



链滴

# DS218 安装 Docker 后跑高密集型 CPU 容器 频繁死机的问题详查和解决

作者: [wyisolo](#)

原文链接: <https://ld246.com/article/1664684629872>

来源网站: [链滴](#)

许可协议: [署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

DS218参数配置如下

CPU: Realtek RTD1296 SoC

CPU时钟频率:1.4 GHz

CPU核心数:4

DS218【Arm64 芯片】在安装了aarch64架构的Docker应用后，发现在部署高密度CPU任务比如青面板的时候，DS218CPU占用时常占满99%，进而导致主板温度升高，小型NAS服务器宕机了.....

ARM Docker目前无法实现CFS机制，通过Docker指令来限制密集型容器的对资源CPU的占用

```
WARNING: No kernel memory TCP limit support
WARNING: No cpu cfs quota support
WARNING: No cpu cfs period support
WARNING: No blkio throttle.read_bps_device support
WARNING: No blkio throttle.write_bps_device support
WARNING: No blkio throttle.read_iops_device support
WARNING: No blkio throttle.write_iops_device support
```

查询了Linux的对于资源调度的机制后，选择了一款CPULIMITS软件，来对Docker容器本身进行资源限制，从而避免了该款NAS机型因为高并发密集CPU资源抢占而频繁死机的情况。

Linux使用cpulimit对CPU使用率进行限制

具体解决办法：

本地shell命令行中采用本地安装的方式，来安装cpulimit软件，具体步骤如下

实操

```
cd /tmp
wget 'https://sunpma.com/other/oss/cpulimit-1.1.tar.gz'
tar cpulimit-1.1.tar.gz
cd cpulimit-1.1
make
cp cpulimit /usr/local/sbin/
rm -rf cpulimit*
```

## 使用实例

限制程序名为xmrig的程序仅使用60%的CPU使用率，并在后台一直运行；

```
cpulimit -e xmrig -l 60 -b
```

限制进程号为10086的程序仅使用60%的CPU利用率，并在后台一直运行；

```
cpulimit -p 10086 -l 60 -b
```

限制绝对路径下的软件仅使用60%的CPU利用率，并在后台一直运行；

```
cpulimit -e /usr/local/nginx/sbin/nginx -l 60 -b
```

**关闭cpulimit后台进程（取消所有限制）**

kill \$(pidof cpulimit)

## 使用说明

-p 程的PID	--pid=N	pid of the process
-e file	--exe=FILE 可执行程序文件名	name of the executable program
-P 程的绝对路径名	--path=PATH	absolute path name of the
-b 台运行	--background	run in background
-l up	--limit=N 允许的CPU百分比, 最低为1%	percentage of cpu allowed from 1
-z s	--lazy 如果目标进程退出或无目标进程则终止	exit if there is no suitable target proce
-h 示帮助并退出	--help	display this help and exit

## 注意事项

- 限制CPU的百分比按照实际的CPU数量而定, 比如单核最高为100%, 双核最高为200%, 三核最高300%, 以此类推即可;
- ROOT用户可以限制所有的进程, 普通用户只能限制用户有权限管理的进程;

**注意: DS218安装了CPULIMIT软件后, 是没有-b的选项的, 所以退出shell后要想让配置生效, 必结合nohup 和&命令使用即可, 机器如果重启限制是否还在, 未测试。**

[linux退出终端运行程序,linux nohup命令实现退出终端后程序继续后台运行](#)

### 摘要

Unix/Linux下一般想让某个程序在后台运行, 很多都是使用&在程序结尾来让程序自动运行; 但如果想在退出终端后, 程序依然还在后台运行, 则要用nohup与&组合来实现。

### nohup 命令

用途: 不挂断地运行命令

语法: nohup Command [ Arg ... ] [& ]

### 描述

nohup 命令运行由 Command参数和任何相关的Arg参数指定的命令, 忽略所有挂断(SIGHUP)信号在注销后使用 nohup 命令运行后台中的程序: 要运行后台中的 nohup 命令, 需要添加&到命令的尾。

### 日志记录

无论是否将 nohup 命令的输出重定向到终端，输出都将附加到当前目录的nohup.out 文件中。如果前目录的nohup.out文件不可写，输出重定向到\$HOME/nohup.out文件中。如果没有文件能创建或开以用于追加，那么 Command 参数指定的命令不可调用。

#### 使用时注意

在当shell中提示了nohup成功后，还需要按终端上键盘任意键退回到shell输入命令窗口，然后通过输入exit来退出终端；如果在nohup执行成功后直接点关闭程序按钮关闭终端的话，这时候会掉该命令所对应的session，导致nohup对应的进程被通知需要一起shutdown，起不到关掉终端后程序继续后台运行的作用。

```
nohup cpulimit -e /volume1/@appstore/Docker/bin/containerd -l 60 &
```

```
nonadmin@DS218:~$ sudo -i
```

```
Password:
```

```
root@DS218:~# echo $(pidof cpulimit)
```

```
24339
```

```
root@DS218:~# ps -ef | grep 24339
```

```
root 24339 1 0 05:37 ? 00:00:00 cpulimit -e /volume1/@appstore/Docker/bin/containerd -l 50
```

```
root 24488 24417 0 05:38 pts/1 00:00:00 grep --color=auto 24339
```

附件如下

[cpulimit1.1.zip](#)