

Linux stress命令详解

[软件包命令下载路径](#)

简介： Linux stress命令详解

stress 命令主要用来模拟系统负载较高时的场景，本文介绍其基本用法。

1 安装 stress

系统默认没有安装 stress，需要通过下面的命令安装：

```
rz -e
ls
stress-1.0.4-16.el7.x86_64.rpm
rpm -ivh stress-1.0.4-16.el7.x86_64.rpm
stress --version
```

```
root@65cfe18dc2e7:/# stress --version
stress 1.0.4
```

2 语法

```
stress <options>
```

3 可选参数

-c, --cpu N 产生 N 个进程，每个进程都反复不停的计算随机数的平方根

-i, --io N 产生 N 个进程，每个进程反复调用 sync() 将内存上的内容写到硬盘上

-m, --vm N 产生 N 个进程，每个进程不断分配和释放内存

-vm-bytes B 指定分配内存的大小

-vm-stride B 不断的给部分内存赋值，让 COW(Copy On Write)发生

-vm-hang N 指示每个消耗内存的进程在分配到内存后转入睡眠状态 N 秒，然后释放内存，一直重复执行这个过程

-vm-keep 一直占用内存，区别于不断的释放和重新分配(默认是不断释放并重新分配内存)

-d, --hadd N 产生 N 个不断执行 write 和 unlink 函数的进程(创建文件，写入内容，删除文件)

-hadd-bytes B 指定文件大小

-t, --timeout N 在 N 秒后结束程序

-backoff N 等待N微妙后开始运行

-q, --quiet 程序在运行的过程中不输出信息

-n, --dry-run 输出程序会做什么而并不实际执行相关的操作

-version 显示版本号

-v, --verbose 显示详细的信息

4 消耗 CPU 资源

stress 消耗 CPU 资源是通过调用 sqrt 函数计算由 rand 函数产生的随机数的平方根实现。下面的命令会产生 4 个这样的进程不断计算：

```
stress -c 4
```

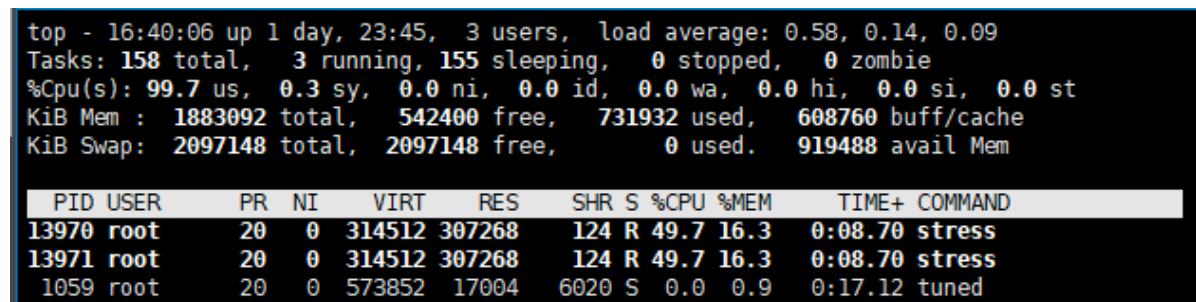
使用 top 命令查看 CPU 的状态如下(CPU 在用户态满负荷运转)：

5 消耗内存资源

下面的命令产生两个子进程，每个进程分配 300M 内存：

```
stress --vm 2 --vm-bytes 300M --vm-keep  
# 也可以替换成单位 G
```

父进程处于睡眠状态，两个子进程负责资源消耗。



```
top - 16:40:06 up 1 day, 23:45, 3 users, load average: 0.58, 0.14, 0.09  
Tasks: 158 total, 3 running, 155 sleeping, 0 stopped, 0 zombie  
%Cpu(s): 99.7 us, 0.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st  
KiB Mem : 1883092 total, 542400 free, 731932 used, 608760 buff/cache  
KiB Swap: 2097148 total, 2097148 free, 0 used, 919488 avail Mem
```

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
13970	root	20	0	314512	307268	124	R	49.7	16.3	0:08.70	stress
13971	root	20	0	314512	307268	124	R	49.7	16.3	0:08.70	stress
1059	root	20	0	573852	17004	6020	S	0.0	0.9	0:17.12	tuned

-vm-keep

一直占用内存，区别于不断的释放和重新分配(默认是不断释放并重新分配内存)。

-vm-hang N

指示每个消耗内存的进程在分配到内存后转入睡眠状态 N 秒，然后释放内存，一直重复执行这个过程。

-vm-keep 和 --vm-hang 都可以用来模拟只有少量内存的机器，但是指定它们时 CPU 的使用情况是不一样的。

```
stress --vm 2 --vm-bytes 500M --vm-keep
```

一直在进行默认的 stride 操作，user 非常高(cpu 在用户态忙碌)。

```
stress --vm 2 --vm-bytes 500M --vm-hang 5
```

上面这两种状态不断切换，但整体上看 CPU 的负载并不高。

-vm-stride B

不断的给部分内存赋值，让 COW(Copy On Write)发生。只要指定了内存相关的选项，这个操作就会执行，只是大小为默认的 4096。赋值内存的比例由参数决定：

```
for (i = 0; i < bytes; i += stride)
    ptr[i] = 'z'; /* Ensure that COW happens. */
```

bytes 为消耗的总内存大小，stride 为间隔。

该参数会影响 CPU 状态 us 和 sy：

```
stress --vm 2 --vm-bytes 500M --vm-stride 64
```

```
top - 16:44:32 up 1 day, 23:50, 3 users, 1
Tasks: 158 total, 3 running, 155 sleeping,
%Cpu(s): 48.2 us, 51.8 sy, 0.0 ni, 0.0 id,
```

```
stress --vm 2 --vm-bytes 500M --vm-stride 1M
```

```
top - 16:45:02 up 1 day, 23:50, 3 users, 1
Tasks: 158 total, 3 running, 155 sleeping,
%Cpu(s): 0.3 us, 99.7 sy, 0.0 ni, 0.0 id,
```

为什么会产生这样的结果？原因是单独的赋值和对比操作可以让 CPU 在用户态的负载占到 99% 以上。-vm-stride 值增大就意味着减少赋值和对比操作，这样就增加了内存的释放和分配次数(cpu在内核空间的负载)。

不指定 --vm-stride 选项就使用默认值是 4096，CPU 负载情况居于前两者之间：

```
stress --vm 2 --vm-bytes 500M
```

6 消耗 IO 资源

下面的命令产生 4 个进程，每个进程都反复调用 sync 函数将内存上的内容写到硬盘上：

```
stress -i 4
```

使用 top 命令查看 CPU 的状态如下：

```
top - 16:46:53 up 1 day, 23:52, 3 users, load average:
Tasks: 160 total, 5 running, 155 sleeping, 0 stopped
%Cpu(s): 3.7 us, 96.3 sy, 0.0 ni, 0.0 id, 0.0 wa, 0.0 st
```

sy 升高

压测磁盘及 IO

下面的命令创建一个进程不断的在磁盘上创建 10M 大小的文件并写入内容：

```
stress -d 1 --hdd-bytes 10M
```

其它选项

<code>-verbose</code>	# 显示 <code>stress</code> 程序运行过程中的详细信息：
<code>-timeout N</code>	# 在 <code>N</code> 秒后结束程序。
<code>-quiet</code>	# <code>stress</code> 程序运行的过程中不输出信息。
<code>-n, --dry-run</code>	# 输出程序会做什么而并不实际执行相关的操作：
<code>-backoff N</code>	# 让新 <code>fork</code> 出来的进程 <code>sleep N</code> 微秒再开始运行。

除了单独指定某一类的选项，还可以同时执行多个类型的任务，比如产生 3 个 CPU 进程、3 个 IO 进程、2 个 10M 的 vm 进程，并且每个 vm 进程中不循环分配释放内存：

```
stress --cpu 3 --io 3 --vm 2 --vm-bytes 10M --vm-keep
```

总结

我在对配置为40C 128GB烤机时，使用的参数如下，每个机器实际性能都有相差可自行调整合适参数进行压测

```
cat test_config.sh
#!/bin/bash

while [ true ]
do
stress --vm 30 --vm-bytes 4G
done

或者 stress --vm 20 --vm-bytes 8G
```