**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»**

**Отчет**

по лабораторной работе «Управление памятью в ОС Linux»

по дисциплине «**Операционные системы**»

Выполнил: Величко Максим Иванович

Факультет: Информационных технологий и программирования

Группа: М32061

Изображение выглядит как текст, коллекция картинок

Автоматически созданное описание

Санкт-Петербург

2022

Текущая конфигурация ОС в аспекте управления памятью:

• Общий объем оперативной памяти – 1.8 Гб

• Объем раздела подкачки – 819 Мб

• Размер страницы виртуальной памяти – 4096 б = 4 Кб

• Объем свободной физической памяти в ненагруженной системе – 1.4 Гб

• Объем свободного пространства в разделе подкачки в ненагруженной системе – 819 Мб

Изображение выглядит как текст

Автоматически созданное описание

**Эксперимент №1**

Подготовительный этап:

скрипт mem.bash, запускающий бесконечный цикл и каждый 100000-ый раз записывающий размер полученного массива в файл report.log

Первый этап:

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

Ход эксперимента:

Запустить скрипт и дождаться аварийной остановки процесса.

Последняя запись в report.log: 30 000 000

Журнал:

Изображение выглядит как текст

Автоматически созданное описание

На одной консоли запустить top, а на другой mem.bash.

Чтобы руками не записывать результаты top, каждую 1-2 секунды запускался скрипт top\_info, который выводил параметры, за которыми надо было следить:

• 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)

Изображение выглядит как текст

Автоматически созданное описание

Второй этап:

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

Ход эксперимента:

Создали аналогичные скрипты mem2.bash, top\_info2 и info2, а также скрипт exp1, который запускает два скрипта mem/mem2.bash.

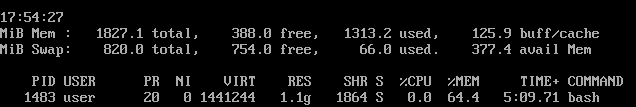
В скрипты info и info2 записывается вся информация об этих двух скриптах, за которыми надо следить.

Последняя запись в

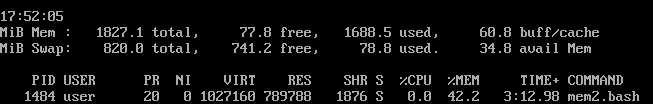
report.log: 15 100 000

report2.log: 30 000 000

info:



info2:



Журнал:

Изображение выглядит как текст

Автоматически созданное описание

Обработка результатов:

Были построены графики для обоих опытов

Опыт 1

Вывод: изначально оперативная память линейно убывает, но когда она достигает совсем низкого значения, начинает линейно убывать память подкачки, и когда подкачка становится равной 0, процесс убивается. В нижней правой части графика видно, что за счет кэша оперативная память немного возраст и наоборот.

Опыт 2

Вывод: изначально все повторяется, как в случае с одним процессом, но когда память подкачки заканчивается, первый процесс убивается, и резко возрастает значение и подкачки, и оперативной памяти, что позволяет еще некоторое время существовать второму процессу.

При низкой величине физической памяти (7 – 10 Mб) происходит переход на файлы подкачки. Как только их значение станет критично, то произойдет аварийное завершение программы.

Если запусти 2 скрипта одновременно, то из графика можно заметить, что один из них продолжил работу, это обусловлено тем, что во время аварийного завершения первого скрипта у нас восстанавливается физическая память, далее идет сценарий как из первого этапа.

**Эксперимент №2**

Подготовительный этап:

Была создана копия mem.bash – newmem.bash, которая принимает на вход значение N – максимальное количество элементов в массиве, после которого скрипт завершает работу.



Был написан обработчик, через который будет производиться запуск



Основной этап:

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

Ход эксперимента:

Был создан вспомогательный скрипт memcall, который запускает скрипт newmem.bash с одним и тем же N K раз.

K = 10

N = 3 000 000 (в 10 раз меньше, чем последнее число в эксперименте 1)



Результат:

В системном журнале записей нет.

K = 30

N = 3 000 000

Результат:

Только 6 процессов успешно завершились.

Изображение выглядит как текст

Автоматически созданное описание

Подберем N такое, что все 30 процессов завершатся успешно:

K = 30

N = 1 000 000

Завершились 30 процессов.

K = 30

N = 2 000 000

Завершились 7 процессов.



K = 30

N = 1 500 000

Завершились 14 процессов.

K = 30

N = 1 250 000

Завершились 30 процессов.

K = 30

N = 1 300 000

Завершились 30 процессов.

K = 30

N = 1 300 000

Завершились 19 процессов.

Вывод: максимальное N, при котором завершаются 30 процессов – 1300000.