|  |  |
| --- | --- |
| Параметры | Значения |
| Общий объем оперативной памяти | 3576,3 Mb |
| Объем раздела подкачки | 3072,0 Mb |
| Размер страницы виртуальной памяти | 4 kB |
| Объем свободной физической памяти в  ненагруженной системе | 1299,4 Mb |
| Объем свободного пространства в разделе  подкачки в ненагруженной системе | 3072,0 Mb |

Эксперимент No1

Подготовительный этап:

mem.bash

Изображение выглядит как текст

Автоматически созданное описание

report.log

Изображение выглядит как стол

Автоматически созданное описание

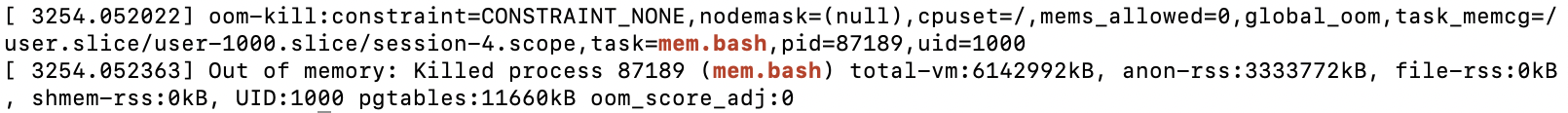
Первый этап:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| MiB Mem | | | | MiB Swap | | | |
| time | free | used | buff/cache | time | free | used | avail Mem |
| 0:01.36 | 2943,8 | 585,5 | 203,4 | 0:01.36 | 2560,7 | 511,3 | 2990,5 |
| 0:04.38 | 2638,6 | 890,9 | 203,4 | 0:04.38 | 2560,7 | 511,3 | 2685,4 |
| 0:07.40 | 2341,8 | 1188,0 | 203,4 | 0:07.40 | 2560,7 | 511,3 | 1515,0 |
| 0.10:41 | 2015,2 | 1514,3 | 203,4 | 0.10:41 | 2560,7 | 511,3 | 2062,0 |
| 0:13.43 | 1702,7 | 1826,9 | 203,4 | 0:13.43 | 2560,7 | 511,3 | 1749,4 |
| 0:16.45 | 1387,9 | 2141,6 | 203,4 | 0:16.45 | 2560,7 | 511,3 | 1434,7 |
| 0:19.48 | 1127,3 | 2402,2 | 203,4 | 0:19.48 | 2560,7 | 511,3 | 1174,1 |
| 0:22.50 | 914,7 | 2614,8 | 203,4 | 0:22.50 | 2560,7 | 511,3 | 961,4 |
| 0:25.51 | 709,5 | 2820,1 | 203,4 | 0:25.51 | 2560,7 | 511,3 | 756,2 |
| 0:28.53 | 504,7 | 3024,8 | 203,4 | 0:28.53 | 2560,7 | 511,3 | 551,4 |
| 0:31.55 | 362,7 | 3166,8 | 203,4 | 0:31.55 | 2560,7 | 511,3 | 409,4 |
| 0:34.57 | 188,0 | 3341,6 | 203,4 | 0:34.57 | 2560,7 | 511,3 | 234,7 |
| 0:37.60 | 64,3 | 3479,9 | 188,7 | 0:37.60 | 2558,2 | 513,8 | 96,3 |
| 0:40.57 | 65,6 | 3549,9 | 75,3 | 0:40.57 | 2548,5 | 523,5 | 26,4 |
| 0:43.64 | 61,6 | 3563,6 | 55,8 | 0:43.64 | 2290,3 | 781,7 | 12,6 |
| 0:46.61 | 67,9 | 3552,5 | 65,5 | 0:46.61 | 1950,2 | 1121,8 | 23,8 |
| 0:49.63 | 61,7 | 3558,1 | 66,7 | 0:49.63 | 1633,9 | 1438,1 | 18,2 |
| 0:52.63 | 62,7 | 3558,3 | 64,4 | 0:52.63 | 1315,4 | 1756,6 | 18,0 |
| 0:55.63 | 100,7 | 3521,3 | 62,2 | 0:55.63 | 972,4 | 2099,6 | 55,0 |
| 0:58.66 | 69,7 | 3553,1 | 60,7 | 0:58.66 | 679,6 | 2392,4 | 23,2 |
| 1:01.67 | 69,7 | 3553,6 | 59,5 | 1:01.67 | 369,6 | 2702,4 | 22,6 |
| 1:04.66 | 72,4 | 3551,4 | 58,6 | 1:04.66 | 60,4 | 3011,6 | 24,9 |

|  |  |  |  |
| --- | --- | --- | --- |
| time | VIRT | RES | %Mem |
| 0:01.36 | 358488,0 | 139280,0 | 3,8 |
| 0:04.38 | 672516,0 | 453176,0 | 12,4 |
| 0:07.40 | 982188,0 | 762848,0 | 20,8 |
| 0.10:41 | 1300176,0 | 1,0g | 29,5 |
| 0:13.43 | 1619484,0 | 1,3g | 38,2 |
| 0:16.45 | 1940904,0 | 1,6g | 47,0 |
| 0:19.48 | 2207676,0 | 1,9g | 54,3 |
| 0:22.50 | 2424420,0 | 2,1g | 60,2 |
| 0:25.51 | 2534036,0 | 2,3g | 65,9 |
| 0:28.53 | 2843784,0 | 2,5g | 71,7 |
| 0:31.55 | 2988324,0 | 2,6g | 75,6 |
| 0:34.57 | 3167316,0 | 2,8g | 80,5 |
| 0:37.60 | 3329544,0 | 3,0g | 84,9 |
| 0:40.57 | 3484644,0 | 3,1g | 89,1 |
| 0:43.64 | 3779532,0 | 3,1g | 89,9 |
| 0:46.61 | 4102008,0 | 3,1g | 89,5 |
| 0:49.63 | 4432008,0 | 3,1g | 89,7 |
| 0:52.63 | 4756332,0 | 3,1g | 89,7 |
| 0:55.63 | 5071416,0 | 3,1g | 89,0 |
| 0:58.66 | 5401548,0 | 3,1g | 89,9 |
| 1:01.67 | 5401548,0 | 3,1g | 89,9 |
| 1:04.66 | 5727852,0 | 3,1g | 90,1 |

Посмотрите с помощью команды dmesg | grep "mem.bash" последние две записи о скрипте в системном журнале и зафиксируйте их в отчете. Также зафиксируйте значение в последней строке файла report.log.

dmesg | grep "mem.bash"



report.log



Второй этап:

Задача – оценить изменения параметров, выводимых утилитой top в процессе работы нескольких экземпляров созданного скрипта.  
Ход эксперимента:  
Создайте копию скрипта, созданного на предыдущем этапе, в файл mem2.bash. Настройте её на запись в файл report2.log. Создайте скрипт, который запустит немедленно друг за другом оба скрипта в фоновом режиме. Подготовьте две консоли. В первой запустите утилиту top. Во второй запустите созданный перед этим скрипт и переключитесь на первую консоль. Убедитесь, что в top появились mem.bash и mem2.bash. Наблюдайте за следующими значениями (и фиксируйте их изменения во времени в отчете):

launch.sh

Изображение выглядит как текст

Автоматически созданное описание

Проводите наблюдения и фиксируйте их в отчете до аварийной остановки последнего из двух скриптов и их исчезновения из перечня процессов в top.

Посмотрите с помощью команды dmesg | grep "mem[2]\*.bash" последние записи о скриптах в системном журнале и зафиксируйте их в отчете. Также зафиксируйте значения в последних строках файлов report.log и report2.log.

Изображение выглядит как текст

Автоматически созданное описание

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| MiB Mem | | | | MiB Swap | | | |
| time | free | used | buff/cache | time | free | used | avail Mem |
| 0:00.31 | 1276,6 | 1094,2 | 1393,5 | 0:00.31 | 3072,0 | 0 | 2482,1 |
| 0:03.08 | 1013,5 | 1357,3 | 1393,5 | 0:03.08 | 3072,0 | 0 | 2219,5 |
| 0:05.82 | 595,5 | 1778,3 | 1393,5 | 0:05.82 | 3072,0 | 0 | 1798,0 |
| 0.08:51 | 121,7 | 2249,1 | 1393,5 | 0.08:51 | 3072,0 | 0,0 | 1327,1 |
| 0:10.43 | 58,2 | 2346,4 | 1361,4 | 0:10.43 | 3071,2 | 0,8 | 1229,9 |
| 0:12.58 | 64,8 | 2410,2 | 1290,9 | 0:12.58 | 3070,7 | 1,3 | 1166,1 |
| 0:15.04 | 61,9 | 2517,0 | 1183,8 | 0:15.04 | 3069,5 | 2,5 | 1059,3 |
| 0:16.92 | 65,9 | 2560,4 | 1140,2 | 0:16.92 | 3069,0 | 3,0 | 1015,9 |
| 0:19.23 | 61,1 | 2690,2 | 1012,7 | 0:19.23 | 3063,7 | 8,3 | 886,1 |
| 0:21.59 | 68,1 | 2769,1 | 924,1 | 0:21.59 | 3052,3 | 19,7 | 807,1 |
| 0:24.03 | 87,5 | 2851,6 | 821,2 | 0:24.03 | 3050,1 | 21,9 | 724,6 |
| 0:26.76 | 82,5 | 3039,0 | 638,3 | 0:26.76 | 3032,4 | 39,6 | 537,3 |
| 0:29.18 | 86,3 | 3186,2 | 486,0 | 0:29.18 | 3017,7 | 54,4 | 390,1 |
| 0:31.60 | 102,5 | 3339,6 | 337,2 | 0:31.60 | 2996,4 | 75,6 | 236,7 |
| 0:33.95 | 74,2 | 3430,9 | 246,6 | 0:33.95 | 2974,1 | 97,8 | 145,3 |
| 0:36.25 | 105,6 | 3478,1 | 150,9 | 0:36.25 | 2797,0 | 275,0 | 98,2 |
| 0:40.57 | 53,1 | 3559,1 | 91,8 | 0:40.57 | 2352,0 | 720,0 | 17,2 |
| 0:41.65 | 75,9 | 3538,4 | 85,8 | 0:41.65 | 2116,0 | 955,9 | 37,9 |
| 0:43.72 | 74,2 | 3541,2 | 82,8 | 0:43.72 | 1791,8 | 1280,2 | 35,1 |
| 0:45.79 | 70,9 | 3545,8 | 78,6 | 0:45.79 | 1412,6 | 1659,4 | 30,5 |
| 0:47.89 | 63,4 | 3554,4 | 75,2 | 0:47.89 | 1125,9 | 1946,1 | 21,8 |
| 0:49.94 | 75,4 | 3543,0 | 73,2 | 0:49.94 | 821,4 | 2250,6 | 33,3 |
| 0:54.16 | 59,3 | 3559,6 | 71,3 | 0:54.16 | 154,8 | 2917,2 | 16,7 |
| 0:58.30 | 49,9 | 3576,3 | 54,0 | 0:58.30 | 0,0 | 3072,0 | 13,2 |
| 1:00.96 | 1484,2 | 2136,5 | 67,9 | 1:00.96 | 1236,4 | 1835,6 | 1439,8 |
| 1:03.77 | 1102,2 | 2517,6 | 69,2 | 1:03.77 | 1236,4 | 1835,6 | 1058,5 |
| 1:06.57 | 825,2 | 2784,1 | 90,6 | 1:06.57 | 1242,9 | 1829,1 | 792,1 |
| 1:09.37 | 627,5 | 2981,7 | 90,7 | 1:09.37 | 1242,9 | 1829,1 | 594,6 |
| 1:12.16 | 400,2 | 3208,9 | 91,0 | 1:12.16 | 1242,9 | 1829,1 | 367,3 |
| 1:14.94 | 180,9 | 3428,1 | 91,1 | 1:14.94 | 1245,9 | 1826,1 | 148,2 |
| 1:17.69 | 62,2 | 3548,9 | 87,0 | 1:17.69 | 1171,6 | 1900,4 | 27,4 |
| 1:20.33 | 62,7 | 3550,7 | 82,2 | 1:20.33 | 961,9 | 2110,1 | 25,6 |
| 1:23.01 | 62,8 | 3557,4 | 68,6 | 1:23.01 | 743,9 | 2328,1 | 18,9 |
| 1:25.76 | 62,5 | 3560,3 | 63,0 | 1:25.76 | 526,4 | 2545,1 | 16,0 |
| 1:28.47 | 81,7 | 3542,1 | 61,2 | 1:28.47 | 259,1 | 2812,9 | 34,2 |
| 1:31.14 | 60,2 | 3564,2 | 59,6 | 1:31.14 | 80,9 | 2991,1 | 12,1 |
| 1:32.73 | 48,1 | 3576,3 | 53,0 | 1:32.73 | 0,0 | 3072,0 | 11,3 |

|  |  |
| --- | --- |
| Mem1 | Mem2 |
| 0,3 | 0,2 |
| 4,0 | 3,9 |
| 9,9 | 9,8 |
| 16,5 | 16,4 |
| 18,2 | 18,3 |
| 19,0 | 19,2 |
| 20,6 | 20,7 |
| 21,1 | 21,3 |
| 22,8 | 23,1 |
| 24,5 | 24,7 |
| 26,0 | 26 |
| 29,0 | 28,7 |
| 31,4 | 30,9 |
| 33,5 | 33,1 |
| 35,4 | 35 |
| 36,9 | 36,3 |
| 39,9 | 38,5 |
| 40,4 | 39,5 |
| 41,4 | 40,6 |
| 42,2 | 41,3 |
| 43,0 | 41,6 |
| 43,4 | 41,9 |
| 44,6 | 43,2 |
| 0,0 | 43,8 |
| 0,0 | 48,5 |
| 0,0 | 59,3 |
| 0,0 | 66 |
| 0,0 | 71,3 |
| 0,0 | 77,7 |
| 0,0 | 83,8 |
| 0,0 | 87,5 |
| 0,0 | 88 |
| 0,0 | 88,6 |
| 0,0 | 89,2 |
| 0,0 | 89 |
| 0,0 | 90,4 |
| 0,0 | 0 |

**Выводы:**

Оперативня память уменьшалась линейно, что можно заметить на графиках

При нехватке ресурсов, будет использован файл подкачки, и в случае нехватки файла подкачки, процесс будет остановлен аварийно

По 2 эксперименту, можно заметить, что во время аварийной остановки процесса, его ресурсы были переданы второму процессу.

Обработка результатов:

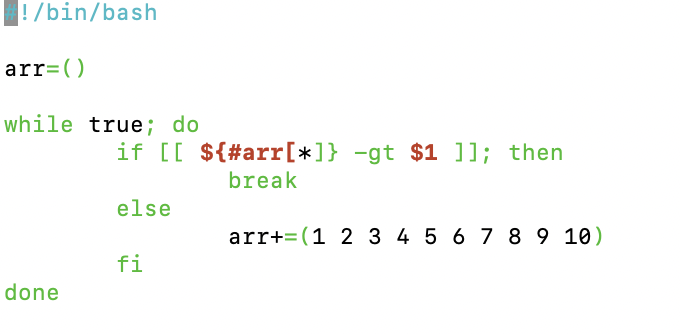
Постройте графики изменения каждой из величин, за которыми производилось наблюдение на каждом из этапов. Объясните динамику изменения этих величин исходя из теоретических основ управления памятью в рамках страничной организации памяти с разделом подкачки. Объясните значения пороговых величин: размер массива, при котором произошла аварийная остановка процесса, параметры, зафиксированные в момент аварийной остановки системным журналом. Сформулируйте письменные выводы.

Эксперимент No2

Подготовительный этап:

Создайте копию скрипта mem.bash в файл newmem.bash. Измените копию таким образом, чтобы она завершала работу, как только размер создаваемого массива превысит значение N, передаваемое в качестве параметра скрипту. Уберите запись данных в файл.

newmem.bash



newmemstart.bash

Изображение выглядит как текст

Автоматически созданное описание

Основной этап:

Задача – определить граничные значения потребления памяти, обеспечивающие безаварийную работу для регулярных процессов, запускающихся с заданной интенсивностью.  
Ход эксперимента:  
Создайте скрипт, который будет запускать newmem.bash каждую секунду, используя один и тот же параметр N так, что всего будет осуществлено K запусков.

Возьмите в качестве значения N, величину, в 10 раз меньшую, чем размер массива, при котором происходила аварийная остановка процесса в первом этапе предыдущего эксперимента. Возьмите в качестве K значение 10. Убедитесь, что все K запусков успешно завершились, и в системном журнале нет записей об аварийной остановке newmem.bash.

Измените значение K на 30 и снова запустите скрипт. Объясните, почему ряд процессов завершился аварийно. Подберите такое максимальное значение N, чтобы при K=30 не происходило аварийных завершений процессов. Укажите в отчете сформулированные выводы по этому эксперименту и найденное значение N.

1. При K=10 и N=8000000 скрипт успешно завершил свою работу
2. При K=30 и N=8000000 скрипт успешно завершил свою работу
3. При K=30 и N=6000000 скрипт аварийно завершил свою работу

Изображение выглядит как текст

Автоматически созданное описание

1. При K=30 и N=4200000 скрипт успешно завершил работу (верхняя граница)

**Вывод:**

Сделав лабораторную работу, было выяснено, что до тех пор пока ресурсов системы хватает, процесс будет использовать оперативную память. Когда же оперативная память закончится, будет использоваться файл подкачки, но когда все ресурсы, включая файл подкачки, будут использованы, процесс будет аварийно завершен, тем самым освобождая ресурсы. Также было замечено, что при запуске одинаково требовательных процессов, ресурсы будут делиться поровну.