**深 圳 大 学 实 验 报 告**

**课程名称： 操作系统**

**实验项目名称：实验三 内存分配与回收**

**学院： 计算机与软件学院**

**专业： 计算机科学与技术**

**指导教师： 罗秋明**

**报告人：谭嘉豪 学号： 2020152087 班级： 数计班**

**实验时间： 2023年5月13日**

**实验报告提交时间： 2023年5月20日**

**教务部制**

|  |
| --- |
| **一、实验目的与要求**   * 1. **加深对内存分配与使用操作的直观认识；**   2. **掌握Linux操作系统的内存分配与使用的编程接口；**   3. **了解Linux操作系统中进程的逻辑编程地址和物理地址间的映射；**   **二、实验内容**   1. **可以使用Linux或其它Unix类操作系统；** 2. **学习该操作系统提供的分配、释放的函数使用方法；** 3. **学习该操作系统提供的进程地址映射情况的工具；**   **三、实验步骤及说明**   1. **地址转换及crash的使用**   **编写如下程序：**  **运行得到全局遍历及自定义函数的地址。**  **通过ps j命令查找进程test1的pid=2969。**  **打开crash软件，获取进程2969的进程信息。**  **一个48位的虚拟地址，由9位pgd的索引+9位pud的索引+9位pmd的索引+9位pte的索引+12位的页内偏移，这里索引的单位是字节。**  **全局变量地址的二进制表示为0000 0000 0，000 0000 00，00 0000 011，0 0000 0001， 0000 0011 1100即：**   * **Pgd：0000 0000 0。** * **Pud：000 0000 00。** * **Pmd：00 0000 011。** * **Pte：0 0000 0001。** * **页内偏移：0000 0011 1100。**   **在关闭随机内存映射的情况下。**  **获取pgd的首地址0xffff953e767fe000，取低32位0x767fe000得到首地址的物理地址。**  **Pud页表的首地址=Pgd首地址的物理地址+偏移量=0x767fe000 + 000000000 \* 8 = 0x767fe000。**  **通过rd -p命令查看0x767fe000处的数据，再通过 pte 命令得到真正的物理页地址0x4431a00。**  **在得到Pud的首地址0x4431a000后，Pmd页表的首地址=Pud首地址的物理地址+偏移量=0x4431a000+000000000 \* 8 = 0x4431a000。**  **通过rd -p命令查看0x4431a000处的数据，再通过 pte 命令得到真正的物理页地址0x177a6000。**  **在得到Pmd的首地址0x177a6000后，Pte页表的首地址= Pmd首地址的物理地址+偏移量=0x177a6000+000000011 \* 8 = 0x177a6018。**  **通过rd -p命令查看0x177a6018处的数据，再通过 pte 命令得到真正的物理页地址0x13968000。**  **在得到Pte的首地址0x13968000后，物理页框的首地址= Pte首地址的物理地址+偏移量=0x13968000 + 000000001 \* 8 = 0x13968008。**  **通过rd -p命令查看0x13968008处的数据，再通过 pte 命令得到真正的物理页地址0x20705000。**  **在得到物理页框的起始地址0x20705000后，加上页内偏移即可得到全局变量val的物理地址，即0x20705000 + 0000 0011 1100 = 0x2070503c。**  **使用rd -p 命令查看0x2070503c处的数据是否是val的值。**  **可用观察到0x2070503c的值为0x7b，转为十进制正好为123，与源代码中的val值一致。**  **接下来使用vtop命令进一步验证上述计算过程，可以看到计算过程及结果完全一致。**  **接下来进行函数地址的转换。**  **函数地址0x40057D的二进制表示为0000 0000 0，000 0000 00，00 0000 010，0 0000 0000，0101 0111 1101，即：**   * **Pgd：0000 0000 0。** * **Pud：000 0000 00。** * **Pmd：00 0000 010。** * **Pte：0 0000 0000。** * **页内偏移：0101 0111 1101。**   **由于Pgd与pud的偏移量与上述是相同的，因此Pmd的首地址也为0x177a6000。**  **在得到Pmd的首地址0x177a6000后，Pte页表的首地址= Pmd首地址的物理地址+偏移量=0x177a6000+000000010 \* 8 = 0x177a6010。**  **通过rd -p命令查看0x177a6010处的数据，再通过 pte 命令得到真正的物理页地址0x13931000。**  **在得到Pte的首地址0x13931000后，物理页框的首地址= Pte首地址的物理地址+偏移量=0x13931000 + 0 0000 0000 \* 8 = 0x13931000。**  **通过rd -p命令查看0x13931000处的数据，再通过 pte 命令得到真正的物理页地址0x1974f000。**  **在得到物理页框的起始地址0x1974f000后，加上页内偏移即可得到函数f的函数地址的物理地址，即0x1974f000 + 0101 0111 1101 = 0x1974F57d。**  **使用通过rd -p命令查看0x1974f57d处的数据，该数据指向函数所在的内存地址。**  **函数地址的物理地址存储的值是指该函数所在的内存地址。在程序运行时，函数在内存中会被分配一个地址，这个地址就是函数的物理地址。函数地址的物理地址存储的值在程序中被用来调用该函数。当程序需要调用一个函数时，它会根据该函数的物理地址在内存中找到对应的函数代码并执行。**  **使用vtop命令解析函数地址。**  **通过比较全局变量的pte与函数地址的pte可以发现，两者的地址差距较大，结合计算机系统2的知识，全局变量地址通常是存储在数据段（.data或.bss）中，而函数地址通常是存储在代码段（.text）中。因此，它们在内存中的位置不同。**   1. **进程空间分配算法**   **编写如下程序test2.c：**  **编译并运行程序，初始并未开始进行内存分配，通过ps命令获得进程号7040。**  **通过cat /proc/7040/maps查看虚拟内存，通过cat /proc/7040/status检查文件中关于内存的情况。**  **初始堆区有一块大小为132KB的内存块，经过查询资料知，这132KB的堆空间叫做arena，此时因为是主线程分配的，所以叫做main arena（每个arena中含有多个chunk，这些chunk以链表的形式加以组织）。由于132KB比1000字节大很多，所以主线程后续再声请堆空间的话，就会先从这132KB的剩余部分中申请，直到用完或不够用的时候，再通过增加program break location的方式来增加main arena的大小。同理，当main arena中有过多空闲内存的时候，也会通过减小program break location的方式来缩小main arena的大小。**  **此时虚拟内存大小为2496KB。**  **现在连续申请6块128MB的内存，注意每一块内存间存在一块大小为4KB的缓冲区。**  **发现多了一块大小为6 \* 128MB = 768M的内存，说明分配的地址是连续的。**  **此时虚拟内存地址变为788952KB（约770MB）。**  **释放2、3、5号的128MB内存。**  **查看maps，发现heap的内存地址不再连续，且由地址可知，剩下的为默认132KB大小地址及1号、4号、6号空间地址，且大小均为128M。**  **查看虚拟内存大小，变为了395724KB，减小了788952-395724=393228KB=3 \* 128MB \* 1024 + 3 \* 4，恰好是3个128MB空间及3个4KB的缓冲区。**  **此时再次分配1024MB空间，查看maps与status。**  **在原6号空间的地址范围为7f2953a8f000-7f295ba90000，大小为128MB，在继续分配1024MB后，发现6号空间的地址变为了7f2913a8e000-7f295ba90000，大小为1152MB=128MB+1024MB，说明后分配的1024MB空间紧跟着6号空间，。**  **虚拟内存大小也变为了1444304KB，又增加了1444304-395724=1048580KB=1024MB \* 1024 + 4KB，恰好是1个1024MB空间和1个4KB缓冲区。**  **最后再分配64MB内存空间，该空间的地址应当与尾地址与未释放的2号空间的尾地址相同，1号空间的大小变为了192MB，实测与预测相同。**  **查看maps，status。**  **可以发现1号空间的大小变为了192MB，在原1号空间的地址范围为7f297ba94000-7f2983a95000，大小为128MB，在继续分配64MB后，发现1号空间的地址变为了7f2977a93000-7f2983a95000，大小为192MB=128MB+64MB，说明最后分配的64MB空间紧跟着1号空间,64MB的尾地址与未释放的2号空间的尾地址相同。**  **实验过程中：**  **用户空间的地址为虚拟地址，通过映射的方式映射到物理地址，因此使用的是离散分配算法。**  **由于在连续申请了6个128MB的空间后，释放了2、3、5号空间后，发现剩下空间不连续，因此用户空间存在内存碎片问题。**  **此时内存分布情况如下：**  **在申请1024MB空间后：**  **在申请64MB后：**  **申请64MB空间后，在2、3号空间与5号空间大小均满足条件的大小的情况下，从该空间占据的是空闲空间较大、更近的2、3号空间，而不是空闲空间较小、更远的5号空间，可以得知用户进程空间分配算法使用的是首次适应算法。**   1. **系统单个进程所能分配的最大虚拟内存**   **首先通过ulimit -v unlimited命令取消对单个进程虚拟内存空间大小的限制，使其能够分配系统最大可用的虚拟内存空间。**  **编写如下程序，以128MB为单位不断申请内存，但是不写入数据，直到程序被killed。**  **在不向内存写入数据的情况下，系统单个进程虚拟内存空间大小最大为46905894MB，但实际上物理内存没有这么多。**  **修改程序，对上述申请内存的同时对其写入数据。**  **可以看到一共分配了3072MB内存，也就是3GB。**  **修改代码，分配3072MB内存，用/proc/PID/maps展示此时进程空间使用情况，观察所分配空间在什么区域，检查进程剩余可用空间有多少。**  **运行程序，在另一个终端查看进程空间使用情况。**  **可以看到进程所分配空间在堆区。**  **进程剩余可用空间为37MB。**   1. **读操作与写操作关于虚拟内存和物理内存的使用情况**   **按照题意编写如下程序。**  **运行程序，在分配内存前，虚拟内存大小（VmSize）为2496KB，物理内存（VmRSS）大小为508KB。**  **在分配256MB内存后，虚拟内存大小（VmSize）为264644KB，增加了262148KB（约256MB），物理内存（VmRSS）大小不变，仍为508KB。**  **在对这256MB空间间隔4KB进行读操作后，虚拟内存大小不变，仍为264644KB，物理内存大小变为1196KB。**  **在对这256MB空间间隔4KB进行写操作后，虚拟内存大小不变，仍为264644KB，物理内存大小也变为263280KB。**  **通过观察发现，在分配完256MB的空间后，读和写内存时的虚拟内存大小不再发生改变，但是在读操作时，物理内存大小相对于写操作变化较小。**  **因为Linux使用了写时复制（Copy-On-Write）机制和页表来管理内存。在读内存时，Linux只需要将需要读取的数据从物理内存中读取到缓存中，并不会对物理内存造成太大的影响。而在写内存时，如果涉及到的内存页没有被修改过，Linux会直接在物理内存中进行修改；但如果需要修改的内存页已经被修改过，为了保证数据的一致性，Linux会先将该内存页复制一份，然后再将修改写入复制出来的页中，这就会导致物理内存大小的变化较大。因此，写时复制机制可以减少内存的复制和分配，提高内存的利用率，但也可能会影响程序的性能。**   1. **分配物理页帧并实现抢占**   **使用free -m命令查看可用内存，可用物理内存有1530MB，虚拟内存交换区有1873MB，可用内存共3403MB。**  **编写程序，先后申请分配1500MB内存空间，总共3000MB。**  **在两个终端运行程序，得到两个进程3790和3791。**  **在两个进程都未申请分配内存前，通过/proc/meminfo | grep Mem查看可用内存有1516976KB。**  **接下来进程3790申请分配1500MB内存并使用。**  **发现可用内存发生减少，只剩大约270MB。**  **使用cat /proc/3790/smaps查看物理内存确实使用了1500MB。**  **同样cat /proc/3790/status也是。**  **此时进程3791开始使用内存，并且运行过程由于需要抢占物理内存，运行时间相较于进程3790需要更长。**  **此时发现可用物理内存只剩下75MB。**  **查看进程3790的smaps，发现使用的物理内存显著减少。**  **查看进程3791的smaps，发现使用的物理内存增加了1500MB，显然进程3791抢占了进程3790的物理内存。**    **四、感想及其他**  **本次实验过程中，安装crash遇到了较多麻烦，最初在Ubuntu20.04上安装不成功，后来在Centos7上安装成功并进行实验。**  **这次实验我主要掌握了页表的地址映射计算，了解了Linux上的内存分配算法等。** |

**深圳大学学生实验报告用纸**

|  |
| --- |
| **指导教师批阅意见：**  **成绩评定：**  **指导教师签字：**  **2023年 月 日** |
| **备注：** |

**注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。**

**2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。**