Linux 性能分析的前 60 秒

<https://linux.cn/article-6924-1-rel.html>

<https://segmentfault.com/a/1190000004104493>

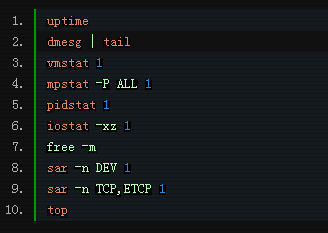
为了解决性能问题，你登入了一台 Linux 服务器，在最开始的一分钟内需要查看什么？

在 Netflix 我们有一个庞大的 EC2 Linux 集群，还有非常多的性能分析工具来监控和调查它的性能。其中包括用于云监控的Atlas，用于实例按需分析的 Vector。即使这些工具帮助我们解决了大多数问题，我们有时还是得登入 Linux 实例，运行一些标准的 Linux 性能工具来解决问题。

在这篇文章里，Netflix Performance Engineering 团队将使用居家常备的 Linux 标准命令行工具，演示在性能调查最开始的60秒里要干的事，

**最开始的60秒......**

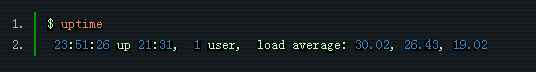
运行下面10个命令，你可以在60秒内就对系统资源的使用情况和进程的运行状况有大体上的了解。无非是先查看错误信息和饱和指标，再看下资源的使用量。这里“饱和”的意思是，某项资源供不应求，已经造成了请求队列的堆积，或者延长了等待时间。



有些命令需要你安装 sysstat 包。（译注：指 mpstat, pidstat, iostat和sar，用包管理器直接安装 sysstat 即可） 这些命令所提供的指标能够帮助你实践 USE 方法：这是一种用于定位性能瓶颈的方法论。你可以以此检查所有资源（CPU，内存，硬盘，等等）的使用量是否饱和，以及是否存在错误。同时请留意上一次检查正常的时刻，这将帮助你减少待分析的对象，并指明调查的方向。（译注：USE 方法，就是检查每一项资源的使用量utilization、饱和saturation、错误error）

接下来的章节里我们将结合实际例子讲解这些命令。如果你想了解更多的相关信息，请查看它们的 man page。

**1. uptime**

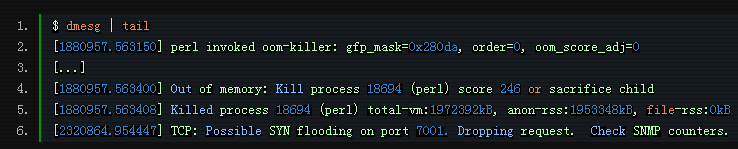


这个命令显示了要运行的任务（进程）数，通过它能够快速了解系统的平均负载。在 Linux 上，这些数值既包括正在或准备运行在 CPU 上的进程，也包括阻塞在不可中断 I/O（通常是磁盘 I/O）上的进程。它展示了资源负载（或需求）的大致情况，不过进一步的解读还有待其它工具的协助。对它的具体数值不用太较真。

最右的三个数值分别是1分钟、5分钟、15分钟系统负载的移动平均值。它们共同展现了负载随时间变动的情况。举个例子，假设你被要求去检查一个出了问题的服务器，而它最近1分钟的负载远远低于15分钟的负载，那么你很可能已经扑了个空。

在上面的例子中，负载均值最近呈上升态势，其中1分钟值高达30，而15分钟值仅有19。这种现象有许多种解释，很有可能是对 CPU 的争用；该系列的第3个和第4个命令——vmstat和mpstat——可以帮助我们进一步确定问题所在。

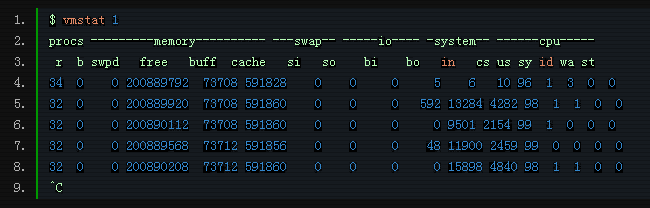
**2. dmesg | tail**



这个命令显示了最新的10个系统信息，如果有的话。注意会导致性能问题的错误信息。上面的例子里就包括对过多占用内存的某进程的死刑判决，还有丢弃 TCP 请求的公告。

不要漏了这一步！检查dmesg总是值得的。

**3. vmstat 1**



vmstat(8)，是 “virtual memory stat” 的简称，几十年前就已经包括在 BSD 套件之中，一直以来都是居家常备的工具。它会逐行输出服务器关键数据的统计结果。

通过指定1作为 vmstat 的输入参数，它会输出每一秒内的统计结果。（在我们当前使用的）vmstat 输出的第一行数据是从启动到现在的平均数据，而不是前一秒的数据。所以我们可以跳过第一行，看看后面几行的情况。

检查下面各列：

r：等待 CPU 的进程数。该指标能更好地判定 CPU 是否饱和，因为它不包括 I/O。简单地说，r 值高于 CPU 数时就意味着饱和。

free：空闲的内存千字节数。如果你数不清有多少位，就说明系统内存是充足的。接下来要讲到的第7个命令，free -m，能够更清楚地说明空闲内存的状态。

si，so：交换分区换入和换出。如果它们不为零，意味着内存已经不足，开始动用交换空间的存粮了。

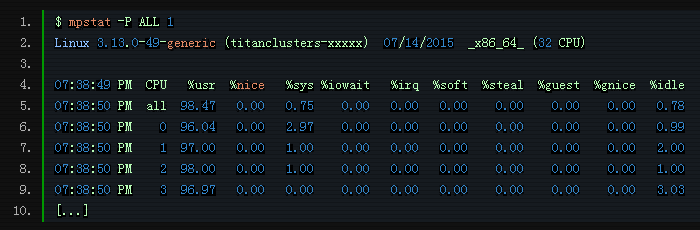
us，sy，id，wa，st：它们是所有 CPU 的使用百分比。它们分别表示 用户态用时user time，系统态用时system time（处于内核态的时间），空闲idle，I/O 等待wait I/O和偷去的时间steal time（被其它租户，或者是租户自己的 Xen 隔离设备驱动域isolated driver domain所占用的时间）。

通过相加 us 和 sy 的百分比，你可以确定 CPU 是否处于忙碌状态。一个持续不变的 I/O 等待意味着瓶颈在硬盘上，这种情况往往伴随着 CPU 的空闲，因为任务都卡在磁盘 I/O 上了。你可以把 I/O 等待当作 CPU 空闲的另一种形式，它额外给出了 CPU 空闲的线索。

I/O 处理同样会消耗系统时间。一个高于20%的平均系统时间，往往值得进一步发掘：也许系统花在 I/O 的时间太长了。

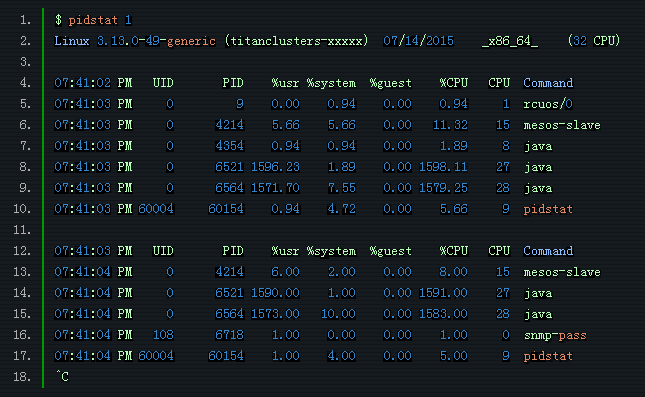
在上面的例子中，CPU 基本把时间花在用户态里面，意味着跑在上面的应用占用了大部分时间。此外，CPU 平均使用率在90%之上。这不一定是个问题；检查下“r”列，看看是否饱和了。

**4. mpstat -P ALL 1**



这个命令显示每个 CPU 的时间使用百分比，你可以用它来检查 CPU 是否存在负载不均衡。单个过于忙碌的 CPU 可能意味着整个应用只有单个线程在工作。

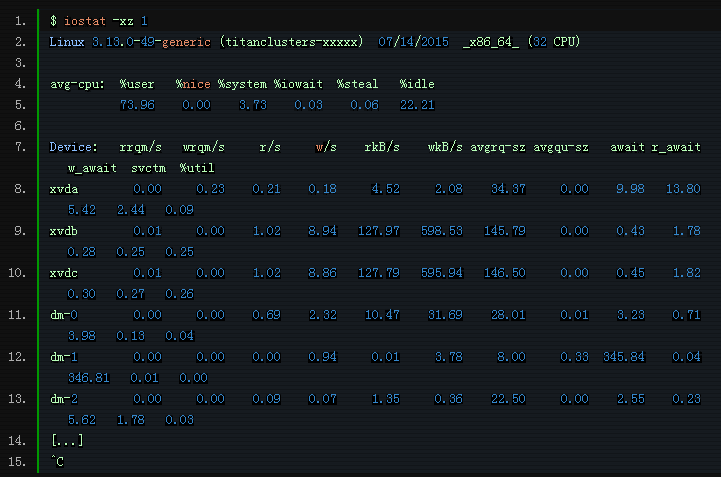
**5. pidstat 1**



pidstat看上去就像top，不过top的输出会覆盖掉之前的输出，而pidstat的输出则添加在之前的输出的后面。这有利于观察数据随时间的变动情况，也便于把你看到的内容复制粘贴到调查报告中。

上面的例子表明，CPU 主要消耗在两个 java 进程上。%CPU列是在各个 CPU 上的使用量的总和；1591%意味着 java 进程消耗了将近16个 CPU。

**6. iostat -xz 1**



这个命令可以弄清块设备（磁盘）的状况，包括工作负载和处理性能。注意以下各项：

r/s，w/s，rkB/s，wkB/s：分别表示每秒设备读次数，写次数，读的 KB 数，写的 KB 数。它们描述了磁盘的工作负载。也许性能问题就是由过高的负载所造成的。

await：I/O 平均时间，以毫秒作单位。它是应用中 I/O 处理所实际消耗的时间，因为其中既包括排队用时也包括处理用时。如果它比预期的大，就意味着设备饱和了，或者设备出了问题。

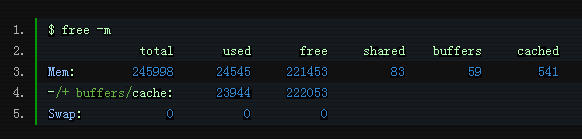
avgqu-sz：分配给设备的平均请求数。大于1表示设备已经饱和了。（不过有些设备可以并行处理请求，比如由多个磁盘组成的虚拟设备）

%util：设备使用率。这个值显示了设备每秒内工作时间的百分比，一般都处于高位。低于60%通常是低性能的表现（也可以从 await 中看出），不过这个得看设备的类型。接近100%通常意味着饱和。

如果某个存储设备是由多个物理磁盘组成的逻辑磁盘设备，100%的使用率可能只是意味着 I/O 占用。

请牢记于心，磁盘 I/O 性能低不一定是个问题。应用的 I/O 往往是异步的（比如预读read-ahead和写缓冲buffering for writes），所以不一定会被阻塞并遭受延迟。

**7. free -m**



右边的两列显示：

buffers：用于块设备 I/O 的缓冲区缓存

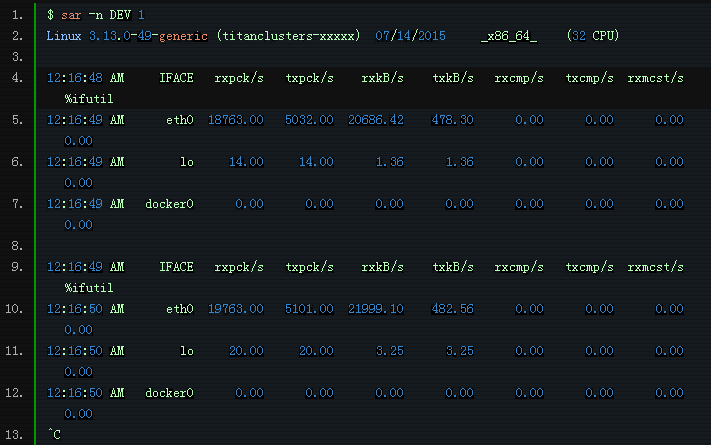
cached：用于文件系统的页缓存

它们的值接近于0时，往往导致较高的磁盘 I/O（可以通过 iostat 确认）和糟糕的性能。上面的例子里没有这个问题，每一列都有好几 M 呢。

比起第一行，-/+ buffers/cache提供的内存使用量会更加准确些。Linux 会把暂时用不上的内存用作缓存，一旦应用需要的时候立刻重新分配给它。所以部分被用作缓存的内存其实也算是空闲内存，第二行以此修订了实际的内存使用量。为了解释这一点， 甚至有人专门建了个网站： LinuxAteMyRam.com。

如果你在 Linux 上安装了 ZFS，正如我们在一些服务上所做的，这一点会变得更加迷惑，因为 ZFS 它自己的文件系统缓存不算入free -m。有时系统看上去已经没有多少空闲内存可用了，其实内存都待在 ZFS 的缓存里呢。

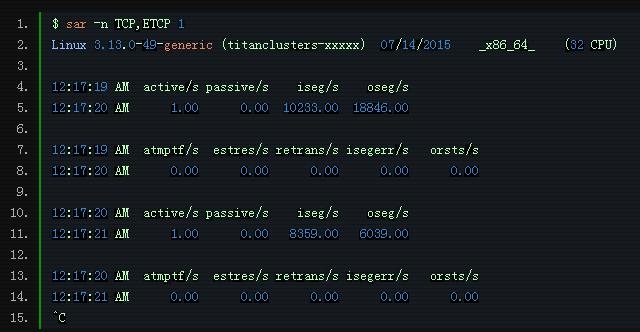
**8. sar -n DEV 1**



这个命令可以用于检查网络流量的工作负载：rxkB/s 和 txkB/s，以及它是否达到限额了。上面的例子中，eth0接收的流量达到 22Mbytes/s，也即 176Mbits/sec（限额是 1Gbit/sec）

我们用的版本中还提供了%ifutil作为设备使用率（接收和发送两者中的最大值）的指标。我们也可以用 Brendan 的 nicstat计量这个值。一如nicstat，sar显示的这个值不一定是对的，在这个例子里面就没能正常工作（0.00）。

**9. sar -n TCP,ETCP 1**



这个命令显示一些关键TCP指标的汇总。其中包括：

active/s：本地每秒创建的 TCP 连接数（比如 concept() 创建的）

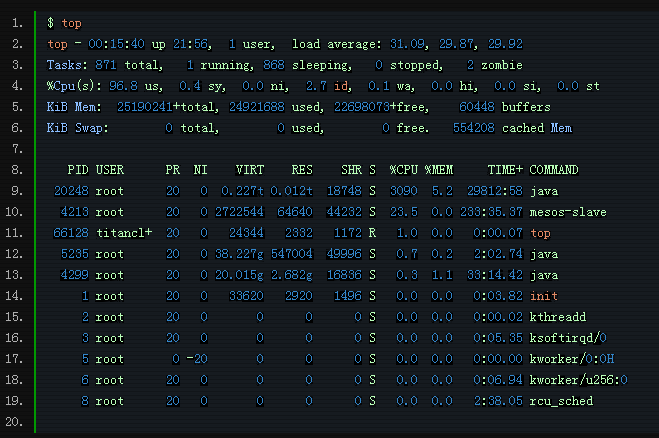
passive/s：远程每秒创建的 TCP 连接数（比如 accept() 创建的）

retrans/s：每秒 TCP 重传次数

主动连接数active和被动连接数passive通常可以用来粗略地描述系统负载。可以认为主动连接是对外的，而被动连接是对内的，虽然严格来说不完全是这个样子。（比如，一个从 localhost 到 localhost 的连接）

重传是网络或系统问题的一个信号；它可能是不可靠的网络（比如公网）所造成的，也有可能是服务器已经过载并开始丢包。在上面的例子中，每秒只创建一个新的 TCP 连接。

**10. top**



top命令包括很多我们之前检查过的指标。它适合用来查看相比于之前的命令输出的结果，负载有了哪些变动。

不能清晰显示数据随时间变动的情况，这是top的一个缺点。相较而言，vmstat和pidstat的输出不会覆盖掉之前的结果，因此更适合查看数据随时间的变动情况。另外，如果你不能及时暂停top的输出（Ctrl-s 暂停，Ctrl-q 继续），也许某些关键线索会湮灭在新的输出中。

**在这之后...**

有很多工具和方法论有助于你深入地发掘问题。Brendan 在2015年 Velocity 大会上的 Linux Performance Tools tutorial 中列出超过40个命令，覆盖了观测、基准测试、调优、静态性能调优、分析profile，和追踪tracing多个方面。