Лабораторная работа №5. Управление памятью в ОС Linux

Рассматриваемые вопросы

1. Использование утилиты top для мониторинга параметров памяти
2. Использование имитационных экспериментов для анализа работы механизмов управления памятью.

Основные источники данных о состоянии памяти вычислительного узла
`команда free`
файл **`/proc/meminfo`** (документация в соответствующем разделе **`man proc`**)
файл **`/proc/[PID]/statm`** (документация в соответствующем разделе **`man proc`**)
утилита **`top`**

Организация управления памятью в ОС Linux
В Linux используется страничная организация виртуальной памяти. Память разбита на страницы. Размер страницы можно посмотреть в параметрах конфигурации с помощью команды **`getconf PAGE_SIZE`**. При обращении к адресу в памяти происходит динамическое преобразование адреса путем замены старших бит виртуального адреса на номер физической страницы с сохранением значения младших бит как смещения на странице.
Обычно, кроме физической памяти используется также раздел подкачки. В этом случае адресное пространство процесса состоит из страниц, находящихся в оперативной памяти и страниц, находящихся в разделе подкачки. Параметры раздела подкачки можно узнать из файла **`/proc/swaps`**. При динамическом выделении памяти процессу, операционная система сначала пытается выделить страницы в физической памяти, но если это невозможно, инициирует страничный обмен, в рамках которого ряд страниц из физической памяти вытесняется на раздел подкачки, а адреса, соответствующие вытесненным страницам выделяются процессу под новые страницы.
Операционная система контролирует выделение памяти процессам. Если процесс попробует запросить расширение адресного пространства, которое невозможно в пределах имеющейся свободной оперативной памяти, его работа будет аварийно остановлена с записью в системном журнале.

Задание на лабораторную работу
Проведите два виртуальных эксперимента в соответствии с требованиями и проанализируйте их результаты. В указаниях ниже описано, какие данные необходимо фиксировать в процессе проведения экспериментов. Рекомендуется написать «следящие» скрипты и собирать данные, например, из вывода утилиты **`top`** автоматически с заданной периодичностью, например, 1 раз в секунду. Можно проводить эксперименты и фиксировать требуемые параметры и в ручном режиме, но в этом случае рекомендуется замедлить эксперимент, например, уменьшив размер добавляемой к массиву последовательности с 10 до 5 элементов.

Требования к проведению экспериментов и содержанию отчета
Зафиксируйте в отчете данные о текущей конфигурации операционной системы в аспекте управления памятью:

- Общий объем оперативной памяти.
- Объем раздела подкачки.
- Размер страницы виртуальной памяти.
- Объем свободной физической памяти в ненагруженной системе.
- Объем свободного пространства в разделе подкачки в ненагруженной системе.

Эксперимент №1
Подготовительный этап:
Создайте скрипт **`mem.bash`**, реализующий следующий сценарий. Скрипт выполняет бесконечный цикл. Перед началом выполнения цикла создается пустой массив и счетчик шагов, инициализированный нулем. На каждом шаге цикла в конец массива добавляется последовательность из 10 элементов, например, (1 2 3 4 5 6 7 8 9 10). Каждый 100000-ый шаг в файл **`report.log`** добавляется строка с текущим значением размера массива (перед запуском скрипта, файл обнуляется).

Первый этап:
Задача – оценить изменения параметров, выводимых утилитой **`top`** в процессе работы созданного скрипта.
Ход эксперимента:
Запустите созданный скрипт **`mem.bash`**. Дождитесь аварийной остановки процесса и вывода в консоль последних сообщений системного журнала. Зафиксируйте в отчете последнюю запись журнала - значения параметров, с которыми произошла аварийная остановка процесса. Также зафиксируйте значение в последней строке файла **`report.log`**.
Подготовьте две консоли. В первой запустите утилиту **`top`**. Во второй запустите скрипт и переключитесь на первую консоль. Убедитесь, что в **`top`** появился запущенный скрипт. Наблюдайте за следующими значениями (и фиксируйте их изменения во времени в отчете):

- 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).

Проводите наблюдения и фиксируйте их в отчете до аварийной остановки процесса скрипта и его исчезновения из перечня процессов в **top**.
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**.

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):

- 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).

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**.

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 №2
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.

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**.