Лабораторная работа №5

Управление памятью в ОС Linux

Выполнила студентка группы М3212

Авсюкевич Анастасия

**Рассматриваемые вопросы:**

1. Использование утилиты top для мониторинга параметров памяти

2. Использование имитационных экспериментов для анализа работы механизмов управления памятью.

**Задание на лабораторную работу:**

**Эксперимент №1**

Данные о текущей конфигурации операционной системы в контексте управления памятью.

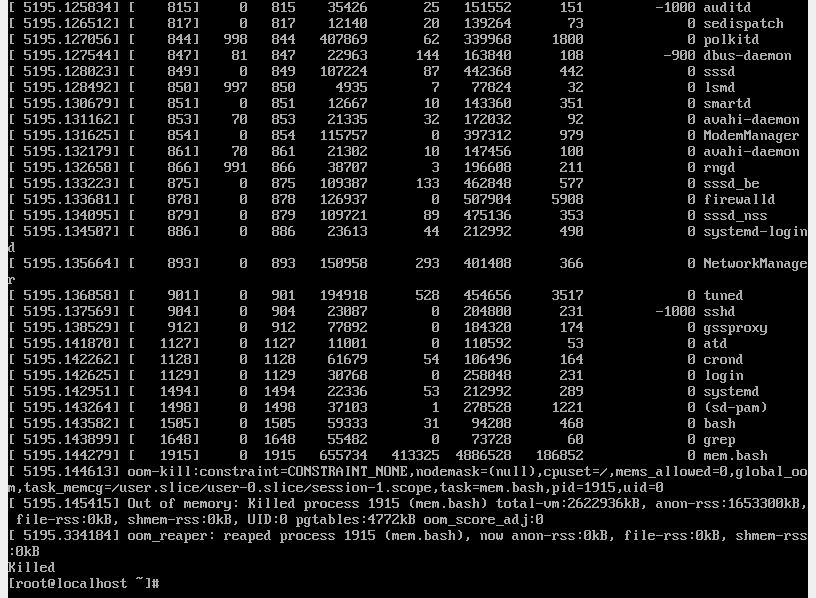
1) Общий объем оперативной памяти – 1,8 Gi (Гб)

2) Объем раздела подкачки - 820 Mi (Мб)

3) Размер страницы виртуальной памяти – 4096 Байт

4) Объем свободной физической памяти в ненагруженной системе – 1,4 Gi (Гб)

5) Объем свободного пространства в разделе подкачки в ненагруженной системе – 696 Mi (Мб)



Параметры аварийной остановки:

total-vm: 2622936 kB

anon-rss: 1653300 kB

file-rss: 0 kB

shmem-rss: 0 kB

UID: 0

pgtables: 4772 kB

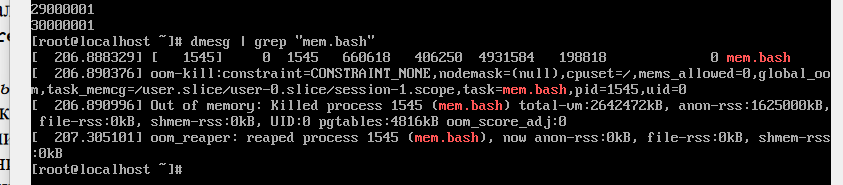
oom\_score\_adj: 0

Последняя строка report.log: 30000001

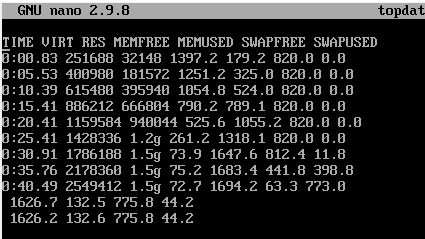
Первый этап:

Последняя строка report.log: 30000001

Последние две записи о скрипте в системном журнале:

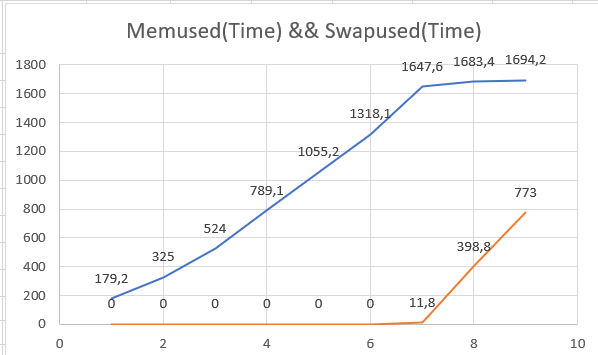


Данные команды top в течение работы скрипта:

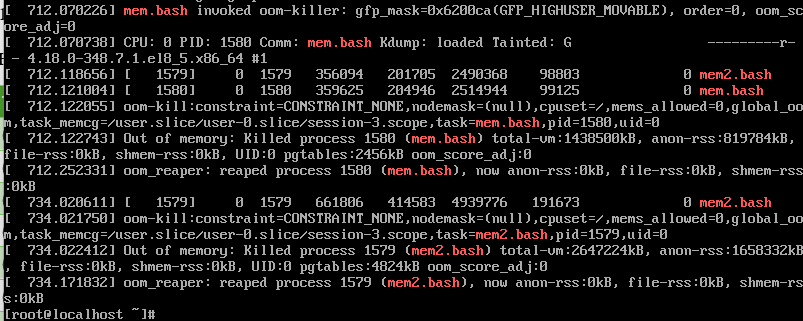


|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIME** | **VIRT** | **RES** | **MEMFREE** | **MEMUSED** | **SWAPFREE** | **SWAPUSED** |
| 0:00.83 | 251688 | 32148 | 1397,2 | 179,2 | 820 | 0 |
| 0:05.53 | 400980 | 181572 | 1251,2 | 325,0 | 820 | 0 |
| 0:10.39 | 615480 | 395940 | 1054,8 | 524,0 | 820 | 0 |
| 0:15.41 | 886212 | 666804 | 790,2 | 789,1 | 820 | 0 |
| 0:20.41 | 1159584 | 940044 | 525,6 | 1055,2 | 820 | 0 |
| 0:25.41 | 1428336 | 1,2g | 261,2 | 1318,1 | 820 | 0 |
| 0:30.91 | 1786188 | 1.5g | 73,9 | 1647,6 | 812,4 | 11,8 |
| 0:35.76 | 2178360 | 1.5g | 75,2 | 1683,4 | 441,8 | 398,8 |
| 0:40.49 | 2549412 | 1.5g | 72,7 | 1694,2 | 63,3 | 773 |

График зависимости используемой оперативной памяти и памяти раздела подкачки от времени:



Второй этап:



Последняя строка report.log: 15000001

Последняя строка report2.log: 30000001

Данные команды top в течение работы скрипта mem.bash:

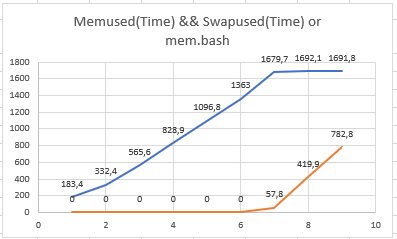
|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIME** | **VIRT** | **RES** | **MEMFREE** | **MEMUSED** | **SWAPFREE** | **SWAPUSED** |
| 0:00.44 | 238488 | 19104 | 1393,8 | 183,4 | 820 | 0 |
| 0:02.83 | 315444 | 95928 | 1243,5 | 332,4 | 820 | 0 |
| 0:05.33 | 440844 | 221592 | 1015,1 | 565,6 | 820 | 0 |
| 0:07.87 | 574428 | 355176 | 752,9 | 828,9 | 820 | 0 |
| 0:10.40 | 711180 | 491928 | 482,7 | 1096,8 | 820 | 0 |
| 0:12.96 | 850176 | 630792 | 217,3 | 1363 | 820 | 0 |
| 0:15.62 | 1032204 | 782120 | 74,6 | 1679,7 | 770 | 57,8 |
| 0:17.82 | 1213572 | 787952 | 80,5 | 1692,1 | 412,1 | 419,9 |
| 0:20.06 | 1399428 | 797196 | 74 | 1691,8 | 55,5 | 782,8 |

Данные команды top в течение работы скрипта mem2.bash:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **TIME** | **VIRT** | **RES** | **MEMFREE** | **MEMUSED** | **SWAPFREE** | **SWAPUSED** |
| 0:00.45 | 238752 | 19300 | 1394,6 | 182,6 | 820 | 0 |
| 0:02.82 | 313596 | 94012 | 1244,1 | 332,8 | 820 | 0 |
| 0:05.29 | 437280 | 217828 | 1017,8 | 563 | 820 | 0 |
| 0:07.83 | 571392 | 351940 | 754,7 | 825,1 | 820 | 0 |
| 0:10.37 | 707484 | 487900 | 488,1 | 1092,2 | 820 | 0 |
| 0:12.94 | 844236 | 624652 | 219,7 | 1361,6 | 820 | 0 |
| 0:15.61 | 1024416 | 774440 | 75 | 1678,9 | 770 | 57,8 |
| 0:17.80 | 1203408 | 782704 | 80.4 | 1691,2 | 412,1 | 419,9 |
| 0:20.04 | 1386228 | 784212 | 74,8 | 1692,3 | 51,8 | 782,8 |
| 0:24.82 | 1776816 | 1.1g | 534.6 | 1238,3 | 387,6 | 432,4 |
| 0:29.79 | 2184696 | 1.5g | 130 | 1638,8 | 387,6 | 432,4 |
| 0:34.22 | 2547168 | 1.5g | 74 | 1692,3 | 97,5 | 749,4 |

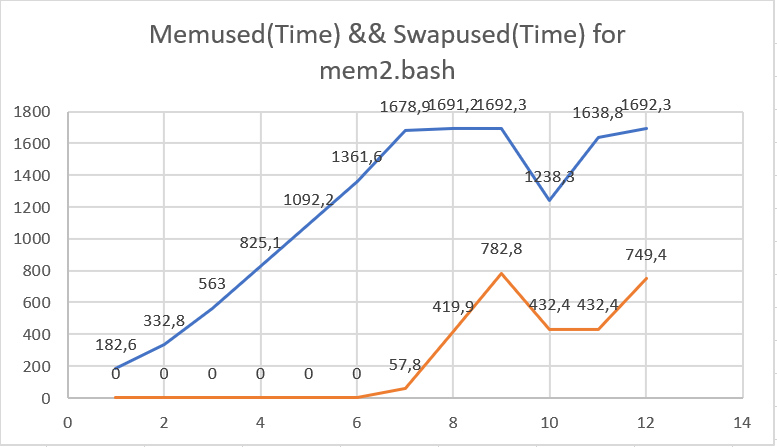
В ходе проведения эксперимента я заметила, что начиная с примерно 0:15 секунды начал использоваться раздел подкачки, так как физическая память уже почти закончилось (ее стало не хватать на хранение и обработку очередной страницы). Это произошло и для скрипта mem.bash и для mem2.bash.

Графики зависимости используемой оперативной памяти и памяти раздела подкачки от времени для mem.bash:



\*for

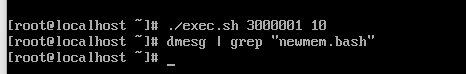
Графики зависимости используемой оперативной памяти и памяти раздела подкачки от времени для mem2.bash:



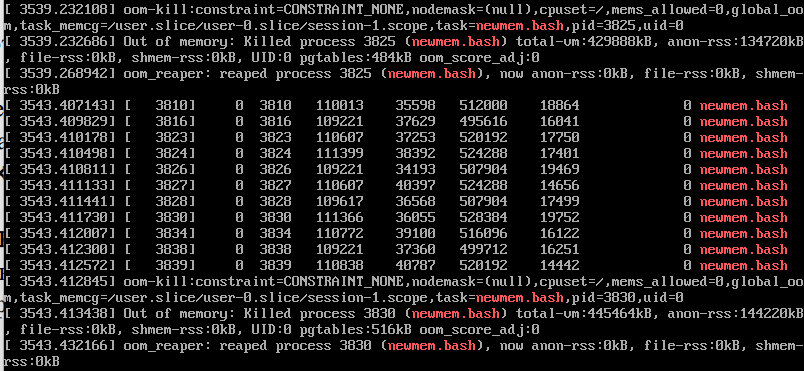
Вывод: любой процесс сначала расходует ресурсы физической (оперативной) памяти, после её заполнения процесс использует процесс подкачки. Когда оба ресурса заканчиваются программа завершается аварийно. Размер максимального массива обусловлен количеством свободной памяти и содержанием массива. Изначально, когда был запущен один скрипт, максимальный размер массива был 30000001, когда же запускались два скрипта, размер одного массива (то есть количество выделенной памяти) для первого из запущенных скриптов стал в два раза меньше – 15000001, и далее, когда он аварийно завершился, второй процесс спокойно закончил работу. Размер массива при этом составил 30000001. На каждой итерации я заполняла массив числами от 1 до 10000. При запуске двух скриптов одновременно можно заметить, что поначалу они работают вместе. Затем 1-й процесс завершается, что приводит к освобождению небольшого количества памяти, вследствие чего второй скрипт продолжает ещё работать какое-то время и также завершается.

**Эксперимент №2**

Нет аварийной остановки при запуске с параметрами n=3000001 и k=10.



При n=3000001 и k=30 ряд процессов завершился аварийно:



При запуске с k=1000000 все процессы завершаются успешно. При k=1100000 часть процессов завершается аварийно.

Вывод: при одновременном запуске 10 скриптов, в которых происходит заполнение массива размером в 10 раз меньше критического процесс не завершается аварийно, однако при запуске 30 таких скриптов не все процессы завершаются успешно из-за переполнения памяти. В ходе двух экспериментов была продемонстрирована механика работы памяти в ОС Linux, найдены критические значения для работы ряда однотипных скриптов для конкретной конфигурации системы.