用十条命令在一分钟内检查Linux服务器性能

作者：Vanessa
原文链接：https://ld246.com/article/1451373018340
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)

如果你的Linux服务器突然负载暴增，告警短信快发爆你的手机，如何在最短时间内找出Linux性问题所在？来看Netflix性能工程团队的这篇博文，看它们通过十条命令在一分钟内对机器性能问题进诊断。
概述
通过执行以下命令，可以在1分钟内对系统资源使用情况有个大致的了解。

uptime
dmesg | tail
vmstat 1
mpstat -P ALL 1
pidstat 1
iostat -xz 1
free -m
sar -n DEV 1
sar -n TCP,ETCP 1
top

其中一些命令需要安装sysstat包，有一些由procps包提供。这些命令的输出，有助于快速定位能瓶颈，检查出所有资源（CPU、内存、磁盘IO等）的利用率（utilization）、饱和度（saturation和错误（error）度量，也就是所谓的<a href="http://www.brendangregg.com/usemethod.html" USE方法。
下面我们来逐一介绍下这些命令，有关这些命令更多的参数和说明，请参照命令的手册。
uptime
<pre>$ uptime
23:51:26 up 21:31, 1 user, load average: 30.02, 26.43, 19.02
</pre>
这个命令可以快速查看机器的负载情况。在Linux系统中，这些数据表示等待CPU资源的进程和塞在不可中断IO进程（进程状态为D）的数量。这些数据可以让我们对系统资源使用有一个宏观的了。
命令的输出分别表示1分钟、5分钟、15分钟的平均负载情况。通过这三个数据，可以了解服务负载是在趋于紧张还是区域缓解。如果1分钟平均负载很高，而15分钟平均负载很低，说明服务器正命令高负载情况，需要进一步排查CPU资源都消耗在了哪里。反之，如果15分钟平均负载很高，1分平均负载较低，则有可能是CPU资源紧张时刻已经过去。
上面例子中的输出，可以看见最近1分钟的平均负载非常高，且远高于最近15分钟负载，因此我需要继续排查当前系统中有什么进程消耗了大量的资源。可以通过下文将会介绍的vmstat、mpstat命令进一步排查。
dmesg | tail
<pre>$ dmesg | tail
[1880957.563150] perl invoked oom-killer: gfp_mask=0x280da, order=0, oom_score_adj=0
[...]
[1880957.563400] Out of memory: Kill process 18694 (perl) score 246 or sacrifice child
[1880957.563408] Killed process 18694 (perl) total-vm:1972392kB, anon-rss:1953348kB, file-rs:0kB
[2320864.954447] TCP: Possible SYN flooding on port 7001. Dropping request. Check SNMP ounters.
</pre>
该命令会输出系统日志的最后10行。示例中的输出，可以看见一次内核的oom kill和一次TCP丢。这些日志可以帮助排查性能问题。千万不要忘了这一步。
vmstat 1
<pre>$ vmstat 1
procs ---------memory---------- ---swap-- -----io---- -system-- ------cpu-----
 r b swpd   free   buff cache   si   so   bi   bo   in   cs us sy id wa st
34 0   0 200889792 73708 591828   0   0   0   5   6   10 96 1 3 0 0

原文链接：用十条命令在一分钟内检查Linux服务器性能

```
32  0    0 200889920  73708 591860    0    0    0   592 13284 4282 98  1  1  0  0
32  0    0 200890112  73708 591860    0    0    0     0 9501 2154 99  1  0  0  0
32  0    0 200889568  73712 591856    0    0    0    48 11900 2459 99  0  0  0  0
32  0    0 200890208  73712 591860    0    0    0     0 15898 4840 98  1  1  0  0
^C
</pre>
<p>vmstat(8) 命令，每行会输出一些系统核心指标，这些指标可以让我们更详细的了解系统状态。面跟的参数1，表示每秒输出一次统计信息，表头提示了每一列的含义，这几介绍一些和性能调优相的列：</p>
<ul>
<li>r：等待在CPU资源的进程数。这个数据比平均负载更加能够体现CPU负载情况，数据中不包含待IO的进程。如果这个数值大于机器CPU核数，那么机器的CPU资源已经饱和。</li>
<li>free：系统可用内存数（以千字节为单位），如果剩余内存不足，也会导致系统性能问题。下文绍到的free命令，可以更详细的了解系统内存的使用情况。</li>
<li>si, so：交换区写入和读取的数量。如果这个数据不为0，说明系统已经在使用交换区（swap）机器物理内存已经不足。</li>
<li>us, sy, id, wa, st：这些都代表了CPU时间的消耗，它们分别表示用户时间（user）、系统（内核时间（sys）、空闲时间（idle）、IO等待时间（wait）和被偷走的时间（stolen，一般被其他虚拟机耗）。</li>
</ul>
<p>上述这些CPU时间，可以让我们很快了解CPU是否出于繁忙状态。一般情况下，如果用户时间和统时间相加非常大，CPU出于忙于执行指令。如果IO等待时间很长，那么系统的瓶颈可能在磁盘IO。/p>
<p>示例命令的输出可以看见，大量CPU时间消耗在用户态，也就是用户应用程序消耗了CPU时间。不一定是性能问题，需要结合r队列，一起分析。</p>
<p><span>mpstat -P ALL 1</span></p>
<pre>$ mpstat -P ALL 1
Linux 3.13.0-49-generic (titanclusters-xxxxx)  07/14/2015  _x86_64_ (32 CPU)
07:38:49 PM  CPU   %usr  %nice   %sys %iowait   %irq  %soft  %steal  %guest  %gnice  %idle
07:38:50 PM  all  98.47   0.00   0.75   0.00   0.00   0.00    0.00    0.00    0.00   0.78
07:38:50 PM    0  96.04   0.00   2.97   0.00   0.00   0.00    0.00    0.00    0.00   0.99
07:38:50 PM    1  97.00   0.00   1.00   0.00   0.00   0.00    0.00    0.00    0.00   2.00
07:38:50 PM    2  98.00   0.00   1.00   0.00   0.00   0.00    0.00    0.00    0.00   1.00
07:38:50 PM    3  96.97   0.00   0.00   0.00   0.00   0.00    0.00    0.00    0.00   3.03
[...]
</pre>
<p>该命令可以显示每个CPU的占用情况，如果有一个CPU占用率特别高，那么有可能是一个单线程用程序引起的。</p>
<p><span>pidstat 1</span></p>
<pre>$ pidstat 1
Linux 3.13.0-49-generic (titanclusters-xxxxx)  07/14/2015    _x86_64_    (32 CPU)
07:41:02 PM   UID       PID    %usr %system  %guest    %CPU   CPU  Command
07:41:03 PM     0         9    0.00    0.94    0.00    0.94     1  rcuos/0
07:41:03 PM     0      4214    5.66    5.66    0.00   11.32    15  mesos-slave
07:41:03 PM     0      4354    0.94    0.94    0.00    1.89     8  java
07:41:03 PM     0      6521 1596.23    1.89    0.00 1598.11    27  java
07:41:03 PM     0      6564 1571.70    7.55    0.00 1579.25    28  java
07:41:03 PM 60004     60154    0.94    4.72    0.00    5.66     9  pidstat
07:41:03 PM   UID       PID    %usr %system  %guest    %CPU   CPU  Command
07:41:04 PM     0      4214    6.00    2.00    0.00    8.00    15  mesos-slave
07:41:04 PM     0      6521 1590.00    1.00    0.00 1591.00    27  java
07:41:04 PM     0      6564 1573.00   10.00    0.00 1583.00    28  java
07:41:04 PM   108      6718    1.00    0.00    0.00    1.00     0  snmp-pass
07:41:04 PM 60004     60154    1.00    4.00    0.00    5.00     9  pidstat
```

^C
</pre>
pidstat命令输出进程的CPU占用率，该命令会持续输出，并且不会覆盖之前的数据，可以方便察系统动态。如上的输出，可以看见两个JAVA进程占用了将近1600%的CPU时间，既消耗了大约16 CPU核心的运算资源。
iostat -xz 1
<pre>$ iostat -xz 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015 _x86_64_ (32 CPU)
avg-cpu: %user   %nice %system %iowait %steal   %idle
           73.96   0.00   3.73   0.03   0.06   22.21
Device: rrqm/s   wrqm/s   r/s   w/s   rkB/s   wkB/s avgrq-sz avgqu-sz   await r_await w_aw
it svctm %util
xvda   0.00   0.23   0.21   0.18   4.52   2.08   34.37   0.00   9.98   13.80   5.42   2.44   0
09
xvdb   0.01   0.00   1.02   8.94   127.97   598.53   145.79   0.00   0.43   1.78   0.28   0.25
 0.25
xvdc   0.01   0.00   1.02   8.86   127.79   595.94   146.50   0.00   0.45   1.82   0.30   0.27
0.26
dm-0   0.00   0.00   0.69   2.32   10.47   31.69   28.01   0.01   3.23   0.71   3.98   0.13
0.04
dm-1   0.00   0.00   0.00   0.94   0.01   3.78   8.00   0.33   345.84   0.04 346.81   0.01
0.00
dm-2   0.00   0.00   0.09   0.07   1.35   0.36   22.50   0.00   2.55   0.23   5.62   1.78   0
03
[...]
^C
</pre>
iostat命令主要用于查看机器磁盘IO情况。该命令输出的列，主要含义是：

r/s, w/s, rkB/s, wkB/s：分别表示每秒读写次数和每秒读写数据量（千字节）。读写量过大，可会引起性能问题。
await：IO操作的平均等待时间，单位是毫秒。这是应用程序在和磁盘交互时，需要消耗的时间包括IO等待和实际操作的耗时。如果这个数值过大，可能是硬件设备遇到了瓶颈或者出现故障。
avgqu-sz：向设备发出的请求平均数量。如果这个数值大于1，可能是硬件设备已经饱和（部分端硬件设备支持并行写入）。
%util：设备利用率。这个数值表示设备的繁忙程度，经验值是如果超过60，可能会影响IO性能可以参照IO操作平均等待时间）。如果到达100%，说明硬件设备已经饱和。

如果显示的是逻辑设备的数据，那么设备利用率不代表后端实际的硬件设备已经饱和。值得注意是，即使IO性能不理想，也不一定意味这应用程序性能会不好，可以利用诸如预读取、写缓存等策略升应用性能。
free –m
<pre>$ free -m
             total   used   free   shared   buffers   cached
Mem: 245998   24545   221453   83   59   541
-/+ buffers/cache: 23944   222053
Swap: 0   0   0
</pre>
free命令可以查看系统内存的使用情况，-m参数表示按照兆字节展示。最后两列分别表示用于I缓存的内存数，和用于文件系统页缓存的内存数。需要注意的是，第二行-/+ buffers/cache，看上缓存占用了大量内存空间。这是Linux系统的内存使用策略，尽可能的利用内存，如果应用程序需要存，这部分内存会立即被回收并分配给应用程序。因此，这部分内存一般也被当成是可用内存。
如果可用内存非常少，系统可能会动用交换区（如果配置了的话），这样会增加IO开销（可以在istat命令中提现），降低系统性能。

sar -n DEV 1
<pre>$ sar -n DEV 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015 _x86_64_ (32 CPU)
12:16:48 AM IFACE rxpck/s txpck/s rxkB/s txkB/s rxcmp/s txcmp/s rxmcst/s %ifuti

12:16:49 AM eth0 18763.00 5032.00 20686.42 478.30 0.00 0.00 0.00 0.00
12:16:49 AM lo 14.00 14.00 1.36 1.36 0.00 0.00 0.00 0.00
12:16:49 AM docker0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
12:16:49 AM IFACE rxpck/s txpck/s rxkB/s txkB/s rxcmp/s txcmp/s rxmcst/s %ifuti

12:16:50 AM eth0 19763.00 5101.00 21999.10 482.56 0.00 0.00 0.00 0.00
12:16:50 AM lo 20.00 20.00 3.25 3.25 0.00 0.00 0.00 0.00
12:16:50 AM docker0 0.00 0.00 0.00 0.00 0.00 0.00 0.00 0.00
^C
</pre>
sar命令在这里可以查看网络设备的吞吐率。在排查性能问题时，可以通过网络设备的吞吐量，断网络设备是否已经饱和。如示例输出中，eth0网卡设备，吞吐率大概在22 Mbytes/s，既176 Mbits sec，没有达到1Gbit/sec的硬件上限。
sar -n TCP,ETCP 1
<pre>$ sar -n TCP,ETCP 1
Linux 3.13.0-49-generic (titanclusters-xxxxx) 07/14/2015 _x86_64_ (32 CPU)
12:17:19 AM active/s passive/s iseg/s oseg/s
12:17:20 AM 1.00 0.00 10233.00 18846.00
12:17:19 AM atmptf/s estres/s retrans/s isegerr/s orsts/s
12:17:20 AM 0.00 0.00 0.00 0.00 0.00
12:17:20 AM active/s passive/s iseg/s oseg/s
12:17:21 AM 1.00 0.00 8359.00 6039.00
12:17:20 AM atmptf/s estres/s retrans/s isegerr/s orsts/s
12:17:21 AM 0.00 0.00 0.00 0.00 0.00
^C
</pre>
sar命令在这里用于查看TCP连接状态，其中包括：

active/s：每秒本地发起的TCP连接数，既通过connect调用创建的TCP连接；
passive/s：每秒远程发起的TCP连接数，即通过accept调用创建的TCP连接；
retrans/s：每秒TCP重传数量；

TCP连接数可以用来判断性能问题是否由于建立了过多的连接，进一步可以判断是主动发起的连，还是被动接受的连接。TCP重传可能是因为网络环境恶劣，或者服务器压力过大导致丢包。
top
<pre>$ top
top - 00:15:40 up 21:56, 1 user, load average: 31.09, 29.87, 29.92
Tasks: 871 total, 1 running, 868 sleeping, 0 stopped, 2 zombie
%Cpu(s): 96.8 us, 0.4 sy, 0.0 ni, 2.7 id, 0.1 wa, 0.0 hi, 0.0 si, 0.0 st
KiB Mem: 25190241+total, 24921688 used, 22698073+free, 60448 buffers
KiB Swap: 0 total, 0 used, 0 free. 554208 cached Mem
  PID USER PR NI VIRT RES SHR S %CPU %MEM TIME+ COMMAND
 20248 root 20 0 0.227t 0.012t 18748 S 3090 5.2 29812:58 java
  4213 root 20 0 2722544 64640 44232 S 23.5 0.0 233:35.37 mesos-slave
 66128 titancl+ 20 0 24344 2332 1172 R 1.0 0.0 0:00.07 top
  5235 root 20 0 38.227g 547004 49996 S 0.7 0.2 2:02.74 java
  4299 root 20 0 20.015g 2.682g 16836 S 0.3 1.1 33:14.42 java
     1 root 20 0 33620 2920 1496 S 0.0 0.0 0:03.82 init
     2 root 20 0 0 0 0 S 0.0 0.0 0:00.02 kthreadd</pre>

```
    3 root      20   0       0      0      0 S   0.0  0.0   0:05.35 ksoftirqd/0
    5 root       0 -20       0      0      0 S   0.0  0.0   0:00.00 kworker/0:0H
    6 root      20   0       0      0      0 S   0.0  0.0   0:06.94 kworker/u256:0
    8 root      20   0       0      0      0 S   0.0  0.0   2:38.05 rcu_sched
</pre>
<p>top命令包含了前面好几个命令的检查的内容。比如系统负载情况（uptime）、系统内存使用情（free）、系统CPU使用情况（vmstat）等。因此通过这个命令，可以相对全面的查看系统负载的来。同时，top命令支持排序，可以按照不同的列排序，方便查找出诸如内存占用最多的进程、CPU占率最高的进程等。</p>
<p>但是，top命令相对于前面一些命令，输出是一个瞬间值，如果不持续盯着，可能会错过一些线。这时可能需要暂停top命令刷新，来记录和比对数据。</p>
<p><span>总结</span></p>
<p>排查Linux服务器性能问题还有很多工具，上面介绍的一些命令，可以帮助我们快速的定位问题例如前面的示例输出，多个证据证明有JAVA进程占用了大量CPU资源，之后的性能调优就可以针对用程序进行。</p>
<p> </p>
<p>还没实践过，但感觉很高级的样子</p>
<p>转自： http://www.infoq.com/cn/news/2015/12/linux-performance</p>
<p> </p>
```

原文链接：用十条命令在一分钟内检查Linux服务器性能