# 浅谈Linux下查看某一进程占用的内存

内核补给站

## 概述

想必在linux上写过程序的同学都有分析进程占用多少内存的经历，或者被问到这样的问题——你的程序在运行时占用了多少内存（物理内存）？通常我们可以通过top命令查看进程占用了多少内存。这里我们可以看到VIRT、RES和SHR三个重要的指标，他们分别代表什么意思呢？这是本文需要跟大家一起探讨的问题。当然如果更加深入一点，你可能会问进程所占用的那些物理内存都用在了哪些地方？这时候top命令可能不能给到你你所想要的答案了，不过我们可以分析proc文件系统提供的smaps文件，这个文件详尽地列出了当前进程所占用物理内存的使用情况。

## 关于内存的两个概念

要理解top命令关于内存使用情况的输出，我们必须首先搞清楚虚拟内存（Virtual Memory）和驻留内存（Resident Memory）两个概念。

## 【虚拟内存】

首先需要强调的是虚拟内存不同于物理内存，虽然两者都包含内存字眼但是它们属于两个不同层面的概念。进程占用虚拟内存空间大并非意味着程序的物理内存也一定占用很大。虚拟内存是操作系统内核为了对进程地址空间进行管理（process address space management）而精心设计的一个逻辑意义上的内存空间概念。我们程序中的指针其实都是这个虚拟内存空间中的地址。比如我们在写完一段C++程序之后都需要采用g++进行编译，这时候编译器采用的地址其实就是虚拟内存空间的地址。因为这时候程序还没有运行，何谈物理内存空间地址？凡是程序运行过程中可能需要用到的指令或者数据都必须在虚拟内存空间中。既然说虚拟内存是一个逻辑意义上（假象的）的内存空间，为了能够让程序在物理机器上运行，那么必须有一套机制可以让这些假象的虚拟内存空间映射到物理内存空间（实实在在的RAM内存条上的空间）。这其实就是操作系统中页映射表（page table）所做的事情了。内核会为系统中每一个进程维护一份相互独立的页映射表。。页映射表的基本原理是将程序运行过程中需要访问的一段虚拟内存空间通过页映射表映射到一段物理内存空间上，这样CPU访问对应虚拟内存地址的时候就可以通过这种查找页映射表的机制访问物理内存上的某个对应的地址。“页（page）”是虚拟内存空间向物理内存空间映射的基本单元。

下图1演示了虚拟内存空间和物理内存空间的相互关系，它们通过Page Table关联起来。其中虚拟内存空间中着色的部分分别被映射到物理内存空间对应相同着色的部分。而虚拟内存空间中灰色的部分表示在物理内存空间中没有与之对应的部分，也就是说灰色部分没有被映射到物理内存空间中。这么做也是本着“按需映射”的指导思想，因为虚拟内存空间很大，可能其中很多部分在一次程序运行过程中根本不需要访问，所以也就没有必要将虚拟内存空间中的这些部分映射到物理内存空间上。
到这里为止已经基本阐述了什么是虚拟内存了。总结一下就是，虚拟内存是一个假象的内存空间，在程序运行过程中虚拟内存空间中需要被访问的部分会被映射到物理内存空间中。虚拟内存空间大只能表示程序运行过程中可访问的空间比较大，不代表物理内存空间占用也大。

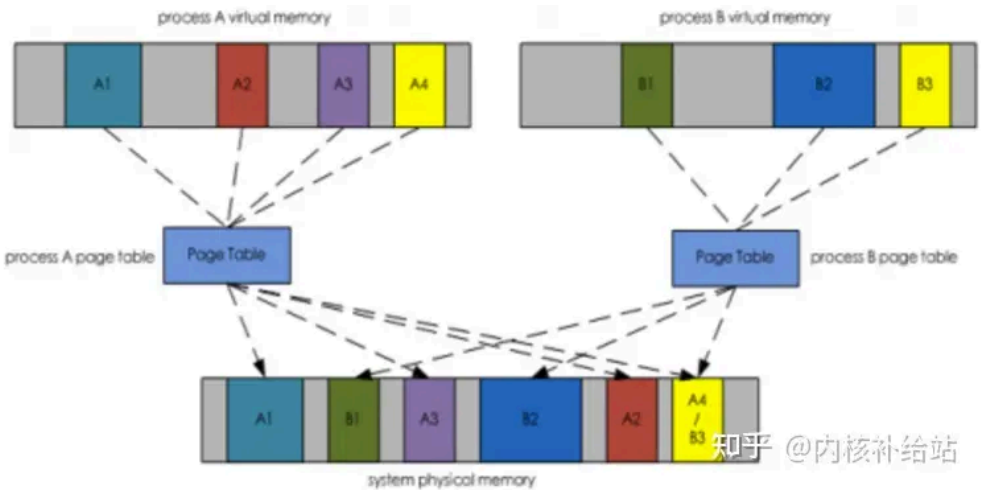



图1. 虚拟内存空间到物理内存空间映射

**【驻留内存】**

驻留内存，顾名思义是指那些被映射到进程虚拟内存空间的物理内存。上图1中，在系统物理内存空间中被着色的部分都是驻留内存。比如，A1、A2、A3和A4是进程A的驻留内存；B1、B2和B3是进程B的驻留内存。进程的驻留内存就是进程实实在在占用的物理内存。一般我们所讲的进程占用了多少内存，其实就是说的占用了多少驻留内存而不是多少虚拟内存。因为虚拟内存大并不意味着占用的物理内存大。

关于虚拟内存和驻留内存这两个概念我们说到这里。下面一部分我们来看看top命令中VIRT、RES和SHR分别代表什么意思。

### top命令中VIRT、RES和SHR的含义

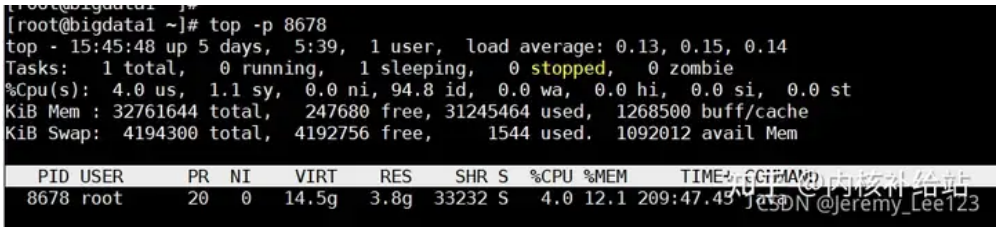
搞清楚了虚拟内存的概念之后解释VIRT的含义就很简单了。VIRT表示的是进程虚拟内存空间大小。对应到图1中的进程A来说就是A1、A2、A3、A4以及灰色部分所有空间的总和。也就是说VIRT包含了在已经映射到物理内存空间的部分和尚未映射到物理内存空间的部分总和。

RES的含义是指进程虚拟内存空间中已经映射到物理内存空间的那部分的大小。对应到图1中的进程A来说就是A1、A2、A3以及A4几个部分空间的总和。所以说，看进程在运行过程中占用了多少内存应该看RES的值而不是VIRT的值。
最后来看看SHR所表示的含义。SHR是share（共享）的缩写，它表示的是进程占用的共享内存大小。在上图1中我们看到进程A虚拟内存空间中的A4和进程B虚拟内存空间中的B3都映射到了物理内存空间的A4/B3部分。咋一看很奇怪。为什么会出现这样的情况呢？其实我们写的程序会依赖于很多外部的动态库（.so），比如libc.so、libld.so等等。这些动态库在内存中仅仅会保存/映射一份，如果某个进程运行时需要这个动态库，那么动态加载器会将这块内存映射到对应进程的虚拟内存空间中。多个进展之间通过共享内存的方式相互通信也会出现这样的情况。这么一来，就会出现不同进程的虚拟内存空间会映射到相同的物理内存空间。这部分物理内存空间其实是被多个进程所共享的，所以我们将他们称为共享内存，用SHR来表示。某个进程占用的内存除了和别的进程共享的内存之外就是自己的独占内存了。所以要计算进程独占内存的大小只要用RES的值减去SHR值即可。

## 一、 ps -ef|grep flink 和 top -p pid 组合

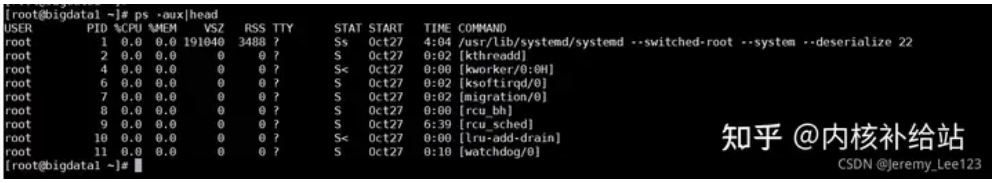
Linux下查看某一个进程所占用的内存，首先可以通过ps命令找到进程id，比如：ps -ef|grep flink，可以看到flink task这个程序的进程id


已知pid是8678，现在可以使用如下命令查看内存：




## 二、直接命令查看 ps -aux|grep flink


第一个标注的地方是CPU和内存占用率，后面的15 209 056是物理内存使用量，单位是k，此时kafka大约占用15G内存




1）USER: 行程拥有者

2）PID: 进程的ID

3）%CPU: 占用的 CPU 使用率

4）%MEM: 占用的记忆体使用率

5）VSZ: 占用的虚拟记忆体大小

6）RSS: 占用的记忆体大小

7）TTY: 终端的次要装置号码 (minor device number of tty)

8）STAT: 该行程的状态:

- D: 不可中断的静止
- R: 正在执行中
- S: 静止状态
- T: 暂停执行
- Z: 不存在但暂时无法消除
- W: 没有足够的记忆体分页可分配
- <: 高优先序的行程
- N: 低优先序的行程
- L: 有记忆体分页分配并锁在记忆体内

9）START: 行程开始时间

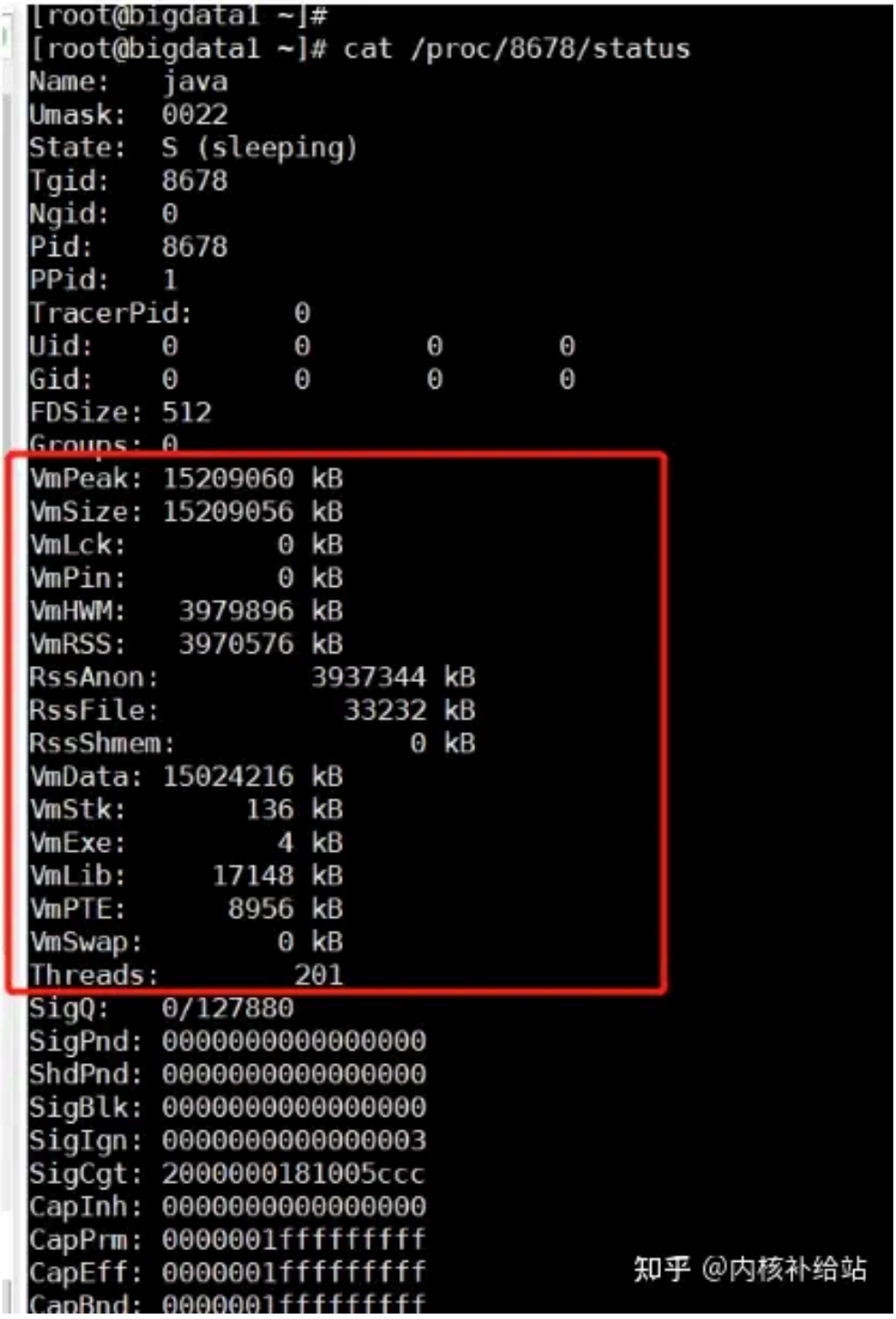
10）TIME: 执行的时间

11）COMMAND:所执行的指令

**ps aux|sort -k4,4nr|head -n 10** 查看内存占用前10名的程序


## 三、查看进程文件cat /proc/pid/status

比如：cat /proc/8678/status




VmSize对应的值就是物理内存占用，大约为943M和刚才一致

在这里我们关注VmSize|VmRSS|VmData|VmStk|VmExe|VmLib 这个6个指标，下面有一些简单的解释。

1. VmSize: 虚拟内存大小。 整个进程使用虚拟内存大小，是VmLib, VmExe, VmData, 和 VmStk 的总和。占所有虚拟内存分配（文件映射，共享内存，堆内存，任何内存）的份额，并且几乎在每次分配新内存时都会增长。几乎，因为如果在数据段中用新的堆内存分配代替了释放的旧分配，则不会分配新的虚拟内存。每当释放虚拟分配时，它都会减少。VmPeak跟踪的最大值VmSize-只能随时间增加。
2. VmLck: 虚拟内存锁。进程当前使用的并且加锁的虚拟内存总数
3. VmRSS: 虚拟内存驻留集合大小。这是驻留在物理内存的一部分。它没有交换到硬盘。它包括代码，数据和栈。随着访问内存的增加而增加，随着将页面调出到交换设备的次数减少。
4. VmData: 虚拟内存数据。堆使用的虚拟内存。随着使用堆的数据段部分而增长。由于当前的堆分配器会保留释放的内存，以防将来的分配需要它，它几乎永远不会收缩。
5. VmStk: 虚拟内存栈。栈使用的虚拟内存

6. VmExe: 可执行的虚拟内存，可执行的和静态链接库所使用的虚拟内存
7. VmLib: 虚拟内存库，动态链接库所使用的虚拟内存

**补充：**

1. VmPeak代表当前进程运行过程中占用内存的峰值.
2. VmSize代表虚拟内存总大小
3. VmLck代表进程已经锁住的物理内存的大小.锁住的物理内存不能交换到硬盘.
4. VmHWM是程序得到分配到物理内存的峰值.
5. VmRSS是程序现在使用的物理内存.
6. VmData:表示进程数据段的大小.
7. VmStk:表示进程堆栈段的大小.
8. VmExe:表示进程代码的大小.
9. VmLib:表示进程所使用LIB库的大小.
10. VmPTE:占用的页表的大小.
11. VmSwap:进程占用Swap的大小.
12. Threads:表示当前进程组的线程数量.
13. SigPnd:屏蔽位,存储了该线程的待处理信号,等同于线程的PENDING信号.
14. ShnPnd:屏蔽位,存储了该线程组的待处理信号.等同于进程组的PENDING信号.
15. SigBlk:存放被阻塞的信号,等同于BLOCKED信号.
16. SigIgn:存放被忽略的信号,等同于IGNORED信号.
17. SigCgt:存放捕获的信号,等同于CAUGHT信号.
18. CapEff:当一个进程要进行某个特权操作时,操作系统会检查cap_effective的对应位是否有效,而不再是检查进程的有效UID是否为0.
19. CapPrm:表示进程能够使用的能力,在cap_permitted中可以包含cap_effective中没有的能力，这些能力是被进程自己临时放弃的,也可以说cap_effective是cap_permitted的一个子集.
20. CapInh:表示能够被当前进程执行的程序继承的能力.
21. CapBnd:是系统的边界能力,我们无法改变它.
22. Cpus_allowed:3指出该进程可以使用CPU的亲和性掩码,因为我们指定为两块CPU,所以这里就是3,如果该进程指定为4个CPU(如果有话),这里就是F(1111).
23. Cpus_allowed_list:0-1指出该进程可以使用CPU的列表,这里是0-1.
24. voluntary_ctxt_switches表示进程主动切换的次数.
25. nonvoluntary_ctxt_switches表示进程被动切换的次数.

---

**原文链接：**浅谈Linux下查看某一进程占用的内存

编辑于 2022-08-05 16:28

Linux　Linux 内核　进程管理