cache: enabled, doesn't support
DPO or FUA
[    1.448504]  sda: sda1 sda2
[    1.448780] sd 0:0:1:0: [sda] Attached SCSI disk
[    1.451475] input: ImExPS/2 Generic Explorer Mouse as /devices/platform/i8042/serio1/
input/input2
[    1.531051] tsc: Refined TSC clocksource calibration: 2100.005 MHz
[    4.922621] usb 1-1: New USB device found, idVendor=0627, idProduct=0001
[    4.922627] usb 1-1: New USB device strings: Mfr=1, Product=3, SerialNumber=5
[    4.922629] usb 1-1: Product: QEMU USB Tablet
[    4.922631] usb 1-1: Manufacturer: QEMU
[    4.922633] usb 1-1: SerialNumber: 42
[    5.823527] input: QEMU QEMU USB Tablet as /devices/pci0000:00/0000:00:01.2/
usb1/1-1/1-1:1.0/input/input3
[    5.823757] hid-generic 0003:0627:0001.0001: input,hidraw0: USB HID v0.01 Pointer [QEMU
QEMU USB Tablet] on usb-0000:00:01.2-1/input0
[    6. 86844] SGI XFS with ACLs, security attributes, large block/inode numbers, no debug
enabled
[    6. 88977] XFS (dm-1): Mounting V4 Filesystem
[    6.155803] XFS (dm-1): Starting recovery (logdev: internal)
[    6.187844] XFS (dm-1): Ending recovery (logdev: internal)
[    6.422561] systemd-journald[106]: Received SIGTERM
[    6.812415] type=1404 audit(1455811051.682:2): enforcing=1 old_enforcing=0
auid=4294967295 ses=4294967295
[    7. 24071] SELinux: 2048 avtab hash slots, 111514 rules.
[    7. 44938] SELinux: 2048 avtab hash slots, 111514 rules.
[    7. 79940] SELinux:  8 users, 102 roles, 4977 types, 295 bools, 1 sens, 1024 cats
[    7. 79944] SELinux:  83 classes, 111514 rules
[    7. 86403] SELinux:  Completing initialization.
[    7. 86406] SELinux:  Setting up existing superblocks.
[    7. 86412] SELinux: initialized (dev sysfs, type sysfs), uses genfs_contexts
[    7. 86417] SELinux: initialized (dev rootfs, type rootfs), uses genfs_contexts
[    7. 86426] SELinux: initialized (dev bdev, type bdev), uses genfs_contexts
[    7. 86431] SELinux: initialized (dev proc, type proc), uses genfs_contexts
[    7. 86480] SELinux: initialized (dev tmpfs, type tmpfs), uses transition SIDs
[    7. 86496] SELinux: initialized (dev devtmpfs, type devtmpfs), uses transition SIDs
[    7. 86976] SELinux: initialized (dev sockfs, type sockfs), uses task SIDs
[    7. 86980] SELinux: initialized (dev debugfs, type debugfs), uses genfs_contexts
[    7. 87872] SELinux: initialized (dev pipefs, type pipefs), uses task SIDs
[    7. 87877] SELinux: initialized (dev anon_inodefs, type anon_inodefs), uses
genfs_contexts
[    7. 87880] SELinux: initialized (dev aio, type aio), not configured for labeling
[    7. 87883] SELinux: initialized (dev devpts, type devpts), uses transition SIDs
[    7. 87895] SELinux: initialized (dev hugetlbfs, type hugetlbfs), uses transition SIDs
[    7. 87900] SELinux: initialized (dev mqueue, type mqueue), uses transition SIDs
[    7. 87905] SELinux: initialized (dev selinuxfs, type selinuxfs), uses genfs_contexts
[    7. 87915] SELinux: initialized (dev securityfs, type securityfs), uses genfs_contexts
[    7. 87919] SELinux: initialized (dev sysfs, type sysfs), uses genfs_contexts
[    7. 88245] SELinux: initialized (dev tmpfs, type tmpfs), uses transition SIDs
[    7. 88252] SELinux: initialized (dev tmpfs, type tmpfs), uses transition SIDs
[    7. 88317] SELinux: initialized (dev tmpfs, type tmpfs), uses transition SIDs
[    7. 88342] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88351] SELinux: initialized (dev pstore, type pstore), uses genfs_contexts
[    7. 88354] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88359] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88364] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88372] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88376] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88380] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
```



```
[    7. 88384] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88391] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88395] SELinux: initialized (dev cgroup, type cgroup), uses genfs_contexts
[    7. 88400] SELinux: initialized (dev configfs, type configfs), uses genfs_contexts
[    7. 88402] SELinux: initialized (dev drm, type drm), not configured for labeling
[    7. 88409] SELinux: initialized (dev dm-1, type xfs), uses xattr
[    7. 98755] type=1403 audit(1455811051.967:3): policy loaded auid=4294967295
ses=4294967295
[    7.110951] systemd[1]: Successfully loaded SELinux policy in 317.874ms.
[    7.212349] systemd[1]: Relabelled /dev and /run in 23.634ms.
[    7.862304] SELinux: initialized (dev autofs, type autofs), uses genfs_contexts
[    8.131675] SELinux: initialized (dev hugetlbfs, type hugetlbfs), uses transition SIDs
[    8.382401] systemd-udevd[503]: starting version 208
[    8.742381] input: PC Speaker as /devices/platform/pcspkr/input/input4
[    8.877826] ppdev: user-space parallel port driver
[    8.961245] piix4_smbus 0000:00:01.3: SMBus Host Controller at 0x700, revision 0
[    9.261314] XFS (sda1): Mounting V4 Filesystem
[    9.518676] Adding 4063228k swap on /dev/mapper/euleros-swap.  Priority:-1 extents:1
across:4063228k FS
[    9.602697] snd_hda_intel 0000:00:04.0: irq 40 for MSI
[    9.636085] sound hdaudioC0D0: autoconfig: line_outs=1 (0x3/0x0/0x0/0x0/0x0) type:line
[    9.636090] sound hdaudioC0D0:    speaker_outs=0 (0x0/0x0/0x0/0x0/0x0)
[    9.636092] sound hdaudioC0D0:    hp_outs=0 (0x0/0x0/0x0/0x0/0x0)
[    9.636094] sound hdaudioC0D0:    mono: mono_out=0x0
[    9.636096] sound hdaudioC0D0:    inputs:
[    9.636098] sound hdaudioC0D0:      Line=0x5
[    9.747980] XFS (sda1): Ending clean mount
[    9.747993] SELinux: initialized (dev sda1, type xfs), uses xattr
[    9.755453] systemd-journald[482]: Received request to flush runtime journal from PID 1
[   10. 37592] type=1305 audit(1455811054.906:4): audit_pid=585 old=0 auid=4294967295
ses=4294967295 subj=system_u:system_r:auditd_t:s0 res=1
[   11.482452] kbox: module verification failed: signature and/or  required key missing -
tainting kernel
[   11.485337] num of online cpus: 4
[   11.485340] KBOX: the config 0x59 is ok.
[   11.485343] KBOX: the config info is:
Product          : euler
Soft version     : V200R002C10
Frame No         : 1
Slot No          : 0
Location No      : 0
Hardware version : Unknown

[   11.604279] console [kbox_console0] enabled
[   11.649070] KBOX: load OK
[   11.858046] ip_tables: (C) 2000-2006 Netfilter Core Team
[   11.899038] kbox_status=0 Changing to=0(maintain(0),working(1))
[   11.918847] hmem_driver: module verification failed: signature and/or  required key
missing - tainting kernel
[   11.930164] reserve_kbox_mem_start sym address: 0xffffffff81beab50, physical start
address: 0x13f000000
[   11.938631] reserve_kbox_mem_len sym address: 0xffffffff81beab48, physical size: 0x1000000
[   11.938634] hmem_init parameters check ok. start_paddr: 0x13f000000, mem_size: 0x1000000
[   11.939410] the manage area checksum.
[   11.939927] no init or error!! magic number=0x0, flag=0x0.
[   11.939929] init the manage area.
[   11.939930] init the manage area finish!.
[   11.944070] kbox:format dev hmem
[   11.944074] file_supports=20
[   11.998256] nf_conntrack version 0.5.0 (16384 buckets, 65536 max)
[   12. 95253] ip6_tables: (C) 2000-2006 Netfilter Core Team
[   12.253913] Ebtables v2.0 registered
[   12.278419] Bridge firewalling registered
[   12.852228] IPv6: ADDRCONF(NETDEV_UP): ens3: link is not ready
[   12.856737] IPv6: ADDRCONF(NETDEV_UP): ens3: link is not ready
[   12.860846] IPv6: ADDRCONF(NETDEV_UP): ens3: link is not ready
[   13.796136] IPv6: ADDRCONF(NETDEV_UP): ens3: link is not ready
[   14.855520] e1000: ens3 NIC Link is Up 1000 Mbps Full Duplex, Flow Control: RX
[   14.856049] IPv6: ADDRCONF(NETDEV_CHANGE): ens3: link becomes ready
```



```
[   15. 14908] kbox_status=0 Changing to=1(maintain(0),working(1))
[   15. 14913] dev_cb:ffff8800368b8680
[   15. 14915] dev_cb: hmem
[   57. 47985] SysRq : Manual OOM execution
```

3. 控制台打印日志

```
---------------area type:8, record num:1, unit_size:8 ------------


###num:0, record len:61
oom time:20160218155821-e0387
[ 2455.550908] Mem-Info:
[ 2455.553173] Node 0 DMA per-cpu:
[ 2455.556302] CPU    0: hi:    0, btch:   1 usd:   0
[ 2455.561064] CPU    1: hi:    0, btch:   1 usd:   0
[ 2455.565825] CPU    2: hi:    0, btch:   1 usd:   0
[ 2455.570586] CPU    3: hi:    0, btch:   1 usd:   0
[ 2455.575347] Node 0 DMA32 per-cpu:
[ 2455.578651] CPU    0: hi:  186, btch:  31 usd:   0
[ 2455.583412] CPU    1: hi:  186, btch:  31 usd:   0
[ 2455.588173] CPU    2: hi:  186, btch:  31 usd:   0
[ 2455.592934] CPU    3: hi:  186, btch:  31 usd:   0
[ 2455.597698] active_anon:315623 inactive_anon:23 isolated_anon:0
[ 2455.597699]  active_file:44 inactive_file:40 isolated_file:0
[ 2455.597700]  unevictable:63860 dirty:22 writeback:0 unstable:0
[ 2455.597701]  free:490 slab_reclaimable:3727 slab_unreclaimable:7388
[ 2455.597703]  mapped:1188 shmem:27 pagetables:976 bounce:0
[ 2455.626584] Node 0 DMA free:24kB min:16kB low:20kB high:24kB active_anon:15852kB
inactive_anon:0kB active_file:0kB inactive_file:0kB unevictable:0kB isolated(anon):0kB
isolated(file):0kB present:15676kB mlocked:0kB dirty:0kB writeback:0kB mapped:0kB shmem:0kB
slab_reclaimable:0kB slab_unreclaimable:0kB kernel_stack:0kB pagetables:0kB unstable:0kB
bounce:0kB writeback_tmp:0kB pages_scanned:0 all_unreclaimable? yes
[ 2455.662953] lowmem_reserve[]: 0 2 3072 128
[ 2455.667074] Node 0 DMA32 free:1936kB min:2028kB low:2532kB high:3040kB active_anon:
1246640kB inactive_anon:92kB active_file:176kB inactive_file:0kB unevictable:255440kB
isolated(anon):0kB isolated(file):0kB present:1764800kB mlocked:0kB dirty:88kB writeback:0kB
mapped:4752kB shmem:108kB slab_reclaimable:14908kB slab_unreclaimable:29552kB kernel_stack:
792kB pagetables:3904kB unstable:0kB bounce:0kB writeback_tmp:0kB pages_scanned:4640
all_unreclaimable? yes
[ 2455.707242] lowmem_reserve[]: 0 0 256 256
[ 2455.711283] Node 0 DMA: 1*4kB 0*8kB 0*16kB 0*32kB 0*64kB 0*128kB 0*256kB 0*512kB 0*1024kB
0*2048kB 0*4096kB = 4kB
[ 2455.721622] Node 0 DMA32: 226*4kB 67*8kB 8*16kB 3*32kB 4*64kB 0*128kB 1*256kB 0*512kB
0*1024kB 0*2048kB 0*4096kB = 2176kB
[ 2455.732650] 64028 total pagecache pages
[ 2455.736460] 0 pages in swap cache
[ 2455.739752] Swap cache stats: add 0, delete 0, find 0/0
[ 2455.744944] Free swap  = 0kB
[ 2455.747804] Total swap = 0kB
[ 2455.755068] 487408 pages RAM
[ 2455.757938] 78981 pages reserved
[ 2455.761143] 7202 pages shared
[ 2455.764089] 397079 pages non-shared
[ 2455.767672] calling kbox_sync :begin
[ 2455.771223] sync kbox :begin
[ 2455.774085] open all redirect device :begin
[ 2455.778242] open all redirect device :end
[ 2455.782224] flush kbox regions :begin
[ 2455.785864] kbox region (oom) is writing into (hmem), action is 202
[ 2455.792525] test write len : 21748
[ 2455.795902] first start addr : ffff88007e686000
[ 2455.800403] second start addr : ffff88007e6a6000
[ 2455.804990] cur_index : 1, offset : 131072, third start addr : ffff88007e6a6008
[ 2455.812252] first length : 21748
[ 2455.815508] bios_write_file_data write data len *ptr_length : 21748
[ 2455.821735] cur_index : 2, record_number : 2, total_number : 4
[ 2455.827538] kbox region (oom) has been written into (hmem)
[ 2455.833420] dev hmem is dirty
oom time:20160218155821-e0387
```


9.2.4 提供系统 die/oops 信息

本节主要介绍内核发生oops时内核黑匣子所记录的异常信息。

功能概述

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访问空指针、非法访问内存空间等，触发oops事件。Kbox能够将oops事件发生的时间、发生oops进程信息、异常进程调用栈等记录到存储设备中，方便维护人员定位。

信息清单

记录的异常die/oops信息包含三个部分内容。

1. oops异常信息，最多记录信息容量为128K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 当前进程的pid、进程名称。
   - 异常进程的调用轨迹及栈信息（栈信息最多打印150行）。
   - 打印系统中模块名称起始地址和长度（最多打印384条模块信息）。

     说明

     若系统中存在大量模块，且各模块的名称很长时，存在冲日志的风险。
2. 内核消息打印日志，最多记录信息容量为64K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 异常发生时最近64K日志信息，即内核打印到循环缓冲区中的最后64K日志信息。
3. 控制台打印日志，最多记录信息容量为32K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 其他内核打印日志消息。

     说明

     如果当前其他进程没有向控制台打印日志，收集的控制台打印日志为空。

信息举例

在日志信息文件中，包含如下内容：

1. oops异常信息

```
*****area type:die - location in die area:0*****
//错误类型及错误号
die info:Oops:0002
---------KBOX_START----------
//die事件发生的时间
die time:20131109041507-e9ea9
//进程的所在CPU号、进程名称
CPU 0 Pid: 7664, comm: bash
//寄存器信息
RIP: 0010:[<ffffffffa0adbcdf>]  [<ffffffffa0adbcdf>] dev_wr_handler+0xa3f/0xce0 [kpgen]
RSP: 0018:ffff88005ea73eb8  EFLAGS: 00010292
RAX: 0000000000000045 RBX: 0000000000000000 RCX: 0000000000006161
RDX: 0000000000000062 RSI: 0000000000000096 RDI: 0000000000000246
RBP: ffff88005ea73f08 R08: ffff88007b001000 R09: 0000000000000000
R10: 0000000000000000 R11: 0000000000000000 R12: ffffffff810f10f0
R13: 0000000000000005 R14: 00007f8bcf81a000 R15: 0000000000000000
FS:  00007f8bcf97f700(0000) GS:ffff880069600000(0000) knlGS:0000000000000000
```

```
CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
CR2: 0000000000000000 CR3: 0000000035c43000 CR4: 00000000001407f0
DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
DR3: 0000000000000000 DR6: 00000000ffff0ff0 DR7: 0000000000000400
Process bash (pid: 7664, threadinfo ffff88005ea72000, task ffff8800373ba6c0)
Stack:
 0000000a32303031 0000000000000000 0000000000000000 0000000000000000
 ffff88005ea73f08 ffffffff81146a08 0000000000000002 0000000000000005
 ffff88004f3b8e80 ffff88005ea73f48 ffff88005ea73f38 ffffffff81146e9b
//调用栈
Call Trace:
 [<ffffffffa0003e43>] kbox_show_registers+0x4c3/0xa30 [kbox]
 [<ffffffffa000e91b>] ? kbox_buffer_write+0xcb/0x110 [kbox]
 [<ffffffffa001012c>] kbox_die_callback+0x15c/0x220 [kbox]
 [<ffffffff81445e0f>] notifier_call_chain+0x3f/0x80
 [<ffffffff81445e5d>] __atomic_notifier_call_chain+0xd/0x10
 [<ffffffff81445e71>] atomic_notifier_call_chain+0x11/0x20
 [<ffffffff81445eae>] notify_die+0x2e/0x30
 [<ffffffff814430c8>] __die+0x88/0x100
 [<ffffffff8102e883>] no_context+0xf3/0x200
 [<ffffffff8102eabd>] __bad_area_nosemaphore+0x12d/0x220
 [<ffffffff8102ec09>] bad_area+0x49/0x60
 [<ffffffff81445d6c>] do_page_fault+0x44c/0x4b0
 [<ffffffff8103aef6>] ? console_unlock+0x246/0x2a0
 [<ffffffff810f10f0>] ? oom_kill_process+0x2a0/0x2a0
 [<ffffffff81442675>] page_fault+0x25/0x30
 [<ffffffff810f10f0>] ? oom_kill_process+0x2a0/0x2a0
 [<ffffffffa0adbcdf>] ? dev_wr_handler+0xa3f/0xce0 [kpgen]
 [<ffffffffa0adbcdf>] ? dev_wr_handler+0xa3f/0xce0 [kpgen]
 [<ffffffff81146a08>] ? rw_verify_area+0x58/0x100
 [<ffffffff81146e9b>] vfs_write+0xcb/0x130
 [<ffffffff81146ff0>] sys_write+0x50/0x90
 [<ffffffff8144a079>] system_call_fastpath+0x16/0x1b
Code: 00 00 0f 85 12 f7 ff ff 48 c7 c7 08 d5 ad a0 31 c0 e8 58 2e 96 e0 cd 04 e9 0b f7 ff ff
48 c7 c7 98 d4 ad a0 31 c0 e8 43 2e 96 e0 <c6> 04 25 00 00 00 00 00 e9 f0 f6 ff ff 48 c7 c7
d0 d4 ad a0 31
mod_name                mod_start             core_size
os_test                 0xffffffffa0015000    0x3df6
bioshmem_driver        0xffffffffa000c000    0x454d
kbox                    0xffffffffa0b81000    0x49a53
agetty_query            0xffffffffa0b7c000    0x3442
cpufreq_powersave       0xffffffffa0b77000    0x314b
signo_catch             0xffffffffa0b6d000    0x30fa
sysalarm_agent_netlink_k0xffffffffa0b72000    0x316d
af_packet               0xffffffffa0b63000    0x8a5e
......
//die发生后的动作（0：不做任何操作；1：调用panic；2：reboot）
action after die is:0
---------KBOX_END------------
```

2. 内核消息打印日志

```
message:
*****area type:message - location in message area:4*****
die time:20131109041507-e9ea9
>[  246.509001] INFO: task sched_work:253 blocked for more than 120 seconds.
<3>[  246.509005] "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message.
<6>[  246.509009] sched_work      D 0000000000000000  5544   253      2 0x00000000
<4>[  246.509018]  ffff88005eb0bdd0 0000000000000046 ffffffff81075026 ffff88005eb0a010
<4>[  246.509025]  00000000000159c0 00000000000159c0 00000000000159c0 00000000000159c0
<4>[  246.509032]  ffff88005eb0bfd8 ffff88005eb0bfd8 00000000000159c0 00000000000159c0
<4>[  246.509038] Call Trace:
<4>[  246.509048]  [<ffffffff81075026>] ? update_curr+0x186/0x1c0
<4>[  246.509055]  [<ffffffff8107607a>] ? dequeue_task_fair+0x6a/0x170
<4>[  246.509060]  [<ffffffff8106ba29>] ? dequeue_task+0x89/0xa0
<4>[  246.509082]  [<ffffffffa00bf7fe>] ? DBG_Log+0x3e/0x340 [vos]
<4>[  246.509090]  [<ffffffff81440499>] schedule+0x29/0x90
<4>[  246.509095]  [<ffffffff8143edad>] schedule_timeout+0x21d/0x2c0
<4>[  246.509102]  [<ffffffff810c1392>] ? call_rcu_sched+0x12/0x20
<4>[  246.509107]  [<ffffffff8106261a>] ? __put_cred+0x3a/0x50
<4>[  246.509111]  [<ffffffff81062d2e>] ? commit_creds+0x12e/0x1e0
```

```
<4>[  246.509117]  [<ffffffff8143f54d>] __down+0x6d/0xb0
<4>[  246.509125]  [<ffffffff81061aa7>] down+0x47/0x50
<4>[  246.509142]  [<ffffffffa00c6009>] LVOS_sema_down+0x9/0x10 [vos]
<4>[  246.509157]  [<ffffffffa00c64cc>] LVOS_SchedWorkThread+0x5c/0x190 [vos]
<4>[  246.509165]  [<ffffffff8144b3d4>] kernel_thread_helper+0x4/0x10
<4>[  246.509180]  [<ffffffffa00c6470>] ? LVOS_SchedWorkInit+0x60/0x60 [vos]
<4>[  246.509186]  [<ffffffff8144b3d0>] ? gs_change+0x13/0x13
<3>[  246.509190] INFO: task sched_work:254 blocked for more than 120 seconds.
<3>[  246.509192] "echo 0 > /proc/sys/kernel/hung_task_timeout_secs" disables this message.
<6>[  246.509196] sched_work      D 0000000000000000  5544   254      2 0x00000000
<4>[  246.509203]  ffff88005eb07dd0 0000000000000046 ffffffff81075026 ffff88005eb06010
<4>[  246.509209]  00000000000159c0 00000000000159c0 00000000000159c0 00000000000159c0
<4>[  246.509214]  ffff88005eb07fd8 ffff88005eb07fd8 00000000000159c0 00000000000159c0
<4>[  246.509220] Call Trace:
<4>[  246.509225]  [<ffffffff81075026>] ? update_curr+0x186/0x1c0
<4>[  246.509231]  [<ffffffff8107607a>] ? dequeue_task_fair+0x6a/0x170
<4>[  246.509236]  [<ffffffff8106ba29>] ? dequeue_task+0x89/0xa0
<4>[  246.509249]  [<ffffffffa00bf7fe>] ? DBG_Log+0x3e/0x340 [vos]
<4>[  246.509256]  [<ffffffff81440499>] schedule+0x29/0x90
<4>[  246.509260]  [<ffffffff8143edad>] schedule_timeout+0x21d/0x2c0
<4>[  246.509265]  [<ffffffff810c1392>] ? call_rcu_sched+0x12/0x20
<4>[  246.509269]  [<ffffffff8106261a>] ? __put_cred+0x3a/0x50
<4>[  246.509274]  [<ffffffff81062d2e>] ? commit_creds+0x12e/0x1e0
<4>[  246.509279]  [<ffffffff8143f54d>] __down+0x6d/0xb0
<4>[  246.509286]  [<ffffffff81061aa7>] down+0x47/0x50
```

3. 控制台打印日志

```
console:
*****area type:console - location in console area:4*****
die time:20131109041507-e9ea9
die time:20131109041507-e9ea9
```

9.2.5 提供系统 rlock 信息

本节主要介绍内核发生rlock(内核软狗和硬狗狗叫)时内核黑匣子所记录的异常信息。

功能概述

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长时间关中断、长时间关抢占、大量中断等，使得内核软件狗得不到调度或者是不能喂硬件狗，造成内核异常。Kbox能够将事件发生的时间、各cpu当前软件狗信息、中断信息等记录到存储设备中，方便维护人员定位。

信息清单

记录的异常rlock信息包含三个部分内容。

1. rlock异常信息，每个CPU最多记录信息容量为8K。总信息容量最多256K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 异常发生的原因
   - 当前进程的pid、进程名称。
   - 异常进程的调用轨迹及栈信息及寄存器信息（栈信息最多打印150行）。
   - 打印系统中模块名称、模块起始地址及长度（最多打印384条模块信息）。

     说明

     若系统中存在大量模块，且各模块的名称很长时，存在冲日志的风险。
2. 内核消息打印日志，最多记录信息容量为64K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。

- 异常发生时最近64K日志信息，即内核打印到循环缓冲区中的最后64K日志信息。

3. 控制台打印日志，最多记录信息容量为32K。该记录区主要内容包括：
   - 内核异常发生的时间(UTC时间)。
   - 其他内核打印日志消息。

信息举例

在日志信息文件中，包含如下内容：

1. rlock异常信息

```
---------------area type:6, record num:1, unit_size:64 ------------

###num:0, record len:68907
---------KBOX_START----------
// rlock时间发生的时间
rlock time:20160217172018-3ceff
// rlock发生的原因
rlock reason:SOFT-WATCHDOG detected! on cpu 0.
// 进程所在CPU号、进程名称
CPU 0 Pid: 3651, comm: r_process/0
// 寄存器信息
RIP: 0010:[<ffffffff8160e426>]  [<ffffffff8160e426>] _raw_spin_lock_bh+0x36/0x50
RSP: 0018:ffff881010bcfe30  EFLAGS: 00000206
RAX: 0000000000000f96 RBX: ffff881010bcfdc0 RCX: 0000000000000000
RDX: 0000000000000002 RSI: 0000000000000002 RDI: ffff881010bcfe4c
RBP: ffff881010bcfe38 R08: ffffffff81d644a8 R09: 0000000000000001
R10: 0000000000032c53 R11: 0318000000000000 R12: ffff881010bcfe50
R13: ffffffff81d62f80 R14: 0000000000000046 R15: ffff88100fb12a84
FS:  0000000000000000(0000) GS:ffff88103ee00000(0000) knlGS:0000000000000000
CS:  0010 DS: 0000 ES: 0000 CR0: 0000000080050033
CR2: 00007ff1587c3cf8 CR3: 000000000190e000 CR4: 00000000001407f0
DR0: 0000000000000000 DR1: 0000000000000000 DR2: 0000000000000000
DR3: 0000000000000000 DR6: 00000000ffff0ff0 DR7: 0000000000000400
Process r_process/0 (pid: 3651, threadinfo ffff881010bcc000, task ffff88100f06e780)
Stack:
 ffff881010bcfe4c ffff881010bcfec0 ffffffffa05385b3 000400007efb8e00
 ffffffff81d644a8 ffffffff81d644a8 0000000100040800 ffffffff81d62f80
 ffffffffa0538040 ffff881010bcfe4c ffffffffffffffff 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 00000000ff213843
 0000000000000079 ffff881024b37df0 ffff881010bcff48 ffffffff810982ff
 0000000000000000 ffff8810239ba748 0000000000000079 ffff881000000000
 ffff881000000000 ffff881010bcfef8 ffff881010bcfef8 ffff882000000000
 ffff882000000000 ffff881010bcff18 ffff881010bcff18 00000000ff213843
 ffffffff81098230 0000000000000000 0000000000000000 ffff881024b37df0
 ffffffff816170d8 0000000000000000 0000000000000000 0000000000000000
 0000000000000000 ffff881024b37df0 ffffffff81098230 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 0000000000000000
 0000000000000000 0000000000000000 0000000000000000 0000000000000000
 ffffffffffffffff 0000000000000000 0000000000000010 0000000000000202
 ffff881010bcff58 0000000000000018
Call Trace:
 [<ffffffffa05385b3>] gen_R_process+0xe3/0x120 [kpgen_kbox]
 [<ffffffffa0538040>] ? dev_release_handler+0x10/0x10 [kpgen_kbox]
 [<ffffffff810982ff>] kthread+0xcf/0xe0
 [<ffffffff81098230>] ? kthread_create_on_node+0x140/0x140
 [<ffffffff816170d8>] ret_from_fork+0x58/0x90
 [<ffffffff81098230>] ? kthread_create_on_node+0x140/0x140
Code: 89 fb e8 1e 93 a6 ff b8 00 00 02 00 f0 0f c1 03 89 c2 c1 ea 10 66 39 c2 75 03 5b 5d c3
83 e2 fe 0f b7 f2 b8 00 80 00 00 0f b7 0b <66> 39 ca 74 ea f3 90 83 e8 01 75 f1 48 89 df 0f
1f 80 00 00 00
---------KBOX_END------------

mod_name                mod_start             core_size
kpgen_kbox              0xffffffffa0538000    0x9883
```

```
ip6t_rpfilter           0xffffffffa0533000    0x3102
ip6t_REJECT             0xffffffffa052e000    0x328b
ipt_REJECT              0xffffffffa0529000    0x30fd
xt_conntrack            0xffffffffa04c7000    0x31d8
hmem_driver             0xffffffffa051e000    0x46c0
kbox                    0xffffffffa04d1000    0x4b1cf
ebtable_nat             0xffffffffa04c2000    0x3207
ebtable_broute          0xffffffffa04cc000    0x31bb
bridge                  0xffffffffa04a4000    0x1c2b9
stp                     0xffffffffa049f000    0x32b0
llc                     0xffffffffa0496000    0x38d8
ebtable_filter          0xffffffffa0491000    0x321b
ebtables                0xffffffffa0484000    0x78c1
ip6table_nat            0xffffffffa047f000    0x3240
nf_conntrack_ipv6       0xffffffffa0479000    0x4932
nf_defrag_ipv6          0xffffffffa03f6000    0x875b
nf_nat_ipv6             0xffffffffa03c7000    0x3733
ip6table_mangle         0xffffffffa03c2000    0x319c
ip6table_security       0xffffffffa03bd000    0x31a6
ip6table_raw            0xffffffffa03a8000    0x318b
ip6table_filter         0xffffffffa0272000    0x320f
ip6_tables              0xffffffffa0380000    0x6991
......
-----------------the event info on catch cpu 0 starting:-------------

// 每个CPU上的看门狗进程信息及中断信息
the watchdog task info on cpu 0:
name: watchdog/0, pid:491, state:R
total runtime:8391220, total rundelay:1766671, total runtimes:114
last arrive time:440648306927, last queue:444651419348
voluntary_ctxt_switches:114, nonvoluntary_ctxt_switches:0

// 定时器信息
the hrtimer info on cpu 0:
until now, the hrtimer happened: 117, saved 116.
hrtimer state = 0x2, softexpires = 0x6def866e80
// 当前CPU中断数
the interrupts on cpu 0:
   0:        137  IR-IO-APIC-edge      timer
   1:          2  IR-IO-APIC-edge      i8042
   8:          1  IR-IO-APIC-edge      rtc0
   9:          2  IR-IO-APIC-fasteoi   acpi
  12:          5  IR-IO-APIC-edge      i8042
  16:         33  IR-IO-APIC-fasteoi   ehci_hcd:usb1
  23:       2796  IR-IO-APIC-fasteoi   ehci_hcd:usb2
  99:       1959  IR-PCI-MSI-edge      megasas
 100:        197  IR-PCI-MSI-edge      megasas
 101:        146  IR-PCI-MSI-edge      megasas
 102:        232  IR-PCI-MSI-edge      megasas
 103:        188  IR-PCI-MSI-edge      megasas
 104:        115  IR-PCI-MSI-edge      megasas
 105:        479  IR-PCI-MSI-edge      megasas
 106:        708  IR-PCI-MSI-edge      megasas
 107:        966  IR-PCI-MSI-edge      megasas
 108:        729  IR-PCI-MSI-edge      megasas
 109:       1431  IR-PCI-MSI-edge      megasas
 110:       1826  IR-PCI-MSI-edge      megasas
 111:         10  IR-PCI-MSI-edge      megasas
 112:         32  IR-PCI-MSI-edge      megasas
 113:         28  IR-PCI-MSI-edge      megasas
 114:         26  IR-PCI-MSI-edge      megasas
 115:         64  IR-PCI-MSI-edge      ahci
 116:          1  IR-PCI-MSI-edge      isci-msix
 218:          2  IR-PCI-MSI-edge      ioat-msix
 219:          2  IR-PCI-MSI-edge      ioat-msix
 220:          2  IR-PCI-MSI-edge      ioat-msix
 221:          2  IR-PCI-MSI-edge      ioat-msix
 222:          2  IR-PCI-MSI-edge      ioat-msix
 223:          2  IR-PCI-MSI-edge      ioat-msix
```



```
 224:          2  IR-PCI-MSI-edge      ioat-msix
 225:          2  IR-PCI-MSI-edge      ioat-msix
 234:          1  IR-PCI-MSI-edge      enp1s0f1-tx-0
 235:          1  IR-PCI-MSI-edge      enp1s0f1-rx-1
 236:          1  IR-PCI-MSI-edge      enp1s0f1-rx-2
 237:          1  IR-PCI-MSI-edge      enp1s0f1-rx-3
 238:          1  IR-PCI-MSI-edge      enp1s0f1-rx-4
 239:          1  IR-PCI-MSI-edge      enp1s0f3-tx-0
 240:          1  IR-PCI-MSI-edge      enp1s0f3-rx-1
 241:          1  IR-PCI-MSI-edge      enp1s0f3-rx-2
 242:          1  IR-PCI-MSI-edge      enp1s0f3-rx-3
 243:          1  IR-PCI-MSI-edge      enp1s0f3-rx-4
 244:         46  IR-PCI-MSI-edge      enp1s0f0-tx-0
 245:        339  IR-PCI-MSI-edge      enp1s0f0-rx-1
 246:         46  IR-PCI-MSI-edge      enp1s0f0-rx-2
 247:          8  IR-PCI-MSI-edge      enp1s0f0-rx-3
 248:         83  IR-PCI-MSI-edge      enp1s0f0-rx-4
 249:          1  IR-PCI-MSI-edge      enp1s0f2-tx-0
 250:          1  IR-PCI-MSI-edge      enp1s0f2-rx-1
 251:          1  IR-PCI-MSI-edge      enp1s0f2-rx-2
 252:          1  IR-PCI-MSI-edge      enp1s0f2-rx-3
 253:          1  IR-PCI-MSI-edge      enp1s0f2-rx-4
 NMI:          3   Non-maskable interrupts
 LOC:      32124   Local timer interrupts
 SPU:          0   Spurious interrupts
 PMI:          3   Performance monitoring interrupts
 IWI:         91   IRQ work interrupts
 RTR:          0   APIC ICR read retries
 PLT:          0   Platform interrupts
 RES:       3074   Rescheduling interrupts
 CAL:       1060   Function call interrupts
 TLB:          7   TLB shootdowns
 TRM:          0   Thermal event interrupts
 THR:          0   Threshold APIC interrupts
 MCE:          0   Machine check exceptions
 MCP:          3   Machine check polls
 ERR:          0
 MIS:          0

the cpu0 info about: usr nice system idle iowait irq softirq steal guest g_nice:
 61 0 2711 43862 134 0 1 0 0 0

-----------------the event info on catch cpu 0 end--------------------
---------------the info of stopped cpu 1 start---------------
<pid:0:0:swapper/1>
Call Trace:
 <NMI>  [<ffffffffa04e62f8>] kbox_rlock_stop_other_cpus_call+0xb8/0x110 [kbox]
 [<ffffffff810414cb>] smp_nmi_call_function_handler+0x4b/0x60
 [<ffffffff8160fa29>] nmi_handle.isra.0+0x69/0xb0
 [<ffffffff8160fba4>] do_nmi+0x134/0x410
 [<ffffffff8160ee90>] end_repeat_nmi+0x1a/0x1e
 [<ffffffff81340cb2>] ? intel_idle+0xd2/0x160
 [<ffffffff81340cb2>] ? intel_idle+0xd2/0x160
 [<ffffffff81340cb2>] ? intel_idle+0xd2/0x160
 <<EOE>>  [<ffffffff814ae0b0>] cpuidle_enter_state+0x40/0xc0
 [<ffffffff814ae1f5>] cpuidle_idle_call+0xc5/0x200
 [<ffffffff8101d15e>] arch_cpu_idle+0xe/0x30
 [<ffffffff810c7ed1>] cpu_startup_entry+0xf1/0x290
 [<ffffffff8104272a>] start_secondary+0x1ba/0x230
the watchdog task info on cpu 1:
name: watchdog/1, pid:492, state:S
total runtime:1525377, total rundelay:24311, total runtimes:122
last arrive time:468669173269, last queue:0
voluntary_ctxt_switches:121, nonvoluntary_ctxt_switches:1


the hrtimer info on cpu 1:
until now, the hrtimer happened: 117, saved 0.
hrtimer state = 0x1, softexpires = 0x6def772c40
```



```
the interrupts on cpu 1:
 105:         10  IR-PCI-MSI-edge      megasas
 246:         11  IR-PCI-MSI-edge      enp1s0f0-rx-2
 247:        316  IR-PCI-MSI-edge      enp1s0f0-rx-3
 NMI:          1   Non-maskable interrupts
 LOC:       2194   Local timer interrupts
 SPU:          0   Spurious interrupts
 PMI:          0   Performance monitoring interrupts
 IWI:        116   IRQ work interrupts
 RTR:          0   APIC ICR read retries
 PLT:          0   Platform interrupts
 RES:       7386   Rescheduling interrupts
 CAL:       1018   Function call interrupts
 TLB:          1   TLB shootdowns
 TRM:          0   Thermal event interrupts
 THR:          0   Threshold APIC interrupts
 MCE:          0   Machine check exceptions
 MCP:          3   Machine check polls
 ERR:          0
 MIS:          0

the cpu1 info about: usr nice system idle iowait irq softirq steal guest g_nice:
 8 0 43 46715 14 0 0 0 0 0

---------------the info of stopped cpu 1 end-----------------
```

2. 内核消息打印日志

```
message:
*****area type:message - location in message area:2*****
###num:0, record len:65568
[    0.      0] CPU0 microcode updated early to revision 0x428, date = 2014-05-29
[    0.      0] Initializing cgroup subsys cpuset
[    0.      0] Initializing cgroup subsys cpu
[    0.      0] Initializing cgroup subsys cpuacct
[    0.      0] Linux version 3.10.0-229.20.1.41.x86_64 (abuild@HGH1000008206) (gcc version
4.8.3 20140911 (EulerOS 4.8.3-10) (GCC) ) #1 SMP Fri Feb 5 01:17:19 UTC 2016
[    0.      0] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.41.x86_64 root=/dev/mapper/
euleros-root ro reserve_kbox_mem=16M crash_kexec_post_notifiers panic=3 rd.lvm.lv=euleros/
swap crashkernel=auto rd.lvm.lv=euleros/root
[    0.      0] e820: BIOS-provided physical RAM map:
[    0.      0] BIOS-e820: [mem 0x0000000000000000-0x000000000009cfff] usable
[    0.      0] BIOS-e820: [mem 0x000000000009d000-0x000000000009ffff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000000e0000-0x00000000000fffff] reserved
[    0.      0] BIOS-e820: [mem 0x0000000000100000-0x000000003fffffff] usable
[    0.      0] BIOS-e820: [mem 0x0000000040000000-0x00000000400fffff] reserved
[    0.      0] BIOS-e820: [mem 0x0000000040100000-0x000000007e4fefff] usable
[    0.      0] BIOS-e820: [mem 0x000000007e4ff000-0x000000007e8fefff] reserved
[    0.      0] BIOS-e820: [mem 0x000000007e8ff000-0x000000007eefefff] ACPI NVS
[    0.      0] BIOS-e820: [mem 0x000000007eeff000-0x000000007effefff] ACPI data
[    0.      0] BIOS-e820: [mem 0x000000007efff000-0x000000007effffff] usable
[    0.      0] BIOS-e820: [mem 0x000000007f000000-0x000000008fffffff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000feb00000-0x00000000feb03fff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000fed18000-0x00000000fed18fff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000fed1c000-0x00000000fed8ffff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[    0.      0] BIOS-e820: [mem 0x00000000ffc00000-0x00000000ffffffff] reserved
[    0.      0] BIOS-e820: [mem 0x0000000100000000-0x000000207fffffff] usable
[    0.      0] e820: reserve kbox memory size: 16M
[    0.      0] e820: reserve kbox memory success. [mem 0x000000207f000000-0x000000207fffffff]
[    0.      0] NX (Execute Disable) protection: active
[    0.      0] e820: user-defined physical RAM map:
[    0.      0] user: [mem 0x0000000000000000-0x000000000009cfff] usable
[    0.      0] user: [mem 0x000000000009d000-0x000000000009ffff] reserved
[    0.      0] user: [mem 0x00000000000e0000-0x00000000000fffff] reserved
[    0.      0] user: [mem 0x0000000000100000-0x000000003fffffff] usable
[    0.      0] user: [mem 0x0000000040000000-0x00000000400fffff] reserved
[    0.      0] user: [mem 0x0000000040100000-0x000000007e4fefff] usable
[    0.      0] user: [mem 0x000000007e4ff000-0x000000007e8fefff] reserved
```

```
[    0.      0] user: [mem 0x000000007e8ff000-0x000000007eefefff] ACPI NVS
[    0.      0] user: [mem 0x000000007eeff000-0x000000007effefff] ACPI data
[    0.      0] user: [mem 0x000000007efff000-0x000000007effffff] usable
[    0.      0] user: [mem 0x000000007f000000-0x000000008fffffff] reserved
[    0.      0] user: [mem 0x00000000feb00000-0x00000000feb03fff] reserved
[    0.      0] user: [mem 0x00000000fec00000-0x00000000fec00fff] reserved
[    0.      0] user: [mem 0x00000000fed18000-0x00000000fed18fff] reserved
[    0.      0] user: [mem 0x00000000fed1c000-0x00000000fed8ffff] reserved
[    0.      0] user: [mem 0x00000000fee00000-0x00000000fee00fff] reserved
[    0.      0] user: [mem 0x00000000ffc00000-0x00000000ffffffff] reserved
[    0.      0] user: [mem 0x0000000100000000-0x000000207effffff] usable
[    0.      0] user: [mem 0x000000207f000000-0x000000207fffffff] reserved
[    0.      0] SMBIOS 2.7 present.
[    0.      0] DMI: Huawei Technologies Co., Ltd. RH2288H V2-8S/BC11SRSG1, BIOS RMIBV396
10/29/2014
[    0.      0] e820: update [mem 0x00000000-0x00000fff] usable ==> reserved
[    0.      0] e820: remove [mem 0x000a0000-0x000fffff] usable
[    0.      0] No AGP bridge found
[    0.      0] e820: last_pfn = 0x207f000 max_arch_pfn = 0x400000000
[    0.      0] MTRR default type: write-back
[    0.      0] MTRR fixed ranges enabled:
[    0.      0]   00000-9FFFF write-back
[    0.      0]   A0000-BFFFF uncachable
[    0.      0]   C0000-EFFFF write-protect
[    0.      0]   F0000-FFFFF write-combining
[    0.      0] MTRR variable ranges enabled:
[    0.      0]   0 base 000080000000 mask 3FFFC0000000 uncachable
[    0.      0]   1 base 0000C0000000 mask 3FFFE0000000 uncachable
[    0.      0]   2 base 0000E0000000 mask 3FFFF0000000 uncachable
[    0.      0]   3 base 0000F0000000 mask 3FFFF8000000 uncachable
[    0.      0]   4 base 0000F8000000 mask 3FFFFC000000 uncachable
[    0.      0]   5 base 0000FC000000 mask 3FFFFE000000 uncachable
[    0.      0]   6 base 0000FEC00000 mask 3FFFFFC00000 uncachable
[    0.      0]   7 base 0000FF000000 mask 3FFFFF000000 write-protect
[    0.      0]   8 disabled
[    0.      0]   9 disabled
[    0.      0] x86 PAT enabled: cpu 0, old 0x7040600070406, new 0x7010600070106
[    0.      0] x2apic enabled by BIOS, switching to x2apic ops
[    0.      0] e820: last_pfn = 0x7f000 max_arch_pfn = 0x400000000
[    0.      0] found SMP MP-table at [mem 0x000fe1b0-0x000fe1bf] mapped at [ffff8800000fe1b0]
[    0.      0] Base memory trampoline at [ffff880000097000] 97000 size 24576
[    0.      0] Using GB pages for direct mapping
[    0.      0] init_memory_mapping: [mem 0x00000000-0x000fffff]
[    0.      0]  [mem 0x00000000-0x000fffff] page 4k
[    0.      0] BRK [0x01eea000, 0x01eeafff] PGTABLE
[    0.      0] BRK [0x01eeb000, 0x01eebfff] PGTABLE
[    0.      0] BRK [0x01eec000, 0x01eecfff] PGTABLE
[    0.      0] init_memory_mapping: [mem 0x207ee00000-0x207effffff]
[    0.      0]  [mem 0x207ee00000-0x207effffff] page 2M
[    0.      0] BRK [0x01eed000, 0x01eedfff] PGTABLE
[    0.      0] init_memory_mapping: [mem 0x207c000000-0x207edfffff]
[    0.      0]  [mem 0x207c000000-0x207edfffff] page 2M
[    0.      0] init_memory_mapping: [mem 0x2000000000-0x207bffffff]
[    0.      0]  [mem 0x2000000000-0x203fffffff] page 1G
[    0.      0]  [mem 0x2040000000-0x207bffffff] page 2M
[    0.      0] init_memory_mapping: [mem 0x1000000000-0x1fffffffff]
[    0.      0]  [mem 0x1000000000-0x1fffffffff] page 1G
[    0.      0] init_memory_mapping: [mem 0x00100000-0x3fffffff]
[    0.      0]  [mem 0x00100000-0x001fffff] page 4k
[    0.      0]  [mem 0x00200000-0x3fffffff] page 2M
[    0.      0] init_memory_mapping: [mem 0x40100000-0x7e4fefff]
[    0.      0]  [mem 0x40100000-0x401fffff] page 4k
[    0.      0]  [mem 0x40200000-0x7e3fffff] page 2M
[    0.      0]  [mem 0x7e400000-0x7e4fefff] page 4k
[    0.      0] BRK [0x01eee000, 0x01eeefff] PGTABLE
[    0.      0] BRK [0x01eef000, 0x01eeffff] PGTABLE
[    0.      0] init_memory_mapping: [mem 0x7efff000-0x7effffff]
[    0.      0]  [mem 0x7efff000-0x7effffff] page 4k
[    0.      0] init_memory_mapping: [mem 0x100000000-0xfffffffff]
```



```
[    0.      0] [mem 0x100000000-0xfffffffff] page 1G
[    0.      0] RAMDISK: [mem 0x35ef7000-0x36f73fff]
[    0.      0] ACPI: RSDP 00000000000fe020 00024 (v02 INSYDE)
[    0.      0] ACPI: XSDT 000000007effe170 000BC (v01 INSYDE Romley   00000001      01000013)
[    0.      0] ACPI: FACP 000000007effb000 000F4 (v04 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: DSDT 000000007efe9000 0DEE2 (v01 INSYDE Romley   00000000 ACPI 00040000)
[    0.      0] ACPI: FACS 000000007edcb000 00040
[    0.      0] ACPI: SPMI 000000007effd000 00040 (v05 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: BOOT 000000007effc000 00028 (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: APIC 000000007eff9000 00BB6 (v03 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: MCFG 000000007eff8000 0003C (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: SLIC 000000007eff7000 00176 (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: SRAT 000000007efe7000 01340 (v02 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: SLIT 000000007efe6000 0003C (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: BDAT 000000007efe5000 00030 (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: PRAD 000000007efe4000 000BC (v02 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: MSDM 000000007efe3000 00055 (v03 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: SSDT 000000007ef0f000 D3050 (v02 INSYDE Romley   00004000 ACPI 00040000)
[    0.      0] ACPI: HPET 000000007effa000 00038 (v01 INSYDE Romley   00000001 ACPI 00040000)
[    0.      0] ACPI: WDAT 000000007ef0e000 00194 (v01 INSYDE INSYDE   00000001 MSFT 01000013)
[    0.      0] ACPI: DMAR 000000007ef0d000 00120 (v01 INSYDE Romley   00000001 ACPI 00000001)
[    0.      0] ACPI: HEST 000000007ef0c000 000A8 (v01 INSYDE Romley   00000001    ? 00000001)
[    0.      0] ACPI: ERST 000000007ef0b000 00230 (v01 INSYDE Romley   00000001    ? 00000001)
[    0.      0] ACPI: BERT 000000007ef0a000 00030 (v01 INSYDE Romley   00000001    ? 00000001)
[    0.      0] ACPI: EINJ 000000007ef09000 00130 (v01 INSYDE Romley   00000001    ? 00000001)
[    0.      0] ACPI: Local APIC address 0xfee00000
[    0.      0] Setting APIC routing to cluster x2apic.
[    0.      0] SRAT: PXM 0 -> APIC 0x00 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x01 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x02 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x03 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x04 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x05 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x06 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x07 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x08 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x09 -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x0a -> Node 0
[    0.      0] SRAT: PXM 0 -> APIC 0x0b -> Node 0
[    0.      0] SRAT: PXM 1 -> APIC 0x20 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x21 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x22 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x23 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x24 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x25 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x26 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x27 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x28 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x29 -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x2a -> Node 1
[    0.      0] SRAT: PXM 1 -> APIC 0x2b -> Node 1
[    0.      0] SRAT: Node 0 PXM 0 [mem 0x00000000-0x107fffffff]
[    0.      0] SRAT: Node 1 PXM 1 [mem 0x1080000000-0x207fffffff]
[    0.      0] ACPI: SLIT table looks invalid. Not used.
[    0.      0] Initmem setup node 0 [mem 0x00000000-0x107fffffff]
[    0.      0]   NODE_DATA [mem 0x107ffd9000-0x107fffffff]
[    0.      0] Initmem setup node 1 [mem 0x1080000000-0x207effffff]
[    0.      0]   NODE_DATA [mem 0x207efd7000-0x207effdfff]
[    0.      0] Reserving 168MB of memory at 688MB for crashkernel (System RAM: 131027MB)
[    0.      0]  [ffffea0000000000-ffffea0041ffffff] PMD -> [ffff88103fe00000-
ffff88107fdfffff] on node 0
[    0.      0]  [ffffea0042000000-ffffea0081ffffff] PMD -> [ffff88203e600000-
ffff88207e5fffff] on node 1
[    0.      0] Zone ranges:
[    0.      0]   DMA      [mem 0x00001000-0x00ffffff]
[    0.      0]   DMA32    [mem 0x01000000-0xffffffff]
[    0.      0]   Normal   [mem 0x100000000-0x207effffff]
[    0.      0] Movable zone start for each node
[    0.      0] Early memory node ranges
```

```
[    0.      0]   node   0: [mem 0x00001000-0x0009cfff]
[    0.      0]   node   0: [mem 0x00100000-0x3fffffff]
[    0.      0]   node   0: [mem 0x40100000-0x7e4fefff]
[    0.      0]   node   0: [mem 0x7efff000-0x7effffff]
[    0.      0]   node   0: [mem 0x100000000-0x107fffffff]
[    0.      0]   node   1: [mem 0x1080000000-0x207effffff]
[    0.      0] On node 0 totalpages: 16769948
[    0.      0]   DMA zone: 64 pages used for memmap
[    0.      0]   DMA zone: 21 pages reserved
[    0.      0]   DMA zone: 3996 pages, LIFO batch:0
[    0.      0]   DMA32 zone: 8016 pages used for memmap
[    0.      0]   DMA32 zone: 513024 pages, LIFO batch:31
[    0.      0]   Normal zone: 253952 pages used for memmap
[    0.      0]   Normal zone: 16252928 pages, LIFO batch:31
[    0.      0] On node 1 totalpages: 16773120
[    0.      0]   Normal zone: 262080 pages used for memmap
[    0.      0]   Normal zone: 16773120 pages, LIFO batch:31
[    0.      0] ACPI: PM-Timer IO Port: 0x408
[    0.      0] ACPI: Local APIC address 0xfee00000
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x00] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x01] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x02] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x03] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x04] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x05] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x06] disabled)
[    0.      0] ACPI: X2APIC (apic_id[0xffffffff] uid[0x07] disabled)
......
[    0.      0] ACPI: LAPIC (acpi_id[0x00] lapic_id[0x00] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x01] lapic_id[0x02] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x02] lapic_id[0x04] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x03] lapic_id[0x06] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x04] lapic_id[0x08] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x05] lapic_id[0x0a] enabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x06] lapic_id[0x20] enabled)
......
[    0.      0] ACPI: LAPIC (acpi_id[0x18] lapic_id[0xff] disabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x19] lapic_id[0xff] disabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x1a] lapic_id[0xff] disabled)
[    0.      0] ACPI: LAPIC (acpi_id[0x1b] lapic_id[0xff] disabled)
......
[    0.      0] ACPI: X2APIC_NMI (uid[0xffffffff] high edge lint[0x1])
[    0.      0] ACPI: LAPIC_NMI (acpi_id[0xff] high edge lint[0x1])
[    0.      0] ACPI: IOAPIC (id[0x08] address[0xfec00000] gsi_base[0])
[    0.      0] IOAPIC[0]: apic_id 8, version 32, address 0xfec00000, GSI 0-23
[    0.      0] ACPI: IOAPIC (id[0x09] address[0xfec01000] gsi_base[24])
[    0.      0] IOAPIC[1]: apic_id 9, version 32, address 0xfec01000, GSI 24-47
[    0.      0] ACPI: IOAPIC (id[0x0a] address[0xfec40000] gsi_base[48])
[    0.      0] IOAPIC[2]: apic_id 10, version 32, address 0xfec40000, GSI 48-71
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 0 global_irq 2 dfl dfl)
[    0.      0] ACPI: INT_SRC_OVR (bus 0 bus_irq 9 global_irq 9 high level)
[    0.      0] ACPI: IRQ0 used by override.
[    0.      0] ACPI: IRQ2 used by override.
[    0.      0] ACPI: IRQ9 used by override.
[    0.      0] Using ACPI (MADT) for SMP configuration information
[    0.      0] ACPI: HPET id: 0x8086a701 base: 0xfed00000
[    0.      0] smpboot: Allowing 240 CPUs, 216 hotplug CPUs
[    0.      0] nr_irqs_gsi: 88
[    0.      0] PM: Registered nosave memory: [mem 0x0009d000-0x0009ffff]
[    0.      0] PM: Registered nosave memory: [mem 0x000a0000-0x000dffff]
[    0.      0] PM: Registered nosave memory: [mem 0x000e0000-0x000fffff]
[    0.      0] PM: Registered nosave memory: [mem 0x40000000-0x400fffff]
[    0.      0] PM: Registered nosave memory: [mem 0x7e4ff000-0x7e8fefff]
[    0.      0] PM: Registered nosave memory: [mem 0x7e8ff000-0x7eefefff]
[    0.      0] PM: Registered nosave memory: [mem 0x7eeff000-0x7effefff]
[    0.      0] PM: Registered nosave memory: [mem 0x7f000000-0x8fffffff]
[    0.      0] PM: Registered nosave memory: [mem 0x90000000-0xfeafffff]
[    0.      0] PM: Registered nosave memory: [mem 0xfeb00000-0xfeb03fff]
[    0.      0] PM: Registered nosave memory: [mem 0xfeb04000-0xfebfffff]
```

```
[    0.      0] PM: Registered nosave memory: [mem 0xfec00000-0xfec00fff]
[    0.      0] PM: Registered nosave memory: [mem 0xfec01000-0xfed17fff]
[    0.      0] PM: Registered nosave memory: [mem 0xfed18000-0xfed18fff]
[    0.      0] PM: Registered nosave memory: [mem 0xfed19000-0xfed1bfff]
[    0.      0] PM: Registered nosave memory: [mem 0xfed1c000-0xfed8ffff]
[    0.      0] PM: Registered nosave memory: [mem 0xfed90000-0xfedfffff]
[    0.      0] PM: Registered nosave memory: [mem 0xfee00000-0xfee00fff]
[    0.      0] PM: Registered nosave memory: [mem 0xfee01000-0xffbfffff]
[    0.      0] PM: Registered nosave memory: [mem 0xffc00000-0xffffffff]
[    0.      0] e820: [mem 0x90000000-0xfeafffff] available for PCI devices
[    0.      0] Booting paravirtualized kernel on bare hardware
[    0.      0] setup_percpu: NR_CPUS:5120 nr_cpumask_bits:240 nr_cpu_ids:240 nr_node_ids:2
[    0.      0] PERCPU: Embedded 28 pages/cpu @ffff88103ee00000 s82816 r8192 d23680 u131072
[    0.      0] pcpu-alloc: s82816 r8192 d23680 u131072 alloc=1*2097152
[    0.      0] pcpu-alloc: [0] 000 001 002 003 004 005 012 013 014 015 016 017 024 026 028
030
[    0.      0] pcpu-alloc: [0] 032 034 036 038 040 042 044 046 048 050 052 054 056 058 060
062
[    0.      0] pcpu-alloc: [0] 064 066 068 070 072 074 076 078 080 082 084 086 088 090 092
094
[    0.      0] pcpu-alloc: [0] 096 098 100 102 104 106 108 110 112 114 116 118 120 122 124
126
[    0.      0] pcpu-alloc: [0] 128 130 132 134 136 138 140 142 144 146 148 150 152 154 156
158
[    0.      0] pcpu-alloc: [0] 160 162 164 166 168 170 172 174 176 178 180 182 184 186 188
190
[    0.      0] pcpu-alloc: [0] 192 194 196 198 200 202 204 206 208 210 212 214 216 218 220
222
[    0.      0] pcpu-alloc: [0] 224 226 228 230 232 234 236 238 --- --- --- --- --- --- ---
---
[    0.      0] pcpu-alloc: [1] 006 007 008 009 010 011 018 019 020 021 022 023 025 027 029
031
[    0.      0] pcpu-alloc: [1] 033 035 037 039 041 043 045 047 049 051 053 055 057 059 061
063
[    0.      0] pcpu-alloc: [1] 065 067 069 071 073 075 077 079 081 083 085 087 089 091 093
095
[    0.      0] pcpu-alloc: [1] 097 099 101 103 105 107 109 111 113 115 117 119 121 123 125
127
[    0.      0] pcpu-alloc: [1] 129 131 133 135 137 139 141 143 145 147 149 151 153 155 157
159
[    0.      0] pcpu-alloc: [1] 161 163 165 167 169 171 173 175 177 179 181 183 185 187 189
191
[    0.      0] pcpu-alloc: [1] 193 195 197 199 201 203 205 207 209 211 213 215 217 219 221
223
[    0.      0] pcpu-alloc: [1] 225 227 229 231 233 235 237 239 --- --- --- --- --- --- ---
---
[    0.      0] Built 2 zonelists in Zone order, mobility grouping on.  Total pages: 33018935
[    0.      0] Policy zone: Normal
[    0.      0] Kernel command line: BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.41.x86_64 root=/dev/
mapper/euleros-root ro reserve_kbox_mem=16M crash_kexec_post_notifiers panic=3
rd.lvm.lv=euleros/swap crashkernel=auto rd.lvm.lv=euleros/root
[    0.      0] PID hash table entries: 4096 (order: 3, 32768 bytes)
[    0.      0] xsave: enabled xstate_bv 0x7, cntxt size 0x340
[    0.      0] Checking aperture...
[    0.      0] No AGP bridge found
[    0.      0] Memory: 131777152k/136298496k available (6254k kernel code, 2126224k absent,
2395120k reserved, 4185k data, 1604k init)
[    0.      0] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=240, Nodes=2
[    0.      0] Hierarchical RCU implementation.
[    0.      0] ?RCU restricting CPUs from NR_CPUS=5120 to nr_cpu_ids=240.
[    0.      0] ?Experimental no-CBs for all CPUs
[    0.      0] ?Experimental no-CBs CPUs: 0-239.
[    0.      0] NR_IRQS:327936 nr_irqs:3416 16
[    0.      0] Console: colour VGA+ 80x25
[    0.      0] console [tty0] enabled
[    0.      0] allocated 536870912 bytes of page_cgroup
[    0.      0] please try 'cgroup_disable=memory' option if you don't want memory cgroups
[    0.      0] Enabling automatic NUMA balancing. Configure with numa_balancing= or the
kernel.numa_balancing sysctl
```



```
[    0.     0] hpet clockevent registered
[    0.     0] tsc: Fast TSC calibration using PIT
[    0.  1000] tsc: Detected 2100.083 MHz processor
[    0.    76] Calibrating delay loop (skipped), value calculated using timer frequency..
4200.16 BogoMIPS (lpj=2100083)
[    0.   362] pid_max: default: 245760 minimum: 1920
[    0.  1218] Security Framework initialized
[    0.  1358] SELinux:  Initializing.
[    0.  1571] SELinux:  Starting in permissive mode
[    0. 12648] Dentry cache hash table entries: 16777216 (order: 15, 134217728 bytes)
[    0. 50933] Inode-cache hash table entries: 8388608 (order: 14, 67108864 bytes)
[    0. 67461] Mount-cache hash table entries: 4096
[    0. 69398] Initializing cgroup subsys memory
[    0. 69591] Initializing cgroup subsys devices
[    0. 69728] Initializing cgroup subsys freezer
[    0. 69864] Initializing cgroup subsys net_cls
[    0. 70000] Initializing cgroup subsys blkio
[    0. 70135] Initializing cgroup subsys perf_event
[    0. 70316] Initializing cgroup subsys hugetlb
[    0. 70591] CPU: Physical Processor ID: 0
[    0. 70726] CPU: Processor Core ID: 0
[    0. 71802] mce: CPU supports 23 MCE banks
[    0. 71973] CPU0: Thermal monitoring enabled (TM1)
[    0. 72126] Last level iTLB entries: 4KB 512, 2MB 0, 4MB 0
Last level dTLB entries: 4KB 512, 2MB 0, 4MB 0
tlb_flushall_shift: 6
[    0. 72582] Freeing SMP alternatives: 24k freed
[    0. 74325] ACPI: Core revision 20130517
[    0.107173] ACPI: All ACPI Tables successfully acquired
[    0.121871] ftrace: allocating 23953 entries in 94 pages
[    0.135973] dmar: Host address width 46
[    0.136383] dmar: DRHD base: 0x000000fbffe000 flags: 0x0
[    0.136791] dmar: IOMMU 0: reg_base_addr fbffe000 ver 1:0 cap d2078c106f0466 ecap f020de
[    0.137529] dmar: DRHD base: 0x000000cfffc000 flags: 0x1
[    0.137935] dmar: IOMMU 1: reg_base_addr cfffc000 ver 1:0 cap d2078c106f0466 ecap f020de
[    0.138666] dmar: ATSR flags: 0x0
[    0.139065] dmar: ATSR flags: 0x0
[    0.139468] dmar: RMRR base: 0x0000007e8b5000 end: 0x0000007e8bcfff
[    0.139996] IOAPIC id 10 under DRHD base  0xfbffe000 IOMMU 0
[    0.140396] IOAPIC id 8 under DRHD base  0xcfffc000 IOMMU 1
[    0.140798] IOAPIC id 9 under DRHD base  0xcfffc000 IOMMU 1
[    0.141197] HPET id 0 under DRHD base 0xcfffc000
[    0.141584] Queued invalidation will be enabled to support x2apic and Intr-remapping.
[    0.142795] Enabled IRQ remapping in x2apic mode
[    0.144021] ..TIMER: vector=0x30 apic1=0 pin1=2 apic2=-1 pin2=-1
[    0.154438] smpboot: CPU0: Intel(R) Xeon(R) CPU E5-2620 v2 @ 2.10GHz (fam: 06, model: 3e,
stepping: 04)
[    0.155381] TSC deadline timer enabled
[    0.155521] Performance Events: PEBS fmt1+, 16-deep LBR, IvyBridge events, full-width
counters, Intel PMU driver.
[    0.156600] ... version:                3
[    0.156985] ... bit width:              48
[    0.157384] ... generic registers:      4
[    0.157781] ... value mask:             0000ffffffffffff
[    0.158184] ... max period:             0000ffffffffffff
[    0.158583] ... fixed-purpose events:   3
[    0.158983] ... event mask:             000000070000000f
[    0.181255] CPU1 microcode updated early to revision 0x428, date = 2014-05-29
[    0.189969] NMI watchdog: enabled on all CPUs, permanently consumes one hw-PMU counter.
[    0.204834] CPU2 microcode updated early to revision 0x428, date = 2014-05-29
[    0.220502] CPU3 microcode updated early to revision 0x428, date = 2014-05-29
[    0.236172] CPU4 microcode updated early to revision 0x428, date = 2014-05-29
[    0.251846] CPU5 microcode updated early to revision 0x428, date = 2014-05-29
[    0.168779] smpboot: Booting Node   0, Processors  #1 #2 #3 #4 #5 OK
[    0.267988] CPU6 microcode updated early to revision 0x428, date = 2014-05-29
[    0.361745] CPU7 microcode updated early to revision 0x428, date = 2014-05-29
[    0.377410] CPU8 microcode updated early to revision 0x428, date = 2014-05-29
[    0.393067] CPU9 microcode updated early to revision 0x428, date = 2014-05-29
[    0.408728] CPU10 microcode updated early to revision 0x428, date = 2014-05-29
```



```
[    0.424371] CPU11 microcode updated early to revision 0x428, date = 2014-05-29
[    0.255911] smpboot: Booting Node   1, Processors  #6 #7 #8 #9 #10 #11 OK
[    0.428794] smpboot: Booting Node   0, Processors  #12 #13 #14 #15 #16 #17 OK
[    0.516003] smpboot: Booting Node   1, Processors  #18 #19 #20 #21 #22 #23
[    0.602460] Brought up 24 CPUs
[    0.603251] smpboot: Total of 24 processors activated (100864.24 BogoMIPS)
[    0.649359] devtmpfs: initialized
[    0.660599] EVM: security.selinux
[    0.660999] EVM: security.ima
[    0.661397] EVM: security.capability
[    0.661986] PM: Registering ACPI NVS region [mem 0x7e8ff000-0x7eefefff] (6291456 bytes)
[    0.664279] atomic64 test passed for x86-64 platform with CX8 and with SSE
[    0.665157] NET: Registered protocol family 16
[    0.673953] ACPI FADT declares the system doesn't support PCIe ASPM, so disable it
[    0.674685] ACPI: bus type PCI registered
[    0.675087] acpiphp: ACPI Hot Plug PCI Controller Driver version: 0.5
[    0.675938] PCI: MMCONFIG for domain 0000 [bus 00-ff] at [mem 0x80000000-0x8fffffff]
(base 0x80000000)
[    0.676668] PCI: MMCONFIG at [mem 0x80000000-0x8fffffff] reserved in E820
[    0.688427] PCI: Using configuration type 1 for base access
[    0.693356] ACPI: Added _OSI(Module Device)
[    0.693757] ACPI: Added _OSI(Processor Device)
[    0.694160] ACPI: Added _OSI(3.0 _SCP Extensions)
[    0.694557] ACPI: Added _OSI(Processor Aggregator Device)
[    0.703192] ACPI: EC: Look up EC in DSDT
[    0.859018] ACPI: Interpreter enabled
[    0.859423] ACPI Exception: AE_NOT_FOUND, While evaluating Sleep State [\_S1_] (20130517/
hwxface-571)
[    0.860293] ACPI Exception: AE_NOT_FOUND, While evaluating Sleep State [\_S2_] (20130517/
hwxface-571)
[    0.861172] ACPI Exception: AE_NOT_FOUND, While evaluating Sleep State [\_S3_] (20130517/
hwxface-571)
[    0.862068] ACPI: (supports S0 S4 S5)
[    0.862471] ACPI: Using IOAPIC for interrupt routing
[    0.865783] HEST: Table parsing has been initialized.
[    0.866188] PCI: Using host bridge windows from ACPI; if necessary, use "pci=nocrs" and
report a bug
[    0.883902] ACPI: PCI Root Bridge [PCI0] (domain 0000 [bus 00-7e])
[    0.884304] acpi PNP0A08:00: _OSC: OS supports [ExtendedConfig ASPM ClockPM Segments MSI]
[    0.892055] acpi PNP0A08:00: _OSC: OS now controls [PCIeHotplug PME AER PCIeCapability]
[    0.893555] PCI host bridge to bus 0000:00
[    0.893962] pci_bus 0000:00: root bus resource [bus 00-7e]
[    0.894368] pci_bus 0000:00: root bus resource [io  0x0000-0x0cf7]
[    0.894772] pci_bus 0000:00: root bus resource [io  0x1000-0xbfff]
[    0.895182] pci_bus 0000:00: root bus resource [mem 0x000a0000-0x000bffff]
[    0.895588] pci_bus 0000:00: root bus resource [mem 0x90000000-0xcfffefff]
[    0.896000] pci 0000:00:00.0: [8086:0e00] type 00 class 0x060000
[    0.896066] pci 0000:00:00.0: PME# supported from D0 D3hot D3cold
[    0.896144] pci 0000:00:01.0: [8086:0e02] type 01 class 0x060400
[    0.896224] pci 0000:00:01.0: PME# supported from D0 D3hot D3cold
[    0.896273] pci 0000:00:01.0: System wakeup disabled by ACPI
[    0.896717] pci 0000:00:02.0: [8086:0e04] type 01 class 0x060400
[    0.896794] pci 0000:00:02.0: PME# supported from D0 D3hot D3cold
[    0.896841] pci 0000:00:02.0: System wakeup disabled by ACPI
[    0.897282] pci 0000:00:02.2: [8086:0e06] type 01 class 0x060400
[    0.897358] pci 0000:00:02.2: PME# supported from D0 D3hot D3cold
[    0.897405] pci 0000:00:02.2: System wakeup disabled by ACPI
[    0.897832] pci 0000:00:04.0: [8086:0e20] type 00 class 0x088000
[    0.897847] pci 0000:00:04.0: reg 0x10: [mem 0x94b00000-0x94b03fff 64bit]
[    0.897974] pci 0000:00:04.1: [8086:0e21] type 00 class 0x088000
[    0.897988] pci 0000:00:04.1: reg 0x10: [mem 0x94b04000-0x94b07fff 64bit]
[    0.898114] pci 0000:00:04.2: [8086:0e22] type 00 class 0x088000
[    0.898128] pci 0000:00:04.2: reg 0x10: [mem 0x94b08000-0x94b0bfff 64bit]
[    0.898258] pci 0000:00:04.3: [8086:0e23] type 00 class 0x088000
[    0.898272] pci 0000:00:04.3: reg 0x10: [mem 0x94b0c000-0x94b0ffff 64bit]
[    0.898400] pci 0000:00:04.4: [8086:0e24] type 00 class 0x088000
[    0.898414] pci 0000:00:04.4: reg 0x10: [mem 0x94b10000-0x94b13fff 64bit]
[    0.898535] pci 0000:00:04.5: [8086:0e25] type 00 class 0x088000
[    0.898549] pci 0000:00:04.5: reg 0x10: [mem 0x94b14000-0x94b17fff 64bit]
```



```
[    0.898674] pci 0000:00:04.6: [8086:0e26] type 00 class 0x088000
[    0.898688] pci 0000:00:04.6: reg 0x10: [mem 0x94b18000-0x94b1bfff 64bit]
[    0.898812] pci 0000:00:04.7: [8086:0e27] type 00 class 0x088000
[    0.898826] pci 0000:00:04.7: reg 0x10: [mem 0x94b1c000-0x94b1ffff 64bit]
[    0.898949] pci 0000:00:05.0: [8086:0e28] type 00 class 0x088000
[    0.899060] pci 0000:00:05.1: [8086:0e29] type 00 class 0x088000
[    0.899186] pci 0000:00:05.2: [8086:0e2a] type 00 class 0x088000
[    0.899297] pci 0000:00:05.4: [8086:0e2c] type 00 class 0x080020
[    0.899307] pci 0000:00:05.4: reg 0x10: [mem 0x94b28000-0x94b28fff]
[    0.899442] pci 0000:00:11.0: [8086:1d3e] type 01 class 0x060400
[    0.899537] pci 0000:00:11.0: PME# supported from D0 D3hot D3cold
[    0.899617] pci 0000:00:16.0: [8086:1d3a] type 00 class 0x078000
[    0.899638] pci 0000:00:16.0: reg 0x10: [mem 0x94b20000-0x94b2000f 64bit]
[    0.899703] pci 0000:00:16.0: PME# supported from D0 D3hot D3cold
[    0.899770] pci 0000:00:16.1: [8086:1d3b] type 00 class 0x078000
[    0.899791] pci 0000:00:16.1: reg 0x10: [mem 0x94b21000-0x94b2100f 64bit]
[    0.899856] pci 0000:00:16.1: PME# supported from D0 D3hot D3cold
[    0.899936] pci 0000:00:1a.0: [8086:1d2d] type 00 class 0x0c0320
[    0.900508] pci 0000:00:1a.0: reg 0x10: [mem 0x94b26000-0x94b263ff]
[    0.903810] pci 0000:00:1a.0: PME# supported from D0 D3hot D3cold
[    0.903860] pci 0000:00:1a.0: System wakeup disabled by ACPI
[    0.904297] pci 0000:00:1c.0: [8086:1d10] type 01 class 0x060400
[    0.904381] pci 0000:00:1c.0: PME# supported from D0 D3hot D3cold
[    0.904427] pci 0000:00:1c.0: System wakeup disabled by ACPI
[    0.904867] pci 0000:00:1c.2: [8086:1d14] type 01 class 0x060400
[    0.904949] pci 0000:00:1c.2: PME# supported from D0 D3hot D3cold
[    0.904996] pci 0000:00:1c.2: System wakeup disabled by ACPI
[    0.905437] pci 0000:00:1c.3: [8086:1d16] type 01 class 0x060400
[    0.905520] pci 0000:00:1c.3: PME# supported from D0 D3hot D3cold
[    0.905567] pci 0000:00:1c.3: System wakeup disabled by ACPI
[    0.906010] pci 0000:00:1d.0: [8086:1d26] type 00 class 0x0c0320
[    0.906559] pci 0000:00:1d.0: reg 0x10: [mem 0x94b25000-0x94b253ff]
[    0.909889] pci 0000:00:1d.0: PME# supported from D0 D3hot D3cold
[    0.909935] pci 0000:00:1d.0: System wakeup disabled by ACPI
[    0.910372] pci 0000:00:1e.0: [8086:244e] type 01 class 0x060401
[    0.910457] pci 0000:00:1e.0: System wakeup disabled by ACPI
[    0.910892] pci 0000:00:1f.0: [8086:1d41] type 00 class 0x060100
[    0.911065] pci 0000:00:1f.2: [8086:1d02] type 00 class 0x010601
[    0.911083] pci 0000:00:1f.2: reg 0x10: [io  0x4048-0x404f]
[    0.911091] pci 0000:00:1f.2: reg 0x14: [io  0x405c-0x405f]
[    0.911099] pci 0000:00:1f.2: reg 0x18: [io  0x4040-0x4047]
[    0.911107] pci 0000:00:1f.2: reg 0x1c: [io  0x4058-0x405b]
[    0.911115] pci 0000:00:1f.2: reg 0x20: [io  0x4020-0x403f]
[    0.911123] pci 0000:00:1f.2: reg 0x24: [mem 0x94b24000-0x94b247ff]
[    0.911169] pci 0000:00:1f.2: PME# supported from D3hot
[    0.911233] pci 0000:00:1f.3: [8086:1d22] type 00 class 0x0c0500
[    0.911249] pci 0000:00:1f.3: reg 0x10: [mem 0x94b22000-0x94b220ff 64bit]
[    0.911270] pci 0000:00:1f.3: reg 0x20: [io  0x4000-0x401f]
[    0.911352] pci 0000:00:1f.6: [8086:1d24] type 00 class 0x118000
[    0.911373] pci 0000:00:1f.6: reg 0x10: [mem 0x94b23000-0x94b23fff 64bit]
[    0.911553] pci 0000:01:00.0: [14e4:1657] type 00 class 0x020000
[    0.911568] pci 0000:01:00.0: reg 0x10: [mem 0x948b0000-0x948bffff 64bit pref]
[    0.911580] pci 0000:01:00.0: reg 0x18: [mem 0x948a0000-0x948affff 64bit pref]
[    0.911592] pci 0000:01:00.0: reg 0x20: [mem 0x94890000-0x9489ffff 64bit pref]
[    0.911657] pci 0000:01:00.0: PME# supported from D0 D3hot D3cold
[    0.911710] pci 0000:01:00.1: [14e4:1657] type 00 class 0x020000
[    0.911724] pci 0000:01:00.1: reg 0x10: [mem 0x94880000-0x9488ffff 64bit pref]
[    0.911736] pci 0000:01:00.1: reg 0x18: [mem 0x94870000-0x9487ffff 64bit pref]
[    0.911748] pci 0000:01:00.1: reg 0x20: [mem 0x94860000-0x9486ffff 64bit pref]
[    0.911812] pci 0000:01:00.1: PME# supported from D0 D3hot D3cold
[    0.911859] pci 0000:01:00.2: [14e4:1657] type 00 class 0x020000
[    0.911873] pci 0000:01:00.2: reg 0x10: [mem 0x94850000-0x9485ffff 64bit pref]
[    0.911885] pci 0000:01:00.2: reg 0x18: [mem 0x94840000-0x9484ffff 64bit pref]
[    0.911896] pci 0000:01:00.2: reg 0x20: [mem 0x94830000-0x9483ffff 64bit pref]
[    0.911960] pci 0000:01:00.2: PME# supported from D0 D3hot D3cold
[    0.912008] pci 0000:01:00.3: [14e4:1657] type 00 class 0x020000
[    0.912022] pci 0000:01:00.3: reg 0x10: [mem 0x94820000-0x9482ffff 64bit pref]
[    0.912035] pci 0000:01:00.3: reg 0x18: [mem 0x94810000-0x9481ffff 64bit pref]
[    0.912047] pci 0000:01:00.3: reg 0x20: [mem 0x94800000-0x9480ffff 64bit pref]
```



```
[    0.912111] pci 0000:01:00.3: PME# supported from D0 D3hot D3cold
[    0.913372] pci 0000:00:01.0: PCI bridge to [bus 01]
[    0.913781] pci 0000:00:01.0:   bridge window [mem 0x94800000-0x948fffff 64bit pref]
[    0.913836] pci 0000:00:02.0: PCI bridge to [bus 03]
[    0.914302] pci 0000:02:00.0: [1000:005b] type 00 class 0x010400
[    0.914311] pci 0000:02:00.0: reg 0x10: [io  0x3000-0x30ff]
[    0.914320] pci 0000:02:00.0: reg 0x14: [mem 0x94a40000-0x94a43fff 64bit]
[    0.914329] pci 0000:02:00.0: reg 0x1c: [mem 0x94a00000-0x94a3ffff 64bit]
[    0.914340] pci 0000:02:00.0: reg 0x30: [mem 0xfffe0000-0xffffffff pref]
[    0.914383] pci 0000:02:00.0: supports D1 D2
[    0.916370] pci 0000:00:02.2: PCI bridge to [bus 02]
[    0.916774] pci 0000:00:02.2:   bridge window [io  0x3000-0x3fff]
[    0.916778] pci 0000:00:02.2:   bridge window [mem 0x94a00000-0x94affff]
[    0.916855] pci 0000:04:00.0: [8086:1d6b] type 00 class 0x010700
[    0.916873] pci 0000:04:00.0: reg 0x10: [mem 0x9447c000-0x9447ffff 64bit pref]
[    0.916887] pci 0000:04:00.0: reg 0x18: [mem 0x94000000-0x943fffff 64bit pref]
[    0.916896] pci 0000:04:00.0: reg 0x20: [io  0x2000-0x20ff]
[    0.916999] pci 0000:04:00.0: reg 0x164: [mem 0x94400000-0x94403fff 64bit pref]
[    0.917067] pci 0000:04:00.3: [8086:1d70] type 00 class 0x0c0500
[    0.917081] pci 0000:04:00.3: reg 0x10: [mem 0x94900000-0x94900fff]
[    0.917117] pci 0000:04:00.3: reg 0x20: [io  0x2100-0x211f]
[    0.917184] pci 0000:04:00.3: PME# supported from D0 D3hot D3cold
[    0.917253] pci 0000:00:11.0: PCI bridge to [bus 04-05]
[    0.917660] pci 0000:00:11.0:   bridge window [io  0x2000-0x2fff]
[    0.917665] pci 0000:00:11.0:   bridge window [mem 0x94900000-0x949fffff]
[    0.917671] pci 0000:00:11.0:   bridge window [mem 0x94000000-0x944fffff 64bit pref]
[    0.917741] pci 0000:00:1c.0: PCI bridge to [bus 06]
[    0.918208] pci 0000:00:1c.2: PCI bridge to [bus 07]
[    0.918712] pci 0000:08:00.0: [126f:0750] type 00 class 0x030000
[    0.918737] pci 0000:08:00.0: reg 0x10: [mem 0x90000000-0x93ffffff pref]
[    0.918755] pci 0000:08:00.0: reg 0x14: [mem 0x94600000-0x947fffff]
[    0.918836] pci 0000:08:00.0: reg 0x30: [mem 0xffff0000-0xffffffff pref]
[    0.918951] pci 0000:08:00.0: supports D1
[    0.918953] pci 0000:08:00.0: PME# supported from D0 D1 D3hot
[    0.920686] pci 0000:00:1c.3: PCI bridge to [bus 08]
[    0.921092] pci 0000:00:1c.3:   bridge window [mem 0x94600000-0x947fffff]
[    0.921098] pci 0000:00:1c.3:   bridge window [mem 0x90000000-0x93ffffff 64bit pref]
[    0.921172] pci 0000:00:1e.0: PCI bridge to [bus 09] (subtractive decode)
[    0.921588] pci 0000:00:1e.0:   bridge window [io  0x0000-0x0cf7] (subtractive decode)
[    0.921589] pci 0000:00:1e.0:   bridge window [io  0x1000-0xbfff] (subtractive decode)
[    0.921591] pci 0000:00:1e.0:   bridge window [mem 0x000a0000-0x000bffff] (subtractive
decode)
[    0.921593] pci 0000:00:1e.0:   bridge window [mem 0x90000000-0xcfffefff] (subtractive
decode)
[    0.921646] pci_bus 0000:00: on NUMA node 0
[    0.921648] acpi PNP0A08:00: Disabling ASPM (FADT indicates it is unsupported)
[    0.922562] ACPI: PCI Root Bridge [UNC0] (domain 0000 [bus 7f])
[    0.922957] acpi PNP0A03:00: _OSC: OS supports [ExtendedConfig ASPM ClockPM Segments MSI]
[    0.923678] acpi PNP0A03:00: _OSC failed (AE_NOT_FOUND); disabling ASPM
[    0.924139] PCI host bridge to bus 0000:7f
[    0.924529] pci_bus 0000:7f: root bus resource [bus 7f]
[    0.924931] pci 0000:7f:08.0: [8086:0e80] type 00 class 0x088000
[    0.924984] pci 0000:7f:09.0: [8086:0e90] type 00 class 0x088000
[    0.925029] pci 0000:7f:0a.0: [8086:0ec0] type 00 class 0x088000
[    0.925076] pci 0000:7f:0a.1: [8086:0ec1] type 00 class 0x088000
[    0.925120] pci 0000:7f:0a.2: [8086:0ec2] type 00 class 0x088000
[    0.925164] pci 0000:7f:0a.3: [8086:0ec3] type 00 class 0x088000
[    0.925207] pci 0000:7f:0b.0: [8086:0e1e] type 00 class 0x088000
[    0.925245] pci 0000:7f:0b.3: [8086:0e1f] type 00 class 0x088000
[    0.925286] pci 0000:7f:0c.0: [8086:0ee0] type 00 class 0x088000
[    0.925326] pci 0000:7f:0c.1: [8086:0ee2] type 00 class 0x088000
[    0.925374] pci 0000:7f:0c.2: [8086:0ee4] type 00 class 0x088000
[    0.925420] pci 0000:7f:0d.0: [8086:0ee1] type 00 class 0x088000
[    0.925458] pci 0000:7f:0d.1: [8086:0ee3] type 00 class 0x088000
[    0.925500] pci 0000:7f:0d.2: [8086:0ee5] type 00 class 0x088000
[    0.925544] pci 0000:7f:0e.0: [8086:0ea0] type 00 class 0x088000
[    0.925588] pci 0000:7f:0e.1: [8086:0e30] type 00 class 0x110100
[    0.925637] pci 0000:7f:0f.0: [8086:0ea8] type 00 class 0x088000
[    0.925694] pci 0000:7f:0f.1: [8086:0e71] type 00 class 0x088000
```



```
[    0.925746] pci 0000:7f:0f.2: [8086:0eaa] type 00 class 0x088000
[    0.925803] pci 0000:7f:0f.3: [8086:0eab] type 00 class 0x088000
[    0.925859] pci 0000:7f:0f.4: [8086:0eac] type 00 class 0x088000
[    0.925914] pci 0000:7f:0f.5: [8086:0ead] type 00 class 0x088000
[    0.925979] pci 0000:7f:10.0: [8086:0eb0] type 00 class 0x088000
[    0.926033] pci 0000:7f:10.1: [8086:0eb1] type 00 class 0x088000
[    0.926089] pci 0000:7f:10.2: [8086:0eb2] type 00 class 0x088000
[    0.926148] pci 0000:7f:10.3: [8086:0eb3] type 00 class 0x088000
[    0.926204] pci 0000:7f:10.4: [8086:0eb4] type 00 class 0x088000
[    0.926260] pci 0000:7f:10.5: [8086:0eb5] type 00 class 0x088000
[    0.926312] pci 0000:7f:10.6: [8086:0eb6] type 00 class 0x088000
[    0.926368] pci 0000:7f:10.7: [8086:0eb7] type 00 class 0x088000
[    0.926424] pci 0000:7f:13.0: [8086:0e1d] type 00 class 0x088000
[    0.926466] pci 0000:7f:13.1: [8086:0e34] type 00 class 0x110100
[    0.926510] pci 0000:7f:13.4: [8086:0e81] type 00 class 0x088000
[    0.926557] pci 0000:7f:13.5: [8086:0e36] type 00 class 0x110100
[    0.926603] pci 0000:7f:16.0: [8086:0ec8] type 00 class 0x088000
[    0.926650] pci 0000:7f:16.1: [8086:0ec9] type 00 class 0x088000
[    0.926693] pci 0000:7f:16.2: [8086:0eca] type 00 class 0x088000
[    0.926886] ACPI: PCI Root Bridge [PCI1] (domain 0000 [bus 80-fe])
[    0.927270] acpi PNP0A08:01: _OSC: OS supports [ExtendedConfig ASPM ClockPM Segments MSI]
[    0.928112] acpi PNP0A08:01: _OSC: OS now controls [PCIeHotplug PME AER PCIeCapability]
[    0.929110] PCI host bridge to bus 0000:80
[    0.929499] pci_bus 0000:80: root bus resource [bus 80-fe]
[    0.929894] pci_bus 0000:80: root bus resource [io  0x0000-0x0cf7]
[    0.930288] pci_bus 0000:80: root bus resource [io  0xc000-0xffff]
[    0.930680] pci_bus 0000:80: root bus resource [mem 0x000a0000-0x000bffff]
[    0.931078] pci_bus 0000:80: root bus resource [mem 0xd0000000-0xfbffffff]
[    0.931487] pci 0000:80:02.0: [8086:0e04] type 01 class 0x060400
[    0.931570] pci 0000:80:02.0: PME# supported from D0 D3hot D3cold
[    0.931604] pci 0000:80:02.0: System wakeup disabled by ACPI
[    0.932029] pci 0000:80:02.2: [8086:0e06] type 01 class 0x060400
[    0.932115] pci 0000:80:02.2: PME# supported from D0 D3hot D3cold
[    0.932152] pci 0000:80:02.2: System wakeup disabled by ACPI
[    0.932578] pci 0000:80:04.0: [8086:0e20] type 00 class 0x088000
[    0.932593] pci 0000:80:04.0: reg 0x10: [mem 0xd1200000-0xd1203fff 64bit]
[    0.932706] pci 0000:80:04.1: [8086:0e21] type 00 class 0x088000
[    0.932721] pci 0000:80:04.1: reg 0x10: [mem 0xd1204000-0xd1207fff 64bit]
[    0.932840] pci 0000:80:04.2: [8086:0e22] type 00 class 0x088000
[    0.932854] pci 0000:80:04.2: reg 0x10: [mem 0xd1208000-0xd120bfff 64bit]
[    0.932969] pci 0000:80:04.3: [8086:0e23] type 00 class 0x088000
[    0.932984] pci 0000:80:04.3: reg 0x10: [mem 0xd120c000-0xd120ffff 64bit]
[    0.933108] pci 0000:80:04.4: [8086:0e24] type 00 class 0x088000
[    0.933122] pci 0000:80:04.4: reg 0x10: [mem 0xd1210000-0xd1213fff 64bit]
[    0.933238] pci 0000:80:04.5: [8086:0e25] type 00 class 0x088000
[    0.933253] pci 0000:80:04.5: reg 0x10: [mem 0xd1214000-0xd1217fff 64bit]
[    0.933364] pci 0000:80:04.6: [8086:0e26] type 00 class 0x088000
[    0.933379] pci 0000:80:04.6: reg 0x10: [mem 0xd1218000-0xd121bfff 64bit]
[    0.933494] pci 0000:80:04.7: [8086:0e27] type 00 class 0x088000
[    0.933508] pci 0000:80:04.7: reg 0x10: [mem 0xd121c000-0xd121ffff 64bit]
[    0.933619] pci 0000:80:05.0: [8086:0e28] type 00 class 0x088000
[    0.933719] pci 0000:80:05.1: [8086:0e29] type 00 class 0x088000
[    0.933835] pci 0000:80:05.2: [8086:0e2a] type 00 class 0x088000
[    0.933932] pci 0000:80:05.4: [8086:0e2c] type 00 class 0x080020
[    0.933942] pci 0000:80:05.4: reg 0x10: [mem 0xd1220000-0xd1220fff]
[    0.934132] pci 0000:81:00.0: [8086:10fb] type 00 class 0x020000
[    0.934144] pci 0000:81:00.0: reg 0x10: [mem 0xd1100000-0xd111ffff 64bit]
[    0.934151] pci 0000:81:00.0: reg 0x18: [io  0xd020-0xd03f]
[    0.934166] pci 0000:81:00.0: reg 0x20: [mem 0xd1140000-0xd1143fff 64bit]
[    0.934211] pci 0000:81:00.0: PME# supported from D0 D3hot
[    0.934238] pci 0000:81:00.0: reg 0x184: [mem 0xd1000000-0xd1003fff 64bit pref]
[    0.934250] pci 0000:81:00.0: reg 0x190: [mem 0xd0f00000-0xd0f03fff 64bit pref]
[    0.934295] pci 0000:81:00.1: [8086:10fb] type 00 class 0x020000
[    0.934307] pci 0000:81:00.1: reg 0x10: [mem 0xd1120000-0xd113ffff 64bit]
[    0.934314] pci 0000:81:00.1: reg 0x18: [io  0xd000-0xd01f]
[    0.934328] pci 0000:81:00.1: reg 0x20: [mem 0xd1144000-0xd1147fff 64bit]
[    0.934372] pci 0000:81:00.1: PME# supported from D0 D3hot
[    0.934395] pci 0000:81:00.1: reg 0x184: [mem 0xd0e00000-0xd0e03fff 64bit pref]
[    0.934407] pci 0000:81:00.1: reg 0x190: [mem 0xd0d00000-0xd0d03fff 64bit pref]
```



```
[    0.936107] pci 0000:80:02.0: PCI bridge to [bus 81-83]
[    0.936500] pci 0000:80:02.0:   bridge window [io  0xd000-0xdfff]
[    0.936504] pci 0000:80:02.0:   bridge window [mem 0xd1100000-0xd11fffff]
[    0.936509] pci 0000:80:02.0:   bridge window [mem 0xd0d00000-0xd10fffff 64bit pref]
[    0.936574] pci 0000:84:00.0: [8086:10fb] type 00 class 0x020000
[    0.936586] pci 0000:84:00.0: reg 0x10: [mem 0xd0400000-0xd07fffff 64bit pref]
[    0.936593] pci 0000:84:00.0: reg 0x18: [io  0xc020-0xc03f]
[    0.936607] pci 0000:84:00.0: reg 0x20: [mem 0xd0c04000-0xd0c07fff 64bit pref]
[    0.936615] pci 0000:84:00.0: reg 0x30: [mem 0xffc00000-0xffffffff pref]
[    0.936654] pci 0000:84:00.0: PME# supported from D0 D3hot
[    0.936681] pci 0000:84:00.0: reg 0x184: [mem 0xd0b04000-0xd0b07fff 64bit pref]
[    0.936693] pci 0000:84:00.0: reg 0x190: [mem 0xd0a04000-0xd0a07fff 64bit pref]
[    0.936737] pci 0000:84:00.1: [8086:10fb] type 00 class 0x020000
[    0.936749] pci 0000:84:00.1: reg 0x10: [mem 0xd0000000-0xd03fffff 64bit pref]
[    0.936756] pci 0000:84:00.1: reg 0x18: [io  0xc000-0xc01f]
[    0.936771] pci 0000:84:00.1: reg 0x20: [mem 0xd0a00000-0xd0a03fff 64bit pref]
[    0.936777] pci 0000:84:00.1: reg 0x30: [mem 0xffc00000-0xffffffff pref]
[    0.936816] pci 0000:84:00.1: PME# supported from D0 D3hot
[    0.936839] pci 0000:84:00.1: reg 0x184: [mem 0xd0900000-0xd0903fff 64bit pref]
[    0.936851] pci 0000:84:00.1: reg 0x190: [mem 0xd0800000-0xd0803fff 64bit pref]
[    0.938113] pci 0000:80:02.2: PCI bridge to [bus 84-86]
[    0.938505] pci 0000:80:02.2:   bridge window [io  0xc000-0xcfff]
[    0.938512] pci 0000:80:02.2:   bridge window [mem 0xd0000000-0xd0cfffff 64bit pref]
[    0.938530] pci_bus 0000:80: on NUMA node 1
[    0.938531] acpi PNP0A08:01: Disabling ASPM (FADT indicates it is unsupported)
[    0.939321] ACPI: PCI Root Bridge [UNC1] (domain 0000 [bus ff])
[    0.939714] acpi PNP0A03:01: _OSC: OS supports [ExtendedConfig ASPM ClockPM Segments MSI]
[    0.940434] acpi PNP0A03:01: _OSC failed (AE_NOT_FOUND); disabling ASPM
[    0.940879] PCI host bridge to bus 0000:ff
[    0.941255] pci_bus 0000:ff: root bus resource [bus ff]
[    0.941644] pci 0000:ff:08.0: [8086:0e80] type 00 class 0x088000
[    0.941698] pci 0000:ff:09.0: [8086:0e90] type 00 class 0x088000
[    0.941756] pci 0000:ff:0a.0: [8086:0ec0] type 00 class 0x088000
[    0.941801] pci 0000:ff:0a.1: [8086:0ec1] type 00 class 0x088000
[    0.941848] pci 0000:ff:0a.2: [8086:0ec2] type 00 class 0x088000
[    0.941888] pci 0000:ff:0a.3: [8086:0ec3] type 00 class 0x088000
[    0.941933] pci 0000:ff:0b.0: [8086:0e1e] type 00 class 0x088000
[    0.941974] pci 0000:ff:0b.3: [8086:0e1f] type 00 class 0x088000
[    0.942016] pci 0000:ff:0c.0: [8086:0ee0] type 00 class 0x088000
[    0.942057] pci 0000:ff:0c.1: [8086:0ee2] type 00 class 0x088000
[    0.942097] pci 0000:ff:0c.2: [8086:0ee4] type 00 class 0x088000
[    0.942143] pci 0000:ff:0d.0: [8086:0ee1] type 00 class 0x088000
[    0.942196] pci 0000:ff:0d.1: [8086:0ee3] type 00 class 0x088000
[    0.942240] pci 0000:ff:0d.2: [8086:0ee5] type 00 class 0x088000
[    0.942286] pci 0000:ff:0e.0: [8086:0ea0] type 00 class 0x088000
[    0.942330] pci 0000:ff:0e.1: [8086:0e30] type 00 class 0x110100
[    0.942386] pci 0000:ff:0f.0: [8086:0ea8] type 00 class 0x088000
[    0.942444] pci 0000:ff:0f.1: [8086:0e71] type 00 class 0x088000
[    0.942503] pci 0000:ff:0f.2: [8086:0eaa] type 00 class 0x088000
[    0.942559] pci 0000:ff:0f.3: [8086:0eab] type 00 class 0x088000
[    0.942615] pci 0000:ff:0f.4: [8086:0eac] type 00 class 0x088000
[    0.942677] pci 0000:ff:0f.5: [8086:0ead] type 00 class 0x088000
[    0.942737] pci 0000:ff:10.0: [8086:0eb0] type 00 class 0x088000
[    0.942793] pci 0000:ff:10.1: [8086:0eb1] type 00 class 0x088000
[    0.942850] pci 0000:ff:10.2: [8086:0eb2] type 00 class 0x088000
[    0.942905] pci 0000:ff:10.3: [8086:0eb3] type 00 class 0x088000
[    0.942961] pci 0000:ff:10.4: [8086:0eb4] type 00 class 0x088000
[    0.943018] pci 0000:ff:10.5: [8086:0eb5] type 00 class 0x088000
[    0.943076] pci 0000:ff:10.6: [8086:0eb6] type 00 class 0x088000
[    0.943137] pci 0000:ff:10.7: [8086:0eb7] type 00 class 0x088000
[    0.943191] pci 0000:ff:13.0: [8086:0e1d] type 00 class 0x088000
[    0.943234] pci 0000:ff:13.1: [8086:0e34] type 00 class 0x110100
[    0.943276] pci 0000:ff:13.4: [8086:0e81] type 00 class 0x088000
[    0.943320] pci 0000:ff:13.5: [8086:0e36] type 00 class 0x110100
[    0.943367] pci 0000:ff:16.0: [8086:0ec8] type 00 class 0x088000
[    0.943407] pci 0000:ff:16.1: [8086:0ec9] type 00 class 0x088000
[    0.943451] pci 0000:ff:16.2: [8086:0eca] type 00 class 0x088000
[    0.948322] ACPI: PCI Interrupt Link [LNKA] (IRQs 1 *3 4 5 6 7 10 11 12 14 15)
[    0.949871] ACPI: PCI Interrupt Link [LNKB] (IRQs 1 3 4 *5 6 7 10 11 12 14 15)
```



```
[    0.951414] ACPI: PCI Interrupt Link [LNKC] (IRQs 1 3 4 5 *6 7 10 11 12 14 15)
[    0.952958] ACPI: PCI Interrupt Link [LNKD] (IRQs 1 3 4 5 6 *7 10 11 12 14 15)
[    0.954504] ACPI: PCI Interrupt Link [LNKE] (IRQs 1 3 4 5 6 7 10 11 12 14 15) *0,
disabled.
[    0.956144] ACPI: PCI Interrupt Link [LNKF] (IRQs 1 3 4 5 6 7 *10 11 12 14 15)
[    0.957690] ACPI: PCI Interrupt Link [LNKG] (IRQs 1 3 4 5 6 7 10 11 12 14 15) *0,
disabled.
[    0.959364] ACPI: PCI Interrupt Link [LNKH] (IRQs 1 3 4 5 6 7 10 11 12 14 15) *9
[    0.960972] ACPI: Enabled 4 GPEs in block 00 to 3F
[    0.962014] vgaarb: device added: PCI:0000:08:00.0,decodes=io+mem,owns=io+mem,locks=none
[    0.962746] vgaarb: loaded
[    0.963137] vgaarb: bridge control possible 0000:08:00.0
[    0.963603] SCSI subsystem initialized
[    0.964012] ACPI: bus type USB registered
[    0.964419] usbcore: registered new interface driver usbfs
[    0.964846] usbcore: registered new interface driver hub
[    0.965286] usbcore: registered new device driver usb
[    0.966131] PCI: Using ACPI for IRQ routing
[    0.971317] PCI: pci_cache_line_size set to 64 bytes
[    0.971519] e820: reserve RAM buffer [mem 0x0009d000-0x0009ffff]
[    0.971521] e820: reserve RAM buffer [mem 0x7e4ff000-0x7fffffff]
[    0.971523] e820: reserve RAM buffer [mem 0x7f000000-0x7fffffff]
[    0.971524] e820: reserve RAM buffer [mem 0x207f000000-0x207fffffff]
[    0.971797] NetLabel: Initializing
[    0.972183] NetLabel:  domain hash size = 128
[    0.972570] NetLabel:  protocols = UNLABELED CIPSOv4
[    0.972973] NetLabel:  unlabeled traffic allowed by default
[    0.973415] hpet0: at MMIO 0xfed00000, IRQs 2, 8, 0, 0, 0, 0, 0, 0
[    0.974352] hpet0: 8 comparators, 64-bit 14.318180 MHz counter
[    0.977788] Switching to clocksource hpet
[    0.987784] pnp: PnP ACPI init
[    0.988177] ACPI: bus type PNP registered
[    0.988758] system 00:00: [mem 0xfed1c000-0xfed1ffff] has been reserved
[    0.989154] system 00:00: [mem 0xcffff000-0xcfffffff] has been reserved
[    0.989553] system 00:00: [mem 0xfed20000-0xfed3ffff] has been reserved
[    0.989947] system 00:00: [mem 0xcfffc000-0xcfffdfff] could not be reserved
[    0.990349] system 00:00: [mem 0xff000000-0xffffffff] could not be reserved
[    0.990743] system 00:00: [mem 0xfee00000-0xfeefffff] could not be reserved
[    0.991137] system 00:00: Plug and Play ACPI device, IDs PNP0c02 (active)
[    0.991174] pnp 00:01: [dma 4]
[    0.991193] pnp 00:01: Plug and Play ACPI device, IDs PNP0200 (active)
[    0.991214] pnp 00:02: Plug and Play ACPI device, IDs INT0800 (active)
[    0.991307] pnp 00:03: Plug and Play ACPI device, IDs PNP0103 (active)
[    0.991346] pnp 00:04: Plug and Play ACPI device, IDs PNP0c04 (active)
[    0.991410] system 00:05: [io  0x0680-0x069f] has been reserved
[    0.991805] system 00:05: [io  0x0400-0x0453] could not be reserved
[    0.992197] system 00:05: [io  0x0458-0x047f] has been reserved
[    0.992595] system 00:05: [io  0x0500-0x057f] has been reserved
[    0.992985] system 00:05: [io  0x164e-0x164f] has beerlock time:20160217172018-3ceff
```

3. 控制台打印日志

```
console:
*****area type:console - location in console area:2*****
rlock time:20160217172018-3ceff
[  468.704136] kbox:catch rlock!end process!
[  468.704508] calling kbox_sync :begin
[  468.704879] sync kbox :begin
[  468.705247] open all redirect device :begin
[  468.705634] open all redirect device :end
[  468.706019] flush kbox regions :begin
[  468.706408] kbox region (rlock) is writing into (hmem), action is 202
[  468.706800] test write len : 68907
[  468.707184] first start addr : ffffc9001ec02000
[  468.707570] second start addr : ffffc9001ec02000
[  468.707958] cur_index : 0, offset : 0, third start addr : ffffc9001ec02000
[  468.708352] first length : 68907
[  468.728001] cur_index : 1, record_number : 1, total_number : 2
[  468.728392] kbox region (rlock) has been written into (hmem)
[  468.728783] dev hmem is dirty
rlock time:20160217172018-3ceff
```


9.3 如何使用 Kbox

本节主要介绍Kbox的使用流程以及使用中的注意事项。

9.3.1 使用流程

本节介绍内核黑匣子应用流程，包括修改配置文件、重启Kbox服务、查看异常信息等。

使用流程

Kbox的使用流程如图9-2所示。

图 9-2 Kbox 使用简单流程

说明

1、在EulerOS系统中，Kbox默认随内核一起发布，系统启动时Kbox服务会自动开启。

2、启动参数中必须预先配置reserve_mem，默认为16M。若未配置或少于16M，则kbox服务启动会失败

步骤说明

具体步骤说明如表9-2所示。

注意

1. 必须配置存储介质，否则捕获的信息不能存储，系统重启之后信息丢失。
2. 配置之后，必须重启黑匣子服务，配置内容才会生效。

表 9-2 Kbox 步骤说明

编号	流程	说明
1	启动系统	在EulerOS系统中，Kbox默认随内核一起发布，系统启动时Kbox服务会自动开启。
2	修改配置文件	配置产品信息、抓取的异常事件、以及可用的存储方式。Kbox配置文件路径为/etc/kbox/config。
3	重启Kbox服务	修改配置文件后，通过重启Kbox服务，使配置文件生效。
4	记录异常信息	当异常事件发生后，Kbox会根据配置文件，将指定的异常事件信息记录到相应的存储设备中。
5	查看异常信息	具有root权限的用户通过Kbox提供的日志导出命令，将存储设备中的异常信息导出到指定目录的文件中，进行查看。

9.3.2 修改配置文件

本节介绍如何正确配置Kbox工具。

前提条件

- 在EulerOS系统中，Kbox正常运行。
- Kbox转储依赖的转储介质正常工作。

注意事项

用户修改完配置文件，重启Kbox后才生效。

配置步骤

以root权限，在shell命令行环境下：

步骤1 打开配置文件。在任一路径下，使用如下命令：

```
vi  /etc/kbox/config
```

配置文件内容如下：

```
#此文件为kbox(黑匣子)的配置文件
#根据各产品的具体情况，完成配置
#*重要说明*：配置参数请按照提示信息正确配置，参数必须包含英文引号，引号内外不能有空格

##############  硬件平台基本信息  ###############
#产品名称：比如"euler" "pangea" "dopra" "cgp"
product="euler"

#产品版本号：比如V100R001C00B251
version="V200R002C10"

#版本所在框号：比如frame=1~9（一般用于级联）
frame="1"

#单板所在槽位号：比如slot=0~12
```

```
slot="0"

#单板所在位置：比如前插板还是后插板(0~1)
locate="0"

#单板的硬件信息：比如pangea-v2
hard_version=""

#############  异常抓取场景,如果选项为空或非法字符，则默认为no  ###############
#panic事件:主要是显式调用内核的panic函数或者异常导致内核触发oops
panic_event="yes"

#正常reboot事件:主要是用户态命令触发reboot、shutdown、halt和init
reboot_event="no"

#异常reboot事件:主要是内核态异常触发machine restart
emerge_event="no"

#Die事件（可选，建议只抓取panic和nmi两种，其他暂不考虑）
die_event="yes"

#Die事件处理后的操作
die_call_panic="no"

#OOM事件
oom_event="yes"
#OOM事件发生后，是否调用panic
oom_call_panic="yes"

#D死锁事件
dlock_event="no"

#R死锁事件
rlock_event="yes"

#Acpi事件:待定义
acpi_event="no"

#Mce事件:待定义，主要是intel cpu异常相关
mce_event="no"

#BMC预中断事件:主要给mccp产品用
bmc_event="no"

#用户自定义事件:增删ip
ipaddr_event="no"

#网络事件:主要是网络接口的up/down事件
net_event="no"

#用户自定义事件:增删路由、增删rule、修改时间（是否拆开待讨论）
route_event="no"
rule_event="no"
time_event="no"

#安全事件:主要是安全加固相关
security_event="no"

#看门狗事件:主要是看门狗超时复位
wtdog_event="no"


###############  异常保存介质  #################
#bmc设备
bmc="off"

#bios设备
bios="off"
```



```
#pcie设备
pcie="off"

#highmem设备
hmem="on"

#logic设备
logic="off"

###############  模块插入参数  ###############
#注意: start_pddr必须是物理地址，且4K对齐，用十六进制表示。
#      mem_size用十六进制，最小为8M字节
#      参数之间用空格分隔
#bmc驱动参数
bmc_param=""

#bios驱动参数
#pangea产品kbox存储在nvram中，可以不用配置。
bios_param=""

#pcie驱动参数
pcie_param="pci_bus_id=00 pci_device_id=00"

#highmem驱动
hmem_param="sym_start_paddr_name=reserve_kbox_mem_start sym_mem_size_name=reserve_kbox_mem_len"

#logic驱动参数
logic_param=""

#############  内部测试设备  #############
#net设备(内部使用)
net="off"

###############  TFTP服务器IP(内部使用)  ################
#配置net设备为on状态时，可将region的消息传到TFTP服务器，默认配置(127.0.0.1)不传输
TGT_IP="127.0.0.1"
TGT_MAC=""

### 死机上报接口
die_notify_func=""
```

步骤2 修改配置文件。

按照当前系统的硬件环境，配置好系统的基本信息、抓取异常的事件以及存储设备。配置主要包含三方面产品信息、异常事件和存储设备三方面。

注意

1. 参数值请严格按照说明进行配置，配置参数必须使用英文引号，引号内不能有空格。如果字符中间需要空格，请以"_"字符代替。配置为yes/no或YES/NO、on/off或ON/OFF的配置项，大小写要统一，且不能包含空格，否则默认处理为no。
2. 异常事件必须至少配置1项，否则Kbox不能加载。必须配置存储设备，否则异常信息不能转储，Kbox重启后信息丢失。
3. 修改配置文件后，请重启Kbox，否则配置不生效。

具体参数说明和配置指导如表9-3、表9-4和表9-5所示：

表 9-3 产品信息

列序号	参数名称	说明	参数值	是否必配
1	product	当前设备名称（或产品名称）。	用户自定义字符串。	否
2	version	当前系统发布的版本号。	用户自定义字符串。	否
3	frame	当前机架号。	大于0的整数。	否
4	slot	当前单板槽位号。	大于0的整数。	否
5	locate	单板位置。	0或1。	否
6	hard_version	硬件版本。	用户自定义字符串。	否

表 9-4 异常事件

列序号	参数名称	说明	参数值	是否支持	是否必配
1	panic_event	panic事件。	yes/no	是	是，至少配置一种异常事件。
2	reboot_event	安全重启事件。	yes/no	否	
3	emerge_event	异常重启事件。	yes/no	否	
4	die_event	die事件。	yes/no	是	
5	dlock_event	D死锁事件。	yes/no	否	
6	rlock_event	R死锁事件。	yes/no	否	
7	acpi_event	电源上下电事件。	yes/no	否	
8	mce_event	硬件错误事件。	yes/no	否	
9	bmc_event	BMC预中断事件。	yes/no	否	
10	ipaddr_event	修改IP事件。	yes/no	否	
11	net_event	Net设备接口状态变化事件。	yes/no	否	
12	route_event	修改路由事件。	yes/no	否	
13	rule_event	用户自定义的acl规避变化事件。	yes/no	否	

列序号	参数名称	说明	参数值	是否支持	是否必配
14	time_event	时区、时间变化事件。	yes/no	否	
15	security_event	用户自定的安全事件。	yes/no	否	
16	wtdog_event	看门狗事件。	yes/no	否	
17	oom_event	oom事件。	yes/no	是	
18	oom_call_panic	oom发生后是否调用panic。	yes/no	是	否

说明

只有在oom_event项设置为yes时，oom_call_panic配置项才能够生效。

表 9-5 存储设备配置

列序号	参数名称	说明	参数值	是否必配
1	bmc	BMC存储。	on/off	是，至少配置其中一种存储方式。
2	bios	bios nvram存储。	on/off	
3	pcie	PCIE存储。	on/off	
4	hmem	普通内存存储或者高端内存存储。	on/off	
5	logic	逻辑存储。	on/off	

说明

1. 表3和表4必须结合使用。
2. EulerOS默认配置hmem="on"。
3. 通用服务器默认使用hmem参数。

表 9-6 存储配置参数

列序号	参数名称	说明	参数值	是否必配
1	bmc_param	BMC存储驱动参数	暂不支持	否
2	bios_param	biosnvram驱动参数	驱动自带默认参数	否
3	pcie_param	pcie驱动参数	pci_bus_id=00 pci_device_id=00	是

列序号	参数名称	说明	参数值	是否必配
4	hmem_param	保留内存驱动参数	该参数有两种取值方式： ● sym_start_paddr_name=reserve_kbox_mem_start sym_mem_size_name=reserve_kbox_mem_len ● start_paddr=0xxxxxx mem_size=0xxxxx。start_paddr和mem_size为16进制整数，start_paddr为起始物理地址，mem_size为内存大小。 当前默认为第一种配置方式，建议用户不要修改。	是
5	logic_param	logic驱动参数	暂不支持	否

注意

如果没有非易失性存储设备，可以通过启动参数保留内存，用普通内存代替非易失性存储介质，然后通过kdump将这段内存保存为内存镜像，该内存镜像可通过**export_kbox_img_to_txt**命令读取。

----结束

9.3.3 重启 Kbox 服务

本节主要介绍Kbox工具运行环境、约束限制及加载流程。

前提条件

- 在EulerOS系统中，Kbox正常运行。
- Kbox转储依赖的转储介质正常工作。

加载步骤

在正常启动的EulerOS系统中，以root权限登录系统。已经按照要求，完成配置文件的修改。

在shell命令行环境中，在任意路径下，使用命令：

```
#systemctl  restart kbox
```

验证加载是否成功

说明

系统关机和重启，Kbox服务不停止。系统关机和重启调用systemctl stop kbox命令，不会停止Kbox服务，该接口为空。

查看Kbox运行状态。使用命令：

```
#systemct  status   kbox
```

- 如果显示active，则表示Kbox模块加载成功，并且已经在运行。
- 如果显示不是active，则表示Kbox模块未加载或未加载成功。请仔细检查Kbox配置文件配置参数，确保无误后重新加载。更多信息请参见**9.5 常见问题处理**。

9.3.4 查看异常信息

本节介绍如何查看保存在存储设备中的异常信息。

前提条件

- 在EulerOS系统中，Kbox正常运行。
- Kbox转储依赖的转储介质正常工作。
- 具有root权限的用户才能正常查看Kbox相关信息。

导出异常信息

系统启动时，Kbox日志被保存到/var/crash(默认位置，kdump配置文件中path参数可以重新指定位置)，每次系统异常的日志都保存到以“ip-时间点”命名的目录中，该目录下的xxx-xxx.img.gz就是kbox日志。

说明

当前kbox日志存放的目录为“127.0.0.1-时间点”，127.0.0.1是指该kbox日志仅存放在本机，不会传输至其他服务器。

```
[root@localhost 127.0.0.1-2015.11.24-12:08:11]# ls
100000000-1000000.img.gz  vmcore-dmesg.txt
```

使用如下命令对日志解压解析：

```
gunzip  100000000-1000000.img.gz
export_kbox_img_to_txt  -i 100000000-1000000.img
```

提示export_kbox_img_to_txt: parse kbox data into /tmp/kbox_log.txt success.

kbox_log.txt即为解析出来的kbox日志

用户可使用vi命令查看异常信息的日志文件。此处以OOM信息为例。详细信息说明请参见**9.2.3 提供系统oom信息**。

```
area type: 0-panic 1-reboot 2-emerge 3-intermit 4-oom 5-oops/die 6-rlock 7-message 8-console
---------------area type:4, record num:1, unit_size:64 ------------


###num:0, record len:34586
---------KBOX_START----------
oom time:20160310212837-d6cf8
Current Pid: 46, comm: kworker/3:1 Tainted: G           O E ---- -------
3.10.0-229.20.1.42.x86_64 #1
Call Trace:
 [<ffffffffa036eb4b>] ? kbox_show_cur_stack+0x6b/0x80 [kbox]
 [<ffffffffa036f4e3>] ? kbox_dump_oom+0x13/0x30 [kbox]
 [<ffffffffa036e630>] ? kbox_oom_callback+0xe0/0x220 [kbox]
 [<ffffffff811ddb00>] ? pollwake+0x20/0x90
 [<ffffffff810aa700>] ? wake_up_state+0x20/0x20
```

```
 [<ffffffff810a1958>] ? __wake_up_common+0x58/0x90
 [<ffffffff81612b8c>] ? notifier_call_chain+0x4c/0x70
 [<ffffffff8109e81d>] ? __blocking_notifier_call_chain+0x4d/0x70
 [<ffffffff8109e856>] ? blocking_notifier_call_chain+0x16/0x20
 [<ffffffff8115d29e>] ? out_of_memory+0x4e/0x4f0
 [<ffffffff813976ac>] ? tty_ldisc_deref+0x3c/0xa0
 [<ffffffff8139bb8d>] ? moom_callback+0x4d/0x50
 [<ffffffff8109015b>] ? process_one_work+0x17b/0x470
 [<ffffffff81090f2b>] ? worker_thread+0x11b/0x400
 [<ffffffff81090e10>] ? rescuer_thread+0x400/0x400
 [<ffffffff8109830f>] ? kthread+0xcf/0xe0
 [<ffffffff81098240>] ? kthread_create_on_node+0x140/0x140
 [<ffffffff816170d8>] ? ret_from_fork+0x58/0x90
 [<ffffffff81098240>] ? kthread_create_on_node+0x140/0x140
Stack:
 ffff88013861bba0 ffff880138582e00 0000000000000000 0000000000000000
 ffff88013861bdc8 0000000000000000 ffff88013861bc08 ffffffffa0375091
 000000042cc68d17 0000000000000000 ffff880138582e00 ffff88013861bbf8
 0000000000000014 ffff88013861bc40 ffffffffa036eabd ffff880100000004
 ffffffffa036eb4b 000000002cc68d17 0000000000000000 ffff88013861bc64
 ffff88013861bc50 ffffffffa036f4e3 ffff88013861bce8 ffffffffa036e630
 363130323861bc70 3832313230313330 38666336642d3733 ffffffff811ddb00
 0000000000000000 ffff8800362f1700 ffffffff810aa700 0000000000000000
 0000000000000000 000000002cc68d17 ffff88013861bcf8 ffffffff810a1958
 0000000100000000 000000002cc68d17 00000000ffffffff 0000000000000000
 0000000000000000 ffff88013861bd20 ffffffff81612b8c ffffffff8197fd00
 0000000000000000 0000000000000000 ffff88013861bdc8 00000000ffffffff
 ffff88013861bd60 ffffffff8109e81d 0000000000000000 0000000000000000
 0000000000000001 ffff88013ed92ec0 ffff88013efd8000 00000000000000d0
 ffff88013861bd70 ffffffff8109e856 ffff88013861be08 ffffffff8115d29e
 ffff88013ed93680 ffff88013ed93680 ffff88013861bde0 0000000000000286
 0000000000000282 ffff880036acd980 ffff88013861bdd0 ffffffff813976ac
 0000000000000282 0000000000000000 ffff88013861be18 000000002cc68d17
 ffffffff819c57e0 ffff8801385ea000 ffff88013ed92ec0 ffff88013ed96f00
 00000000000000c0 ffff88013861be18 ffffffff8139bb8d ffff88013861be60
 ffffffff8109015b 000000003ed92ed8 0000000000000000 ffff88013ed92ed8
 ffff8801385ea030 ffff880138582e00 ffff8801385ea000 ffff88013ed92ec0
 ffff88013861bec0 ffffffff81090f2b ffff88013861bfd8 ffff88013861bfd8
 ffff880138582e00 ffff880138582e00 ffff880138582e00 ffff880138417d38
```

9.4 查看 Kbox 相关信息

本节介绍如何查看Kbox相关的信息。

前提条件

- 在EulerOS系统中，Kbox正常运行。
- Kbox转储依赖的转储介质正常工作。

查看操作

- 查看Kbox的运行状态。使用命令：

```
systemctl  status  kbox
kbox.service - Crash message collector
   Loaded: loaded (/usr/lib/systemd/system/kbox.service; disabled)
   Active: active (exited) since Thu 2015-11-26 10:39:47 EST; 13s ago
  Process: 26545 ExecStart=/usr/sbin/kbox start (code=exited, status=0/SUCCESS)
 Main PID: 26545 (code=exited, status=0/SUCCESS)

Nov 26 10:39:46 localhost.localdomain kbox[26545]: kbox: kbox module has been loaded successfully!
Nov 26 10:39:46 localhost.localdomain kbox[26545]: driver: drivers have been loaded successfully!
Nov 26 10:39:47 localhost.localdomain systemd[1]: Started Crash message collector.
```

- 显示

- 如果Kbox正常运行，显示active状态。如下图所示：

```
kbox.service - Crash message collector
   Loaded: loaded (/usr/lib/systemd/system/kbox.service; disabled)
   Active: active (exited) since Thu 2015-11-26 10:39:47 EST; 13s ago
```

- 如果显示failed状态，则表示Kbox未加载到系统中。如下图所示：

```
kbox.service - Crash message collector
   Loaded: loaded (/usr/lib/systemd/system/kbox.service; disabled)
   Active: failed (Result: exit-code) since Thu 2015-11-26 10:44:43 EST; 5s ago
  Process: 29347 ExecStart=/usr/sbin/kbox start (code=exited, status=1/FAILURE)
 Main PID: 29347 (code=exited, status=1/FAILURE)
```

- 查看当前系统中的临时存储区(region)。

  Kbox启动后，默认会创建多个临时存储区，保存对应类型的日志信息。查看当前系统中已有的临时存储区命令如下

```
the effective info in /etc/kbox/config:

------------- product      info -------------
product       name: euler
version     number: V200R002C10
frame       number: 1
slot        number: 0
locate        info: 0

------------- enabled    events -------------
panic_event    die_event      oom_event
rlock_event    oom_call_panic


------------- enabled  storages -------------
hmem


the regions and storages list:

------------- regions      list -------------
console die message oom panic rlock

------------- storages     list -------------
hmem
```

  临时存储区域说明如表9-7所示。

表 9-7 存储区域说明

存储区域名称	说明
console	用来存储异常发生后，打印到控制台的日志信息，如果信息过多，会产生回绕，覆盖最先保存的信息。该临时存储区大小为32K byte。
die	用来存储由于内核态进程发生oops(非法访问内存空间、使用空指针等)事件所抓取的异常信息。该临时存储区大小为128K byte。
message	用来存储异常发生时，最近的64K byte内核打印信息。
oom	用来存放oom事件抓取的信息，该临时存储区大小为256K byte。
panic	用来存放panic事件发生时，抓取的异常信息。该临时存储区大小为32K byte。

存储区域名称	说明
rlock	用来存放发生rlock(内核软狗和硬狗狗叫)时抓取到的异常信息，该临时存储区大小为256K byte。

- 查看存储设备的全局信息

  显示当前系统中注册的存储设备

  所谓的存储设备就是注册到Kbox模块中的存储介质，包括NVRAM、PCIE设备内存、BMC设备内存等。当系统发生异常后，Kbox将临时存储区的日志信息刷新到这些设备中。目前默认存储设备为hmem(普通内存)。使用举例如下：

```
localhost:~ # kboxstatus
the effective info in /etc/kbox/config:

------------ product      info ------------
product       name: euler
version     number: V200R002C10
frame       number: 1
slot        number: 0
locate        info: 0

------------ enabled    events ------------
panic_event    die_event      oom_event      oom_call_panic


------------ enabled  storages ------------
hmem


the regions and storages list:

------------ regions      list ------------
console die message oom panic

------------ storages     list ------------
hmem mem_info
localhost:~ #
```

9.5 常见问题处理

本节介绍使用Kbox常见的问题以及处理方法。

Kbox 导出日志失败

1. 通过命令systemctl status kbox，检查Kbox是否启动。如果没有启动，运行systemctl start kbox命令启动服务，启动失败，请检查kbox配置是否正确。
2. 通过命令systemctl status kdump，检查kdump服务是否启动正常。如果没有正常启动，运行systemctl start kdump命令启动kdump服务，如果启动失败，请检查kdump配置是否正确。

   说明

   kbox日志都是通过kdump保存。

Kbox 不能抓取异常信息

通过命令systemctl status kbox，确认Kbox服务是否正常启动，如果没启动服务，运行systemctl start kbox命令启动服务。

Kbox 服务启动失败

请按照如下步骤逐一排查：

1. 确定当前操作系统是否为EulerOS，使用命令“cat /etc/os-release”进行查询。Kbox当前仅支持EulerOS版本，在其它版本的Linux操作系统上无法使用。
2. 通过命令systemctl status kbox和journalctl -l，确认Kbox服务启动失败提示信息。
3. 检查硬件结构，使用命令“uname -m”进行查询。确认Kbox宿主主机是否是x86架构，Kbox只支持x86架构。
4. Kbox配置文件修改有误。请核对修改的配置项，确保修改正确。
5. 预留内存出现错误也会导致kbox启动失败。

   系统在启动阶段会为kbox调测工具预留分配16M的内存。这是通过在grub配置文件的cmdline参数中增加reserve_kbox_mem=16M来实现。

   内存预留算法分两步：

   a. 在e820图中，从高端地址到低端地址遍历每个usable状态的内存段，直到有能满足16MB大小段为止。

   b. 在找到的内存段中再从高端地址向低端地址reserved 16MB内存。

   注意事项：

   （1）如果系统可用内存小于16M，则kbox调测工具启动会失败。

   （2）使用memmap预留内存需要注意避免跟reserve_kbox_mem内存段冲突，memmap在预留内存时不判断内存段状态。这种强制分配的方式可能会对已经reserved的内存段再次reserved而造成冲突。

   下图所示，正常情况下reserve_kbox_mem预留内存段前后的地址段范围和内存状态的改变。

```
[    0.000000] Command line: BOOT_IMAGE=/vmlinuz-3.10.0-229.20.1.20.hulk.x86_64 root=/dev/mapper/euleros-root ro reserve_kbox_mem=16M
crashkernel=auto rd.lvm.lv=euleros/root console=tty0 console=ttyS0,115200
[    0.000000] e820: BIOS-provided physical RAM map:
[    0.000000] BIOS-e820: [mem 0x0000000000000000-0x000000000009afff] usable
[    0.000000] BIOS-e820: [mem 0x000000000009b000-0x000000000009ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000000e0000-0x00000000000fffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000000100000-0x00000000753fcfff] usable
[    0.000000] BIOS-e820: [mem 0x00000000753fd000-0x0000000075c2dfff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000075c2e000-0x0000000075d2dfff] ACPI data
[    0.000000] BIOS-e820: [mem 0x0000000075d2e000-0x0000000079d2dfff] ACPI NVS
[    0.000000] BIOS-e820: [mem 0x0000000079d2e000-0x000000007bd46fff] reserved
[    0.000000] BIOS-e820: [mem 0x000000007bd47000-0x000000007bd47fff] usable
[    0.000000] BIOS-e820: [mem 0x000000007bd48000-0x000000007bdcdfff] reserved
[    0.000000] BIOS-e820: [mem 0x000000007bdce000-0x000000007bffffff] usable
[    0.000000] BIOS-e820: [mem 0x0000000080000000-0x0000000008ffffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000fed1c000-0x00000000fed1ffff] reserved
[    0.000000] BIOS-e820: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[    0.000000] BIOS-e820: [mem 0x0000000100000000-0x000000c07fffffff] usable
[    0.000000] e820: reserve kbox memory size: 16M
[    0.000000] e820: reserve kbox memory success. [mem 0x000000c07f000000-0x000000c07fffffff]
[    0.000000] NX (Execute Disable) protection: active
[    0.000000] e820: user-defined physical RAM map:
[    0.000000] user: [mem 0x0000000000000000-0x000000000009afff] usable
[    0.000000] user: [mem 0x000000000009b000-0x000000000009ffff] reserved
[    0.000000] user: [mem 0x00000000000e0000-0x00000000000fffff] reserved
[    0.000000] user: [mem 0x0000000000100000-0x00000000753fcfff] usable
[    0.000000] user: [mem 0x00000000753fd000-0x0000000075c2dfff] reserved
[    0.000000] user: [mem 0x0000000075c2e000-0x0000000075d2dfff] ACPI data
[    0.000000] user: [mem 0x0000000075d2e000-0x0000000079d2dfff] ACPI NVS
[    0.000000] user: [mem 0x0000000079d2e000-0x000000007bd46fff] reserved
[    0.000000] user: [mem 0x000000007bd47000-0x000000007bd47fff] usable
[    0.000000] user: [mem 0x000000007bd48000-0x000000007bdcdfff] reserved
[    0.000000] user: [mem 0x000000007bdce000-0x000000007bffffff] usable
[    0.000000] user: [mem 0x0000000080000000-0x0000000008ffffff] reserved
[    0.000000] user: [mem 0x00000000fed1c000-0x00000000fed1ffff] reserved
[    0.000000] user: [mem 0x00000000ff000000-0x00000000ffffffff] reserved
[    0.000000] user: [mem 0x0000000100000000-0x000000c07effffff] usable
[    0.000000] user: [mem 0x000000c07f000000-0x000000c07fffffff] reserved
```

9.6 附录

9.6.1 命令参考

说明

/usr/sbin/kbox命令为kbox内部使用命令，用来启动kbox，请用户不要使用。

启动 Kbox 服务

```
systemctl  start  kbox
```

重启 Kbox 服务

```
systemctl  restart  kbox
```

查看 Kbox 状态

```
systemctl  status  kbox
```

查看 Kbox 有效配置信息、临时存储区名称及已注册的存储设备

```
kboxstatus
```

说明

如果kbox服务未启动，则不会显示当前临时存储区(region)以及注册的存储设备

导出日志信息

os_kbox_config命令用来导出非易失性存储里面的日志。

说明

如果系统中没有非易失性存储设备，则执行此命令无效。

os_kbox_config -o dev=[存储设备名称]，type=[导出日志类型]，index=[第几条] | -h | -v

详细的参数说明请参见表9-8。

表 9-8 参数说明

参数名称	描述
dev=[存储设备名称]	当前系统中注册的存储设备名称（biosnvram、hmem、pcie），保存异常信息。
type=[导出日志类型]	导出日志类型，与region名称相同，可取oom/console/message/die/panic/rlock。
index=[第几条]	查看第几条信息，取值[1,32]。取1时为最近的一条记录，最多记录32条信息。
-h	查看帮助。
-v	查看kbox版本信息。

解析日志文件

由于通用服务器没有非易失性存储系统，因此kbox只能配置为在普通内存中存储日志。当系统crash时，kdump会保存这段普通内存为内存镜像文件，通过export_kbox_img_to_txt来解析这个镜像文件，输出文本格式的日志文件。

export_kbox_img_to_txt -i kbox文件

说明

- kbox文件默认使用gzip压缩，解析时请先解压，参考命令gunzip xxx-xxx.img.gz。
- 该命令只适用于通用服务器上kbox日志解析。

10 kdump

Kdump是一种内核崩溃转储机制，在系统崩溃时收集整个内存信息，用于异常事件的定位分析。

10.1 特性描述

当程序或操作人员通过魔术键c或其他原因导致标准内核（即正常运行的内核）崩溃时，Kdump会切换到crash内核，该内核用于捕获标准内核崩溃时的内存信息。

注意

使用默认配置，Kdump在系统崩溃时是不会保存用户数据的。如果用户修改相关配置，可能会保存部分用户数据，请谨慎使用。如需关闭kdump功能，请参考**10.4 管理kdump服务**。

10.2 约束限制

- 当前仅支持EulerOS操作系统。
- kdump过程受到磁盘写速度、内存使用率、逻辑狗叫时间等因素限制，kdump过程产生的vmcore可能存在不完整的情况。
- kdump时间与内存规格强相关，内存越大，dump时间越长，保存的vmcore越大。
- kdump配置文件修改后，必须重启kdump服务。如果系统时间发生跳变（跳变成比修改时间早的时间），重启服务检测不到配置文件有修改，修改不会生效。

- 创建虚拟机时，必须在虚拟机配置文件中配置

```
<features>
<acpi/>
</features>
```

否则虚拟机中kdump功能不可用。

10.3 配置 kdump 参数

获取 Kdump 信息

说明

系统启动后，Kdump服务默认处于工作状态。

每发生一次内核崩溃，Kdump便会在/var/crash目录(默认)下生成一个以127.0.0.1-时间戳命名的文件夹。/var/crash目录下默认只能存放1个文件夹，即上一次内核崩溃的Kdump信息。当再次发生内核crash时，则会先删除之前保存的目录，再保存本次crash的信息。

默认情况下，kdump保存了两个文件：

- xxx-xxx.img.gz

  该文件为kbox存储日志文件。

```
gunzip  xxx-xxx.img.gz  //解压
export_kbox_img_to_txt -i xxx-xxx.img.img  //解析
```

  解析后的文件默认存放在/tmp/kbox_log.txt，查看该文件即可。

- vmcore-dmesg.txt

  该文件是内核崩溃时的dmesg信息。

配置 Kdump 参数

注意

- Kdump配置文件用户使用默认配置即可，随意修改可能导致异常。
- 系统启动后的cmdline参数中必须包含保留内核参数crashkernel=auto或者crashkernel=256M@64M，不要修改(用户可通过命令“cat /proc/cmdline”进行查看)。
- kdump配置文件修改后，必须重启kdump服务。如果系统时间发生跳变（跳变成比修改时间早的时间），重启服务检测不到配置文件有修改，修改不会生效。解决办法：删除/boot/initramfs-3.10.0-xxxx.x86_64kdump.img文件后，重启kdump服务。

Kdump各参数在“/etc/kdump.conf ”文件中配置，这里介绍几个重要的参数配置，如表10-1。

表 10-1 Kdump 参数说明

参数	含义	取值
default	kdump失败后的默认操作	● shell：kdump失败后启动bash。 ● reboot | halt | poweroff：重启或者关机。 当前默认设置为reboot。
path	保存kdump日志绝对路径 **说明** 该路径必须在根（/）分区	默认保存在/var/crash路径下，可设置为/分区下任意绝对路径。
dracut_args	dracat打包kdump initrd的参数	默认为--install "dump_mem_imag gzip awk set_reboot_timer.sh" --nofscks，用户可了解dracat功能后设置。
core_collector	保存kdump日志工具	默认为makedumpfile，其中-d项指定过滤内存级别为31，用户无需修改。
kdump_obj	设置kdump保存kbox日志或者vmcore	● vmcore：保存vmcre。 ● kbox：保存kbox，默认。 ● all：保存vmcore和kbox。
kbox_mem	kbox日志在内存中的存放区域	kbox自填，用户慎用。
keep_old_dumps	历史kdump日志保存个数	● -1：只保存当前日志。 ● 1 | 2 | 3 | ...：支持设置>0个历史日志。
watchdog_time	设置kdump过程自动复位时间（秒）	默认为30秒，如果kdump_obj配置项设置为vmcore或者all，该项设置为>45秒。

10.4 管理 kdump 服务

以下命令用于启动、停止或查询Kdump服务：

- 停止Kdump服务
  ```
  #systemctl  stop  kdump
  ```
- 启动Kdump服务
  ```
  #systemctl  start  kdump
  ```
- 重启Kdump服务
  ```
  #systemctl  restart  kdump
  ```
- 查询Kdump服务
  ```
  #systemctl  status  kdump
  ```


10.5 使用 crash 工具解析 vmcore 文件

用户可使用Euler linux提供的crash工具，对Kdump保存的vmcore文件进行分析，进而定位问题。工具使用流程如下：

1. 在VMP发布包的“Software\02.Tools\DebugTools\”目录下，获取crash工具的发布包vmcore_debug_tool.tar.gz。
2. 在EulerOS 2.1环境下，将vmcore_debug_tool.tar.gz解压到任意目录，然后将Kdump保存的vmcore文件拷贝至vmcore_debug_tool.tar.gz解压后的目录下。
3. 在crash工具解压的目录下执行如下命令：

```
./crash vmlinux-3.4.24.15-0.11-default vmlinux-3.4.24.15-0.11-default.debug vmcore
```

打印系统的相关信息，如内核、CPU、内存、PANIC信息等：

```
      KERNEL: vmlinux-3.4.24.15-0.11-default
   DEBUGINFO: vmlinux-3.4.24.15-0.11-default.debug
    DUMPFILE: vmcore
        CPUS: 4
        DATE: Mon Nov 24 15:35:44 2014
      UPTIME: 854015929139 days, 18:47:25
LOAD AVERAGE: 5.99, 6.00, 6.04
       TASKS: 108
    NODENAME: Storage
     RELEASE: 3.4.24.15-0.11-default
     VERSION: #1 SMP Tue Nov 12 06:28:53 UTC 2013 (0c85715)
     MACHINE: x86_64  (1800 Mhz)
      MEMORY: 4 GB
       PANIC: ""
         PID: 60724
     COMMAND: "exhaustmem"
        TASK: ffff8800593cc500  [THREAD_INFO: ffff880062046000]
         CPU: 1
       STATE: TASK_RUNNING (PANIC)

crash>
```

Euler linux提供的crash工具提供多种命令，帮助用户查看堆栈、日志、内存等信息，使用“help”查看crash支持的命令：

```
crash> help
*              files          mach           repeat         timer ascii
fuser          mount          search         vm    bt             gdb
net            set            vtop  btop           help           p
sig            waitq dev            ipcs           ps             struct
whatisdis            irq            pte            swap           wr    eval
kmem           ptob           sym            q     exit           list
ptov           sys
extend         log            rd             task
```

说明

如果循环缓冲区日志被重定向到nvram，则log命令不可用。Nvram内存段目前不能保存到vmcore中。

各命令的具体说明可通过“man xxx”或者“help xxx”进行查看。例如：

```
crash> man bt
NAME
  bt - backtrace
SYNOPSIS
  bt [-a|-g|-r|-t|-T|-l|-e|-E|-f|-F|-o|-O] [-R ref] [-I ip] [-S sp] [pid | task]
DESCRIPTION
  Display a kernel stack backtrace.  If no arguments are given, the stack
  trace of the current context will be displayed.
       -a  displays the stack traces of the active task on each CPU.
           (only applicable to crash dumps)
       -g  displays the stack traces of all threads in the thread group of
```

```
          the target task; the thread group leader will be displayed first.
      -r  display raw stack data, consisting of a memory dump of the two
          pages of memory containing the task_union structure.
……
```

10.6 常见问题分析方法

踩堆栈问题

1. 编写代码模拟stack被踩。

```
#include <linux/kernel.h>
#include <linux/module.h>


static int __init a_init(void)
{
        char msg[10] = {0};

        printk("a init\n");

        strcpy(msg, "this modules is testing module,
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
aaaa");


        return 0;
}

static void __exit a_exit(void)
{
        printk("a exit\n");
}

module_init(a_init)
module_exit(a_exit)
MODULE_LICENSE("GPL");
```

2. 分析方法：

   使用“bt”命令显示出问题进程堆栈，发现无法解析堆栈。

```
crash> bt
PID: 13360  TASK: ffff8801e386c140  CPU: 2   COMMAND: "insmod"
 #0 [ffff8801fddabc60] machine_kexec at ffffffff8101fc9b
 #1 [ffff8801fddabcb0] crash_kexec at ffffffff81082115
 #2 [ffff8801fddabd80] show_registers at ffffffff81006979
 #3 [ffff8801fddabde0] __die at ffffffff8138e3e7
 #4 [ffff8801fddabe10] die at ffffffff81007623
 #5 [ffff8801fddabe40] do_general_protection at ffffffff8138e0a9
 #6 [ffff8801fddabe70] general_protection at ffffffff8138d7cf
    [exception RIP: init_module+60]
    RIP: ffffffffa087e03c  RSP: ffff8801fddabf20  RFLAGS: 00010246
    RAX: 0000000000000000  RBX: 0000000000614010  RCX: 0000000000000000
    RDX: 0000000000000007  RSI: ffffffffa0a4c117  RDI: ffff8801fddabfe7
    RBP: 20676e6974736574   R8: 0000000000000000   R9: 0000000000000400
    R10: 0000000000000000  R11: 0000000000000000  R12: ffffffffa087e000
    R13: 0000000000000000  R14: 00007f139d2f6000  R15: 0000000000000002
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
    RIP: 6161616161616161  RSP: 00007fff37c795f8  RFLAGS: 00010202
    RAX: 6161616161616161  RBX: 6161616161616161  RCX: 6161616161616161
    RDX: 6161616161616161  RSI: 6161616161616161  RDI: 6161616161616161
    RBP: 6161616161616161   R8: 6161616161616161   R9: 6161616161616161
    R10: 6161616161616161  R11: 6161616161616161  R12: 6161616161616161
    R13: 6161616161616161  R14: 6161616161616161  R15: 6161616161616161
    ORIG_RAX: 6161616161616161  CS: 616161616161  SS: 002b
bt: WARNING: possibly bogus exception frame
```

根据RSP: ffff8801fddabf20，读取stack的内存，根据内存内容规律推出代码位置。

```
crash> rd ffff8801fddabe00 100
ffff8801fddabe00:  0000000000000292 ffff8801fddabe38   ........8.......
ffff8801fddabe10:  ffffffff81007623 ffff8801e386c140   #v......@.......
ffff8801fddabe20:  ffff8801fddabe78 0000000000000000   x...............
ffff8801fddabe30:  00007f139d2f6000 ffff8801fddabe68   .`/.....h.......
ffff8801fddabe40:  ffffffff8138e0a9 ffffffff8156f610   ..8.......V.....
ffff8801fddabe50:  0000000000000001 ffffffffa087e000   ................
ffff8801fddabe60:  0000000000000000 20676e6974736574   ........testing
ffff8801fddabe70:  ffffffff8138d7cf 0000000000000002   ..8.............
ffff8801fddabe80:  00007f139d2f6000 0000000000000000   .`/.............
ffff8801fddabe90:  ffffffffa087e000 20676e6974736574   ........testing
ffff8801fddabea0:  0000000000614010 0000000000000000   .@a.............
ffff8801fddabeb0:  0000000000000000 0000000000000400   ................
ffff8801fddabec0:  0000000000000000 0000000000000000   ................
ffff8801fddabed0:  0000000000000000 0000000000000007   ................
ffff8801fddabee0:  ffffffffa0a4c117 ffff8801fddabfe7   ................
ffff8801fddabef0:  ffffffffffffffff ffffffffa087e03c   ........<.......
ffff8801fddabf00:  0000000000000010 0000000000010246   ........F.......
ffff8801fddabf10:  ffff8801fddabf20 0000000000000018    ...............
ffff8801fddabf20:  202c656c75646f6d 6161616161616161   module, aaaaaaaa
ffff8801fddabf30:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf40:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf50:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf60:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf70:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf80:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabf90:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabfa0:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabfb0:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabfc0:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabfd0:  6161616161616161 6161616161616161   aaaaaaaaaaaaaaaa
ffff8801fddabfe0:  0000616161616161 0000000000010202   aaaaaa..........
ffff8801fddabff0:  00007fff37c795f8 000000000000002b   ...7....+.......
```

踩页表问题

1. 编写代码模拟页表被踩。

```
#include <linux/kernel.h>
#include <linux/module.h>
#include <linux/kprobes.h>

#include <linux/kallsyms.h>
#include <linux/syscalls.h>
#include <linux/slab.h>
#include <linux/kdebug.h>
#include <asm/apic.h>
#include <asm/pgalloc.h>

static int __init jprobe_init(void)
{
        unsigned long address;
        pgd_t *pgd;
        pud_t *pud;
        pmd_t *pmd;

        printk(KERN_INFO "zk--- in \n");

        address = (unsigned long)kmalloc(512 ,GFP_KERNEL);
        pgd = pgd_offset(current->active_mm, address);
        pud = pud_offset(pgd, address);
        pmd = pmd_offset(pud, address);

        memset(pmd, 0, 16);

        printk("test: %x \n", *(int*)address);

        return 0;
}

static void __exit jprobe_exit(void)
{
        printk(KERN_INFO "zk--- out \n");
}

module_init(jprobe_init)
module_exit(jprobe_exit)
MODULE_LICENSE("GPL");
```

2. 分析方法：

   观察出错的堆栈，可以发现是一个地址错误。


```
crash> bt
PID: 13016  TASK: ffff88003595c080  CPU: 0   COMMAND: "insmod"
 #0 [ffff88008b0f7b10] machine_kexec at ffffffff81021a54
 #1 [ffff88008b0f7b60] crash_kexec at ffffffff8108baf8
 #2 [ffff88008b0f7c30] show_registers at ffffffff81006bf8
 #3 [ffff88008b0f7c90] __die at ffffffff81400f2e
 #4 [ffff88008b0f7cc0] no_context at ffffffff8102f733
 #5 [ffff88008b0f7d00] __bad_area_nosemaphore at ffffffff8102f925
 #6 [ffff88008b0f7dd0] bad_area_nosemaphore at ffffffff8102f9fe
 #7 [ffff88008b0f7de0] do_page_fault at ffffffff81402b0e
 #8 [ffff88008b0f7e30] page_fault at ffffffff8140029f
    [exception RIP: acct_collect+88]
    RIP: ffffffff8108af68  RSP: ffff88008b0f7ee8  RFLAGS: 00010286
    RAX: ffff8800658ba6b8  RBX: 0000000000002000  RCX: ffff88006b050480
    RDX: 0000000000000000  RSI: 0000000000000015  RDI: ffffffff81732a20
    RBP: ffff88008b0f7f08   R8: 0000000000000000   R9: fffffffffffffa8
    R10: 00007f7c07212700  R11: 0000000000000246  R12: ffff8800346ac400
    R13: 0000000000000000  R14: 0000000000000000  R15: 00007f7c071d0010
    ORIG_RAX: ffffffffffffffff  CS: 0010  SS: 0018
 #9 [ffff88008b0f7f10] do_exit at ffffffff81052b11
#10 [ffff88008b0f7f40] do_group_exit at ffffffff81052c0d
#11 [ffff88008b0f7f70] sys_exit_group at ffffffff81052c92
#12 [ffff88008b0f7f80] system_call_fastpath at ffffffff81002ffb
    RIP: 00007f7c06d2c2a8  RSP: 00007fffd5e227a8  RFLAGS: 00010246
    RAX: 00000000000000e7  RBX: ffffffff81002ffb  RCX: 0000000000000000
    RDX: 0000000000000000  RSI: 000000000000003c  RDI: 0000000000000000
    RBP: 0000000000000000   R8: 00000000000000e7   R9: fffffffffffffa8
    R10: 00007f7c07212700  R11: 0000000000000246  R12: ffffffff81052c92
    R13: ffff88008b0f7f78  R14: 00007fffd5e237d4  R15: 0000000000000001
    ORIG_RAX: 00000000000000e7  CS: 0033  SS: 002b
```

观察RIP: ffffffff8108af68的回报指令，该指令用到(RAX+0x10)的地址。

```
crash> dis ffffffff8108af68
0xffffffff8108af68 <acct_collect+88>:   add    0x10(%rax),%rbx
```

观察这个地址的页表，可以发现PMD指向了0，说明这个地址的页表有问题。

```
crash> vtop ffff8800658ba6c8
VIRTUAL           PHYSICAL
ffff8800658ba6c8  658ba6c8

PML4 DIRECTORY: ffffffff81a04000
PAGE DIRECTORY: 1a05063
   PUD: 1a05008 => 100067
   PMD: 100960 => 0

      PAGE        PHYSICAL      MAPPING       INDEX CNT FLAGS
fffffea0001962e80  658ba000                0        0  1 100000000000100
```

由此，初步分析出页表被踩后，可以通过被非法访问内存空间的内容，推出可能的模块；或者通过crash时间点的附近日志分析可疑模块。

踩静态变量问题

分析方法：

通过“sym -l”命令可以看到内核符号，先找到被踩静态变量的地址，然后观察其附近地址还有哪些静态变量。常见的场景是其前面的某个变量，操作过程中溢出，造成周围变量都被踩。


```
ffffffff81a16100 (d) vm_table
ffffffff81a16d80 (d) fs_table
ffffffff81a17320 (d) debug_table
ffffffff81a173c0 (d) min_sched_granularity_ns
ffffffff81a173c4 (d) max_sched_granularity_ns
ffffffff81a173c8 (d) max_wakeup_granularity_ns
ffffffff81a173cc (d) min_sched_shares_ratelimit
ffffffff81a173d0 (d) max_sched_shares_ratelimit
ffffffff81a173d4 (d) max_sched_tunable_scaling
```

硬件 DMA 非法访问内存空间问题

1. 问题现象

   堆栈没有任何规律，唯一的相同点就是code区域是一样的，而且数值是有规律的，全部是“fe 0b ad ca”。

   ```
   [ 3051.054204] Call Trace:
   [ 3051.057035]  [<ffffffff8115faa9>] path_openat+0xd9/0x420
   [ 3051.063054]  [<ffffffff8115ff2c>] do_filp_open+0x4c/0xc0
   [ 3051.069103]  [<ffffffff81150b71>] do_sys_open+0x171/0x1f0
   [ 3051.075223]  [<ffffffff8144fc53>] ia32_do_call+0x13/0x13
   [ 3051.081261] Code: fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad <ca> fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca fe 0b ad ca
   ```

2. 分析方法：

   由于内核的code区是只读的，通过线性地址无法去修改，因此可以得出造成这个修改的必然使用的是物理地址，硬件DMA有这个可能。但是目标地址的内容与硬件设备的DMA没有直接联系，除了硬件DMA之外，BIOS有直接操作物理内存的能力，需要分析BIOS代码。

   最终发现驱动代码错误，DMA过程踩到BIOS，造成BIOS运行过程踩到内核内存。

死锁问题

1. 分析方法

   死锁问题，关键点是当时每个CPU的堆栈是什么。 通过“bt -a”命令可以打印出系统当时所有CPU运行的堆栈，通过解析锁的数据结构，可以分析出哪个线程持有锁。

2. 常见死锁类型：
   - spinlock保护区中执行了非原子性的流程，如sleep schedule流程。
   - spinlock保护区中执行逻辑消耗的时间太长。
   - AB-BA死锁。
   - AA死锁，重复上锁。
   - 环形锁，在某些复杂架构设计中，模块间的等待可能出现环形，这会造成死锁。
   - 上锁和解锁不对称，锁指针运行中会被修改。

11 RAS

11.1 概述

本节介绍RAS的一些基础概念，让您了解RAS的作用。

背景描述

随着信息化技术的广泛应用，大型的数据中心、网络中心，如股票证券交易所，电信机房，银行的数据库中心等应用环境对IT的依赖程度日益加深。针对上述情况，如何确保整个系统尽可能长期可靠的正常运行而不下线，并且具备足够强大的容错机制，已经变得越来越重要，RAS机制已成为不可或缺的一部分。

RAS 简介

RAS是Reliability, Availability and Serviceability的缩写，即可靠性，可用性和可服务性。

RAS设计的核心指导理念就是“最大程度保证客户业务可持续正常运行”。换言之，RAS设计就是“尽量降低宕机的可能性”。高可用的单机系统，必须具有高可靠的底层设计（包括硬件和底层软件）、高容错性、快速修复的能力以及快速服务的能力。

RAS 的重要性

对于关键业务服务器来说，其核心要求就是系统具备提供不中断的持续服务的能力，其原因就在于在不同的应用中，服务器宕机导致业务中断带来的损失不完全相同，业务越关键，宕机所带来的损失越大。随着IT影响的不断快速深入，企业对IT系统的依赖程度日益加深，宕机成本正在变得越来越高。

RAS设计中，最基础的要求是保证器件应用的可靠性，即要求具有“硬件尽量不要出故障”的能力。

器件级的可靠性设计的方法，归根结底是两个基本要求：“使用正确的器件”和“正确的使用器件”。前者对器件选型和引入有较高的要求；后者要求更多从设计上考虑，比如降额应用等。器件级的可靠性质量保证包括供应商物料可靠性管理、产品可靠性设计、生产可靠性筛选三个环节，必须各司其职，相互配合，综合考虑。

11.2 约束限制

本节介绍使用RAS时的一些约束限制，避免用户使用时出现异常。

硬件约束

当前仅支持支持KunLun9016，9032服务器。

软件约束

- 操作系统版本：EulerOS V2.0SP1。
- 内核版本：3.10.0-229.20.1.30.hulk 及以上。

使用限制

- 热插拔会使大页（hugeTLB）的overcommit_memory参数不起作用。

  为了保证内存下线过程中大页迁移不会失败，大页迁移时，OS会跳过overcommit_memory这个判断，即使用户分配的大页不足，OS在下线过程中的内存迁移时也能分配超过阈值的大页。执行过一次节点下线（节点或单独的内存）后，overcommit_memory参数会失效，用户不能通过该接口限制大页的分配。
- 热插拔会破坏内存和node的绑定策略。mbind函数通过参数nodemask限制分配节点。执行过一次节点下线（节点或单独的内存）后，该功能失效。

11.3 热插拔说明

前提条件

- 版本：EulerOS V2.0SP1。
- 内核：3.10.0-229.20.1.30.hulk 及以上。

硬件依赖

支持KunLun9016，9032服务器。

相关配置

- 启动参数配置：配置movable_node参数，开启支持热插拔功能。

  修改“/boot/grub2/grub.cfg”（EFI系统的配置文件在“/boot/efi/EFI/euleros/grub.cfg”），在内核启动选项中增加movable_node numa_zonelist_order=zone。
- 支持内存镜像（address range mirroring），在3.10.0-229.20.1.30.内核版本及其之后的版本已默认支持，早期版本需要添加mirrorable启动参数。

OS 处理热插拔的原则

- 热插拔要保证业务不能中断。即不能因为内存下线触发OOM而导致业务中断，此时热插拔没有意义。
- 热插拔要满足多数典型场景（如oracle场景），确保每种场景下都能热插拔成功。
- 实际应用热插拔时，多数情况是硬件故障或者硬件可能有问题的场景，需要更换硬件。

热插拔须知

- 开启热插拔支持，会将OS内核限定在不可热插拔的（unmovable）node上运行。这样会带来内核跨节点访存的增加，可能导致性能下降。
- 热插拔会破坏内存和node的绑定策略。

  比如，某些应用绑到node1上，如果node1下线了，则绑定关系由bind退化为prefer。即如果该节点内存不足，就会去其他节点分配内存，以防止因为内存下线，而将绑到node1上的进程kill掉。

- 热插拔会使大页（hugeTLB）的overcommit_memory参数不起作用。

  为了保证内存下线过程中大页（HUGETLB）迁移不会失败，大页迁移时OS会跳过overcommit_memory这个判断。即如果用户分配的大页不足，OS会在下线过程中的内存迁移时分配超过阈值的大页，以防止在下线过程中出现因分配不到大页而引起的OOM。

- 热插拔时，系统负载不宜过大。

  以16P为例，内存使用率（memused/（total - mem_tobe_removed））一般应在60%以下。内存使用率高，会使内存迁移时间延长，而且迁移失败概率增大。

- 热插拔过程中，下线有可能会失败。

  应用程序在运行过程中，可能会因锁住内存，或过于频繁访问文件而引起filemap_fault，从而导致下线失败。下线失败不是错误，可以多次尝试重新下线。

  下线失败，可以通过nodectl工具查看失败原因。该工具可以查看哪些进程在使用该节点内存。如查看哪些进程在使用node1的内存，代码如下：

```
[root@localhost nodectl]# nodectl --procs 1
This following process(es) have memory page(s) allocated on node 1
9757,           watch: pages = 119 (476 kb)
169177,         nodectl: pages = 27 (108 kb)
173780,         nodectl: pages = 3 (12 kb)

# 加 -v 参数, 可以查看详情
[root@localhost nodectl]# nodectl --procs -v 1
This following process(es) have memory page(s) allocated on node 1
[  3690,        libvirtd] 7f9c7c000000 default anon=287 dirty=286 swapcache=1 active=282
N1=1 N4=1 N6=1 N7=284
[  3690,        libvirtd] 7f9c863d2000 default file=/usr/lib64/libvirt/connection-driver/
libvirt_driver_nodedev.so anon=1 dirty=1 N1=1
[  3690,        libvirtd] 7f9c9943f000 default file=/usr/lib64/libc-2.17.so anon=2 dirty=2
N0=1 N1=1
[  3690,        libvirtd] 7f9c9e3e4000 default heap anon=20 dirty=20 active=19 N0=12 N1=3
N4=2 N7=3
[  9757,           watch] 00e27000 default heap anon=437 dirty=437 N1=106 N2=1 N3=148 N4=8
N6=170 N7=4
[  9757,           watch] 7fc80568b000 default file=/usr/lib64/libc-2.17.so anon=2 dirty=2
N1=1 N3=1
[  9757,           watch] 7fc805abf000 default file=/usr/lib64/libtinfo.so.5.9 anon=1
dirty=1 N1=1
[  9757,           watch] 7ffc546fb000 default stack anon=3 dirty=3 N1=1 N3=2
[ 32838,   systemd-udevd] 7f5df58a0000 default heap anon=78 dirty=69 mapmax=5 swapcache=9
active=69 N1=1 N5=24 N6=19 N7=34
```



```
[ 32838,    systemd-udevd] 7ffd97b72000 default stack anon=14 dirty=14 mapmax=5 N1=2 N4=1
N5=7 N7=4
```

11.4 内存镜像说明

背景

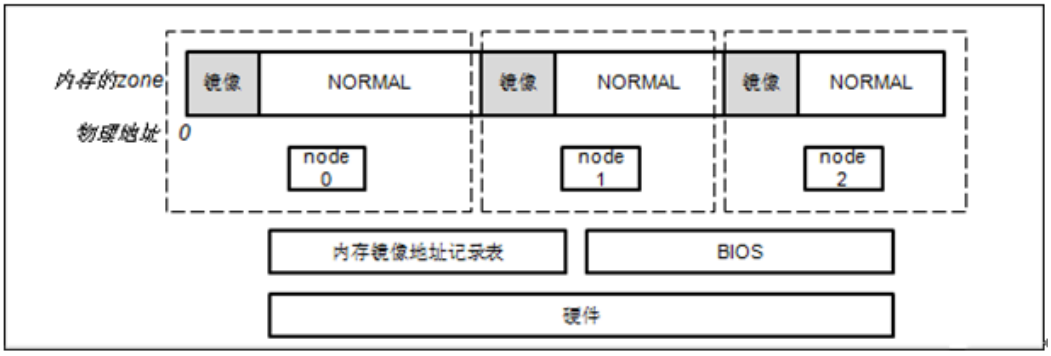
目前大多高端服务器都支持内存镜像特性，其原理是把整个系统的一半内存作为另一半内存的镜像，在主内存数据发生错误时自动从镜像内存中读取，从而提高系统的可靠性。但现有技术的缺点是整个系统有一半的内存OS无法使用，浪费严重。

目前较新的CPU（如Intel Xeon系列）已支持分段内存镜像技术（即部分内存镜像特性），它可以人为指定某块区域做镜像，然后把重要的数据放在镜像内存区，那些不重要的数据所占的内存区域或者是空闲内存区域不做镜像，这样一来就能在保证高可靠性的同时，减少内存浪费。但这一新特性目前只有硬件层面支持，而软件层面并未支持该特性。

原理简介

部分内存镜像特性如图1所示。

图 11-1 部分内存镜像特性示意图



- 如图1所示，一个具有三个node的NUMA系统，每个node都有一段物理地址做内存镜像，OS访问时并无区别，即OS在读写这段内存时和其他区域一样。硬件层面会做镜像处理，如果发生ECC等内存错误，由硬件自动纠正。简单来说就是OS实现了一个能在指定物理地址分配内存的特性。
- 内存镜像地址条目由启动时BIOS的SRAT表提供，OS在初始化时建立好相关的结构，并提供全局的是否使用镜像内存的控制接口，供管理员配置。并且OS提供单个进程是否使用镜像内存的接口和用户态可编程接口，以供用户更精细化的使用镜像内存。

场景描述

部分内存镜像特性作为一个非常重要的RAS新特性，在不过分减少可用内存的前提下，有效提高了计算机系统的可靠性。该特性可集成到高端服务器中，部署在一些关键业务行业，如银行交易中心、军工科研等涉及高可靠性计算机需求的领域。

本特性应用于配置部分内存镜像的计算机系统。主要通过修改操作系统内存分配策略，使操作系统内核数据、以及关键用户进程数据优先使用可靠性更高的镜像内存。通过保护操作系统内核数据和关键用户进程数据，来提高系统整体可靠性。

说明

- 默认所有内核态数据都是关键数据。用户可根据业务的重要性来制定某些用户进程的内存分配全部使用内存镜像，或者使用API接口部分使用内存镜像；
- 用户态进程若要使用镜像内存需要用户指定；若不指定，默认先分配普通内存。

规格限制

1. 计算机系统硬件平台（如cpu）必须支持部分内存镜像特性。
2. 部分内存镜像只能在系统启动时配置，无法在系统起来后动态配置，如改变镜像区域大小。
3. 目前只支持华为9032系列服务器。
4. 镜像内存最小支持64M，内存64M对齐，软件没做最大限制。
5. 地址范围0~4G的内存，软件不做任何特殊处理，建议硬件默认设置为镜像内存。
6. 建议预留全部物理内存的1/64以上的容量，作为镜像内存。
7. 建议合理设置内存镜像的大小：不要覆盖整个node的内存大小，也不要没有设置镜像内存但OS开启使用标识，导致反复fallback性能下降。

影响

1. 配置了内存镜像，系统内存容量减小了。
2. 镜像内存部分的可靠性相对较高。
3. 硬件如果不支持内存镜像，则无影响。
4. 与热插拔特性兼容，即movable node的镜像内存只分配用户态的。
5. 在镜像内存不足，而又开启默认使用内存镜像时，可能会在内存分配时反复fallback而导致性能有所降低。
6. 在配置了镜像内存，但又不开启使用镜像内存的标识时，系统只会从普通内存分配，即此部分镜像内存相当于reserved。

12 Docker

12.1 简介

虚拟化技术使应用在各自的空间内运行相互不影响，为人们提供了极大的便利，同时减少了对实体服务器的需求，降低了成本。目前虚拟化技术包括KVM、XEN、VMWare等，全虚拟化技术或半虚拟化技术存在对主机开销大的缺点，而Linux container技术是通过与主机共用内核，结合内核的cgroup和namespace实现的一种虚拟化技术，极大的减少了对主机资源的占用而且具有较快的启动速度。docker就是一个Linux container引擎技术，实现应用的打包、快速部署等。

Docker的英文本意是码头工人，码头工人的工作就是将商品打包到Container（集装箱）并且搬运Container、装载Container。从docker字面上的解释就可以看出docker是干什么的，对应到Linux中，docker就是将App打包到Container，通过Container实现App在各种平台上的部署，运行。Docker通过Linux Container技术将App变成一个标准化的、可移植的、自管理的组件，实现了应用的build once、run everywhere。Docker技术特点是：应用快速发布、应用部署和扩容简单、更高的应用密度、应用管理更简单。

12.2 安装 Docker

安装

对于EulerOS，docker rpm的安装包在iso或者repo中，可以通过配置yum源为公共仓库地址的repo文件，使用yum来安装。

1. 创建yum源，在/etc/yum.repos.d/目录下配置EulerOS-base.repo文件，添加如下信息：

```
[base]
name=EulerOS-base
baseurl=http://developer.huawei.com/ict/site-euleros/euleros/
repo/yum/2.1/os/x86_64/
```



```
enabled=1
gpgcheck=1
gpgkey=http://developer.huawei.com/ict/site-euleros/euleros/
repo/yum/2.1/os/RPM-GPG-KEY-EulerOS
```

2. 安装

```
yum install -y docker-engine
```

3. 启动

说明

docker默认状态是开启的。

启动时的参数（默认只有daemon），可以通过下面的命令查看：

```
root@euler:~/workspace# ps -eaf|grep docker
root295210 16:55 ?00:00:00 /usr/bin/docker daemon -H fd://
root334212130 17:11 pts/000:00:00 grep docker
```

卸载

```
yum remove docker-engine
```

12.3 配置 Docker

Docker服务可以通过手动输入命令启动，它的配置就是传给docker的命令行参数，也可以通过服务的方式来管理，这时的配置是通过配置文件来完成的。

- 通过命令行启动

```
docker daemon -D #命令行启动，-D打开调试,若要停止，kill掉docker进程
```

- 通过systemd服务启动

```
systemctl start docker   #服务的方式来启动，停止为stop
```

用户可以通过修改 /usr/lib/systemd /system/docker.service 文件来修改docker服务的启动参数。

```
# vim /usr/lib/systemd/system/docker.service
[Unit]
Description=Docker Application Container Engine
Documentation=https://docs.docker.com
After=network.target

[Service]
Type=notify
EnvironmentFile=-/etc/sysconfig/docker
EnvironmentFile=-/etc/sysconfig/docker-storage
EnvironmentFile=-/etc/sysconfig/docker-network
Environment=GOTRACEBACK=crash
# the default is not to use systemd for cgroups because the delegate issues still
# exists and systemd currently does not support the cgroup feature set required
# for containers run by docker
ExecStart=/usr/bin/docker daemon $OPTIONS \
          $DOCKER_STORAGE_OPTIONS \
          $DOCKER_NETWORK_OPTIONS \
          $ADD_REGISTRY \
          $BLOCK_REGISTRY \
          $INSECURE_REGISTRY
LimitNOFILE=1048576
LimitNPROC=1048576
LimitCORE=infinity
TimeoutStartSec=0
# set delegate yes so that systemd does not reset the cgroups of docker containers
Delegate=yes
# kill only the docker process, not all processes in the cgroup
KillMode=process
```



```
[Install]
WantedBy=multi-user.target
```

说明

1. 通过“systemctl restart docker”重启服务使配置生效。
2. EnvironmentFile=-/etc/sysconfig/docker，定义环境变量所在的文件。该文件中定义的环境变量可以在/usr/lib/systemd /system/docker.service文件中进行引用。
3. docker服务程序的启动参数需要加到“ExecStart”的后面。

- 配置docker的外网代理

  若要从docker hub上pull镜像，还需要设置代理。

  a. 在/usr/lib/systemd/system/docker.service中添加内容如下：
  ```
  Environment="HTTP_PROXY=http://count:passwd@proxyhk.huawei.com:8080"
  ```
  b. 重启daemon使配置生效
  ```
  systemctl restart docker
  ```

12.4 使用 Docker

约束限制

docker使用crgoup kmem限制：EulerOS 2.2及以下的版本，内核不支持cgroup kmem功能。

使用方法

docker安装完成后，会自动启动，若是直接使用docker的可执行程序，可以使用docker daemon &启动docker的后台进程。docker后台进程的启动需要root权限。这一章介绍docker中一些简单应用，使读者对docker有一个初步了解和印象，对于命令的使用不作详细解释，对于docker更多的使用，请参考docker --help。

- 下载镜像
  ```
  docker pull ubuntu
  ```
  上面的命令将会从dockerhub（需要连接到外网）上去下载ubuntu镜像

  若是有自己的hub，也可以用下面的方式下载：
  ```
  docker daemon -D  --insecure-registry=rnd-dockerhub.huawei.com
  docker pull rnd-dockerhub.huawei.com/library/ubuntu
  ```

  说明

  上面的rnd-dockerhub.huawei.com是自己的hub地址，library是hub上用户的账号。

- 启动容器
  ```
  docker run -ti ubuntu bash
  ```
- 关于docker的使用，可以使用--help获取当前几乎所有关于docker的使用
  ```
  docker  --help        #显示docker子命令
  docker cmd --help     #cmd为docker了命令，如run，显示docker cmd 的详细信息
  docker daemon --help  #显示docker daemon的详细信息
  ```

13 FAQ

13.1 rpm 安装冲突问题解决方法

现象描述

某些 rpm 包同时被安装的时候会报冲突。以下几种场景会导致冲突：

- 部分 x86_64 与 i686 包冲突；
- 提供了类似功能的包；

- rpm 的 spec 定义不能兼容的包。

解决方法

rpm/yum 安装的时候会提示冲突，用户需要根据提示，删除掉冲突的 rpm 包，如果冲突的只是文档文件，也可以强制安装。

- eg1：

```
Transaction check error:
file /usr/share/doc/dracut-033/dracut.html from install of dracut-033-360.h1.i686 conflicts
with file from package dracut-033-360.h1.x86_64
```

只能选择 x86_64 或 i686 一个包安装，或者强制安装。

```
rpm  -ivh  xxx.rpm --force
```

- eg2：

```
Error: tog-pegasus-libs conflicts with libcmpiCppImpl0-2.0.3-5.x86_64
```

先卸载掉冲突的包：

```
yum remove libcmpiCppImpl0
```

13.2 调整中断负载均衡策略

中断均衡优化简介

Host os自带的irqbalance可以动态（隔一定时间间隔）平衡多个cpu上面的中断数量，来做到全部中断能均衡绑定到各个cpu，但是它不能完全解决CPU中断负载绝对均衡，在虚拟化环境下就会影响vcpu性能，具体就是如果各个vcpu对应的cpu的中断负载不同，那么vcpu的计算性能就会表现较大差异，最终导致虚拟机看到的cpu计算性能不同，会影响虚拟机内部的业务。

需要在原有irqbalance功能的基础上做特性增强，保证中断负载在各个cpu上动态平均。

应用场景

- 在中断负载不均衡时，例如某个具有多队列（有多个中断号）功能的网卡的所有中断全部让cpu0处理，其它cpu空闲，增强irqbalance会将所有中断分散配置到各个cpu，并按照一定时间间隔进行调整，以保证在一段时间内，所有cpu的中断负载均值基本相等。
- 在某些应用场景中，希望能够实现对某些cpu和中断的隔离功能，这就需要将对应的中断号和cpu加到irqbalance的黑名单中，实现cpu和中断对irqbalance隐藏，脱离irqbalance的管理。

注意事项

- irqbalance特性增强功能在服务配置文件（/etc/sysconfig/irqbalance）中提供了开关，如果要关闭，请设置IRQBALANCE_ABSOLUTE_BALANCE=disable，设置生效必须重启irqbalance服务，此时irqbalance服务只具备原生功能，建议在版本中保持默认开启。
- irqbalance提供了中断黑名单、cpu黑名单功能，如果不期望某个中断或cpu被irqbalance管理，可以使用下边的接口将中断号或cpu加到黑名单中。

  说明

  - CPU黑名单功能，设置需要提供系统单板中存在的cpu。
  - 中断黑名单功能，需配置当前系统中存在的中断，并且黑名单中中断数量最大支持4096个中断。

- 如果要irqbalance黑名单的设置生效，必须重启irqbalance服务。
- 对于服务配置文件读写和重启服务操作都需要root用户权限，所以需要保证调用接口的程序具有root执行权限。
- 中断需要与CPU的中断向量绑定后才能设置中断亲和性，一个CPU的中断向量表共有256项，功能详见下表，可以用于绑定通用外设的为32~47、64~127、129~203和205~238，共189个。当系统中断过多时，如果将全部的中断绑定到一个CPU上时，会因为中断向量不足而导致设置中断亲和性失败，如果中断多为msi/msix中断（通过cat /var/log/dmesg | grep -w irq命令查看），可以多分配几个CPU用于绑定中断，因为根据msi机制，不同的中断在不同的CPU上，可以申请相同的中断向量，这样就解决了单一CPU上中断向量不足的问题。

表 13-1

中断向量号	用途
0~31	系统预留中断向量
32~47	可以用于绑定通用外设中断
48~63	通常绑定ISA外设中断
64~127	可以用于绑定通用外设中断
128	系统调用
129~203	可以用于绑定通用外设中断
204	系统预留
205~238	可以用于绑定通用外设中断
239	时钟中断
240~255	为SMP系统预留中断向量

互斥特性

- 与迁移特性互斥

  包括虚拟机的冷、热迁移，整机的冷、热迁移。

  黑名单不会随热迁移迁移到目的端，而且目的端对应的中断号可能与源端不同，所以需要上层重新调用相关接口进行设置。

涉及 API

- addCpuBanned：将一系列CPU添加到irqbalance服务的CPU黑名单中。
- delCpuBanned：将一系列CPU从irqbalance的CPU黑名单中删除。
- getCpuBanned：获取当前的irqbalance CPU黑名单列表。
- addIRQBanned：将一系列中断号添加到irqbalance服务的中断黑名单中。
- delIRQBanned：将一系列中断号从irqbalance服务的中断黑名单中删除。
- getIRQBanned：获取当前的irqbalance中断黑名单列表。
- setIRQBind：将中断绑定到指定的CPU范围内。

示例代码

1. 添加CPU到黑名单示例。

```
import libuvp_compute, sys
import os
try:
   libuvp_compute.addCpuBanned("1,4-5,12-15")
   os.system('systemctl restart irqbalance > /dev/null')
except:
    print 'addCpuBanned failed.'
```

2. 从CPU黑名单移除cpu示例。

```
import libuvp_compute, sys
import os
try:
   libuvp_compute.delCpuBanned("1,4-5")
   os.system('systemctl restart irqbalance > /dev/null')
except:
    print 'delCpuBanned failed.'
```

3. 获取CPU黑名单列表示例。

```
import libuvp_compute, sys
try:
   cpubanned = libuvp_compute.getCpuBanned(128)
   print cpubanned
except:
    print 'getCpuBanned failed.'
```

4. 添加中断到黑名单示例。

```
import libuvp_compute, sys
import os
try:
   libuvp_compute.addIRQBanned("1,4,8,9")
   os.system('systemctl restart irqbalance > /dev/null')
except:
    print 'addIRQBanned failed.'
```

5. 从黑名单中移除中断示例。

```
import libuvp_compute, sys
import os
try:
   libuvp_compute.delIRQBanned("1,8,9")
   os.system('systemctl restart irqbalance > /dev/null')
except:
    print 'delIRQBanned failed.'
```

6. 获取中断黑名单列表示例。

```
import libuvp_compute, sys
try:
   irqbanned = libuvp_compute.getIRQBanned(128)
   print irqbanned
except:
    print 'getIRQBanned failed.'
```

7. 将中断绑定到指定的CPU范围内

```
import libuvp_compute, sys
try:
    libuvp_compute.setIRQBind("4,8","1-5");
except:
    print 'setIRQBind failed.'
```

说明

- 这个接口主要应用在CPU和中断隔离场景中，在cpu和中断隔离场景中，会首先使用上述addCpuBanned和addIRQBanned接口，将CPU和中断加入到irqbalance的黑名单中，然后调用setIRQBind将隔离出来的中断绑定到隔离出来的cpu中。
- setIRQBind接口是通过修改/proc/irq/XX/smp_affinity_list文件来实现中断绑定的，绑定关系在Host重启后会失效，因此需要在Host重启后重新调用此接口进行中断绑定。

13.3 通过 NetworkManager 给 bond 设置 mac 地址不生效

问题现象

编写如下配置文件ifcfg-bond0，指定NM_CONTROLLED=yes，通过NetworkManager给bond设置mac地址。

```
# /etc/sysconfig/network-scripts/ifcfg-bond0
MACADDR=1A:2B:3C:2A:2C:1d
USERCTL=no
BONDING_MASTER=yes
ONBOOT=yes
NM_CONTROLLED=yes
BOOTPROTO=dhcp
BONDING_OPTS="mode=1 miimon=100"
DEVICE=bond0
TYPE=Bond
```

重启network后，ifconfig查看，发现bond实际地址与配置中不一致，mac的设置并未生效。

原因分析

EulerOS 2.0 SP1中NetworkManager不支持给bond设置mac地址。

规避方案

- 系统升级至EulerOS 2.0 SP2及以上版本。
- 不通过NetworkManager设置mac地址，配置文件中指定NM_CONTROLLED=no，或者关闭NetworkManager。

13.4 文件类型和文件名

在Linux系统中，文件命名通常与它的文件类型相关，下面列出了常见文件名和它们所属文件类型的对应关系。

表 13-2 压缩文件

文件名	说明
.bz2	使用bzip2程序压缩的文件。
.gz	使用gzip程序压缩的文件。
.tar	使用tar程序压缩的文件，又称tar文件。
.tbz	用tar和bzip压缩的文件。
.tgz	用tar和gzip压缩的文件。
.zip	使用ZIP压缩的文件，常见于MS-DOS应用程序中。

表 13-3 系统文件

文件名	说明
.conf	一种配置文件，有时也用.cfg。
.lock	锁（lock）文件，用来判定程序或设备是否正在被使用。
.rpm	软件包格式文件。

表 13-4 编程和脚本文件

文件名	说明
.c	C语言的源码文件
.cpp	C++语言的源码文件
.h	C或C++语言的头文件
.o	程序的对象文件
.pl	Perl脚本
.py	Python脚本
.so	库文件
.sh	shell脚本
.tcl	TCL脚本

13.5 常用目录说明

通过对系统目录组织结构的了解，可以在进行文件操作和系统管理时方便地知道所要的东西在什么地方。

文件系统采用分层的树形目录结构。即在一个根目录（通常用“/”表示）中，包含多个下级子目录或文件；子目录中又可含有更下级的子目录或文件信息，这样逐层地延伸下去，构成一棵倒置的树。树中的“根”与“权”代表的是目录或称为文件夹，而“叶子”则是每个文件。

下面列出了主要的系统目录及其简单描述：

- /bin：存放普通用户可以使用的命令文件。目录/usr/bin也可用来贮存用户命令。
- /sbin：一般存放非普通用户使用的命令（有时普通用户也可能会用到）。目录/usr/sbin中也包括了许多系统命令。
- /etc：系统的配置文件。
- /root：系统管理员（root或管理员账户）的主目录。
- /usr：包括与系统用户直接相关的文件和目录，一些主要应用程序也保存在该目录下。
- /home：用户主目录的位置，保存了用户文件（用户自己的配置文件、文档、数据等）。

- /dev：设备文件。在Linux中设备以文件形式表现，从而可以按照操作文件的方式简便地对设备进行操作。
- /media：文件系统挂载点。一般用于安装移动介质、其他文件系统（如DOS）的分区、网络共享文件系统或任何可安装文件系统。
- /lib：包含许多由/bin和/sbin中的程序使用的共享库文件。目录/usr/lib/中含有更多用于用户程序的库文件。
- /boot：包括内核和其他系统启动时使用的文件。
- /var：包含一些经常改变的文件。例如假脱机（spool）目录、文件日志目录、锁文件、临时文件等。
- /proc：操作系统的内存映像文件系统，是一个虚拟的文件系统（没有占用磁盘空间）。当您查看它们时，看到的是内存里的信息，这些文件有助于了解系统内部信息。
- /initrd：在计算机启动时挂载initrd.img映像文件的目录以及载入所需设备模块的目录。
- /opt：存放可选择安装的文件和程序。主要由第三方开发者用于安装和卸装软件包。
- /tmp：用户和程序的临时目录，该目录中的文件被系统自动清空。
- /lost+found：在系统修复过程中恢复的文件。

13.6 NetworkManager 内存占用说明

问题现象

使用ip命令批量创建大量虚拟网卡设备并删除，NetworkManager的内存占用不回落，再次创建同等数量的虚拟网络设备不上涨。

原因分析

NetworkManager大量使用g_ptr_array，g_array以及ghash等glib2动态库的数据结构来管理设备信息。当大量地并发执行创建、删除、修改设备时会产生内存碎片，这种内存占用不是内存泄露，是NetworkManager设计上的缺陷。

对于不同的虚拟网卡，以1000个数量为例，通过批量添加，配置ip，路由，设置网卡信息等操作，内存占用情况如下：

虚拟网卡类型	数量	内存占用
veth	1000	192M
vlan	1000	128M
dummy	1000	192M
macvlan	1000	128M
bridge	1000	192M
bond	1000	192M

解决方法

可以根据系统实际网卡情况来监控NetworkManager的内存占用。如果所有虚拟网卡被删除而NetworkManager内存不回落，可通过重启NetworkManager服务释放内存占用。

重启服务命令：

```
systemctl restart NetworkManager
```

13.7 逻辑卷创建后被自动挂载且挂载点不可用现象说明

问题现象

当使用lvcreate命令创建一个新的逻辑卷后，使用"df -h"查看挂载情况：

```
[root@localhost lvtest]# df -h
Filesystem                Size  Used Avail Use% Mounted on
/dev/mapper/euleros-root   13G  2.5G  9.7G  21% /
devtmpfs                  1.9G     0  1.9G   0% /dev
tmpfs                     1.9G     0  1.9G   0% /dev/shm
tmpfs                     1.9G   41M  1.9G   3% /run
tmpfs                     1.9G     0  1.9G   0% /sys/fs/cgroup
/dev/sda1                 477M   98M  351M  22% /boot
tmpfs                     378M     0  378M   0% /run/user/0
/dev/mapper/vgtest-lvtest  64Z   64Z  958M 100% /home/lvtest
[root@localhost lvtest]#
```

发现新建的逻辑卷被自动挂载了，但逻辑卷的size变得无限大，并且挂载点无法写入。

原因分析

引起该问题需要以下四个要素：

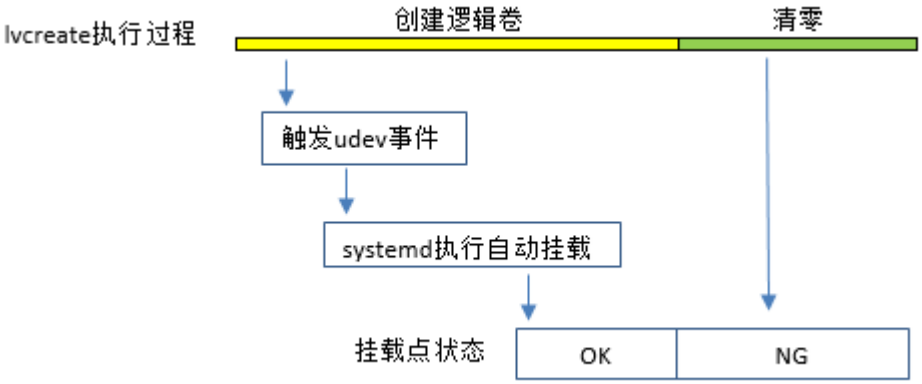
1. 在/etc/fstab中配置了逻辑卷自动挂载项，且已生效。
2. 逻辑卷的文件系统为ext文件系统（fat文件系统已验证不存在此问题，其他文件系统未验证）。
3. udev相关的服务开启：

   systemd-udevd

   systemd-udevd-control.socket

   systemd-udevd-kernel.socket
4. 在同一个卷组中，对名为xxx的逻辑卷执行“创建-删除-再创建”操作（再创建的逻辑卷的大小不能小于前者）。

当删除逻辑卷时，原逻辑卷中文件系统相关的信息仍保留在磁盘上。在磁盘的相同位置新建一个同名同大小的逻辑卷时，该文件系统的信息会被新的逻辑卷复用。lvcreate执行过程如下：



由于lvcreate执行清零的操作在文件系统挂载操作之后，已经挂载的文件系统被清零操作破坏，从而导致了"df -h"显示异常，挂载点不可用。

解决方法

在删除逻辑卷之前，先将逻辑卷中的文件系统系统信息清除。以删除/dev/vgtest/lvtest逻辑卷为例：

```
#dd if=/dev/zero of=/dev/vgtest/lvtest bs=1M count=32
#lvremove /dev/vgtest/lvtest
```

13.8 openLDAP 服务压力规格说明

在客户端使用SSSD的场景下，openLDAP 服务端短时间内收到一个客户端的10000次查询请求，所需CPU资源大约在60%～80%。

当openLDAP 服务端进程可用的CPU资源在20%以下时，SSSD客户端有极大概率会出现请求超时。

由于openLDAP服务端所处的环境存在不确定性，建议将SSSD客户端的ldap_search_timeout配置项设置为10s，以保证在openLDAP服务端所处OS压力过大，其服务进程能使用的CPU资源较小的情况下，SSSD也能正常工作（减小超时的可能性）。

此外，大量的IO读写也会拖慢服务端的响应速度（openLDAP服务需要读取磁盘上的数据库），大量写日志既会产生大量IO也会使得journal服务的CPU使用率极大上升抢占IO和CPU资源，建议服务端的日志级别配置项olcLogLevel不要小于32（数值越小日志量越大）。

13.9 极端情况下 nslcd 服务 OOM 导致 host 解析失败的现象说明

nslcd在高并发（1000个并发）时内存使用率会持续增加，当系统内存使用率达到一定负荷时，nslcd服务有可能发生OOM进而导致进程终止（panic_on_oom=0时），此后nslcd服务进入失败状态。经过一段时间（默认一小时）后，nscd缓存中的host记录开始过期，此时ping命令在解析域名对应的IP地址时由于缓存已过期且nslcd服务已经失败，导致无法从本地缓存和openLDAP服务端获取任何信息，最终提示“unknown host”。建议根据系统内存大小合理限制并发数量，避免内存耗尽。

13.10 使用 authconfig 命令修改 PAM 配置失败

问题现象

使用authconfig命令，针对PAM配置进行修改，修改失败 。

原因分析

EulerOS针对PAM的配置，默认进行了安全加固，为了防止加固配置被覆盖，用户通过authconfig命令自动配置无法直接生效。

解决方法

如果用户要在现有PAM配置下进行变更，有以下两种方法。

- 方法一：使用authconfig命令进行配置，命令执行之后，需要将软连接/etc/pam.d/system-auth和/etc/pam.d/password-auth指向/etc/pam.d/system-auth-ac和/etc/pam.d/password-auth-ac文件。
- 方法二：手动配置/etc/pam.d/system-auth-local和/etc/pam.d/password-auth-local文件。

13.11 scp 传送大文件速度逐渐变为 0，传送缓慢问题的解决方法

问题现象

当出现这三个条件共存的情况下会出现scp速度变为0问题。

1. 设置防火墙规则过滤掉sack-permitted、sack 报文头信息：
```
[root@EulerOS-BaseTemplate ~]# iptables -L -t mangle
......
Chain INPUT (policy ACCEPT)
target prot opt source destination
TCPOPTSTRIP tcp -- anywhere anywhere TCPOPTSTRIP options mss,sack-
permitted,sack,timestamp,md5
......
```
2. 设置 tcp_sack = 1
```
[root@EulerOS-BaseTemplate ~]# sysctl -a | grep tcp_sack
net.ipv4.tcp_sack = 1
```
3. 在链路上发生丢包。

原因分析

- net.ipv4.tcp_sack与防火墙规则冲突导致。
- tcp_sack 设置快速重传，但是防火墙规则把sack需要的信息给strip掉了，导致重传的数据失效，只能慢速传输，速度到最低。
- 在不发生丢包的情况下，没有触发重传场景，则不复现该问题。

解决方法

有两种修改方法，解决冲突问题：

- 打开sack功能，把防火墙过滤sack-permitted,sack规则去掉，在丢包场景下tcp性能较好。
- 关闭sack功能，设置net.ipv4.tcp_sack = 0，在丢包乱序场景下tcp性能较差。

13.12 nfs 的客户端上 mount 相关进程卡住

问题现象

使用nfs时，mount默认采用hard模式挂载。在mount成功以后，服务端发生重启。但在服务端重启完成之后，客户端在访问之前挂载的目录时，仍然卡住。

原因分析

服务端重启之前配置了多个ip地址，客户端采用hard模式挂载了多个ip的共享目录。在服务端重启完成之后，部分ip没有恢复配置。导致客户端在访问原先挂载的共享目录时，访问失败，nfs request会一直重试，导致客户端mount相关进程卡住。

解决方法

在使用hard模式挂载时，如果客户端出现mount相关进程卡住，需要排查服务端的网络、nfs服务、rpcbind服务等是否已经恢复正常。在服务端恢复正常以后，客户端进程即可自动恢复。 另外，建议在使用hard模式时增加intr选项来硬挂载目录，这时当有某个进程进入了重试循环，则允许用户使用键盘将其中断。

说明

1. mount命令在挂载时默认采用hard模式。
2. 在nfs客户端采用hard模式挂载时，可以保证客户端和服务端数据的一致性，不会出现静默数据错误。但当服务端出现异常的时候，客户端会一直向服务端发出请求，直到服务端恢复正常，进而导致客户端进程一直阻塞。
3. 在nfs客户端采用soft模式挂载时，可以通过timeo和retry参数配置超时时间。服务端出现异常时，客户端会向服务器端重发请求。当超过配置的超时时间时，则返回错误，不会一直阻塞。但在soft模式下可能会出现静默数据错误。

13.13 EulerOS 2.1 及 EulerOS 2.2 上 logrotate 配置文件中 su user group 缺省配置时的执行结果差异分析

问题现象

EulerOS 2.2环境上，使用如下配置切割文件报错：

```
/var/log/console/nginx/access.log {
su root  #此处不设置group
create 0600 root root
compress
maxage 90
rotate 1000
size 50M
notifempty
```

```
missingok
copytruncate
postrotate
chmod 600 /var/log/console/nginx/access.log
endscript
}
```

执行结果例：

```
[root@localhost ~]# logrotate -f /root/test.conf
error: error switching euid to 0 and egid to -1: Invalid argument
```

同样的配置在EulerOS 2.1即可执行成功，无上述报错，日志被正常切割转储 。

原因分析

对比两个版本上logrotate的执行流程，EulerOS 2.2上logrotate 版本比EulerOS 2.1高，社区对su [user] [group] 配置项的处理做了优化， user id 及group id默认初始化为-1。

- EulerOS 2.1中，logrotate版本比较低，配置项的处理上对于group id 的默认缺省没有做限制，所以使用缺省配置不会出现报错。
- EulerOS 2.2及以上的版本该值缺省配置时，就出现“error: error switching euid to 0 and egid to -1: Invalid argument”的报错。

解决方法

- EulerOS 2.1的环境上虽然可以配置成功，但建议用户按照man手册要求进行规范配置。

  man手册配置规范如下：

  ```
  su user group
        Rotate log files set under this user and group instead of using default user/group
  (usually root). user specifies the user name used for rotation and group specifies the group
  used for rotation.
  ```

- EulerOS 2.2及以上的版本出现上述报错时，是高版本logrotate的处理机制。出现上述报错请按照man手册中的规范进行配置。

13.14 如何解决配置 2G 内存虚拟机热插内存到 128G 后重启卡住的问题

问题现象

UVP KVM的企业虚拟化场景，使用EulerOS（64位）作为虚拟机操作系统，虚拟机配置2G内存启动后，分别依次热插2G、4G、8G、16G、32G、64G内存且依次上线热插的内存，直到虚拟机总内存为128G时重启虚拟机，虚拟机卡死，VNC或者串口日志有如下消息输出：

```
page      allocation failure
```

原因分析

该问题是操作系统内核自身机制。热插内存后，虚拟机自动上线内存或者用户手动上线内存，需要初始化页表空间，这些页表空间由虚拟机配置内存中的可用内存提供。当虚拟机可用内存不足时，虚拟机在上线热插的内存时会卡死。

在虚拟机重启过程中，第一阶段初始化虚拟机配置的初始内存，第二阶段上线热插的设备内存。在第二阶段中，当前虚拟机配置初始内存的可用内存不足2G（内核或其他

用户进程会占用掉部分内存），无法为128G热插的设备内存的上线提供所需的页表空间，导致虚拟机卡死。

解决方法

为虚拟机配置更多的内存解决该问题，建议虚拟机配置初始内存的可用内存与热插的内存的比例至少为1：32。

以虚拟机名为“*EulerOS_64*”为例，具体步骤如下：

步骤1 登录宿主机执行如下命令，强制关闭虚拟机：

```
virsh destroy EulerOS_64
```

步骤2 执行如下命令，修改虚拟机内存为4G或更多（建议虚拟机配置初始内存的可用内存与热插的内存的比例至少为1：32，即要热插128G内存虚拟机至少需要配置4G内存）：

```
virsh edit EulerOS_64
```

将以下加粗的数值分别修改为如下数值或者更大：

```
<memory      unit='KiB'>4194304</memory>
<currentMemory      unit='KiB'>4194304</currentMemory>
```

步骤3 执行如下命令，启动虚拟机：

```
virsh start EulerOS_64
```

----结束

13.15 术语

account

指允许个人连接到系统的登录名称、个人目录、密码以及shell的组合。

alias

别名。在shell中为了能在执行命令时将某一字符串替换成另一个的一种机制。在提示符中键入alias可了解当前所定义的全部别名。

ARP

Address Resolution Protocol（地址解析协议）。该网际网络协议用于将网际网络地址动态地对应到局域网络的硬件地址上。

batch

批处理。将工作按顺序送到处理器，处理器一个接一个执行直到最后一个完成并准备好接受另一组处理清单的一种处理模式。

boot

引导。即发生在按下计算机的电源开关，机器开始检测接口设备的状态，并把操作系统加载到内存中的整个过程。

bootdisk

引导盘。包含来自硬盘（有时也可从其本身）加载操作系统的必要程序代码的可开机软磁盘。

BSD

Berkeley Software Distribution（伯克利软件发行套件）。一套由美国伯克利大学信息相关科系所发展的Unix分支。

buffer

缓冲区。指内存中固定容量一个小区域，其中的内容可以加载区域模式文件，系统分区表，以及执行中的进程等等。所有缓冲区的连贯性都是由缓冲区内存来维护的。

buffer cache

缓冲区存取。这是操作系统核心中甚为重要的一部份，负责让所有的缓冲区保持在最新的状态，在必要时可以缩小内存空间，清除不需要的缓冲区。

CHAP

Challenge-Handshake Authentication Protocol（询问交互式身份验证协议）：ISP验证其客户端所采用的通信协议。它与PAP的不同处在于：进行最初的判别后，每隔固定的时间周期它将会重新再验证一次。

client

客户端。是指能够短暂地连接到其他程序或计算机上并对其下达命令或要求信息的一个程序或一部计算机。它是服务器/客户端系统组件的一部分。

client/server system

服务器/客户端系统。由一个server（服务器端）与一个或多个client（客户端）所组成的系统架构或通信协议。

compilation

编译。指把人们读得懂的以某种程序语言（例如C语言）书写的程序源代码转换成机器可读的二进制文件的一种过程。

completion

自动补齐。只要系统内有能与之配合对象，shell将自动把一个不完全的子字符串，延展扩大成一个已存在的文件名、用户名。

compression

压缩。这是一种在通信连接的传送过程中缩小文件或减少字符数目的方法。压缩程序通常包含有compress，zip，gzip及bzip2。

console

控制台。也就是人们一般使用并称为终端的概念。它们是连接到一部巨型中央计算机的使用者操作的机器。对PC而言，实际的终端就是指键盘与屏幕。

cookies

由远程web服务器写入到本地硬盘的临时文件。它让服务器可以在使用者再次连上网站的时候可以知道其个人偏好。

DHCP

Dynamic Host Configuration Protocol（动态主机配置协议）。一种以局域网络机器为设计基础，能从DHCP服务器动态取得IP地址的通信协议。

DMA

Direct Memory Access。一种运用在PC架构上的技术，它允许接口设备可以从主存储器存取或读写资料而无须通过CPU联系。

DNS

Domain Name System（网络域名系统）。用来负责分配名称/地址的机制。它可以将机器名称对应到IP地址。同样DNS也允许反向搜寻，也就是说可以从IP地址得知其机器名称。

DPMS

Display Power Management System（显示器电源管理系统）。用于所有现今生产的显示器以管理其电源使之能够延长使用年限的协议。

editor

编辑器。一般而言是指编辑文本文件所使用的程序（也就是文字编辑器）。最为人所熟知的GNU/Linux编辑器有Emacs以及VIM。

email

电子邮件。是处于相同网络里的人们互相传送电子信息的一种方式。与定期邮件相同，email需要收件人以及寄件人地址以便正确地传送信息。

13.16 Linux 和 DOS 常用命令对照表

表 13-5 常用命令对照表

功能说明	linux命令	DOS命令
复制文件	cp	copy
列举文件	ls	dir
清除屏幕	clear	cls
移动文件	mv	move
删除文件	rm	del
创建目录	mkdir	mkdir
查看文件	less	more
文件重命名	mv	ren
查看当前路径	pwd	chdir
把输出回显到屏幕	echo	echo
在文件中寻找字串	grep	find
关闭和退出	exit	exit
显示或设置日期	date	date
显示已被使用的内存	free	mem

