10 【实操篇-进程管理类】

无论是 Linux 系统管理员还是普通用户，监视系统进程的运行情况并适时终止一些失控的进程，是每天的例行事务。和 Linux 系统相比，进程管理在 Windows 中更加直观，它主要是使用"任务管理器"来进行进程管理的。

通常，使用"任务管理器"主要有 3 个目的：

1. 利用"应用程序"和"进程"标签来查看系统中到底运行了哪些程序和进程；
2. 利用"性能"和"用户"标签来判断服务器的健康状态；
3. 在"应用程序"和"进程"标签中强制中止任务和进程；

Linux 中虽然使用命令进行进程管理，但是进程管理的主要目的是一样的，即查看系统中运行的程序和进程、判断服务器的健康状态和强制中止不需要的进程。

那么，到底什么是进程呢？它和我们平时所说的“程序”又有什么联系呢？

1.什么是进程和程序

进程是正在执行的一个程序或命令，每个进程都是一个运行的实体，都有自己的地址空间，并占用一定的系统资源。程序是人使用计算机语言编写的可以实现特定目标或解决特定问题的代码集合。

这么讲很难理解，那我们换一种说法。程序是人使用计算机语言编写的，可以实现一定功能，并且可以执行的代码集合。而进程是正在执行中的程序。当程序被执行时，执行人的权限和属性，以及程序的代码都会被加载入内存，操作系统给这个进程分配一个 ID，称为 PID（进程 ID）。

也就是说，在操作系统中，所有可以执行的程序与命令都会产生进程。只是有些程序和命令非常简单，如 ls 命令、touch 命令等，它们在执行完后就会结束，相应的进程也就会终结，所以我们很难捕捉到这些进程。但是还有一些程和命令，比如 httpd 进程，启动之后就会一直驻留在系统当中，我们把这样的进程称作常驻内存进程。

某些进程会产生一些新的进程，我们把这些进程称作子进程，而把这个进程本身称作父进程。比如，我们必须正常登录到 Shell 环境中才能执行系统命令，而 Linux 的标准 Shell 是 bash。我们在 bash 当中执行了 ls 命令，那么 bash 就是父进程，而 ls 命令是在 bash 进程中产生的进程，所以 ls 进程是 bash 进程的子进程。也就是说，子进程是依赖父进程而产生的，如果父进程不存在，那么子进程也不存在了。

2.进程管理的作用

在使用 Windows 系统的过程中，使用任务管理器，很大程度上是为了强制关闭“未反应”的软件，也就是杀死进程。的确，这是很多使用进程管理工具或进程管理命令的人最常见的使用方法。不过，杀死进程（强制中止进程）只是进程管理工作中最不常用的手段，因为每个进程都有自己正确的结束方法，而杀死进程是在正常方法已经失效的情况下的后备手段。

那么，进程管理到底应该是做什么的呢？我以为，进程管理主要有以下 3 个作用。

1) 判断服务器的健康状态

运维工程师最主要的工作就是保证服务器安全、稳定地运行。理想的状态是，在服务器出现问题，但是还没有造成服务器宕机或停止服务时，就人为干预解决了问题。

进程管理最主要的工作就是判断服务器当前运行是否健康，是否需要人为干预。如果服务器的 CPU 占用率、内存占用率过高，就需要人为介入解决问题了。这又出现了一个问题，我们发现服务器的 CPU 或内存占用率很高，该如何介入呢？是直接终止高负载的进程吗？

当然不是，应该判断这个进程是否是正常进程，如果是正常进程，则说明你的服务器已经不能满足应用需求，你需要更好的硬件或搭建集群了；如果是非法进程占用了系统资源，则更不能直接中止进程，而要判断非法进程的来源、作用和所在位置，从而把它彻底清除。

当然，如果服务器数量很少，我们完全可以人为通过进程管理命令来进行监控与干预，但如果服务器数量较多，那么人为手工监控就变得非常困难了，这时我们就需要相应的监控服务，如 cacti 或 nagios。总之，进程管理工作中最重要的工作就是判断服务器的健康状 态，最理想的状态是服务器宕机之前就解决问题，从而避免服务器的宕机。

2) 查看系统中所有的进程

我们需要查看看系统中所有正在运行的进程，通过这些进程可以判断系统中运行了哪些服务、是否有非法服务在运行。

3) 杀死进程

这是进程管理中最不常用的手段。当需要停止服务时，会通过正确关闭命令来停止服务（如 apache 服务可以通过 service httpd stop 命令来关闭）。只有在正确终止进程的手段失效的情况下，才会考虑使用 kill 命令杀死进程。

其实，进程管理和 Windows 中任务管理器的作用非常类似，不过大家在使用任务管理器时是为了杀死进程，而不是为了判断服务器的健康状态。

3.ps 查看当前系统进程状态

ps 命令是最常用的监控进程的命令，通过此命令可以查看系统中所有运行进程的详细信息。

ps 命令有多种不同的使用方法，这常常给初学者带来困惑。在各种 Linux 论坛上，询问 ps 命令语法的帖子屡见不鲜，而出现这样的情况，还要归咎于 UNIX 悠久的历史和庞大的派系。在不同的 Linux 发行版上，ps 命令的语法各不相同，为此，Linux 采取了一个折中的方法，即融合各种不同的风格，兼顾那些已经习惯了其它系统上使用 ps 命令的用户。

ps 命令的基本格式如下：

```
[root@localhost ~]# ps aux
#查看系统中所有的进程
[root@localhost ~]# ps -ef
#可以查看子父进程之间的关系
```

选项：

- a：显示一个终端的所有进程，除会话引线外；
- u：显示进程的归属用户及内存的使用情况；
- x：显示没有控制终端的进程；
- -l：长格式显示更加详细的信息；
- -e：显示所有进程；
- -f：显示完整格式的进程列表

可以看到，ps 命令有些与众不同，它的部分选项不能加入"-"，比如命令"ps aux"，其中"aux"是选项，但是前面不能带“-”。

大家如果执行 "man ps" 命令，则会发现 ps 命令的帮助为了适应不同的类 UNIX 系统，可用格式非常多，不方便记忆。所以，我建议大家记忆几个固定选项即可。比如：

- "ps aux" 可以查看系统中所有的进程；
- "ps -ef" 可以查看系统中所有的进程，而且还能看到进程的父进程的 PID 和进程优先级；

- "ps -l" 只能看到当前 Shell 产生的进程；

有这三个命令就足够了，下面分别来查看。

【例 1】

```
[root@localhost ~]# ps aux
#查看系统中所有的进程
USER PID %CPU %MEM  VSZ  RSS   TTY STAT START TIME COMMAND
root   1  0.0  0.2 2872 1416   ?   Ss   Jun04 0:02 /sbin/init
root   2  0.0  0.0    0    0   ?    S   Jun04 0:00 [kthreadd]
root   3  0.0  0.0    0    0   ?    S   Jun04 0:00 [migration/0]
root   4  0.0  0.0    0    0   ?    S   Jun04 0:00 [ksoftirqd/0]
...省略部分输出...
```

以上输出信息中各列的具体含义。

表头	含义
USER	该进程是由哪个用户产生的。
PID	进程的 ID。
%CPU	该进程占用 CPU 资源的百分比，占用的百分比越高，进程越耗费资源。
%MEM	该进程占用物理内存的百分比，占用的百分比越高，进程越耗费资源。
VSZ	该进程占用虚拟内存的大小，单位为 KB。
RSS	该进程占用实际物理内存的大小，单位为 KB。
TTY	该进程是在哪个终端运行的。其中，tty1 ~ tty7 代表本地控制台终端（可以通过 Alt+F1 ~ F7 快捷键切换不同的终端），tty1~tty6 是本地的字符界面终端，tty7 是图形终端。pts/0 ~ 255 代表虚拟终端，一般是远程连接的终端，第一个远程连接占用 pts/0，第二个远程连接占用 pts/1，依次增长。
STAT	进程状态。常见的状态有以下几种：-D：不可被唤醒的睡眠状态，通常用于 I/O 情况。-R：该进程正在运行。-S：该进程处于睡眠状态，可被唤醒。-T：停止状态，可能是在后台暂停或进程处于除错状态。-W：内存交互状态（从 2.6 内核开始无效）。-X：死掉的进程（应该不会出现）。-Z：僵尸进程。进程已经中止，但是部分程序还在内存当中。-<：高优先级（以下状态在 BSD 格式中出现）。-N：低优先级。-L：被锁入内存。-s：包含子进程。-l：多线程（小写 L）。-+：位于后台。
START	该进程的启动时间。
TIME	该进程占用 CPU 的运算时间，注意不是系统时间。
COMMAND	产生此进程的命令名。

【例 2】"ps aux"命令可以看到系统中所有的进程，"ps -ef"命令也能看到系统中所有的进程。

```
[root@localhost ~]# ps -ef
F S UID  PID  PPID C  PRI  NI ADDR  SZ WCHAN TTY      TIME  CMD
4 S   0    1    0 0  80   0 -    718 -     ?    00:00:02  init
1 S   0    2    0 0  80   0 -      0 -     ?    00:00:00  kthreadd
1 S   0    3    2 0 -40   - -      0 -     ?    00:00:00  migration/0
1 S   0    4    2 0  80   0 -      0 -     ?    00:00:00  ksoflirqd/0
1 S   0    5    2 0 -40   - -      0 -     ?    00:00:00  migration/0
…省略部分输出…
```

以上输出信息中各列的含义。

表头	含义
F	进程标志，说明进程的权限，常见的标志有两个: 1：进程可以被复制，但是不能被执行；4：进程使用超级用户权限；
S	进程状态。具体的状态和"psaux"命令中的 STAT 状态一致；
UID	运行此进程的用户的 ID；
PID	进程的 ID；
PPID	父进程的 ID；
C	该进程的 CPU 使用率，单位是百分比；
PRI	进程的优先级，数值越小，该进程的优先级越高，越早被 CPU 执行；
NI	进程的优先级，数值越小，该进程越早被执行；
ADDR	该进程在内存的哪个位置；
SZ	该进程占用多大内存；
WCHAN	该进程是否运行。"-"代表正在运行；
TTY	该进程由哪个终端产生；
TIME	该进程占用 CPU 的运算时间，注意不是系统时间；
CMD	产生此进程的命令名；

【例 3】如果不想看到所有的进程，只想查看一下当前登录产生了哪些进程，那只需使用 "ps -l" 命令就足够了：

```
[root@localhost ~]# ps -l
#查看当前登录产生的进程
F S UID   PID  PPID C PRI NI ADDR SZ WCHAN TTY        TIME CMD
4 S 0   18618 18614 0  80  0 - 1681  -     pts/1 00:00:00 bash
4 R 0   18683 18618 4  80  0 - 1619  -     pts/1 00:00:00 ps
```

可以看到，这次从 pts/1 虚拟终端登录，只产生了两个进程：一个是登录之后生成的 Shell，也就是 bash；另一个是正在执行的 ps 命令。

我们再来说说僵尸进程。僵尸进程的产生一般是由于进程非正常停止或程序编写错误，导致子进程先于父进程结束，而父进程又没有正确地回收子进程，从而造成子进程一直存在于内存当中，这就是僵尸进程。

僵尸进程会对主机的稳定性产生影响，所以，在产生僵尸进程后，一定要对产生僵尸进程的软件进行优化，避免一直产生僵尸进程；对于已经产生的僵尸进程，可以在查找出来之后强制中止。

4.kill 终止进程

4.1 kill 终止进程

kill 从字面来看，就是用来杀死进程的命令，但事实上，这个或多或少带有一定的误导性。从本质上讲，kill 命令只是用来向进程发送一个信号，至于这个信号是什么，是用户指定的。

也就是说，kill 命令的执行原理是这样的，kill 命令会向操作系统内核发送一个信号（多是终止信号）和目标进程的 PID，然后系统内核根据收到的信号类型，对指定进程进行相应的操作。

kill 命令的基本格式如下：

```
[root@localhost ~]# kill [信号] PID
```

kill 命令是按照 PID 来确定进程的，所以 kill 命令只能识别 PID，而不能识别进程名。Linux 定义了几十种不同类型的信号，读者可以使用 kill -l 命令查看所有信号及其编号，这里仅列出几个常用的信号。

信号编号	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。

需要注意的是，表中省略了各个信号名称的前缀 SIG，也就是说，SIGTERM 和 TERM 这两种写法都对，kill 命令都可以理解。

下面，我们举几个例子来说明一下 kill 命令。

【例 1】 标准 kill 命令。

```
[root@localhost ~]# kill 2248
#杀死PID是2248的httpd进程，默认信号是15，正常停止
#如果默认信号15不能杀死进程，则可以尝试-9信号，强制杀死进程
```

【例 2】使用“-1”信号，让进程重启。

```
[root@localhost ~]# kill -1 2246
使用“-1 (数字1)”信号，让httpd的主进程重新启动
```

学会如何使用 kill 命令之后，再思考一个问题，使用 kill 命令一定可以终止一个进程吗？

答案是否定的。文章开头说过，kill 命令只是“发送”一个信号，因此，只有当信号被程序成功“捕获”，系统才会执行 kill 命令指定的操作；反之，如果信号被“封锁”或者“忽略”，则 kill 命令将会失效。

4.2 killall 终止特定的一类进程

killall 也是用于关闭进程的一个命令，但和 kill 不同的是，killall 命令不再依靠 PID 来杀死单个进程，而是通过程序的进程名来杀死一类进程，也正是由于这一点，该命令常与 ps、pstree 等命令配合使用。

killall 命令的基本格式如下：

`[root@localhost ~]# killall [选项] 进程名`

注意，此命令的信号类型同 kill 命令一样，因此这里不再赘述，此命令常用的选项有如下 2 个：

- -i：交互式，询问是否要杀死某个进程；
- -I：忽略进程名的大小写；

接下来，给大家举几个例子。

【例 1】杀死 httpd 进程。

```
[root@localhost ~]# killall httpd
#杀死所有进程名是httpd的进程
[root@localhost ~]# ps aux | grep "httpd" | grep -v "grep"
#查询发现所有的httpd进程都消失了
```

5.pstree 查看进程树

pstree 命令是以树形结构显示程序和进程之间的关系，此命令的基本格式如下：

`[root@localhost ~]# pstree [选项] [PID或用户名]`

pstree 命令常用选项以及各自的含义。

选项	含义
-a	显示启动每个进程对应的完整指令，包括启动进程的路径、参数等。
-c	不使用精简法显示进程信息，即显示的进程中包含子进程和父进程。
-n	根据进程 PID 号来排序输出，默认是以程序名排序输出的。
-p	显示进程的 PID。
-u	显示进程对应的用户名称。

需要注意的是，在使用 pstree 命令时，如果不指定进程的 PID 号，也不指定用户名称，则会以 init 进程为根进程，显示系统中所有程序和进程的信息；反之，若指定 PID 号或用户名，则将以 PID 或指定命令为根进程，显示 PID 或用户对应的所有程序和进程。

init 进程是系统启动的第一个进程，进程的 PID 是 1，也是系统中所有进程的父进程。
【例 1】

```
[root@1ocalhost ~]# pstree
init─┬─abrc-dump-oopa
├─abrtd
├─acpid
...省略部分输出...
├─rayslogd───3[{rsyslogrd}]
#有3个rsyslogd进程存在
├─sshd───sshd───bash───pstree
#Pstree命令进程是在远程连接中被执行的
├─udevd───2[udevd
└─xinecd
```

【例 2】如果想知道某个用户都启动了哪些进程，使用 pstree 命令可以很容易实现，以 mysql 用户为例：

```
[root@1ocalhost ~]# pstree mysql
mysqid---6*[{mysqid}]
```

此输出结果显示了 mysql 用户对应的进程为 mysqid，并且 mysqid 进程拥有 5 个子进程（外加 1 个父进程，共计 6 个进程）。

6.top 实时监控系统进程状态

ps 命令可以一次性给出当前系统中进程状态，但使用此方式得到的信息缺乏时效性，并且，如果管理员需要实时监控进程运行情况，就必须不停地执行 ps 命令，这显然是缺乏效率的。

为此，Linux 提供了 top 命令。top 命令可以动态地持续监听进程地运行状态，与此同时，该命令还提供了一个交互界面，用户可以根据需要，人性化地定制自己的输出，进而更清楚地了进程的运行状态。

使用权限：所有使用者。

top 命令的基本格式如下：

`[root@localhost ~]#top [选项]`

选项：

- -d 秒数：指定 top 命令每隔几秒更新。默认是 3 秒；
- -b：使用批处理模式输出。一般和"-n"选项合用，用于把 top 命令重定向到文件中；
- -n 次数：指定 top 命令执行的次数。一般和"-"选项合用；
- -p 进程PID：仅查看指定 ID 的进程；
- -s：使 top 命令在安全模式中运行，避免在交互模式中出现错误；
- -u 用户名：只监听某个用户的进程；

在 top 命令的显示窗口中，还可以使用如下按键，进行一下交互操作：

- ? 或 h：显示交互模式的帮助；
- P：按照 CPU 的使用率排序，默认就是此选项；
- M：按照内存的使用率排序；
- N：按照 PID 排序；
- T：按照 CPU 的累积运算时间排序，也就是按照 TIME+ 项排序；
- k：按照 PID 给予某个进程一个信号。一般用于中止某个进程，信号 9 是强制中止的信号；
- r：按照 PID 给某个进程重设优先级（Nice）值；
- q：退出 top 命令；

我们看看 top 命令的执行结果，如下：

```
[root@localhost ~]# top
top - 12:26:46 up 1 day, 13:32, 2 users, load average: 0.00, 0.00, 0.00
Tasks: 95 total, 1 running, 94 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.1%us, 0.1%sy, 0.0%ni, 99.7%id, 0.1%wa, 0.0%hi, 0.1%si, 0.0%st
Mem: 625344k total, 571504k used, 53840k free, 65800k buffers
Swap: 524280k total, 0k used, 524280k free, 409280k cached
PID   USER PR NI VIRT  RES  SHR S %CPU %MEM   TIME+ COMMAND
19002 root 20  0 2656 1068  856 R  0.3  0.2 0:01.87 top
1     root 20  0 2872 1416 1200 S  0.0  0.2 0:02.55 init
2     root 20  0    0    0    0 S  0.0  0.0 0:00.03 kthreadd
3     root RT  0    0    0    0 S  0.0  0.0 0:00.00 migration/0
4     root 20  0    0    0    0 S  0.0  0.0 0:00.15 ksoftirqd/0
5     root RT  0    0    0    0 S  0.0  0.0 0:00.00 migration/0
6     root RT  0    0    0    0 S  0.0  0.0 0:10.01 watchdog/0
7     root 20  0    0    0    0 S  0.0  0.0 0:05.01 events/0
8     root 20  0    0    0    0 S  0.0  0.0 0:00.00 cgroup
9     root 20  0    0    0    0 S  0.0  0.0 0:00.00 khelper
10    root 20  0    0    0    0 S  0.0  0.0 0:00.00 netns
11    root 20  0    0    0    0 S  0.0  0.0 0:00.00 async/mgr
12    root 20  0    0    0    0 S  0.0  0.0 0:00.00 pm
13    root 20  0    0    0    0 S  0.0  0.0 0:01.70 sync_supers
14    root 20  0    0    0    0 S  0.0  0.0 0:00.63 bdi-default
15    root 20  0    0    0    0 S  0.0  0.0 0:00.00 kintegrityd/0
16    root 20  0    0    0    0 S  0.0  0.0 0:02.52 kblockd/0
17    root 20  0    0    0    0 S  0.0  0.0 0:00.00 kacpid
18    root 20  0    0    0    0 S  0.0  0.0 0:00.00 kacpi_notify
```

我们解释一下命令的输出。top 命令的输出内容是动态的，默认每隔 3 秒刷新一次。命令的输出主要分为两部分：

1. 第一部分是前五行，显示的是整个系统的资源使用状况，我们就是通过这些输出来判断服务器的资源使用状态的；
2. 第二部分从第六行开始，显示的是系统中进程的信息；

我们先来说明第一部分的作用。

- 第一行为任务队列信息。

内 容	说 明
12:26:46	系统当前时间
up 1 day, 13:32	系统的运行时间.本机已经运行 1 天 13 小时 32 分钟
2 users	当前登录了两个用户
load average: 0.00,0.00, 0.00	系统在之前 1 分钟、5 分钟、15 分钟的平均负载。如果 CPU 是单核的，则这个数值超过 1 就是高负载：如果 CPU 是四核的，则这个数值超过 4 就是高负载 （这个平均负载完全是依据个人经验来进行判断的，一般认为不应该超过服务器 CPU 的核数）

- 第二行为进程信息。

内 容	说 明
Tasks: 95 total	系统中的进程总数
1 running	正在运行的进程数
94 sleeping	睡眠的进程数
0 stopped	正在停止的进程数
0 zombie	僵尸进程数。如果不是 0，则需要手工检查僵尸进程

- 第三行为 CPU 信息。

内 容	说 明
Cpu(s): 0.1 %us	用户模式占用的 CPU 百分比
0.1%sy	系统模式占用的 CPU 百分比
0.0%ni	改变过优先级的用户进程占用的 CPU 百分比
99.7%id	空闲 CPU 占用的 CPU 百分比
0.1%wa	等待输入/输出的进程占用的 CPU 百分比
0.0%hi	硬中断请求服务占用的 CPU 百分比
0.1%si	软中断请求服务占用的 CPU 百分比
0.0%st	st（steal time）意为虚拟时间百分比，就是当有虚拟机时，虚拟 CPU 等待实际 CPU 的时间百分比

- 第四行为物理内存信息。

内 容	说 明
Mem: 625344k total	物理内存的总量，单位为KB
571504k used	已经使用的物理内存数量
53840k&ee	空闲的物理内存数量。我们使用的是虚拟机，共分配了 628MB内存，所以只有53MB的空闲内存
65800k buffers	作为缓冲的内存数量

- 第五行为交换分区（swap）信息。

内 容	说 明
Swap: 524280k total	交换分区（虚拟内存）的总大小
Ok used	已经使用的交换分区的大小
524280k free	空闲交换分区的大小
409280k cached	作为缓存的交换分区的大小

我们通过 top 命令的第一部分就可以判断服务器的健康状态。如果 1 分钟、5 分钟、15 分钟的平均负载高于 1，则证明系统压力较大。如果 CPU 的使用率过高或空闲率过低，则证明系统压力较大。如果物理内存的空闲内存过小，则也证明系统压力较大。

这时，我们就应该判断是什么进程占用了系统资源。如果是不必要的进程，就应该结束这些进程；如果是必需进程，那么我们该增加服务器资源（比如增加虚拟机内存），或者建立集群服务器。

我们还要解释一下缓冲（buffer）和缓存（cache）的区别：

- 缓存（cache）是在读取硬盘中的数据时，把最常用的数据保存在内存的缓存区中，再次读取该数据时，就不去硬盘中读取了，而在缓存中读取。
- 缓冲（buffer）是在向硬盘写入数据时，先把数据放入缓冲区,然后再一起向硬盘写入，把分散的写操作集中进行，减少磁盘碎片和硬盘的反复寻道，从而提高系统性能。

简单来说，缓存（cache）是用来加速数据从硬盘中"读取"的，而缓冲（buffer）是用来加速数据"写入"硬盘的。

再来看 top 命令的第二部分输出，主要是系统进程信息，各个字段的含义如下：

- PID：进程的 ID。
- USER：该进程所属的用户。
- PR：优先级，数值越小优先级越高。
- NI：优先级，数值越小、优先级越高。
- VIRT：该进程使用的虚拟内存的大小，单位为 KB。
- RES：该进程使用的物理内存的大小，单位为 KB。
- SHR：共享内存大小，单位为 KB。
- S：进程状态。
- %CPU：该进程占用 CPU 的百分比。
- %MEM：该进程占用内存的百分比。
- TIME+：该进程共占用的 CPU 时间。
- COMMAND：进程的命令名。

这部分和 ps 命令的输出比较类似，只是如果在终端执行 top 命令，则不能看到所有的进程，而只能看到占比靠前的进程。接下来我们举几个 top 命令常用的实例。

【例 1】如果只想让 top 命令查看某个进程，就可以使用 "-p 选项"。命令如下：

```
[root@localhost ~]# top -p 15273
#只查看 PID为 15273的apache进程
top - 14:28:47 up 1 day, 15:34, 3 users, load average: 0.00,0.00,0.00
Tasks: 1 total, 0 running, 1 sleeping, 0 stopped, 0 zombie
Cpu(s): 0.0%us, 0.0%sy, 0.0%ni,100.0%id, 0.0%wa, 0.0%hi, 0.0%si, 0.0%st
Mem: 625344k total, 574124k used, 51220k free, 67024k buffers
Swap: 524280k total, 0k used, 524280k free, 409344k cached
PID     USER PR NI VIRT  RES SHR S %CPU %MEM  TIME+  COMMAND
15273 daemon 20 0  4520 1192 580 S  0.0  0.2 0:00.00   httpd
```

7.netstat 显示网络状态和端口占用信息

1）基本语法

`netstat -anp | grep 进程号 （功能描述：查看该进程网络信息）`

`netstat –nlp | grep 端口号 （功能描述：查看网络端口号占用情况`

2）选项说明

- -a 显示所有正在监听（listen）和未监听的套接字（socket）
- -n 拒绝显示别名，能显示数字的全部转化成数字
- -l 仅列出在监听的服务状态
- -p 表示显示哪个进程在调用

3）案例实操

（1）通过进程号查看sshd进程的网络信息

```
[root@hadoop101 hadoop-2.7.2]# netstat -anp | grep sshd
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
951/sshd
tcp 0 0 192.168.202.100:22 192.168.202.1:57741
ESTABLISHED 3380/sshd: root@pts
tcp 0 52 192.168.202.100:22 192.168.202.1:57783
ESTABLISHED 3568/sshd: root@pts
tcp 0 0 192.168.202.100:22 192.168.202.1:57679
ESTABLISHED 3142/sshd: root@pts
tcp6 0 0 :::22 :::* LISTEN
951/sshd
unix 2 [ ] DGRAM 39574 3568/sshd:
root@pts
unix 2 [ ] DGRAM 37452 3142/sshd:
root@pts
unix 2 [ ] DGRAM 48651 3380/sshd:
root@pts
unix 3 [ ] STREAM CONNECTED 21224 951/sshd
```

（2）查看某端口号是否被占用

```
[root@hadoop101 桌面]# netstat -nltp | grep 22
tcp 0 0 192.168.122.1:53 0.0.0.0:* LISTEN
1324/dnsmasq
tcp 0 0 0.0.0.0:22 0.0.0.0:* LISTEN
951/sshd
tcp6 0 0 :::22 :::* LISTEN
951/sshd
```