理解virt res shr之间的关系 - linux

無名 (/user/99/show) 发表于: 2016-07-04　最后更新时间: 2019-08-13

本文将分为三个部分。

第一部分简要阐述虚拟内存和驻留内存这两个重要的概念；

第二部分解释top命令中VIRT、RES以及SHR三个参数的实际参考意义；

最后一部分向大家介绍一下smaps文件的格式，通过分析smaps文件我们可以详细了解进程物理内存的使用情况，比如mmap文件占用了多少空间、动态内存开辟消耗了多少空间、函数调用栈消耗了多少空间等等。

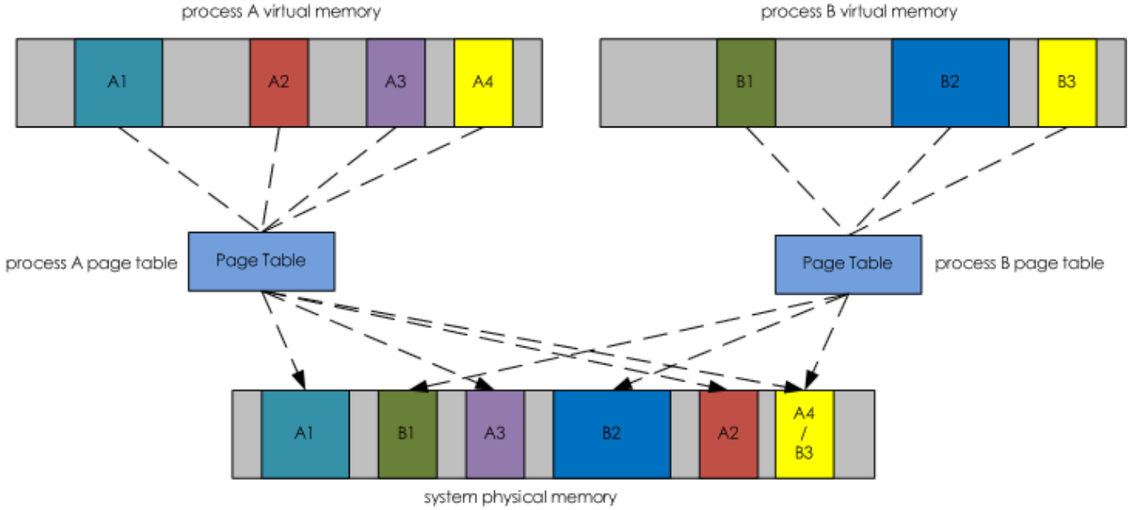
关于内存的两个概念

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。



Copy

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。

进程的smaps文件

通过top命令我们已经能看出进程的虚拟空间大小（VIRT）、占用的物理内存（RES）以及和其他进程共享的内存（SHR）。但是仅此而已，如果我想知道如下问题：

进程的虚拟内存空间的分布情况，比如heap占用了多少空间、文件映射（mmap）占用了多少空间、stack占用了多少空间？
进程是否有被交换到swap空间的内存，如果有，被交换出去的大小？
mmap方式打开的数据文件有多少页在内存中是脏页（dirty page）没有被写回到磁盘的？
mmap方式打开的数据文件当前有多少页面已经在内存中，有多少页面还在磁盘中没有加载到page cahe中？
等等

以上这些问题都无法通过top命令给出答案，但是有时候这些问题正是我们在对程序进行性能瓶颈分析和优化时所需要回答的问题。所幸的是，世界上解决问题的方法总比问题本身要多得多。linux通过proc文件系统为每个进程都提供了一个smaps文件，通过分析该文件我们就可以一一回答以上提出的问题。

在smaps文件中，每一条记录（如下图2所示）表示进程虚拟内存空间中一块连续的区域。其中第一行从左到右依次表示地址范围、权限标识、映射文件偏移、设备号、inode、文件路径。详细解释可以参见understanding-linux-proc-id-maps。

接下来8个字段的含义分别如下：

- Size：表示该映射区域在虚拟内存空间中的大小。
- Rss：表示该映射区域当前在物理内存中占用了多少空间
- Shared_Clean：和其他进程共享的未被改写的page的大小
- Shared_Dirty： 和其他进程共享的被改写的page的大小
- Private_Clean：未被改写的私有页面的大小。
- Private_Dirty： 已被改写的私有页面的大小。
- Swap：表示非mmap内存（也叫anonymous memory，比如malloc动态分配出来的内存）由于物理内存不足被swap到交换空间的大小。
- Pss：该虚拟内存区域平摊计算后使用的物理内存大小(有些内存会和其他进程共享，例如mmap进来的)。比如该区域所映射的物理内存部分同时也被另一个进程映射了，且该部分物理内存的大小为1000KB，那么该进程分摊其中一半的内存，即Pss=500KB。

```
374d602000-374d603000 r--p 00002000 08:03 192563                         /lib64/libdl-2.5.so
Size:                  4 kB
Rss:                   4 kB
Shared_Clean:          0 kB
Shared_Dirty:          0 kB
Private_Clean:         0 kB
Private_Dirty:         4 kB
Swap:                  0 kB
Pss:                   4 kB
```

有了smap如此详细关于虚拟内存空间到物理内存空间的映射信息，相信大家已经能够通过分析该文件回答上面提出的4个问题。