openEuler

openEuler操作系统进程管理

课程介绍

- 本课程主要介绍了进程管理的概念和基础操作。

- 进程（Process）是计算机中已运行程序的实体，是程序的一个具体实现。当程序被系统调用到内存以后，系统会给程序分配一定的资源（内存，设备等等）然后进行一系列的复杂操作，使程序变成进程以供系统调用。进程除包含程序的静态代码（text），还包含堆（heap）、栈（stack）、数据（data）段，堆用来在进程正在运行时分配所需的内存，栈用来保存临时数据（如函数参数、返回地址、局部变量），数据段用来保存已初始化的变量。

- 学完本课程后，您将能够：理解进程的状态及变化，掌握查看和结束进程的方法，了解前台和后台进程及进程优先级，掌握如何通过定时任务管理进程。

目录

CONTENT

openEuler

进程状态

Linux上进程有5种状态：

运行（TASK_RUNNING）：

正在运行或在运行队列中等待。

中断（TASK_INTERRUPTIBLE）：

休眠中， 受阻， 在等待某个条件的形成或接受到信号。

不可中断（TASK_UNINTERRUPTIBLE）：收到信号不唤醒和不可运行， 进程必须等待直到有中断发生。

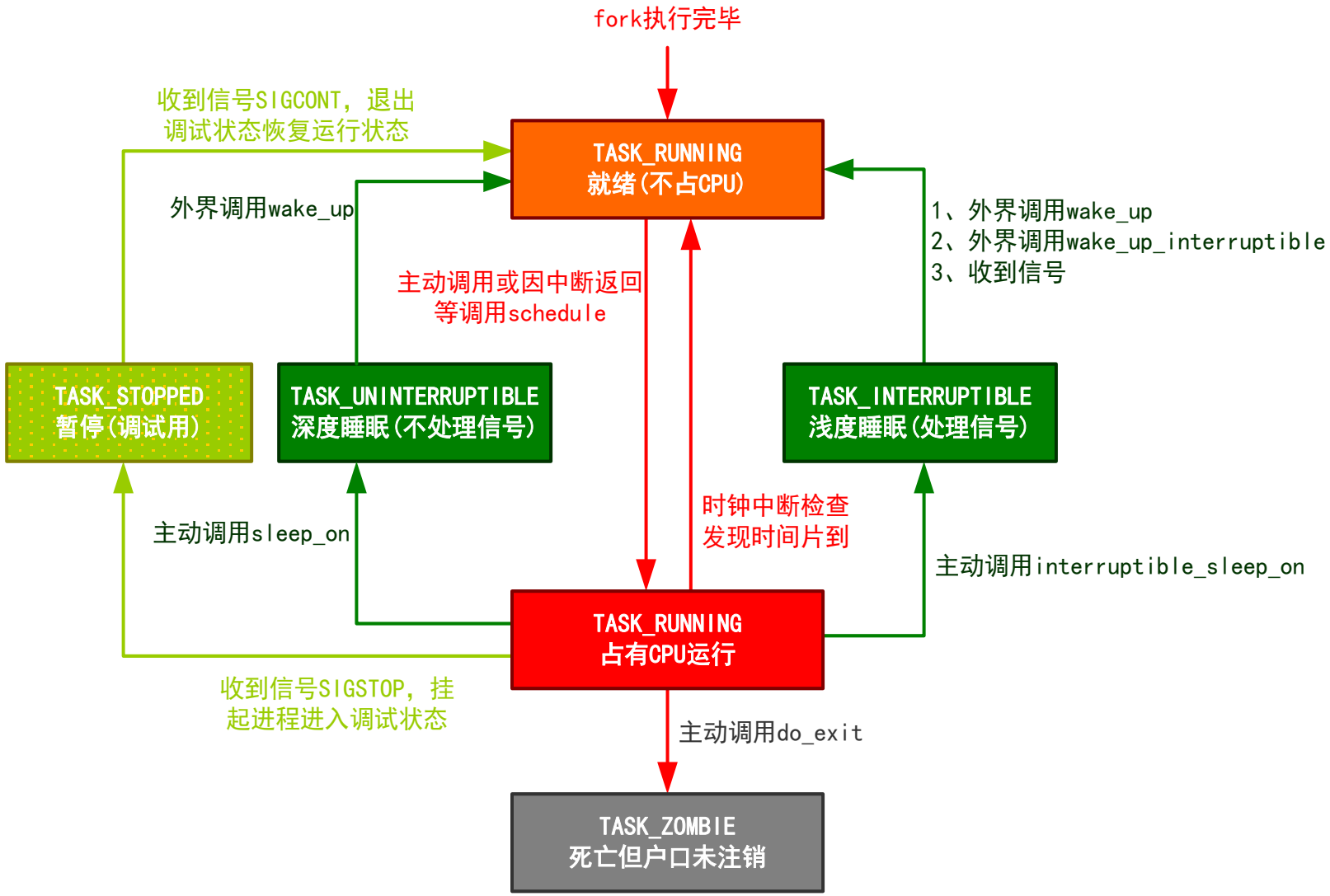
僵死(TASK_ZOMBIE)：

进程已终止， 但进程描述符存在， 直到父进程调用wait4()系统调用后释放。

停止（TASK_STOPED）：

进程收到SIGSTOP， SIGSTP， SIGTIN， SIGTOU信号后停止运行运行。

进程状态变化



openEuler

随堂测

- 1.以下进程状态正确的是（多选题）

  A. TASK_RUNNING

  B. TASK_INTERRUPTIBLE

  C. TASK_UNINTERRUPTIBLE

  D. TASK_STOPED

- 2. kill命令可以结束所有进程。 （判断题）

openEuler

目录

CONTENT

openEuler

使用PS命令查看进程

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。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
2029564 pts/0    00:00:00 vi
2467777 pts/2    00:00:00 ps
```

常用参数

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。

openEuler

使用TOP命令查看进程

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。top命令输出的实例：

```
top - 10:34:09 up 16 days, 51 min,  4 users,  load average: 0.00, 0.00, 0.00
Tasks: 125 total,   1 running, 124 sleeping,   0 stopped,   0 zombie
%Cpu(s):  0.0 us,  0.1 sy,  0.0 ni, 99.8 id,  0.0 wa,  0.0 hi,  0.1 si,  0.0 st
MiB Mem :   3409.5 total,   2429.5 free,    179.5 used,    800.5 buff/cache
MiB Swap:   4052.0 total,   4052.0 free,      0.0 used.   2862.7 avail Mem

   PID USER      PR  NI    VIRT    RES    SHR S  %CPU  %MEM     TIME+ COMMAND
2467836 root      20   0       0      0      0 I   0.3   0.0   0:00.13 kworker/0:1-events
2468241 root      20   0    9720   3964   3336 R   0.3   0.1   0:00.02 top
      1 root      20   0  168608  17136   8688 S   0.0   0.5  10:34.38 systemd
      2 root      20   0       0      0      0 S   0.0   0.0   0:00.30 kthreadd
      3 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_gp
      4 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 rcu_par_gp
      6 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 kworker/0:0H-events_highpri
      8 root       0 -20       0      0      0 I   0.0   0.0   0:00.00 mm_percpu_wq
      9 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_rude_
     10 root      20   0       0      0      0 S   0.0   0.0   0:00.00 rcu_tasks_trace
     11 root      20   0       0      0      0 S   0.0   0.0   0:00.19 ksoftirqd/0
     12 root      20   0       0      0      0 I   0.0   0.0   1:01.89 rcu_sched
     13 root      rt   0       0      0      0 S   0.0   0.0   0:01.51 migration/0
     14 root      20   0       0      0      0 S   0.0   0.0   0:00.00 cpuhp/0
```

openEuler

使用kill命令结束进程

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键来终止，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。发送的前提是，当前用户必须是进程的属主或 root 用户。

例如下面创建一个后台执行sleep进程并使用kill命令结束：

```
# sleep 99 &
[1] 2470292
# ps
   PID TTY          TIME CMD
2465648 pts/2    00:00:00 bash
2470292 pts/2    00:00:00 sleep
2470293 pts/2    00:00:00 ps
# kill 2470292
# ps
   PID TTY          TIME CMD
2465648 pts/2    00:00:00 bash
2470314 pts/2    00:00:00 ps
[1]+  Terminated              sleep 99
# ps
   PID TTY          TIME CMD
2465648 pts/2    00:00:00 bash
2470332 pts/2    00:00:00 ps
```

openEuler

Kill命令常用信号

kill命令的语法格式有以下两种方式：

格式一：kill [-s 信号 | -p] [-a] 进程号...　　格式二：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。例如列出所有信号：

```
# 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
63) SIGRTMAX-1  64) SIGRTMAX
```

编号	信号名	含义
0	EXIT	程序退出时收到该信息。
1	HUP	挂掉电话线或终端连接的挂起信号，这个信号也会造成某些进程在没有终止的情况下重新初始化。
2	INT	表示结束进程，但并不是强制性的，常用的 "Ctrl+C" 组合键发出就是一个 kill -2 的信号。
3	QUIT	退出。
9	**KILL**	**杀死进程，即强制结束进程。**
11	SEGV	段错误。
15	TERM	正常结束进程，是 kill 命令的默认信号。

openEuler

随堂测

1. 下列哪个命令可以动态查看主机进程信息？（单选题）

   A. ps

   B. top

   C. free

   D. lscpu

- 2. kill命令可以结束所有进程。（判断题）

openEuler

目录

CONTENT

openEuler

控制前台和后台进程

- 前台进程和用户交互，需要较高的响应速度，优先级较高;前台进程就是用户使用的有控制终端的进程。
- 后台进程几乎不和用户交互，优先级略低。Linux 的守护进程（Daemon）是一种特殊的后台进程，其独立于终端并周期性地执行任务或等待唤醒

openEuler 中常用以下命令管理进程，包括前后台切换及启停：

- &：将 & 放在命令的最后，可以把这个命令放到后台执行
- Ctrl + z：当前台正在执行一个命令时，按键 Ctrl + z，将其放到后台并暂停
- Ctrl + c：中断当前正在执行的命令
- jobs：查看当前有多少正在后台运行的命令
- fg：将后台中的命令调至前台继续运行
- bg：将后台任务唤醒，并在后台运行

前台启动进程

启动一个比较耗时的进程，然后再把该进程挂起，并使用 ps 命令查看，就会看到该进程在 ps 显示列表中。

例如在执行持续时间比较久的sleep命令：

```
# sleep 99
^Z
[1]+  Stopped                 sleep 999
```

#按“CTRL+Z”组合键挂起进，或在其它登录终端查看：

```
# ps
    PID TTY          TIME CMD
2465648 pts/2    00:00:00 bash
2465834 pts/2    00:00:00 sleep
2465852 pts/2    00:00:00 ps
```

openEuler

后台启动进程

进程需要长时间在后台执行，用户不关注命令的输出结果时，可以将进程直接从后台运行，不占用当前shell终端。

从后台启动进程时，在命令结尾处添加 “ &” 符号（注意，& 前面有空格）。输入命令并运行之后，Shell 会提供给我们一个数字，此数字就是该进程的进程号（PID）。然后直接就会出现提示符，用户可以继续完成其他工作，例如：

```
# jobs -l
[1]+ 2467348 Running              sleep 99 &
```

说明：“[1] ” 是工作号，“2466281 ” 是进程号（PID）。

jobs 命令可以用来查看当前终端放入后台的工作，工作管理的名字也来源于 jobs 命令。

```
# jobs -l
[1]+ 2467348 Running              sleep 99 &
```

说明：以上输出表示当前终端有1个后台工作进程sleep，工作号为 1，状态是运行标志是"+"。"+"号代表最近一个放入后台的工作，也是工作恢复时默认恢复的工作。"-"号代表倒数第二个放入后台的工作，而第三个以后的工作就没有"+-"标志了。

随堂测

- 1.以下哪个符号用于创建后台执行进程（单选题）

  A. @

  B. &

  C. |

  D. $

- 2. 后台进程可以使用Ctrl+C快捷键终止。(判断题)

openEuler

目录

CONTENT

openEuler

进程优先级

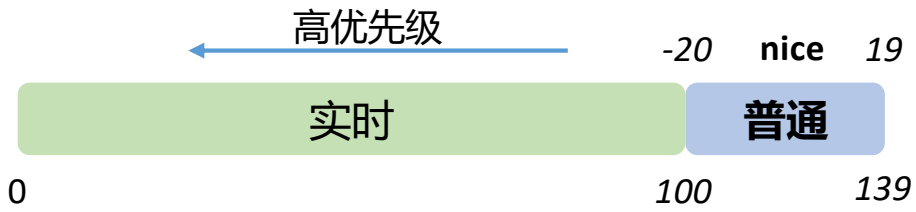
- 进程的 CPU 资源（时间片）分配就是指进程的优先级（priority）
- 优先级高的进程有优先执行权利
- 配置进程的优先级对多任务环境下的 Linux 很有用，可以改善系统性能

使用ps命令查看进程优先级：

```
# ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY        TIME CMD
0 S     0 2465648 2465645  0  80   0 -  1905 -      pts/2    00:00:00 bash
4 R     0 2472951 2465648  0  80   0 -  2249 -      pts/2    00:00:00 ps
```

- PRI，即进程的优先级，表示程序被CPU执行的先后顺序，值越小进程的优先级别越高。
- NI，即 nice 值 ，表示进程可被执行的优先级的修正数值，可理解为“谦让度”。
- 进程的 nice 值不是进程的优先级，但是可以通过调整 nice 值可以影响进程的优先值。

优先级（PRI）范围	描述
0—99	实时进程
100—139	非实时进程



openEuler

设置进程的优先级

openEuler 中使用 nice 和 renice 调整进程的 nice 值，进而影响进程优先级。

- nice 命令语法，优先序（adjustment 取值范围 -19 ~ 20），例如，将 vi 运行的优先级设置为 -18，并在后台运行：

```
# nice -n -18 vi &
[1] 2473024
# ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY        TIME CMD
0 S     0 2465648 2465645  0  80   0 -  1905 -      pts/2    00:00:00 bash
4 T     0 2473024 2465648  0  62 -18 -  2008 -      pts/2    00:00:00 vi
4 R     0 2473026 2465648  0  80   0 -  2249 -      pts/2    00:00:00 ps
```

- 例如，使用renice命令将 vi 运行的优先级设置为 10：

```
# renice -n 12 -p 2473024
2473024 (process ID) old priority -18, new priority 12
[root@localhost ~]# ps -l
F S   UID     PID    PPID  C PRI  NI ADDR SZ WCHAN  TTY        TIME CMD
0 S     0 2465648 2465645  0  80   0 -  1905 -      pts/2    00:00:00 bash
4 T     0 2473024 2465648  0  92  12 -  2008 -      pts/2    00:00:00 vi
4 R     0 2473344 2465648  0  80   0 -  2249 -      pts/2    00:00:00 ps
```

openEuler

随堂测

- 1.以下进程状态正确的是对于进程优先级说明正确的是。 （多选题）

  A. 实时进程优先级是：0-99

  B.实时进程优先级是： 100-139

  A. 非实时进程优先级是：0-99

  B.非实时进程优先级是： 100-139

- 2.进程优先级PRI值越大说明优先级越低。 （判断题）

openEuler

目录

CONTENT

openEuler

定时运行一批程序

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...]

- 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。

例如在6月8日上午10点执行slocate -u命令。在root权限下执行命令如下：

```
# at  10:00  6/8/22
at> slocate -u
at> <EOT>
job 1 at Wed Jun  8 10:00:00 2022
```

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。

openEuler

周期性运行一批程序

crond 服务Linux 下用来周期地执行某种任务或等待处理某些事件的一个守护进程，crond 进程每分钟会定期检查是否有要执行的任务，如果有，则会自动执行该任务。

crond 服务状态查询

```
# systemctl status crond
● crond.service - Command Scheduler
   Loaded: loaded (/usr/lib/systemd/system/crond.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2022-05-17 09:42:55 CST; 2 weeks 2 days ago
 Main PID: 947 (crond)
```

crond 服务设置和查询开机启动

```
# systemctl enable crond
# systemctl is-enabled crond
enabled
```

crond 服务重启

```
# systemctl restart crond
```

openEuler

定时任务配置

crond服务的系统定时任务可以通过crontab 命令管理，基本格式和示例如下：

crontab [选项] [file]

-e　　编辑某个用户的 crontab 文件内容。如果不指定用户，则表示编辑当前用户的 crontab 文件。

-l　　显示某用户的 crontab 文件内容，如果不指定用户，则表示显示当前用户的 crontab 文件内容。

```
# crontab -l
0 2 * * * su - oracle -c "$ORACLE_HOME/user-defined-operate/bin/backup_db.sh" >/ dev/null 2>&1
 * 2 * * * cd /opt/warnView; ./restart.sh >/dev/null 2>&1
10 9-18 * * * cd /ramber/shell/test/date > /dev/null 2>&1
30 02 * * * /var/adm/autoinstall/scripts/parse_atop/parse_atop_daily.sh
0,10,20,30,40,50 * * * * /opt/breeze/monitor/bin/CronAmosStartup.sh #* * * * *  /opt/netstat.sh >/dev/null 2>&1
```

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。

配置格式：minute hour day-of-month month-of-year day-of-week commands

参数	说明	范围
minute	一小时当中的第几分钟（minute）	0~59
hour	一天当中的第几小时（hour）	0~23
day-of-month	一个月当中的第几天（day）	1~31
month-of-year	一年当中的第几个月（month）	1~12
day-of-week	一周当中的星期几（week）	0~7（0和7都代表星期日）
commands	需要执行的命令。	

定时任务输出

定时任务中的命令执行结束后，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。如果用户命令执行很频繁将会有大量输出结果，并产生大量mail邮件文件，长时间运行会导致空间被占满，对执行的命令输出结果可以通过在 crontab 定时任务中都会在末尾带增加 >/dev/null 2>&1，来避免以上问题。

“>/dev/null 2>&1”作用如下：

- > 是重定向
- /dev/null 代表空设备文件
- 1 表示stdout标准输出,系统默认值是1,所以 ">/dev/null" 等同于 "1>/dev/null “
- 2 表示stderr标准错误
- & 表示等同于的意思,2>&1,表示2的输出重定向等同于1

表示标准输出重定向到空设备文件,即不输出任何信息到终端，此时就不会产生邮件文件。

openEuler

随堂测

- 1.定时任务参数时间设置支持以下哪些时间间隔设置。（多选题）

  A. minute

  B. hour

  C. day

  D. month

- 2. 对于cron定时任务一般会接结尾加上“>/dev/null 2>&1”。（判断题）

openEuler

Thank you

openEuler