Enabling an Intelligent Planet

日志管理

Date： 09 / 15 / 2021

Approved by	Checked by	Prepared by
		WuXiaoLin

目录

背景

一、对于 yocto2.5 跟 yocto3.0 的系统

系统默认设置，重启后/var/volatile 的内容会清空。原则上不会出现系统日志把磁盘占满，导致不能开机的问题。

但如果有些客户想保持日志，或者是保存日之后，限定日志大小等操作。就需要参考文档后面的内容进行配置。

二、对于 Debian，ubuntu 文件系统

系统默认配置，日志是保存在/var/log 目录下的，随着系统运行时间的增加，log 日志会越来越大，甚至会把磁盘占满，导致系统出错，开不了机。

所以，对于 Debian 和 ubuntu 必须要做系统日志的大小限制。

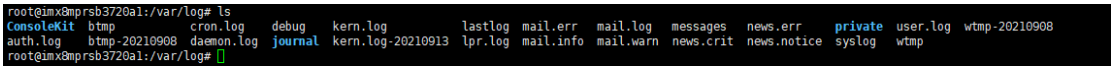
详细，可以参考文档的内容进行配置。

系统日志管理的方法介绍

日志管理有两种方法：第一种是通过 logrotate 日志管理工具管理系统日志，第二种是通过 systemd-journald 管理日志。以下为详细介绍：

一、通过 logrotate 日志管理工具管理系统日志 syslog

logrotate 是一个 Linux 系统默认安装了的日志文件管理工具，用来把旧文件轮转、压缩、删除，并且创建新的日志文件。我们可以根据日志文件的大小、天数等来转储，便于对日志文件管理。log 文件主要存放在/var/log 文件夹中。以 RSB-3720 为例，/var/log 文件夹下面有如下图所示文件。



logrotate 是基于 crond 服务来运行的，其 crond 服务的脚本是/etc/cron.daily/logrotate，日志转储是系统自动完成的。实际运行时，logrotate 会调用配置文件 /etc/logrotate.conf，可以在 /etc/logrotate.d 目录里放置自定义好的配置文件，用来覆盖 logrotate 的缺省值。

/etc/logrotate.conf 是主配置文件，/etc/logrotate.d 是一个目录，该目录下的所有文件都会被主动的读到 /etc/logrotate.conf 中执行。

/var/log 下常见日志文件解释如下表。

日志文件	内容
/var/log/messages	包括整体系统信息，其中也包含系统启动期间的日志。此外，mail，cron，daemon，kern 和 auth 等内容也记录在 var/log/messages 日志中。

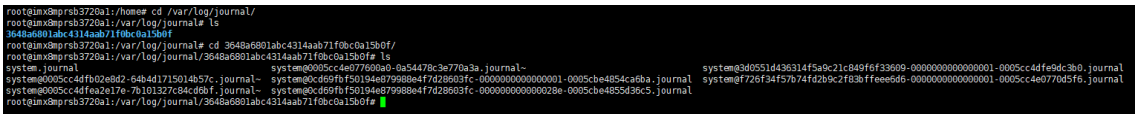
/var/log/dmesg	包含内核缓冲信息（kernel ring buffer）。在系统启动时，会在屏幕上显示许多与硬件有关的信息。可以用 dmesg 查看它们。
/var/log/auth.log	包含系统授权信息，包括用户登录和使用的权限机制等。
/var/log/boot.log	包含系统启动时的日志。
/var/log/daemon.log	包含各种系统后台守护进程日志信息。
/var/log/dpkg.log	包括安装或 dpkg 命令清除软件包的日志。
/var/log/kern.log	包含内核产生的日志，有助于在定制内核时解决问题。
/var/log/lastlog	记录所有用户的最近信息。这不是一个 ASCII 文件，因此需要用 lastlog 命令查看内容。
/var/log/maillog /var/log/mail.log	包含来着系统运行电子邮件服务器的日志信息。例如，sendmail 日志信息就全部送到这个文件中。
/var/log/user.log	记录所有等级用户信息的日志。
/var/log/alternatives.log	更新替代信息都记录在这个文件中。
/var/log/btmp	记录所有失败登录信息。使用 last 命令可以查看 btmp 文件。例如，"last -f /var/log/btmp | more"。
/var/log/cups	涉及所有打印信息的日志。
/var/log/wtmp 或 /var/log/utmp	包含登录信息。使用 wtmp 可以找出谁正在登陆进入系统，谁使用命令显示这个文件或信息等。

二、通过 systemd-journald 管理日志

Journald 是 systemd 引入的用于收集和存储日志数据的系统服务。journald 系统会将每一次启动的信息保存起来。journald 系统主要由三个主要的系统日记服务组件组成：

- 守护程序：systemd 日志服务由 systemd-journald 守护程序处理。
- 配置文件：日志服务的配置在/etc/systemd/journald.conf 里面设置。
- 日志搜索程序：用于搜索日记日志文件的程序是 journalctl。

对于 yocto 和 debian 系统，默认情况下，systemd 的日志保存在内存中，即/run/log/journal 目录下，系统重启就会清除。那么关机后再次开机只能看到本次开机之后的日志，上一次关机之前的日志是无法查看的。如果想要管理日志，保存一定天数或者一定存储容量的日志的话，需要设置相关配置文件。下图为 RSB-3720 开发板/var/log/journal 目录下的文件。因为 systemd journal 是二进制 journal，不能直接用 cat 命令查看相应的日志，需要用 Journalctl 命令查看。



可以使用以下命令查看 Journald 中已经记录的引导信息。

```
$ journalctl --list-boots
```

```
root@imx8mprsb3720a1:/# journalctl --list-boots
-7 012204be9c2545a58f9813d49633e1ab Mon 2021-09-13 18:20:32 UTC<E2><80><94>Mon 2021-09-13 19:30:01 UTC
-6 79dcbcca6e484fe8957b23b7bcba714f Mon 2021-09-13 21:28:19 UTC<E2><80><94>Mon 2021-09-13 21:30:01 UTC
-5 e13a315e74f14afb8247118855078c47 Mon 2021-09-13 21:43:17 UTC<E2><80><94>Mon 2021-09-13 21:43:50 UTC
-4 142416e79ef64b848ec422222a5e08ab Mon 2021-09-13 21:55:31 UTC<E2><80><94>Mon 2021-09-13 22:10:43 UTC
-3 ebe72c0a364d4849892313a007a27ae0 Sun 2021-09-19 00:09:39 UTC<E2><80><94>Sun 2021-09-19 00:09:44 UTC
-2 bd9493a87bd84f3da0dc09b5280ab9a2 Sun 2021-09-19 00:10:39 UTC<E2><80><94>Sun 2021-09-19 00:11:11 UTC
-1 1307cb66fd4e46f09462aefe742b4723 Sun 2021-09-19 00:13:07 UTC<E2><80><94>Sun 2021-09-19 00:13:10 UTC
 0 d5ef4354b34a47cb924bcec666fd1ac8 Thu 2021-09-30 00:17:41 UTC<E2><80><94>Thu 2021-09-30 00:18:14 UTC
root@imx8mprsb3720a1:/#
```

上图为 RSB-3720 开发板 8 次启动的 log 信息，第一列为序号可用于在 journalctl 中引用该次引导，第二列为引导 ID，末尾记录的两次时间为当次引导的开始与结束时间。

可使用以下命令查看上次引导的 journal 记录

```
$ journalctl -b -1
```

```
root@imx8mprsb3720a1:/# journalctl -b -1
-- Logs begin at Mon 2021-09-13 18:20:32 UTC, end at Thu 2021-09-30 00:18:14 UTC. --
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Booting Linux on physical CPU 0x0000000000 [0x410fd034]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Linux version 5.4.70-3720A1AIM30LIVA0271 (oe-user@oe-host) (gcc version 9.2.0 (GCC)) #1 SMP PREEMPT Sat Ju
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Machine model: Advantech i.MX8MPlus RSB3720A1 board
Sep 19 00:13:07 imx8mprsb3720a1 kernel: efi: Getting EFI parameters from FDT:
Sep 19 00:13:07 imx8mprsb3720a1 kernel: efi: UEFI not found.
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Reserved memory: created CMA memory pool at 0x00000000c4000000, size 960 MiB
Sep 19 00:13:07 imx8mprsb3720a1 kernel: OF: reserved mem: initialized node linux,cma, compatible id shared-dma-pool
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: No NUMA configuration found
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: Faking a node at [mem 0x0000000040000000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: NODE_DATA [mem 0x1bf3d5500-0x1bf3d6fff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Zone ranges:
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   DMA32    [mem 0x0000000040000000-0x00000000ffffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   Normal   [mem 0x0000000100000000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Movable zone start for each node
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Early memory node ranges
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   node   0: [mem 0x0000000040000000-0x000000000557fffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   node   0: [mem 0x0000000058000000-0x00000000923fffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   node   0: [mem 0x0000000094400000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Initmem setup node 0 [mem 0x0000000040000000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: On node 0 totalpages: 1554432
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   DMA32 zone: 12288 pages used for memmap
```

也可以使用引导 ID

```
root@imx8mprsb3720a1:/# journalctl --list-boots
-7 012204be9c2545a58f9813d49633e1ab Mon 2021-09-13 18:20:32 UTC<E2><80><94>Mon 2021-09-13 19:30:01 UTC
-6 79dcbcca6e484fe8957b23b7bcba714f Mon 2021-09-13 21:28:19 UTC<E2><80><94>Mon 2021-09-13 21:30:01 UTC
-5 e13a315e74f14afb8247118855078c47 Mon 2021-09-13 21:43:17 UTC<E2><80><94>Mon 2021-09-13 21:43:50 UTC
-4 142416e79ef64b848ec422222a5e08ab Mon 2021-09-13 21:55:31 UTC<E2><80><94>Mon 2021-09-13 22:30:01 UTC
-3 ebe72c0a364d4849892313a007a27ae0 Sun 2021-09-19 00:09:39 UTC<E2><80><94>Sun 2021-09-19 00:09:44 UTC
-2 bd9493a87bd84f3da0dc09b5280ab9a2 Sun 2021-09-19 00:10:39 UTC<E2><80><94>Sun 2021-09-19 00:11:11 UTC
-1 1307cb66fd4e46f09462aefe742b4723 Sun 2021-09-19 00:13:07 UTC<E2><80><94>Sun 2021-09-19 00:13:10 UTC
 0 d5ef4354b34a47cb924bcec666fd1ac8 Thu 2021-09-30 00:17:41 UTC<E2><80><94>Thu 2021-09-30 00:18:14 UTC
root@imx8mprsb3720a1:/# journalctl -b 1307cb66fd4e46f09462aefe742b4723
-- Logs begin at Mon 2021-09-13 18:20:32 UTC, end at Thu 2021-09-30 00:18:14 UTC. --
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Booting Linux on physical CPU 0x0000000000 [0x410fd034]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Linux version 5.4.70-3720A1AIM30LIVA0271 (oe-user@oe-host) (gcc version 9.2.0 (GCC)) #1 SMP PREEMPT S
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Machine model: Advantech i.MX8MPlus RSB3720A1 board
Sep 19 00:13:07 imx8mprsb3720a1 kernel: efi: Getting EFI parameters from FDT:
Sep 19 00:13:07 imx8mprsb3720a1 kernel: efi: UEFI not found.
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Reserved memory: created CMA memory pool at 0x00000000c4000000, size 960 MiB
Sep 19 00:13:07 imx8mprsb3720a1 kernel: OF: reserved mem: initialized node linux,cma, compatible id shared-dma-pool
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: No NUMA configuration found
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: Faking a node at [mem 0x0000000040000000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: NUMA: NODE_DATA [mem 0x1bf3d5500-0x1bf3d6fff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Zone ranges:
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   DMA32    [mem 0x0000000040000000-0x00000000ffffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel:   Normal   [mem 0x0000000100000000-0x00000001bfffffff]
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Movable zone start for each node
Sep 19 00:13:07 imx8mprsb3720a1 kernel: Early memory node ranges
```

“yocto2.5/3.0”与“Debian/Ubuntu”日志管理的差别

对于 yocto2.5 和 yocto3.0

由于日志文件夹挂载在临时文件夹/var/volatile 下，因此，开发板掉电之后日志会丢失，每次只保存本次启动的日志。

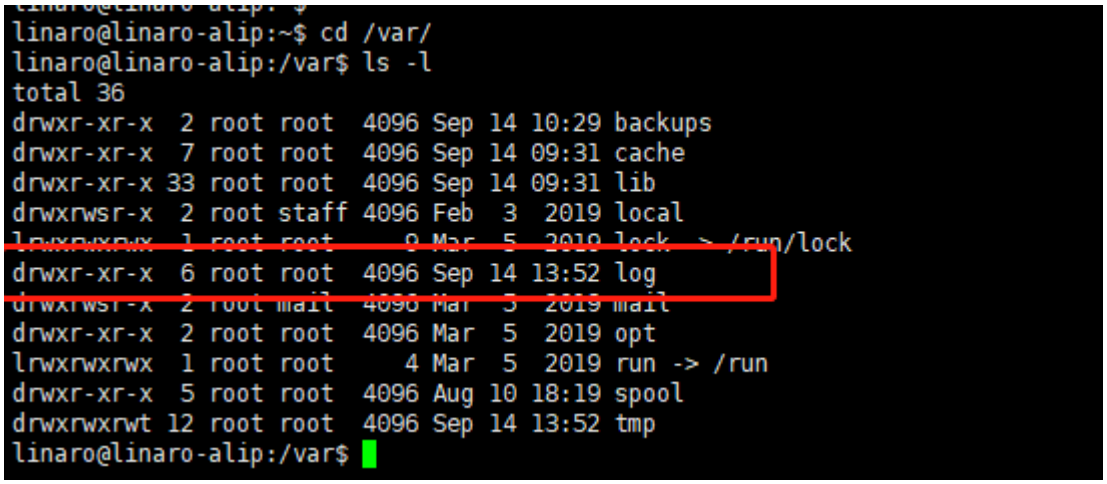
```
root@imx8mprsb3720a1:/var# l
total 24
drwxr-xr-x  2 root root 4096 Jun 19 21:40 backups
drwxr-xr-x  6 root root 4096 Sep  7 07:15 cache
drwxr-xr-x  2 root root 4096 Jun 19 22:57 db
drwxr-xr-x 15 root root 4096 Sep  7 07:16 lib
drwxr-xr-x  2 root root 4096 Jun 19 21:40 local
lrwxrwxrwx  1 root root   11 Jun 19 21:40 lock -> ../run/lock
lrwxrwxrwx  1 root root   12 Jun 19 21:40 log -> volatile/log
lrwxrwxrwx  1 root root    6 Jun 19 21:40 run -> ../run
drwxr-xr-x  5 root root 4096 Jun 19 22:57 spool
lrwxrwxrwx  1 root root   12 Jun 19 21:40 tmp -> volatile/tmp
drwxrwxrwt  4 root root   80 Sep  7 07:15 volatile
```

如果想进行日志管理，保存一定天数或者容量的日志，需要先修改文件/etc/fstab，将对应行注释掉。

```
# stock fstab - you probably want to override this with a machine specific one

/dev/root            /                    auto       defaults              1  1
proc                 /proc                proc       defaults              0  0
devpts               /dev/pts             devpts     mode=0620,gid=5       0  0
tmpfs                /run                 tmpfs      mode=0755,nodev,nosuid,strictatime 0  0
#tmpfs               /var/volatile        tmpfs      defaults              0  0

# uncomment this if your device has a SD/MMC/Transflash slot
#/dev/mmcblk0p1       /media/card          auto       defaults,sync,noauto  0  0

~
~
~
```

对于 debian 系统:

系统日志存放的日志的文件夹没有像 yocto 一样挂载到临时文件夹/var/volatile 下，/var/log 文件夹下的文件都会一直存储 log，因此对于 debian 系统，不定期清理日志的话，日志会占用磁盘容量，磁盘会爆满。因此必须进行日志管理。



如何修改系统日志管理配置

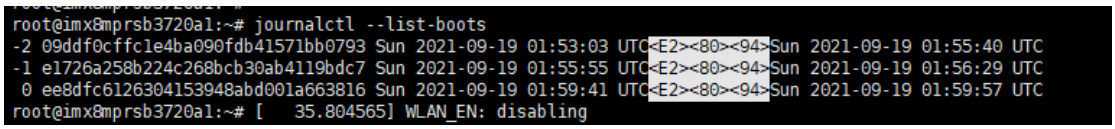
通过修改/etc/systemd/journald.conf

1、首先要设置 systemd-journald 日志持久化，执行以下命令.创建相关的目录来存放 journal 日志，修改权限，重启 systemd-journal 服务。

```
root@imx8mprsb3720a1:~# mkdir /var/log/journal
root@imx8mprsb3720a1:~# chgrp systemd-journal /var/log/journal
root@imx8mprsb3720a1:~# chmod g+s /var/log/journal
root@imx8mprsb3720a1:~# systemctl restart systemd-journald
```

2、重启数次观察日志记录结果。

journalctl --list-boots



3、修改配置文件/etc/systemd/journald.conf

```
#  This file is part of systemd.
#
#  systemd is free software; you can redistribute it and/or modify it
#  under the terms of the GNU Lesser General Public License as published by
#  the Free Software Foundation; either version 2.1 of the License, or
#  (at your option) any later version.
#
# Entries in this file show the compile time defaults.
# You can change settings by editing this file.
# Defaults can be restored by simply deleting this file.
#
# See journald.conf(5) for details.

[Journal]
#Storage=auto
#Compress=yes
#Seal=yes
#SplitMode=uid
#SyncIntervalSec=5m
#RateLimitIntervalSec=30s
#RateLimitBurst=10000
SystemMaxUse=50M
#SystemKeepFree=
#SystemMaxFileSize=
#SystemMaxFiles=100
#RuntimeMaxUse=
#RuntimeKeepFree=
#RuntimeMaxFileSize=
#RuntimeMaxFiles=100
MaxRetentionSec=5day
#MaxFileSec=1month
#ForwardToSyslog=no
#ForwardToKMsg=no
#ForwardToConsole=no
#ForwardToWall=yes
#TTYPath=/dev/console
#MaxLevelStore=debug
#MaxLevelSyslog=debug
#MaxLevelKMsg=notice
#MaxLevelConsole=info
#MaxLevelWall=emerg
#LineMax=48K
#ReadKMsg=yes
```

该配置文件参数详解如下：

`[Journal]`

`#Storage=auto`

Storage 支持的值为 volatile，persistent，auto 和 none，默认是 auto，所有值的含义如下

- 如果为 volatile，则日志数据将仅存储在内存中，即在/run/log/journal 目录下（根据需要创建）。
- 如果是 persistent，则数据将会存储在磁盘上，即/var/log/journal 目录下，并且在早期引导阶段磁盘不可写的时候把数据保存到/run/log/journal 目录下。
- auto 值意味着把日志数据存储在/var/log/journal/目录中。但是该目录必须已经存在并且设置了适当的权限。如果不存在，则日记数据将存储在易失性/run/log/journal/目录中，并且在系统关闭时会删除该数据。
- none 关闭所有存储，所有接收到的日志数据将被丢弃。

`#SystemMaxUse=`

`#SystemKeepFree=`

`#SystemMaxFileSize=`

`#SystemMaxFiles=100`

`#RuntimeMaxUse=`

`#RuntimeKeepFree=`

```
#RuntimeMaxFileSize=
#RuntimeMaxFiles=100
#MaxRetentionSec=
#MaxFileSec=1month
```

这些设置将会限制日志文件的大小上限。 以 System 开头的选项用于限制磁盘使用量，也就是 /var/log/journal 的使用量。以 Runtime 开头的选项用于限制内存使用量，也就是 /run/log/journal 的使用量。

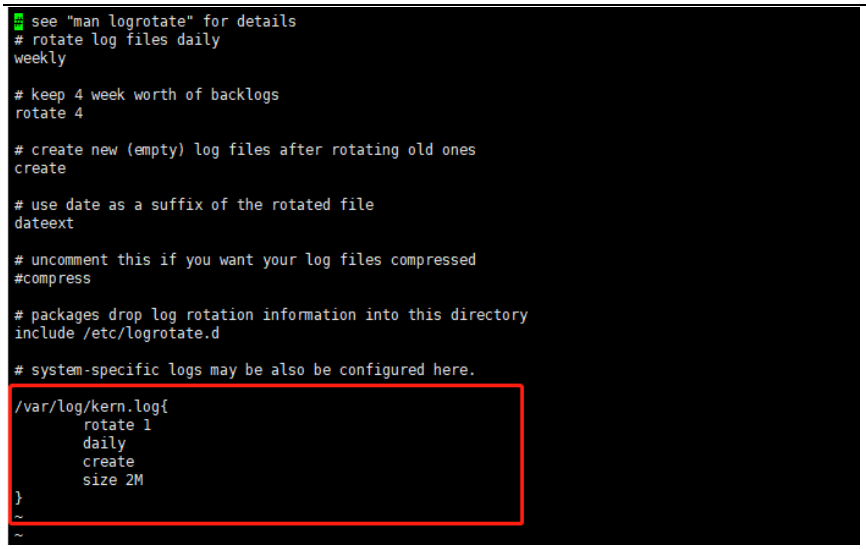
- RuntimeMaxUse/SystemMaxUse= 控制日志最大可使用多少磁盘空间，然后对日志文件执行 systemd-journald logrotate。默认为分配给节点的总物理内存的 10%
- RuntimeKeepFree/SystemKeepFree= 控制 systemd-journald 将为其他用途保留多少磁盘空间，之后将对日志文件执行 systemd-journald logrotate。默认为分配给节点的总物理内存的 15%
- SystemMaxFileSize=/RuntimeMaxFileSize= 限制单个日志文件的最大体积， 到达此限制后日志文件将会自动滚动。 默认值是对应的 SystemMaxUse=/RuntimeMaxUse= 值的 1/8 ， 这也意味着日志滚动 默认保留 7 个历史文件。
- SystemMaxFiles/RuntimeMaxFiles= 限制最多允许同时存在多少个日志文件，超出此限制后，最老的日志文件将被删除，而当前的活动日志文件 则不受影响。默认值为 100 个。
- MaxRetentionSec=日志滚动的时间间隔。通常并不需要使用基于时间的日志滚动策略，因为由 SystemMaxFileSize/RuntimeMaxFileSize= 控制的基于文件大小的日志滚动策略已经可以确保日志文件的大小不会超标。 默认值是一个月， 设为零表示禁用基于时间的日志滚动策略。
- MaxRetentionSec=日志文件的最大保留期限。当日志文件的最后修改时间(mtime)与当前时间之差，大于此处设置的值时，日志文件将会被删除。 通常并不需要使用基于时间的日志删除策略。

通过修改/etc/logrotate.conf 的方法

以 kern.log 为例，若需要保存 2M 或者 1 天的 log 信息，执行命令

```
$ vi /etc/logrotate.conf
```

添加配置信息，如下图所示



在/etc/logrotate.conf 配置文件中加入相关的文件路径以及对应的配置（配置参数参照下表），本例的配置参数解释如下：

rotate 1 //保存 1 份备份

daily //1 天转存一次

create //创建新的日志文件

size 2M //当日志文件到达 2M 时转储

配置项说明：

配置项	说明
compress	通过 gzip 压缩转储旧的日志
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，是先拷贝再清空的方式，拷贝和清空之间有一个时间差，可能会丢失部分日志数据
nocopytruncate	备份日志文件但是不截断
create mode owner group	使用指定的文件模式创建新的日志文件，如:create 0664 root utmp
nocreate	不建立新的日志文件
delaycompress	和 compress 一起使用时，转储的日志文件到下一次转储时才压缩

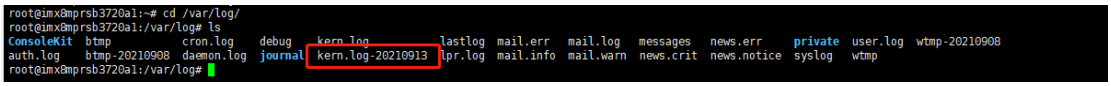
配置项	说明
nodelaycompress	覆盖 delaycompress 选项，转储同时压缩
missingok	在日志转储期间,任何错误将被忽略
errors address	转储时的错误信息发送到指定的 Email 地址
ifempty	即使日志文件是空文件也转储，这个是 logrotate 的缺省选项
notifempty	如果日志文件是空文件的话，不转储
mail E-mail	把转储的日志文件发送到指定的 E-mail 地址
nomail	转储时不发送日志文件到 E-mail 地址
olddir directory	转储后的日志文件放入指定的目录，必须和当前日志文件在同一个文件系统
noolddir	转储后的日志文件和当前日志文件放在同一个目录下
prerotate/endscript	在转储之前需要执行的命令可以放入这个对中，这两个关键字必须单独成行
postrotate/endscript	在转储之后需要执行的命令可以放入这个对中，这两个关键字必须单独成行
sharedscripts	所有的日志文件都转储完毕后统一执行一次脚本
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
size(minsize) logsize	当日志文件到达指定的大小时才转储，size 可以指定单位为 k 或 M，如:size 500k,size 100M
dateext	指定转储后的日志文件以当前日期为格式结尾，如

配置项	说明
dateformat dateformat	配合 dateext 使用，紧跟在下一行出现，定义日期格式，只支持%Y %m %d %s 这 4 个参数，如:dateformat -%Y%m%d%s

添加好配置信息后保存退出，执行命令

```
$ logrotate /etc/logrotate.conf
```

当日志信息超过 2M 或者一天之后，系统会自动转储该 log 信息



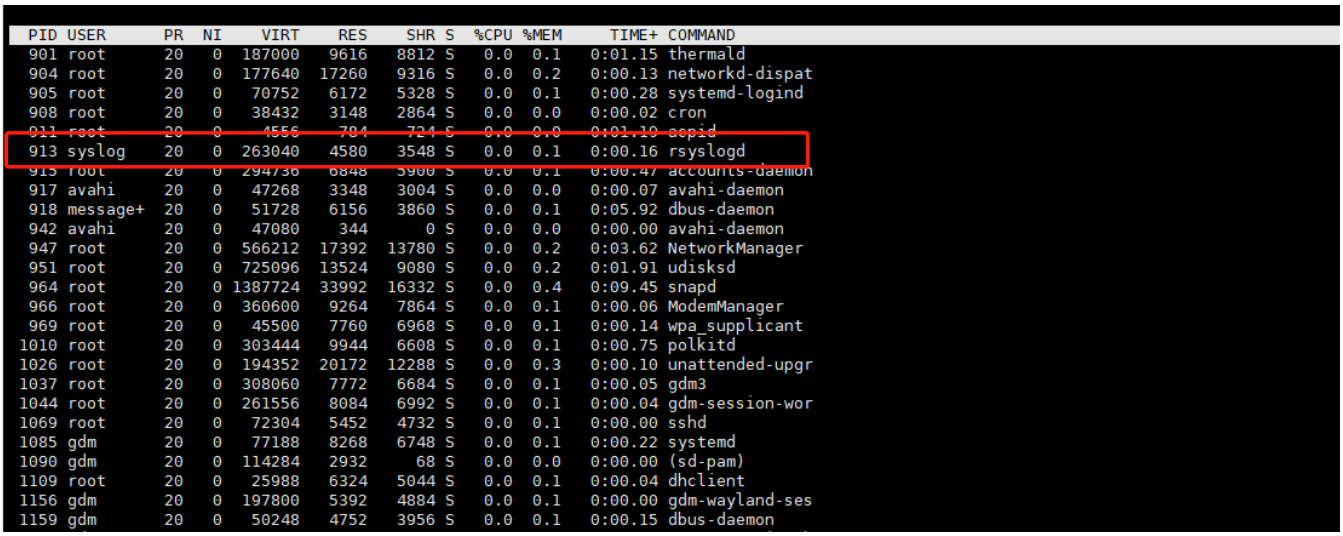
备注：如果是多个文件同种配置的话可以这么写

```
/var/log/mail.info
/var/log/mail.warn
/var/log/mail.err
/var/log/mail.log
/var/log/daemon.log
/var/log/kern.log
/var/log/auth.log
/var/log/user.log
/var/log/lpr.log
/var/log/cron.log
/var/log/debug
/var/log/messages
{
        rotate 4
        weekly
        missingok
        notifempty
        compress
        delaycompress
        sharedscripts
        postrotate
                /usr/lib/rsyslog/rsyslog-rotate
        endscript
}
```

补充说明

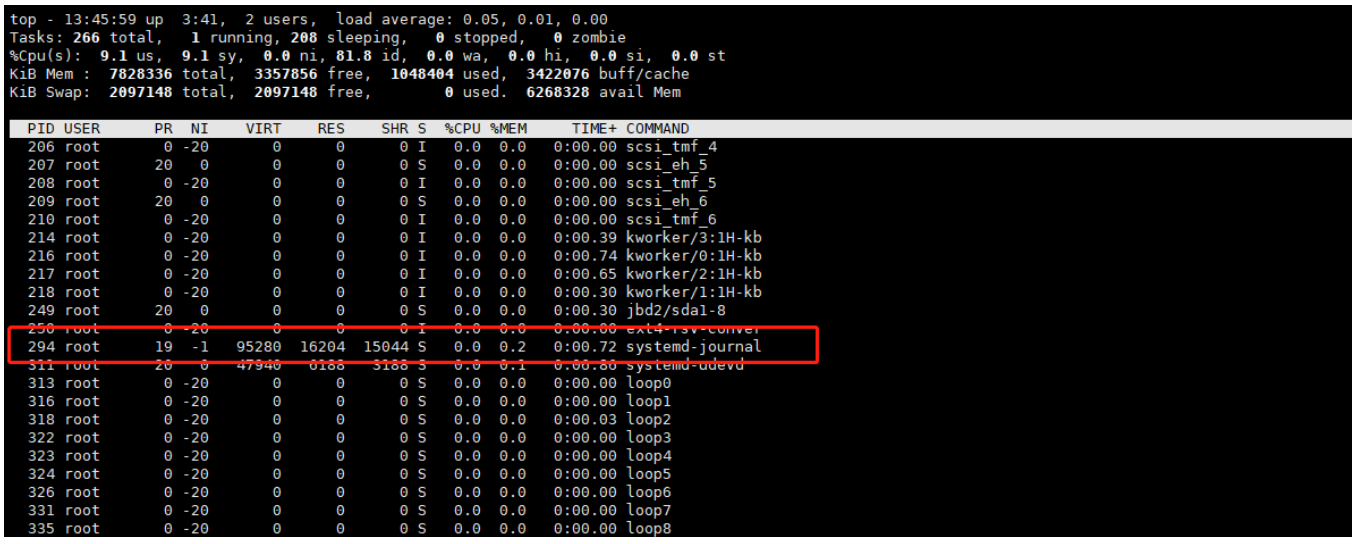
相关进程

系统日志进程



```
  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  901 root      20   0  187000   9616   8812 S   0.0  0.1   0:01.15 thermald
  904 root      20   0  177640  17260   9316 S   0.0  0.2   0:00.13 networkd-dispat
  905 root      20   0   70752   6172   5328 S   0.0  0.1   0:00.28 systemd-logind
  908 root      20   0   38432   3148   2864 S   0.0  0.0   0:00.02 cron
  911 root      20   0    4556    784    724 S   0.0  0.0   0:01.10 acpid
  913 syslog    20   0  263040   4580   3548 S   0.0  0.1   0:00.16 rsyslogd
  915 root      20   0  294736   6848   5900 S   0.0  0.1   0:00.47 accounts-daemon
  917 avahi     20   0   47268   3348   3004 S   0.0  0.0   0:00.07 avahi-daemon
  918 message+  20   0   51728   6156   3860 S   0.0  0.1   0:05.92 dbus-daemon
  942 avahi     20   0   47080    344      0 S   0.0  0.0   0:00.00 avahi-daemon
  947 root      20   0  566212  17392  13780 S   0.0  0.2   0:03.62 NetworkManager
  951 root      20   0  725096  13524   9080 S   0.0  0.2   0:01.91 udisksd
  964 root      20   0 1387724  33992  16332 S   0.0  0.4   0:09.45 snapd
  966 root      20   0  360600   9264   7864 S   0.0  0.1   0:00.06 ModemManager
  969 root      20   0   45500   7760   6968 S   0.0  0.1   0:00.14 wpa_supplicant
 1010 root      20   0  303444   9944   6608 S   0.0  0.1   0:00.75 polkitd
 1026 root      20   0  194352  20172  12288 S   0.0  0.3   0:00.10 unattended-upgr
 1037 root      20   0  308060   7772   6684 S   0.0  0.1   0:00.05 gdm3
 1044 root      20   0  261556   8084   6992 S   0.0  0.1   0:00.04 gdm-session-wor
 1069 root      20   0   72304   5452   4732 S   0.0  0.1   0:00.00 sshd
 1085 gdm       20   0   77188   8268   6748 S   0.0  0.1   0:00.22 systemd
 1090 gdm       20   0  114284   2932     68 S   0.0  0.0   0:00.00 (sd-pam)
 1109 root      20   0   25988   6324   5044 S   0.0  0.1   0:00.04 dhclient
 1156 gdm       20   0  197800   5392   4884 S   0.0  0.1   0:00.00 gdm-wayland-ses
 1159 gdm       20   0   50248   4752   3956 S   0.0  0.1   0:00.15 dbus-daemon
```

systemd-journal 进程



```
top - 13:45:59 up  3:41,  2 users,  load average: 0.05, 0.01, 0.00
Tasks: 266 total,   1 running, 208 sleeping,   0 stopped,   0 zombie
%Cpu(s):  9.1 us,  9.1 sy,  0.0 ni, 81.8 id,  0.0 wa,  0.0 hi,  0.0 si,  0.0 st
KiB Mem :  7828336 total,  3357856 free,  1048404 used,  3422076 buff/cache
KiB Swap:  2097148 total,  2097148 free,        0 used.  6268328 avail Mem

  PID USER      PR  NI    VIRT    RES    SHR S  %CPU %MEM     TIME+ COMMAND
  206 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 scsi_tmf_4
  207 root      20   0       0      0      0 S   0.0  0.0   0:00.00 scsi_eh_5
  208 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 scsi_tmf_5
  209 root      20   0       0      0      0 S   0.0  0.0   0:00.00 scsi_eh_6
  210 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 scsi_tmf_6
  214 root       0 -20       0      0      0 I   0.0  0.0   0:00.39 kworker/3:1H-kb
  216 root       0 -20       0      0      0 I   0.0  0.0   0:00.74 kworker/0:1H-kb
  217 root       0 -20       0      0      0 I   0.0  0.0   0:00.65 kworker/2:1H-kb
  218 root       0 -20       0      0      0 I   0.0  0.0   0:00.30 kworker/1:1H-kb
  249 root      20   0       0      0      0 S   0.0  0.0   0:00.30 jbd2/sda1-8
  250 root       0 -20       0      0      0 I   0.0  0.0   0:00.00 ext4-rsv-conver
  294 root      19  -1   95280  16204  15044 S   0.0  0.2   0:00.72 systemd-journal
  311 root      20   0   47940   6188   3188 S   0.0  0.1   0:00.88 systemd-udevd
  313 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop0
  316 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop1
  318 root       0 -20       0      0      0 S   0.0  0.0   0:00.03 loop2
  322 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop3
  323 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop4
  324 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop5
  326 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop6
  331 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop7
  335 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 loop8
```

syslog 和 systemd-journal 的不同

syslog 保存日志的形式是以文件夹划分的，比如内核的消息保存在 kern.log，用户的消息保存在 user.log，而 systemd-journal 的保存的是，从本次启动到本次关机的 log 内容。

syslog 的日志管理主要是，配置了相应文件下的转储周期（每天或者每周或者每月）、备份的文件数以及文件大小，系统会在相应.log 文件超过时间或者文件大小的情况下保存一份.log 文件后创建一份新的 log 文件。
systemd-journal 的日志管理则是，在设置好相应配置文件的保存天数和存储容量大小的情况下，系统会将设定天数之前或者超出存储容量的情况下，将最旧的系统 log 信息清除。

如果想要单独保存内核信息、用户信息的话推荐使用 syslog 管理日志，修改/etc/logrotate.conf 配置文件即可，systemd-journal 默认情况下不保存在磁盘，每次只保存本次启动的 log 信息，无需改动。

如果希望以启动次数来保存 log 信息，推荐使用 systemd-journal 管理日志，若要关闭 syslog 的话请参照以下方法。

如何关闭 syslog

修改/etc/syslog.conf 配置文件，将对应的行注释掉，不让其保存 log 信息。

```
#  /etc/syslog.conf     Configuration file for syslogd.
#
#  Ported from debian by Yu Ke <ke.yu@intel.com>
#


#
# First some standard logfiles.  Log by facility.
#

#auth,authpriv.*                        /var/log/auth.log
#*.*;auth,authpriv.none         -/var/log/syslog
#cron.*                         /var/log/cron.log
#daemon.*                       -/var/log/daemon.log
#kern.*                         -/var/log/kern.log
#lpr.*                          -/var/log/lpr.log
#mail.*                         -/var/log/mail.log
#user.*                         -/var/log/user.log

#
# Logging for the mail system.  Split it up so that
# it is easy to write scripts to parse these files.
#
mail.info                       -/var/log/mail.info
mail.warn                       -/var/log/mail.warn
mail.err                        /var/log/mail.err

# Logging for INN news system
#
news.crit                       /var/log/news.crit
news.err                        /var/log/news.err
news.notice                     -/var/log/news.notice

#
# Some `catch-all' logfiles.
#
*.=debug;\
auth,authpriv.none;\
```

附录

系统日志详细参考连接:
https://www.cnblogs.com/balaamwe/archive/2012/02/28/2371306.html
systemd-journald 参考链接:
https://blog.csdn.net/weixin_33836223/article/details/89692175
https://blog.csdn.net/baidu_23959681/article/details/82625611
https://www.cnblogs.com/morgan363/p/13957565.html