

КОНТРОЛЬНАЯ РАБОТА

по дисциплине «Операционные системы»

Цель работы:

Получение и закрепление практических навыков администрирования процессов и управления памятью в ОС семейства CNU/Linux с применением системных утилит.

В разделе «Порядок выполнения работы» представлены задания 1 – 8 контрольной работы. В разделе «Теоретические сведения» приводятся основная информация для выполнения заданий. «Приложение А» содержит указания по оформлению отчета. «Приложение Б» содержит критерии оценки контрольной работы.

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

Задание 1. Мониторинг производительности системы

1. Выполните проверку системы при помощи команды **top**. Отсортируйте процессы по: объему используемой памяти; времени работы; идентификатору; проценту использования времени процессора.
2. Запустите новый процесс в системе, найдите его в выводе команды **top**. Измените приоритет запущенного процесса командой **renice** и убедитесь, что изменения отражены в выводе **top**.
3. С помощью команд **atop** и **atopsar** выполните сортировку процессов минимум по трем параметрам общей нагрузки.
4. При помощи команды **mpstat** и опции **P** выведите информацию по: определенному процессору, всем процессорам.
5. Выполните проверку системы при помощи команды **pidstat**. Выведите по четыре отчета статистики использования процессора для каждой активной задачи в системе с интервалом в три секунды.

В отчете приведите ответ в виде снимков экрана по каждому пункту задания.

Задание 2. Создание скрипта для получения статистики

Напишите скрипт, осуществляющий сбор статистики использования процессоров и статистики работы процессов средствами **mpstat** и **pidstat**, выводящий данные в текстовый файл с именем **уууу-mm-dd_h:m:s_syslog**.

В отчет включите текст скрипта, содержимое файла лога и вывод команды **ls -l** для директории сохранения логов скрипта.

Задание 3. Создание автозапускаемой службы

С помощью утилиты **systemd** зарегистрируйте свою службу по сбору статистики, запускающую скрипт из задания 2 при старте системы. Запустите службу и проверьте ее работу.

В отчете приведите снимок экрана с выводом команды **ls -l** для директории сохранения логов скрипта, статистику выполнения службы и текст файла `.service`.

Задание 4. Запуск процесса по таймеру

Для службы из задания 3 создайте таймер для запуска каждую минуту (отдельный юнит, для службы `my_service.service` – `my_service.timer`). Запустите таймер и проверьте его статус. После нескольких (5-7) минут работы, остановите таймер. Отрадите в отчете результат работы таймера (листинг юнита таймера, снимок экрана со статистикой работы таймера, вывод команды **ls -l** для директории сохранения логов скрипта, вывод `journalctl` для вашей службы и таймера).

Примечание: для таймера создания таймера в секции [Timer] может быть полезным использовать параметр `AccuracySec`, установив его значение, например, `30s`

Задание 5. Получение карты виртуальной памяти процесса

С помощью **rmap** выведите карту виртуальной памяти любого работающего процесса, например, экземпляра терминала. Изучите структуру карты, сделайте вывод, какая информация может быть получена в результате работы **rmap**. В отчет включите снимок экрана с полученной картой.

Задание 6. Настройка файла подкачки

1. Проверьте текущее использование памяти с помощью команды **free**.
2. Создайте файл подкачки размером 1 ГБ.
3. Установите требуемые разрешения и отформатируйте файл подкачки с применением **mkswap**.
4. Подключите созданный файл подкачки с применением команды **swapon**.
5. Убедитесь, что размер swar-файла равен 1 ГБ .
6. Добавьте настройку для автоматического подключения swar-файла при перезагрузке виртуальной машины (необходимо внести изменения в файл `/etc/fstab`), перезагрузите машину и убедитесь в корректности работы.
7. Выполните команды **free -h -t** и **swapon -s**.
8. Деактивируйте созданный файл подкачки и удалите его.
9. Проверьте доступное количество свободной памяти.

В отчет включите снимки экрана для каждого пункта задания.

Задание 7. Тестирование памяти

С помощью утилиты Memtester проведите тестирование 1 Гб свободной памяти (если данный объем недоступен, выберите меньшее количество

свободной памяти), выполнив не менее двух проходов. Отобразите в отчете процесс и результаты тестирования.

Задание 8. Изменение параметра **swappiness**

Получите текущее значение параметра **swappiness**. Измените его. Сделайте скриншот внесенного изменения.

На что влияет значение этого параметра? В отчете приведите ответ в произвольной форме.

ТЕОРЕТИЧЕСКИЕ СВЕДЕНИЯ

Процесс – это программа в стадии выполнения или, с точки зрения операционной системы, сущность, представляющая понятие активности/работы. Каждый процесс обладает уникальным идентификатором (PID) и контекстом выполнения, включающим в себя стек вызовов, снимок содержимого регистров процессора и прочие данные, необходимые для корректного выполнения в многозадачной системе. Операционная система осуществляет управление процессами, планирует их выполнение, выделяет ресурсы процессам, осуществляет контроль за их работой.

Процесс является потребителем ресурсов вычислительной системы, таких как процессорное время, память, дисковое пространство, устройства ввода-вывода, сетевые ресурсы и прочее. Мониторинг ресурсов позволяет выявлять узкие места и проблемы в работе системы, на основе чего принимаются решения по оптимизации настроек, улучшению производительности и предотвращению возможных сбоев и проблем.

Утилиты для мониторинга процессов

Для получения более подробной информации по рассматриваемым утилитам здесь и далее рекомендуется использовать команду **man**.

Утилита **ps** позволяет просмотреть запущенные процессы.

Утилита **top** динамически отображает статистику системы, интерактивная. При запуске работает с окном терминала как динамический процесс. Позволяет получить как общую информацию по загрузке системы, так и детальнее изучить работающие процессы.

Утилита **atop** работает в интерактивном режиме, так же, как и утилита **top**. Atop позволяет получить загрузку по процессору, памяти, дискам и сети.

Команда **atopsar** — это встроенный анализатор логов atop, который позволит локализовать проблему путем получения лога за требуемый период времени. Утилита по умолчанию снимает показания каждые 10 минут, и сохраняет их в лог `/var/log/atop-YYYYMMDD`. Файл конфигураций находится `/etc/default/atop`.

Примеры команд для логирования:

нагрузки на RAM с 09:35:01 до 10:55:01 16 января

atopsar -r /var/log/atop/atop_20230116 -b 09:35:01 -e 10:55:01 -m

нагрузки по сетевым интерфейсам:

atopsar -r /var/log/atop/atop_20230116 -i -b 00:40:00 -e 01:30:00

нагрузки на диски:

atop ***sar -r /var/log/atop/atop_20230116 -d -b 01:10:00 -e 01:30:00***

Пакет **sysstat**

Sysstat – это набор инструментов мониторинга производительности для Linux, установка: ***sudo apt install sysstat***

В состав пакета входят утилиты: **mpstat**, **pidstat**, **vmstat**, **iostat**.

Утилита **mpstat** выводит отчет о использовании процессоров.

Утилита **pidstat** выполняет мониторинг выбранного процесса в реальном времени.

Загрузка служб в Linux

Systemd позволяет планировать выполнение служб, процессов и скриптов и автоматизировать их запуск в фоновом режиме.

Утилита **systemctl** запускает сервисы, описанные в его конфигурации, состоящей из файлов (юнитов).

Системные сервисы расположены в директориях:

/usr/lib/systemd/system/ – юниты из установленных пакетов RPM;

/run/systemd/system/ – динамически создаваемые юниты;

/etc/systemd/system/ – юниты, созданные системным администратором.

Юнит – текстовый файл, содержащий секции вида:

[Название секции в квадратных скобках]

имя_переменной = значение

Для написания простейшего юнита нужно использовать секции:

[Unit] – для описания службы;

[Service] – необходимые для запуска команды;

[Install] – секция используется при установке службы (при enable/disable).

Пример простого юнита, запускающего (после запуска network.target) службу, выполняющую скрипт my_service.sh:

[Unit]

Description=Sysstat service to collect data by my_service.sh

After=network.target

[Service]

Type=simple

ExecStart=/etc/systemd/system/my_service.sh

[Install]

WantedBy=multi-user.target

Для обновления конфигурации **systemctl** выполните команду **daemon-reload**: ***sudo systemctl daemon-reload***.

Для добавления юнита в список служб, запускаемых при загрузке системы, используйте команду **enable**: ***sudo systemctl enable my_service.service***.

Для запуска службы используйте команду **start**: ***sudo systemctl start my_service.service***.

Убедиться в успешности запуска можно с помощью команды **status**, выполненной с указанием имени службы.

Управление памятью в Linux

Управление памятью осуществляется ядром операционной системы и включает в себя выделение памяти процессам, освобождение памяти по их завершении, управление таблицей страниц, кеширование данных, управление файлом подкачки, обработку ошибок памяти и др.

В Linux реализован механизм виртуальной памяти. Это означает, что процессы не работают с физическими адресами напрямую, у каждого процесса имеется свое закрытое виртуальное адресное пространство, что позволяет им располагать большим объемом памяти, чем физически доступно в системе, а также обеспечивает изоляцию адресных пространств одновременно выполняющихся процессов. Этот механизм основан на организации совместного функционирования оперативной и внешней памяти. Процесс выполняет обращение к памяти, используя виртуальный адрес, а механизм виртуальной памяти определяет соответствующий ему физический адрес или выполняет операцию подкачки в случае, если требуемая страница отсутствует в оперативной памяти. Фактическое отображение виртуальных адресов на физические выполняет модуль управления памятью (MMU, Memory Management Unit).

Наименьшей единицей управления памятью с точки зрения ядра является страница. Размер страницы зависит от архитектуры процессора, но исторически сложилось так, что размер страницы в Linux по умолчанию составляет 4 КБ. Позднее, начиная с версии ядра 2.6.3, была внедрена поддержка огромных страниц (huge pages) размером в 2 МБ. Ядро использует структуру *struct page* для отслеживания состояния каждой страницы физической памяти в системе.

Информацию о состоянии виртуальных страниц процесса можно получить с помощью команды **psmap**, вызванной с указанием pid процесса.

Преобразование виртуальных адресов в физические

Для каждого процесса в Linux создается отдельная таблица страниц, состоящая из дескрипторов (записей) для каждой виртуальной страницы процесса. В дескрипторе содержится информация о номере физической страницы, в которую загружена данная виртуальная страница, права доступа к странице и другие атрибуты. Для уменьшения расхода оперативной памяти на хранение служебной информации применяются многоуровневые таблицы страниц.

Виртуальный адрес состоит из нескольких частей, соответствующих иерархии таблицы страниц. Количество индексов и битов, отводимых для каждого индекса в структуре виртуального адреса, зависят от архитектуры процессора и реализации ядра Linux.

Микропроцессоры с архитектурой x86-64 поддерживают четыре уровня таблицы страниц. В настоящее время из 64 разрядов виртуального адреса задействованы только младшие 48 бит. Структура виртуального адреса и иерархия таблицы страниц представлены на рисунке 1:

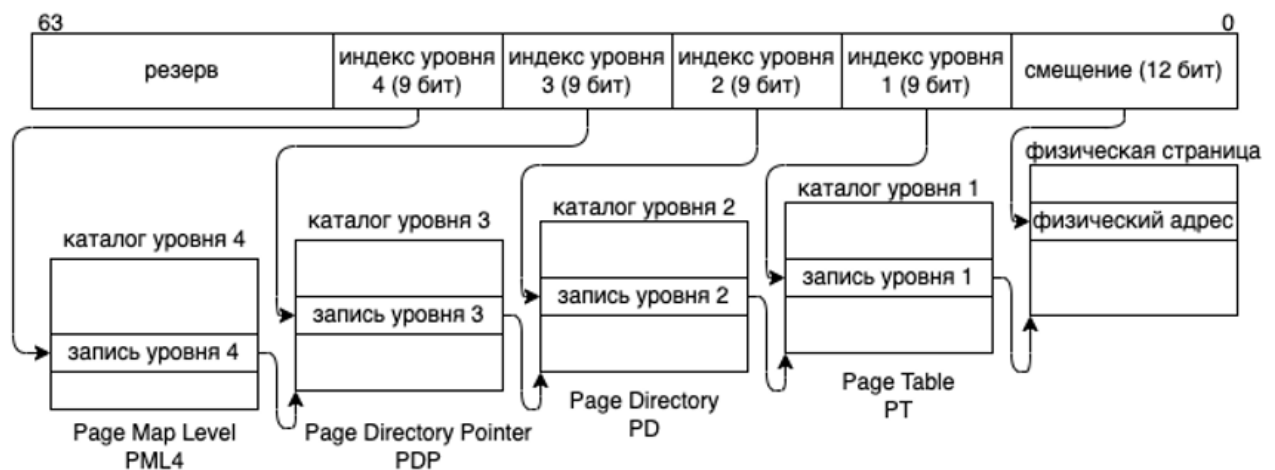


Рис.1. Схема преобразования виртуального адреса в физический в МП x86-64

При трансляции виртуального адреса в физический происходит последовательный переход от уровня к уровню иерархии таблицы страниц, начиная с PML4 и до собственно таблицы страниц, в которой находится дескриптор страницы, соответствующий данному виртуальному адресу. Информация, содержащаяся в этом дескрипторе, используется для преобразования виртуального адреса в физический.

Команды Linux для работы с памятью

top - отображает потребление ресурсов процессами в реальном времени (и др.);

free - утилита для отображения объема свободной и использованной оперативной памяти;

/proc/meminfo — раздел файловой системы /proc содержащий информацию о состоянии памяти в системе:

Пример: **cat /proc/meminfo | grep Mem.**

SWAP

SWAP (или область подкачки) - это часть жесткого диска (отдельный раздел или файл на диске), которая используется операционной системой для хранения временно неиспользуемых фрагментов памяти процессов, перемещенных из оперативной памяти во вторичное хранилище. Дополнительно SWAP используется при организации режима сна (hibernation или suspend to disk). В этом случае в SWAP сохраняется образ всей оперативной памяти.

Необходимый объем SWAP определяется в зависимости от объема оперативной памяти вычислительной системы и настроек перехода в спящий режим. Так для систем с ОЗУ до 2 Гб без перехода в спящий режим под SWAP необходимо выделить 2*ОЗУ, для систем с переходом - 3*ОЗУ; для систем с ОЗУ от 64 Гб не менее 4 Гб, причем спящий режим для таких систем не рекомендуется.

Размер раздела подкачки устанавливается автоматически во время установки системы. Для того, чтобы разрешить спящий режим, необходимо

отредактировать пространство подкачки на этапе пользовательского разбиения.

Утилиты Linux для работы с SWAP

swapon -s; grep Swap /proc/meminfo; free -h – показывают использование и настройки файла подкачки;

mkswap /swapfile - размечает файл /swapfile как файловую систему swap;

swapoff\swapon /swapfile - отключает/подключает файл подкачки в систему;

cat /etc/fstab - выводит список устройств для монтирования. Можно добавить swap раздел.

Стресс-тестирование памяти

Memtester - утилита, предназначенная для стресс-тестирования памяти компьютера на наличие сбоев. Для корректной работы необходимо запускать утилиту с правами суперпользователя и указывать объем свободной памяти, т.к. при работе утилита перезаписывает память.

Для установки используйте: ***apt-get install memtester***.

Запуск проверки свободного блока памяти размера 1024 Мбайт с одним проходом осуществляется командой: ***memtester 1024 1***

Во время работы утилиты Memtester имеется возможность отслеживания выходных данных в режиме реального времени в другом окне терминала: ***tail -f /var/log/syslog***.

После завершения тестирования данные доступны в системном журнале: ***cat /var/log/syslog | grep memtester***.

Приложение А. Требования к оформлению отчета

Выполненная контрольная работа (в виде отчета) в электронной форме загружается в личный кабинет студента в соответствующий элемент курса в ЭИОС МТУСИ.

Отчет по контрольной работе выполняется на стандартных листах бумаги формата А4. Поля: левое поле - 30 мм, правое – 10 мм, верхнее и нижнее – 20 мм. Примерное количество знаков на странице – 2000. Шрифт Times New Roman размером 12, межстрочный интервал 1,5. Страницы текста и рисунки имеют сквозную нумерацию.

Первой страницей является титульный лист, на котором номер страницы не проставляется.

Содержание и список использованных источников оформляются на отдельных страницах.

Отчет должен содержать следующие структурные элементы в указанной последовательности:

- титульный лист;
- содержание;
- цель работы;
- задание 1;
- задание 2;
- ...;
- задание 8;
- выводы;
- список использованных источников.

Приложение Б. Критерии оценки

Критерии оценки контрольной работы	Оценка
Задания контрольной работы полностью выполнены студентом на высоком качественном уровне, все этапы выполнения заданий отражены в отчете. Отчет оформлен в соответствии с предъявленными требованиями. Даны грамотные и точные ответы на поставленные вопросы, продемонстрировано уверенное владение системными утилитами.	отлично
Задания контрольной работы выполнены студентом на хорошем качественном уровне, все этапы выполнения заданий отражены в отчете. Допустимы незначительные недочеты в оформлении отчета. Допустимы несущественные неточности в ответе на вопросы. Продemonстрировано правильное применение системных утилит при решении поставленных задач.	хорошо
Задания контрольной работы выполнены студентом на достаточном качественном уровне, этапы выполнения заданий отражены в отчете. Допустимы недочеты в оформлении отчета. Допустимы неточности в ответе на вопросы. Продemonстрировано в целом правильное применение системных утилит при решении поставленных задач.	удовлетворительно
Допущены существенные ошибки в ходе выполнения контрольной работы, незнание теоретического материала, относящегося к выполненной работе, неумение сформулировать правильные ответы на вопросы, много неточностей в оформлении пояснительной записки	неудовлетворительно