Крайнов Павел Алексеевич

Пашина Софья Александровна

Студенты

г. Москва, РГУ Нефти и Газа им. И.М. Губкина

**Мониторинг системы контрольных групп процессов в ОС Альт**

**Monitoring the process control group system in OS Alt**

***Аннотация:***

В статье рассматриваются основы системы управления контрольных групп процессов (cgroups) в операционной системе Linux, которая позволяет детально регулировать использование системных ресурсов, таких как процессорное время, память и операции ввода-вывода. В ходе исследования продемонстрированы методы настройки cgroups, включая создание групп, распределение ресурсов и мониторинг их использования с помощью утилит командной строки и системных инструментов, таких как `top` и `htop`. Проведен эксперимент с ограничением ресурсов для тестового процесса, результаты которого подтвердили эффективность cgroups в обеспечении стабильной работы системы при высоких нагрузках.

***Abstract:***

The article discusses the basics of the process control group management system (cgroups) in the Linux operating system, which allows for detailed control of the use of system resources such as processor time, memory, and I/O operations. The research demonstrates methods for configuring cgroups, including creating groups, allocating resources, and monitoring their use using command-line utilities and system tools such as 'top` and `htop'. An experiment was conducted with limited resources for the test process, the results of which confirmed the effectiveness of cgroups in ensuring stable system operation under high loads.

***Ключевые слова:*** Linux, ОС Альт, групповые процессы, cgroups, время, память, операции ввода-вывода.

***Key-words:*** Linux, OS Alt, group processes, cgroups, time, memory, I/O operations

**Актуальность**

В последнее время, с ростом многопроцессорных архитектур, растет и количество компьютеров, которые управляются операционными системами на основе Линукс. В ОС Linux, работающей как на серверах, так и на персональных устройствах, обширная поддержка многопроцессорных систем позволяет эффективно регулировать распределение ресурсов, что делает ее удобной для создания различных решений для оптимизации производительности.

Одним из таких механизмов является система управления контрольных групп процессов **cgroups**, которая позволяет детально управлять ресурсами: процессором, памятью, а также операциями ввода-вывода.

**Объект, предмет и цель**

Объект исследования — система контроля ресурсов в операционной системе Linux.

Предмет исследования — утилита cgroups, благодаря которой можно выделять и ограничивать доступ к системным ресурсам для отдельных процессов или для целых групп.

Цель исследования — изучить возможности и ознакомиться с утилитой cgroups для управления ресурсами в Linux, включая настройку и мониторинг использования процессора, памяти и дисковых операций.

**Литературный обзор**

Контрольные группы (cgroups) были впервые представлены в ядре Linux в версии 2.6.24 для реализации функций распределения, ограничения и мониторинга использования системных ресурсов. С тех пор, cgroups активно используется администраторами для повышения производительности, защиты и оптимизации ресурсов. Опишем термины, которые будут использованы в этой статье:

* cgroups — это системная утилита управления ресурсами, используемая для управления процессами и их группами.
* Ограничения ЦП, памяти и ввода-вывода являются основными подсистемами, которые поддерживают рабочие группы и регулируют потребление ресурсов процессами.

**Методы исследования**

Тип исследования: эмпирическое исследование возможностей cgroups для управления ресурсами.

Методы сбора данных: для анализа применяются команды в командной строке и системные утилиты (например, mount, mkdir, echo).

Процедура: создаются виртуальные файловые системы иерархии cgroups для выделения ресурсов и настроек, таких как выбор процессоров и объем доступной памяти. Проводится мониторинг производительности с помощью top и htop.

В этой статье предлагается ознакомиться с основами работы с cgroups - получить основную информацию, научиться создавать групповые процессы, настраивать их и проверять корректность настройки с помощью различных утилит.

**Основы работы с cgroups**

Контрольные группы (cgroups) представляют собой встроенный механизм Linux, предназначенный для управления системными ресурсами. С его помощью можно выделять и ограничивать ресурсы, доступные отдельным процессам или их группам. Эта технология особенно актуальна для повышения производительности, обеспечения безопасности и рационального использования ресурсов системы.

Контрольные группы предоставляют следующие возможности:

* Контрольные группы позволяют распределять процессорное время между разными группами процессов, что помогает избежать перегрузки системы. Например, можно выделить больше процессорного времени для задач с высоким приоритетом, чем для фоновых процессов. Если какой-то процесс, запущенный от имени конкретного пользователя, начинает потреблять более 50% процессорных ресурсов, его активность может быть ограничена, чтобы уменьшить общее потребление и сохранить стабильность системы.
* С помощью cgroups можно задавать ограничения на использование памяти для групп процессов. Благодаря этому можно избежать недостатка памяти в системе и позволить приложениям работать постоянно, без сбоя. Например, ограничение объема доступной памяти предотвращает ситуацию, когда один процесс использует слишком много ресурсов и мешает работе остальных.
* Cgroups помогают управлять дисковыми операциями ввода-вывода, путем ограничения их количества. Это позволяет уменьшить нагрузку на дисковую подсистему и избежать замедления работы в важных ситуациях. Например, можно задать ограничение на число операций чтения или записи в секунду (IOPS), чтобы система имела доступ к дисковым ресурсам для более важных процессов.

Для работы с cgroups можно использовать как специальные утилиты, так и воспользоваться базовыми средствами, такими как команды mount и mkdir. Например, смонтируем виртуальную файловую систему tmpfs, которая далее нам понадобится для создания и управления иерархии cgroups:

*# Монтируем файловую систему tmpfs*

*mount -t tmpfs cgroup\_root /sys/fs/cgroup*

Далее создадим каталоги для подсистем cpu, memory, и cpuset внутри /sys/fs/cgroup, чтобы можно было отдельно настраивать эти ресурсы

*# Создаем подкаталоги для различных подсистем cgroup*

*mkdir -p /sys/fs/cgroup/{cpu,memory,cpuset}*

После, смонтируем виртуальную файловую систему типа cgroup, указав в параметре -o команды mount список имен подсистем, которые будут приписаны иерархии контрольных групп. Например, для подсистемы cpuset код должен быть:

*# Монтируем систему cgroup с подсистемой cpuset*

*mount -t cgroup -o cpuset mycgroup /sys/fs/cgroup/cpuset*

**Создание и настройка контрольной группы**

После того как система cgroup смонтирована, можно создавать иерархии cgroups. Например, для создания группы mycgroup и ограничения ее ресурсов выполним команду:

*# Создаем каталог для новой контрольной группы*

*mkdir /sys/fs/cgroup/cpuset/mycgroup*

Для того, чтобы убедиться, что cgroup успешно смонтирована и доступна для дальнейшей настройки, выведем список всех смонтированных cgroups:

*# Проверяем смонтированные cgroups*

*mount | grep cgroup*

В результате выполнения команды mkdir автоматически создаются псевдофайлы, которые позволяют задавать параметры ресурсов для этой группы. Например, чтобы разрешить группе mycgroup использовать только логические процессоры 2 и 3 и узел памяти 0, пропишем команды:

*# Настройка процессоров и памяти для группы*

*echo 2-3 > /sys/fs/cgroup/cpuset/mycgroup/cpuset.cpus*

*echo 0 > /sys/fs/cgroup/cpuset/mycgroup/cpuset.mems*

Для того чтобы добавить процесс в данную группу, достаточно передать его идентификатор (PID) в соответствующий псевдофайл:

*# Добавляем процесс с ID myPID в группу mycgroup*

*echo myPID > /sys/fs/cgroup/cpuset/mycgroup/tasks*

### **Проверка работоспособности настроек**

Чтобы убедиться, что mycgroup корректно создана и применяет ограничения, используем следующие шаги:

**Создание тестового процесса:**

Для проверки создадим процесс, который будет занимать процессорное время. Например, воспользуемся командой sleep:

*sleep 1000 &*

**Добавление процесса в группу:**

Добавим полученный PID в группу mycgroup, подставив его в команду:

*echo 3635 > /sys/fs/cgroup/cpuset/mycgroup/tasks*

**Проверка параметров cpuset для группы:**

Чтобы убедиться, что ограничения на процессоры и память установлены правильно, выполним следующие команды:

*cat /sys/fs/cgroup/cpuset/mycgroup/cpuset.cpus*

*cat /sys/fs/cgroup/cpuset/mycgroup/cpuset.mems*

Если значения 2-3 и 0 отображаются, то параметры настроены корректно.

**Проверка добавления процесса в группу:**

Далее убедимся, что процесс находится в группе, просмотрев файл tasks:

*cat /sys/fs/cgroup/cpuset/mycgroup/tasks*

PID 3635 отображается в этом файле, что подтверждает его принадлежность к mycgroup.

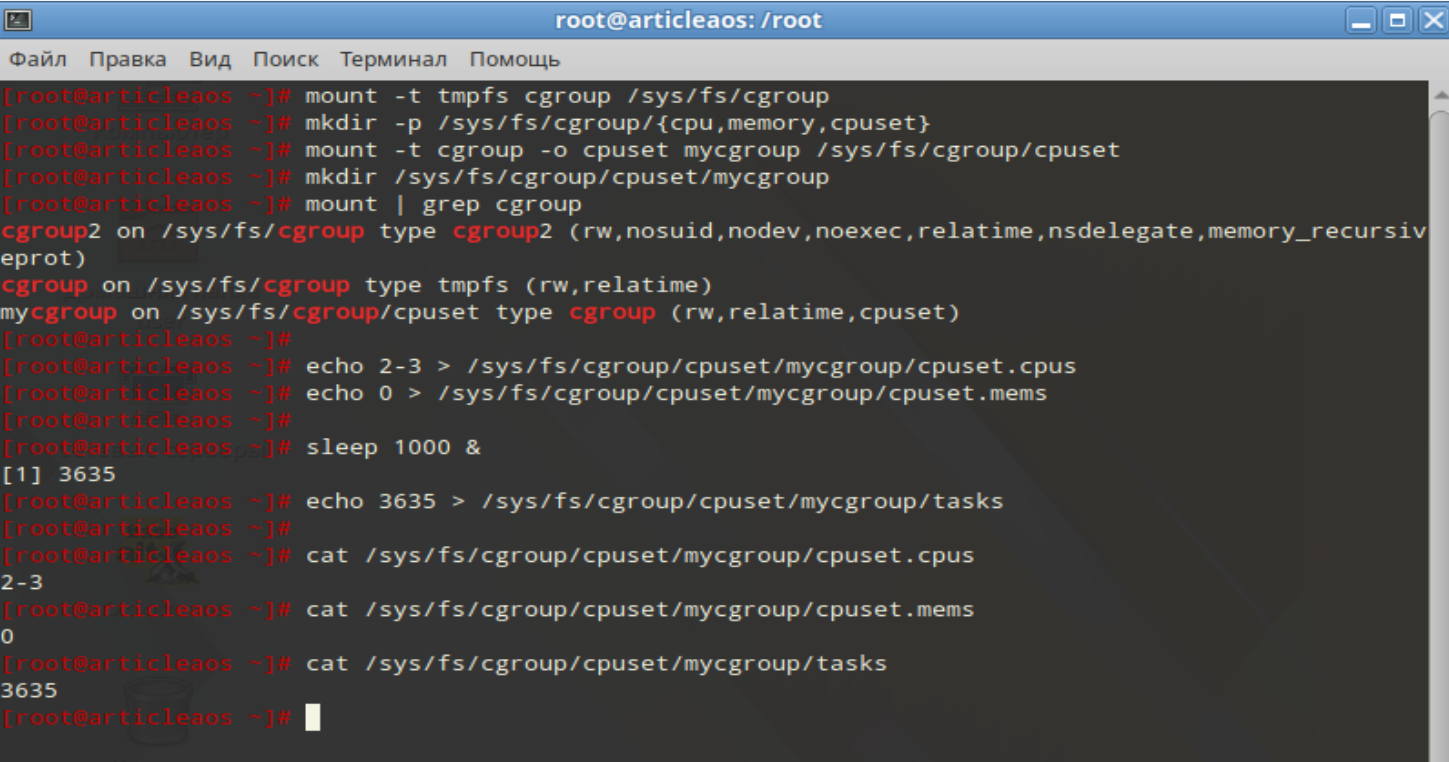


Рисунок 1 – Создание и настройка контрольной группы

**Проверка ранее созданного процесса с помощью утилит top и htop**

***top***

Введем команду top для отслеживания процесса:

*top -p [номер процесса]*

Нажмем “f” для открытия меню настроек колонок и добавим пробелом столбец “Last used CPU”, нажмем “q” и увидим цифры 2 или 3, что означает корректность нашей настройки:

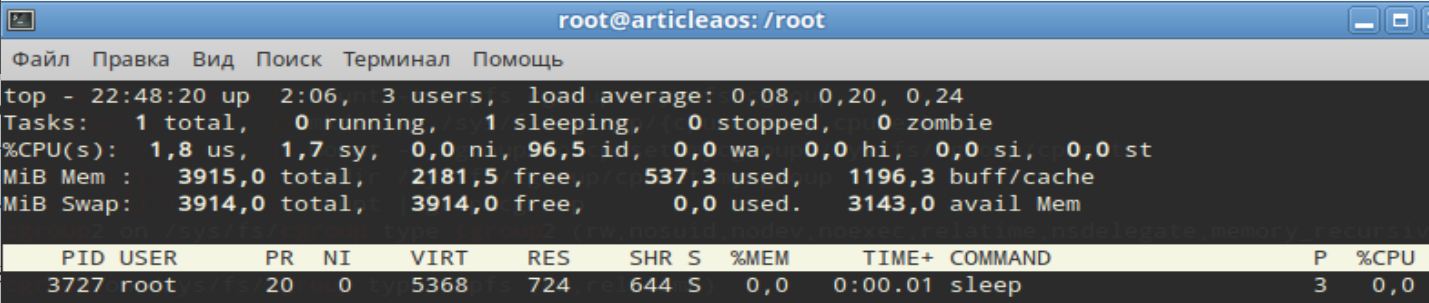


Рисунок 2 – Мониторинг с помощью утилиты top

***htop***

Введем команду htop для отслеживания процесса:

*htop -p [номер процесса]*

В интерфейсе htop нажмем “F2” для перехода в настройки, перейдем к Screens, выберем Processor и затем добавим колонку “Last used CPU”, нажмем F10 для выхода из меню и проверим также, что добавленная колонка показывает 2 или 3, что подтверждает корректность настройки ограничения на использование процессоров:

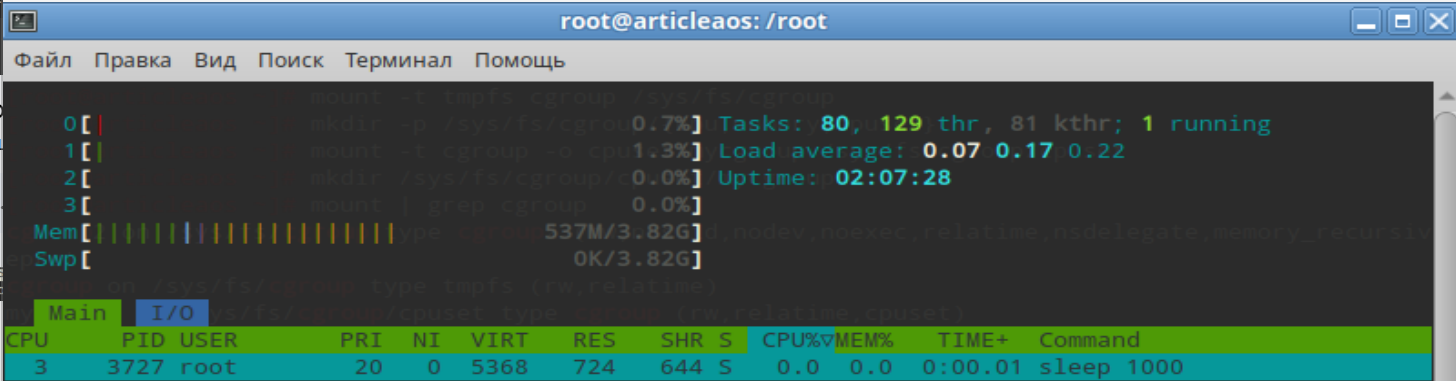


Рисунок 3 – Мониторинг с помощью утилиты htop

**Ограничение нагрузки процессора и памяти с использованием cgroups**

С помощью утилиты можно проверить эффективность ограничения ресурсов процессора и памяти. Для эксперимента были взяты показатели 20% и 50 Мб соответственно. В ходе проведения эксперимента можно оценить, как ограничения влияют на выполнение тестового процесса, который пытается использовать больше ресурсов, чем ему разрешено. Благодаря этому можно наглядно увидеть, что система способна сохранять стабильность работы при высоких нагрузках, предотвращая монополизацию ресурсов отдельными процессами.

**Создание тестовой группы:**

Выполняются команды для создания контрольной группы и задания ограничений:

*mkdir /sys/fs/cgroup/testgroup*

*echo "+cpu +memory" > /sys/fs/cgroup/cgroup.subtree\_control*

*# Ограничение процессора до 20%:*

*echo "20000 100000" > /sys/fs/cgroup/testgroup/cpu.max*

*# Ограничение памяти до 50 МБ:*

*echo $((50 \* 1024 \* 1024)) > /sys/fs/cgroup/testgroup/memory.max*

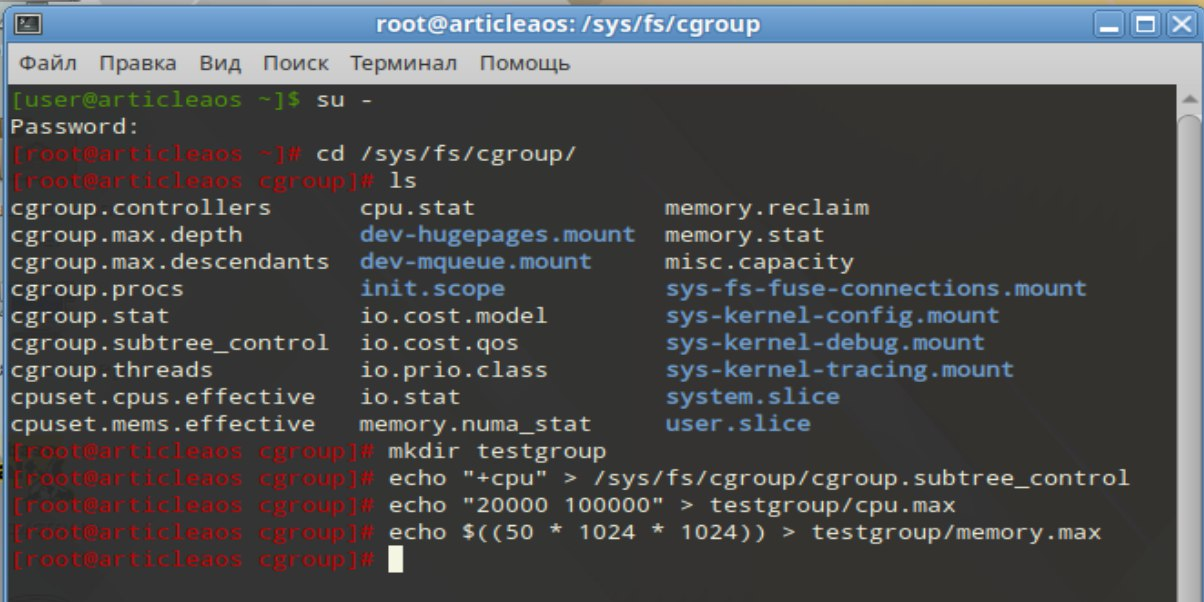


Рисунок 4 – Создание тестовой группы для проведения эксперимента

**Запуск тестового процесса:**

Запускается процесс, который генерирует высокую нагрузку на процессор и память:

*# Генерация высокой нагрузки:*

*dd if=/dev/zero of=/dev/shm/testfile bs=10M count=1024 &*

*# Добавление процесса в группу:*

*echo $! > /sys/fs/cgroup/testgroup/cgroup.procs*

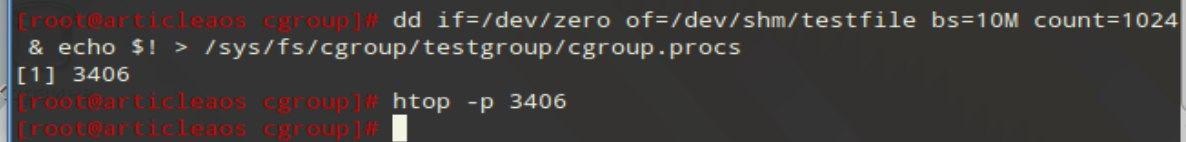


Рисунок 5 – Запуск тестового процесса

**Мониторинг:**

Выполняется мониторинг использования ресурсов с помощью утилит top и htop. Сравниваются фактические показатели нагрузки процессора и памяти с установленными лимитами.

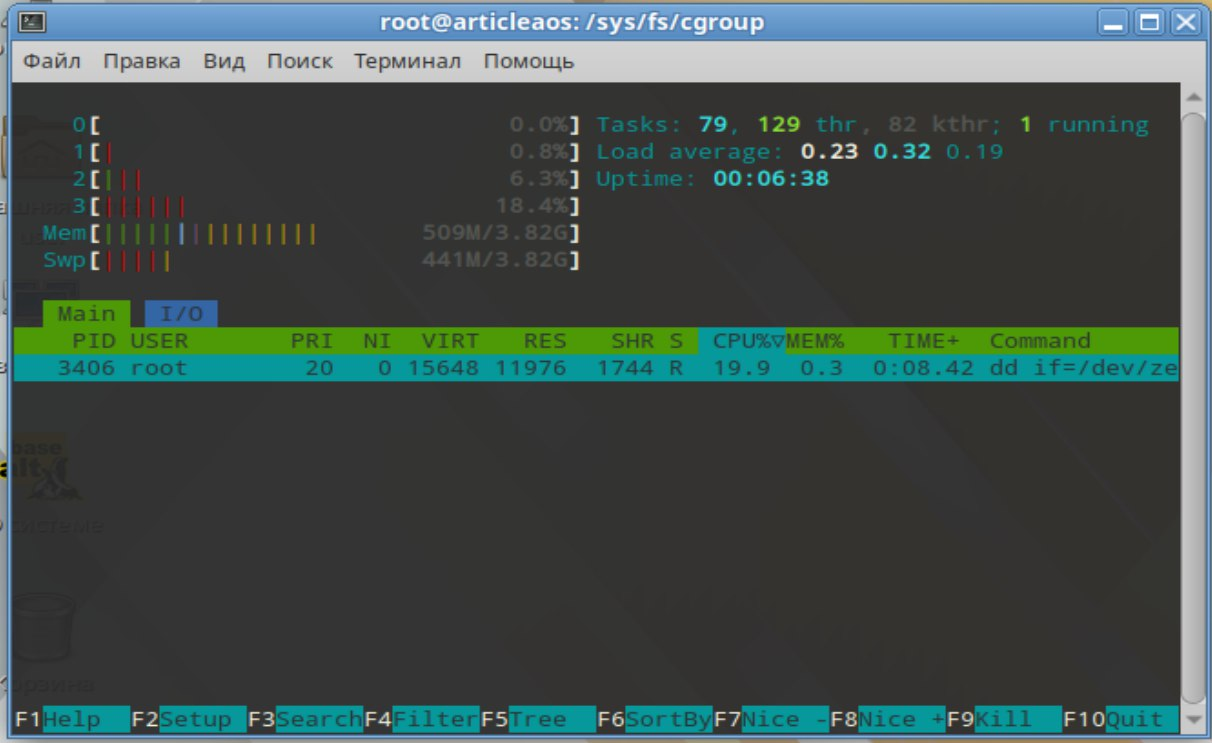


Рисунок 6 – Проверка нагрузки по помощи утилиты htop

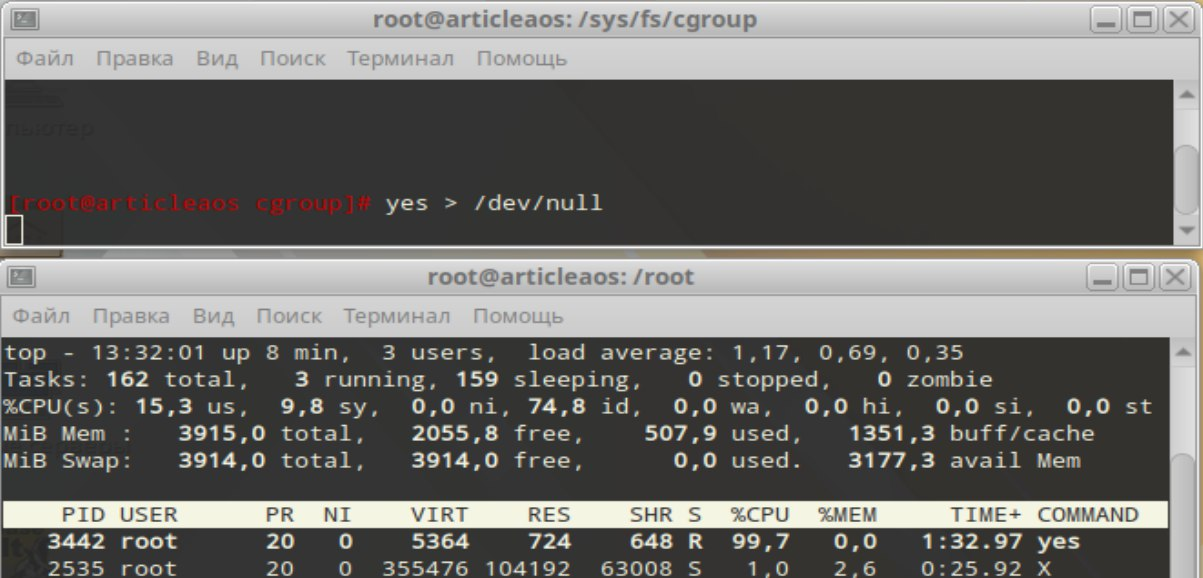


Рисунок 7 – Показатели при нагрузке 100%

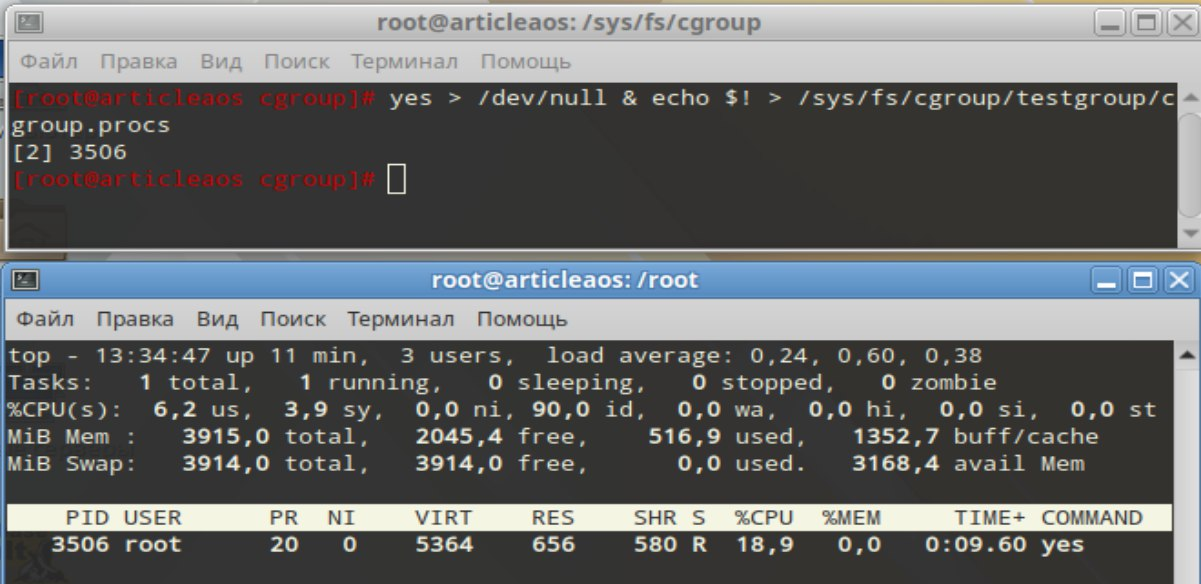


Рисунок 8 – Показатели при нагрузке 20% (ограничение)

Таблица 1 – Показатели тестового процесса при мониторинге

| **Параметр** | **Ожидаемое значение** | **Фактическое значение** | **Примечание** |
| --- | --- | --- | --- |
| **Нагрузка процессора (%)** | До 20% | 18.9% | Процесс не может превысить установленный лимит |
| **Использование памяти (МБ)** | До 50 МБ | 49,8 МБ | Ограничение памяти сработало корректно |
| **Работа процесса** | В рамках ограничений | В рамках ограничений | Процесс завершает работу с учетом лимитов |

В результате проведения эксперимента можем сделать вывод, что:

* Процесс пытался использовать все доступные ресурсы, но контрольная группа ограничила его нагрузку на процессор до 20%. Это позволило системе оставаться стабильной, отзывчивой и доступной для других процессов и задач.
* Лимит в 50 МБ на использование памяти сработал корректно. Процесс работал в рамках заданных ограничений, не влияя на доступность памяти для других процессов.
* Использование контрольных групп доказало свою эффективность в управлении ресурсами. Тестовый процесс не смог превысить заданные ограничения, несмотря на попытки потреблять больше процессорного времени и памяти.

**Таблица сравнения показателей при разном ограничении**

Таблица 2 – Сравнение показателей при разном ограничении

| **Параметр** | **Ограничение 20%** | **Ограничение 100%** | **Примечание** |
| --- | --- | --- | --- |
| **Нагрузка процессора (%)** | 18.9% | 99.7% | При 20% процесс ограничен, система остается стабильной; при 100% — полное использование выделенных ресурсов. |
| **Использование памяти (МБ)** | 49.8 МБ | 1744 МБ | При 20% процесс ограничен до 50 МБ; при 100% использует доступную память без ограничений. |
| **Число активных потоков** | 1 | 1 | Ограничение ресурсов не влияет на количество потоков. |
| **Производительность процесса** | Умеренная | Максимальная | При 20% процесс работает в пределах выделенных ресурсов, при 100% использует все доступные ресурсы. |
| **Влияние на другие процессы** | Минимальное | Высокое | При 20% ресурсы остаются доступными для других задач, при 100% — процесс доминирует. |

* При ограничении 20%:
  + Процесс стабильно работает в рамках выделенных ресурсов.
  + Остальные задачи системы не испытывают дефицита ресурсов, что сохраняет стабильность всей системы.
* При ограничении 100%:
  + Процесс получает максимум доступных ресурсов, но это может негативно повлиять на другие задачи.
  + В условиях серверной нагрузки такое использование может привести к снижению общей производительности.

Использование cgroups с различными уровнями ограничений позволяет гибко адаптировать систему под конкретные задачи и балансировать ресурсы для обеспечения стабильной работы.

**Заключение**

Использование cgroups в Linux дает возможность более гибкого управления и распределения ресурсов, как на серверных системах, так и на стационарных рабочих компьютерах. Данный процесс включает в себя взаимодействие cgroups с разнообразными утилитами мониторинга, такими как top, htop. Это помогает администраторам и простым пользователям получать актуальную информацию о состоянии системы и эффективно использовать ресурсы для выполнения сложных задач, оптимизации нагрузки и улучшения стабильности работы.

**Список использованных источников**

1. Борьба за ресурсы, часть 1: Основы Cgroups [Электронный ресурс]. – URL: https://habr.com/ru/companies/redhatrussia/articles/423051/ (дата обращения: 12.11.2024).
2. Качалов, В. А. Что такое «мониторинг» и «измерение процесса»? // Методы менеджмента качества. – 2008. – № 1-2.
3. Контрольная группа (Linux) [Электронный ресурс]. – URL: https://en.wikipedia.org/wiki/Cgroups (дата обращения: 12.11.2024).
4. Матяш, Д. В. Организация системы мониторинга в процессе управления проектами компании // Материалы конференции.
5. Механизмы контейнеризации: cgroups [Электронный ресурс]. – URL: https://habr.com/ru/companies/selectel/articles/303190/ (дата обращения: 12.11.2024).
6. Уймин, А. Г. Цифровые двойники сетевых инфраструктур: точность, методы и практические решения / А. Г. Уймин // Радиотехнические и телекоммуникационные системы. – 2023. – № 3(51). – С. 44-52. – DOI 10.24412/2221-2574-2023-3-44-52. – EDN QUSITK.