**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное автономное образовательное учреждение высшего образования**

**«Национальный исследовательский университет ИТМО»**

Факультет информационных технологий и программирования

Операционные системы

Лабораторная работа № 5

**Управление памятью в ОС Linux**

**Выполнил студент группы № M32101**

Косовец Роман

Санкт-Петербург

2022

**Эксперимент №1**

***Данные о конфигурации ОС:***

1. Общий объем оперативной памяти: 2048 Мб

2. Объем раздела подкачки: 820 Мб

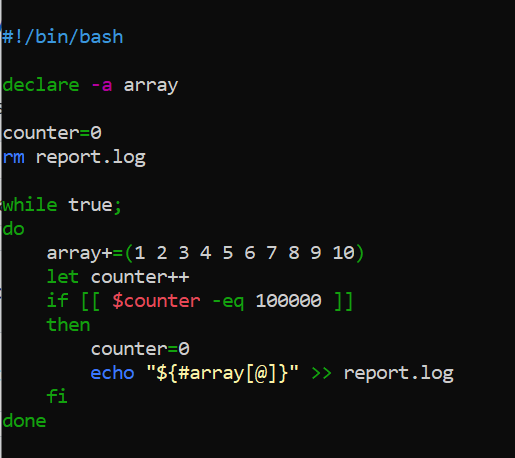
3. Размер страницы виртуальной памяти: 4 Кб

4. Объем свободной физической памяти в ненагруженной системе: 1.6 Гб

5. Объем свободного пространства в разделе подкачки в ненагруженной системе: 820 Мб

***Подготовительный этап:***

Написан скрипт *mem.bash*, который на каждом шаге цикла в конец массива добавлял последовательность из 10 элементов. Далее, каждый 10000-ый шаг в файл report.log добавляет строку с текущим значением размера массива



***Первый этап:***

*Задача* – оценить изменения параметров, выводимых утилитой top в процессе работы созданного скрипта.

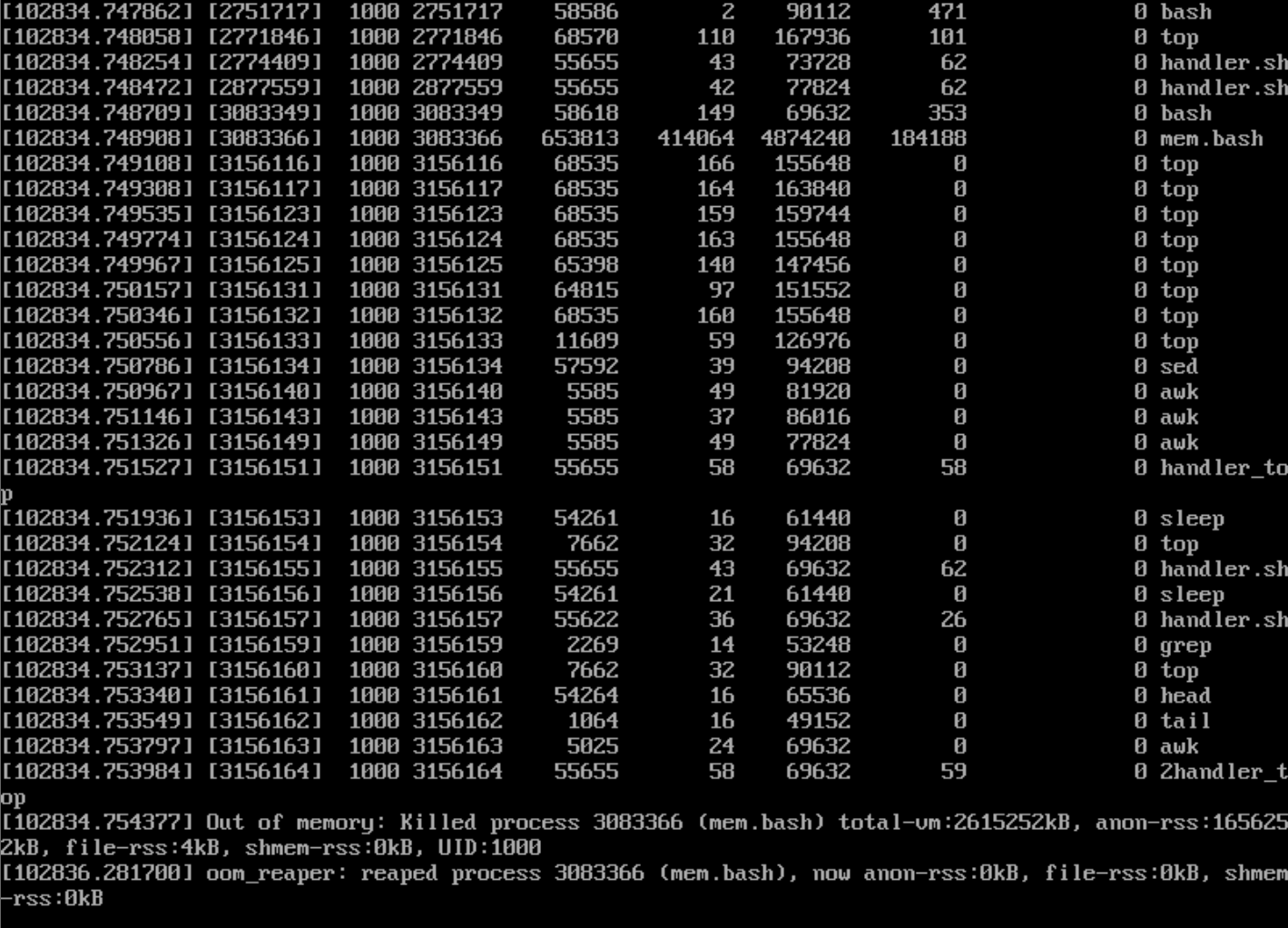
Были подготовлены две консоли. В первой запустили утилиту ***top***, а во второй запустили созданные скрипты ***mem.bash*** и ***handler.sh*** для отслеживания изменений во времени, и переключились на первую консоль. Дождавшись аварийной остановки процесса, зафиксировали:

* последние сообщения системного журнала
* последнюю запись журнала – значения параметров, с которыми произошла аварийная остановка процесса
* значение в последней строке файла ***report.log***

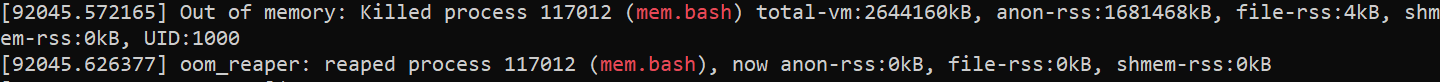
Также с помощью парсера сохранили следующие значения для последующей их обработки:

* значения параметров памяти системы (верхние две строки над основной таблицей);
* значения параметров в строке таблицы, соответствующей работающему скрипту;
* изменения в верхних пяти процессах (как меняется состав и позиции этих процессов).

1. Последние сообщения системного журнала:



1. Последняя запись журнала – значения параметров, с которыми произошла аварийная остановка процесса:



1. Значение в последней строке файла ***report.log:*** 30000000

1. Значения, полученные при использовании парсера:

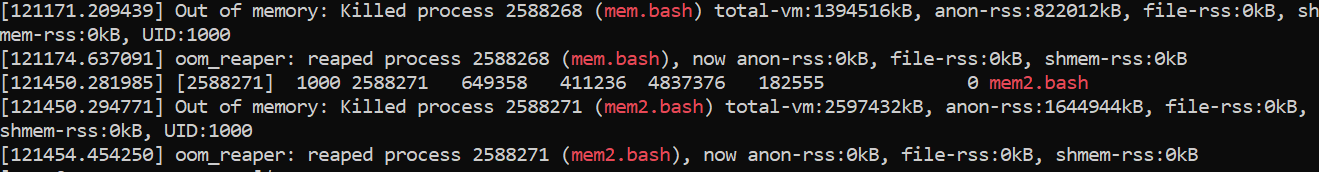
|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | MiB Mem | | | | MiB Swap | | | |
| Time | total | free | used | buff/cache | total | free | used | avail Mem |
| 0:00.54 | 1827,1 | 1572,90 | 184,20 | 69,90 | 820,00 | 715,50 | 104,50 | 1534,30 |
| 0:01.11 | 1513,60 | 243,50 | 69,90 | 715,50 | 104,50 | 1475,00 |
| 0:01.68 | 1452,70 | 304,40 | 70,00 | 715,50 | 104,50 | 1414,10 |
| 0:02.26 | 1393,90 | 363,20 | 70,00 | 715,50 | 104,50 | 1355,30 |
| 0:02.84 | 1330,70 | 426,30 | 70,00 | 715,50 | 104,50 | 1292,10 |
| 0:03.42 | 1268,30 | 488,70 | 70,10 | 715,50 | 104,50 | 1229.7 |
| 0:04.01 | 1210,70 | 546,20 | 70,20 | 715,50 | 104,50 | 1172,10 |
| 0:04.58 | 1150,10 | 606,80 | 70,20 | 715,50 | 104,50 | 1111,60 |
| 0:05.15 | 1091,10 | 665,70 | 70,20 | 715,50 | 104,50 | 1052,60 |
| 0:05.73 | 1025,70 | 731,10 | 70,30 | 715,50 | 104,50 | 987,20 |
| 0:06.31 | 965,70 | 791,00 | 70,30 | 715,50 | 104,50 | 927,30 |
| 0:06.89 | 901,60 | 855,10 | 70,20 | 715,50 | 104,50 | 863,20 |
| 0:07.47 | 847,10 | 909,60 | 70,40 | 715,50 | 104,50 | 808,70 |
| 0:08.05 | 782,10 | 974,60 | 70,40 | 715,50 | 104,50 | 743,60 |
| 0:08.63 | 725,10 | 1031,60 | 70,40 | 715,50 | 104,50 | 686,70 |
| 0:09.20 | 665,30 | 1091,30 | 70,50 | 715,50 | 104,50 | 625,90 |
| 0:09.78 | 606,50 | 1150,00 | 70,50 | 715,50 | 104,50 | 568,10 |
| 0:10.37 | 539,70 | 1216,80 | 70,50 | 715,50 | 104,50 | 501,30 |
| 0:10.94 | 480,30 | 1276,20 | 70,50 | 715,50 | 104,50 | 441,90 |
| 0:11.52 | 422,90 | 1333,60 | 70,60 | 715,50 | 104,50 | 384,50 |
| 0:12.08 | 357,00 | 1399,40 | 70,60 | 715,50 | 104,50 | 318,70 |
| 0:12.68 | 302,00 | 1454,80 | 70,70 | 715,50 | 104,50 | 263,70 |
| 0:13.27 | 232,20 | 1514,10 | 70,70 | 715,50 | 104,50 | 193,90 |
| 0:13.83 | 162,90 | 1583,20 | 80,90 | 715,50 | 104,50 | 129,60 |
| 0:14.41 | 105,90 | 1640,10 | 81,10 | 715,50 | 104,50 | 72,70 |
| 0:14.98 | 72,50 | 1695,50 | 59,10 | 715,50 | 104,50 | 28,30 |
| 0:15.48 | 71,90 | 1711,60 | 43,50 | 673,30 | 146,60 | 20,50 |
| 0:16.03 | 67,60 | 1715,00 | 44,40 | 615,20 | 204,80 | 16,90 |
| 0:16.59 | 67,80 | 1713,30 | 46,00 | 561,30 | 258,70 | 17,80 |
| 0:17.11 | 69,10 | 1709,20 | 48,80 | 502,60 | 317,40 | 20,50 |
| 0:17.66 | 85,80 | 1692,90 | 48,40 | 431,50 | 388,50 | 37,00 |
| 0:18.16 | 93,70 | 1686,00 | 47,30 | 368,90 | 451,10 | 44,40 |
| 0:18.72 | 72,00 | 1708,90 | 46,00 | 339,30 | 480,70 | 22,10 |
| 0:19.27 | 73,00 | 1708,50 | 45,60 | 280,10 | 539,00 | 22,80 |
| 0:19.82 | 66,20 | 1713,90 | 47,00 | 233,40 | 586,60 | 16,70 |
| 0:20.37 | 66,20 | 1715,40 | 45,50 | 175.1 | 644,90 | 16,00 |
| 0:20.91 | 76,90 | 1703,50 | 46,60 | 106,30 | 713,70 | 27,20 |
| 0:21.49 | 65,50 | 1715,30 | 46,30 | 56,40 | 763,60 | 15,70 |
| 0:22.03 | 71,20 | 1710,10 | 45,80 | 0,00 | 820,00 | 21,10 |

***Второй этап:***

*Задача* – оценить изменения параметров, выводимых утилитой top в процессе работы нескольких экземпляров созданного скрипта.

Для выполнения второго этапа были созданы копия скрипта ***mem2.bash*** и скрипт ***runner2.sh***, который запускает немедленно друг за другом оба скрипта в фоновом режиме. Также были подготовлены две консоли. В первой запустили утилиту ***top***, а во второй запустили созданные скрипты ***runner2.sh*** и ***handler.sh*** для отслеживания изменений во времени, и переключились на первую консоль. Дождавшись аварийной остановки процесса, зафиксировали те же данные, что и в первом этапе.

1. Последняя запись журнала – значения параметров, с которыми произошла аварийная остановка процесса:



1. Значение в последней строке файла ***report.log:*** 30000000

Значение в последней строке файла ***report2.log:*** 15000000

1. Значения, полученные при использовании парсера:

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | MiB Mem | | | | MiB Swap | | | |
| Time | total | free | used | buff/cache | total | free | used | avail Mem |
| 0:00.84 | 1827,1 | 1435,3 | 321,1 | 70,6 | 820,0 | 715,5 | 104,5 | 1396,6 |
| 0:01.53 | 1291,6 | 464,7 | 70,7 | 715,5 | 104,5 | 1253,0 |
| 0:01.85 | 1217,2 | 539,1 | 70,8 | 715,5 | 104,5 | 1178,6 |
| 0:02.51 | 1078,5 | 677,7 | 70,9 | 715,5 | 104,5 | 1039,9 |
| 0:03.18 | 931,1 | 823,9 | 70,9 | 715,5 | 104,5 | 893,7 |
| 0:03.83 | 798,9 | 957,2 | 70,9 | 715,5 | 104,5 | 760,4 |
| 0:04.51 | 646,6 | 1109,4 | 71,1 | 715,5 | 104,5 | 608,2 |
| 0:04.85 | 570,3 | 1185,7 | 71,1 | 715,5 | 104,5 | 531,9 |
| 0:05.88 | 356,5 | 1399,3 | 71,1 | 715,5 | 104,5 | 318,1 |
| 0:06.55 | 209,6 | 1546,2 | 71,3 | 715,5 | 104,5 | 171,3 |
| 0:07.24 | 65,1 | 1698,1 | 63,8 | 715,5 | 104,5 | 23,1 |
| 0:07.86 | 70,5 | 1709,4 | 47,1 | 589,8 | 230,4 | 21,1 |
| 0:08.16 | 68,9 | 1710,9 | 47,3 | 521,1 | 298,9 | 19,6 |
| 0:08.79 | 74,5 | 1705,9 | 46,7 | 392,7 | 427,3 | 24,8 |
| 0:09.42 | 69,6 | 1712,1 | 45,4 | 258,2 | 561,8 | 19,3 |
| 0:09.72 | 75,6 | 1705,5 | 46,9 | 187,7 | 632,8 | 26,1 |
| 0:10.03 | 64,8 | 1714,8 | 47,6 | 135,9 | 684,1 | 15,6 |
| 0:10.34 | 83,9 | 1698,1 | 45,1 | 55,5 | 764,4 | 33,5 |
| 0:10.65 | 68,5 | 1713,4 | 45,1 | 0,0 | 820,0 | 18,1 |
| 0:10.86 | 51,1 | 1745,0 | 30,9 | 0,0 | 820,0 | 11,5 |
| 0:11.42 | 787,4 | 977,5 | 62,2 | 358,9 | 461,1 | 744,6 |
| 0:12.00 | 707,3 | 1057,3 | 62,5 | 358,9 | 461,1 | 664,7 |
| 0:12.38 | 663,9 | 1100,7 | 62,5 | 358,9 | 461,1 | 621,2 |
| 0:13.16 | 583,0 | 1181,9 | 62,6 | 358,9 | 461,1 | 540,4 |
| 0:13.92 | 505,3 | 1257,7 | 64,1 | 358,9 | 461,1 | 463,5 |
| 0:14.33 | 457,8 | 1305,3 | 64,1 | 358,9 | 461,1 | 415,8 |
| 0:14.73 | 415,1 | 1347,7 | 64,2 | 358,9 | 461,1 | 373,3 |
| 0:15.51 | 333,8 | 1429,1 | 64,2 | 358,9 | 461,1 | 292,0 |
| 0:16.71 | 206,8 | 1555,9 | 64,4 | 358,9 | 461,1 | 165,1 |
| 0:17.49 | 120,5 | 1642,0 | 64,5 | 358,9 | 461,1 | 78,9 |
| 0:18.25 | 74,0 | 1697,4 | 55,5 | 358,9 | 461,1 | 27,9 |
| 0:18.63 | 69,5 | 1707,4 | 50,1 | 303,1 | 516,9 | 20,7 |
| 0:19.38 | 79,4 | 1698,0 | 49,0 | 220,9 | 599,1 | 30,0 |
| 0:20.14 | 72,4 | 1706,9 | 47,8 | 145,3 | 674,7 | 22,6 |
| 0:20.88 | 83,2 | 1695,6 | 48,2 | 54,6 | 765,4 | 33,9 |
| 0:21.83 | 54,6 | 1721,3 | 51,2 | 0,0 | 820,0 | 6,8 |

|  |  |
| --- | --- |
| Продолжение таблицы | |
| %Mem1 | %Mem2 |
| 43,0 | 43,2 |
| 43,8 | 0,0 |
| 45,9 | 0,0 |
| 50,4 | 0,0 |
| 52,5 | 0,0 |
| 57,1 | 0,0 |
| 61,5 | 0,0 |
| 64,0 | 0,0 |
| 66,2 | 0,0 |
| 70,8 | 0,0 |
| 77,7 | 0,0 |
| 82,3 | 0,0 |
| 85,3 | 0,0 |
| 85,7 | 0,0 |
| 85,6 | 0,0 |
| 85,9 | 0,0 |
| 85,3 | 0,0 |
| 86,5 | 0,0 |

|  |  |
| --- | --- |
| %Mem1 | %Mem2 |
| 5,2 | 5,1 |
| 9,1 | 9,0 |
| 10,9 | 11,8 |
| 14,9 | 14,9 |
| 18,8 | 18,8 |
| 22,6 | 22,5 |
| 26,7 | 26,6 |
| 28,7 | 28,6 |
| 34,8 | 34,6 |
| 38,8 | 38,8 |
| 42,9 | 42,6 |
| 43,0 | 43,4 |
| 43,1 | 42,8 |
| 42,9 | 43,2 |
| 43,1 | 42,9 |
| 42,7 | 43,0 |
| 43,4 | 43,0 |
| 42,9 | 42,6 |

***Выводы:***

После проделанного эксперимента, можно сделать вывод, количество свободной оперативной памяти уменьшалось линейно, как в случае одного процесса ***mem.bash***, так и с двумя.

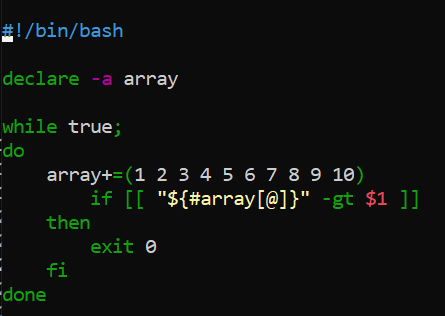
При низкой величине оперативной памяти происходит переход на файл подкачки. Как только значение становится критичным, то произойдет аварийное завершение подкачки.

При этом, если запустить 2 скрипта одновременно, то можно заметить, что, один из скриптов продолжил работу после убийства первого, это обусловлено тем, что во время аварийного завершения первого скрипта у нас восстанавливается физическая память.

**Эксперимент №2**

***Подготовительный этап:***

Создадим копию скрипта ***mem.bash*** в файл ***newmem.bash***, изменив копию таким образом, чтобы она завершала работу, как только размер создаваемого массива превысит значение N, передаваемое в качестве параметра скрипту.

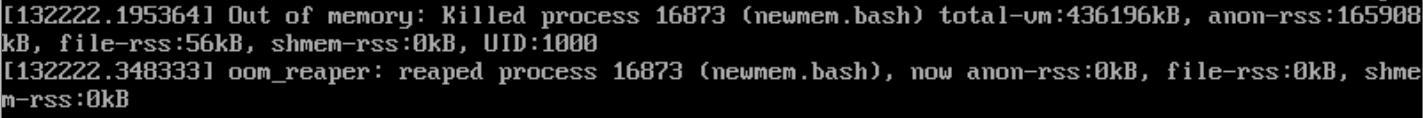
****

***Основной этап:***

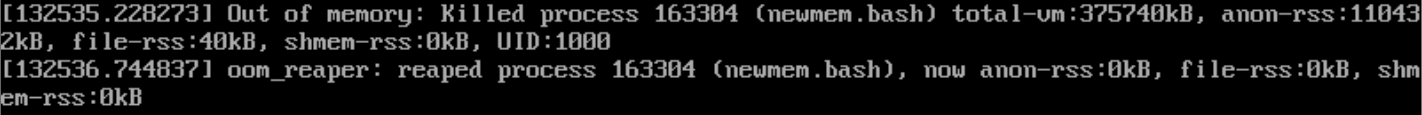
*Задача* – определить граничные значения потребления памяти, обеспечивающие безаварийную работу для регулярных процессов, запускающихся с заданной интенсивностью.

Создали скрипт, который запускает ***newmem.bash*** каждую секунду, используя один и тот же параметр N так, что всего будет осуществлено K запусков.

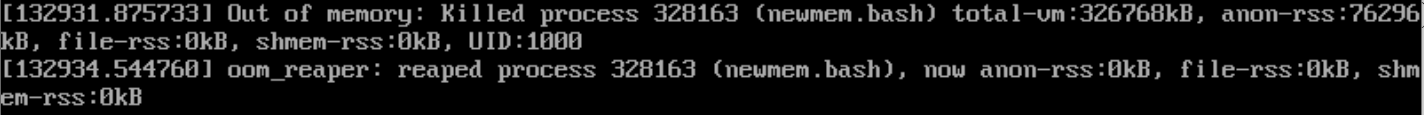
1. При K=10 и N=3000000 скрипт успешно завершает свою работу.
2. При K=30 и N=3000000 скрипт аварийно завершает свою работу.



1. При K=30 и N=2000000 скрипт аварийно завершает свою работу.



1. При K=30 и N=1300000 скрипт аварийно завершает свою работу.



1. При K=30 и N=1290000 скрипт успешно завершает свою работу.

***Вывод:***

Проделав данную лабораторную работу, я убедился, что во время работы с оперативной памятью данные заполняют физическую память, пока не дойдет до критического значения. Далее ОС начинает заполнять файлы подкачки. В случае, когда файлов подкачки будет не хватать – произойдет аварийная остановка процесса. Также при запуске нескольких процессов, одинаково требовательных к ресурсам, эти ресурсы делятся пополам.