**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

**“Большие данные и распределенная цифровая платформа”**

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Системное программирование в Linux»**

**на тему «Создание демона для регулярного резервного копирования данных»**

**Студент гр. 23Б15-пу**

**Абрахин Е.Д.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc178975364)
2. [Описание задачи (формализация задачи) 4](#_Toc178975365)
3. [Теоретическая часть 6](#_Toc178975366)
4. [Основные шаги программы 9](#_Toc178975367)
5. [Описание программы 11](#_Toc178975368)
6. [Рекомендации пользователя 13](#_Toc178975369)
7. [Рекомендации программиста 14](#_Toc178975370)
8. [Исходный код программы 15](#_Toc178975371)
9. [Контрольный пример 16](#_Toc178975372)
10. [Вывод 16](#_Toc178975373)

# Цель работы

Целью данной работы является разработка системного процесса (демона) для автоматического и регулярного резервного копирования данных на основе заданных параметров конфигурации. Программа должна обеспечивать автоматическое копирование файлов с одного каталога в другой в фоновом режиме, с возможностью задания частоты копирования, мониторинга работы демона и журналирования операций.

# Описание задачи (формализация задачи)

Необходимо создать программное обеспечение, работающее в качестве демона, которое будет выполнять регулярное резервное копирование файлов и каталогов с одного места хранения в другое. Программа должна поддерживать настройку параметров через конфигурационный файл, корректно работать в фоновом режиме, управляться через командную строку и обеспечивать защиту от сбоев.

Задача делится на несколько подзадач:

1. Создание демона:

* Реализовать запуск программы в качестве фонового процесса (демона), который будет выполняться независимо от пользовательских сессий и поддерживать автоматическое выполнение резервного копирования.
* Процесс должен запускаться вместе с операционной системой и выполнять свои функции, даже если пользовательская сессия завершена.

1. Чтение конфигурационного файла:

* Программа должна уметь считывать конфигурационный файл, в котором будут указаны исходный каталог для резервного копирования, каталог для хранения резервных копий, а также периодичность выполнения копий (в минутах или часах).
* Конфигурационный файл должен содержать гибкие настройки, позволяющие управлять параметрами работы демона без необходимости изменения кода программы.

1. Процесс резервного копирования:

* Демон должен следить за временем, чтобы выполнять резервное копирование файлов из исходного каталога в каталог назначения согласно заданному в конфигурации расписанию.
* При каждом копировании к имени каталога резервной копии необходимо добавлять временную метку, чтобы различать версии копий.
* Исключить копирование системных директорий, таких как /dev, /proc, /sys, /mnt и другие, которые не нуждаются в резервировании.

1. Журналирование операций:

* Реализовать логирование всех действий программы в системный журнал, включая успешные и неудачные попытки резервного копирования, чтение конфигурации и запуск/остановку демона.
* Логи должны быть доступны для анализа и позволять отслеживать ход выполнения резервного копирования.

# Теоретическая часть

Современные вычислительные системы часто сталкиваются с необходимостью регулярного резервного копирования данных. Это особенно актуально в условиях постоянного увеличения объемов информации, когда традиционные методы ручного копирования становятся недостаточно эффективными и требуют автоматизации. Создание фоновых процессов (демонов), отвечающих за выполнение регулярного резервного копирования данных, позволяет автоматизировать эту задачу, снижая риск потери данных и минимизируя участие пользователя.

Демоны в системах Unix-подобных операционных систем

Демоны — это фоновые процессы, которые работают в системе независимо от взаимодействия с пользователем. Они автоматически выполняют задачи, назначенные на определенные интервалы времени или при наступлении определенных условий. Основные характеристики демонов включают:

* Фоновое выполнение: Демоны продолжают свою работу даже после завершения пользовательских сессий.
* Автономность: Демоны часто выполняют свои задачи без прямого контроля со стороны пользователя.
* Журналирование (логирование): Для отслеживания работы демоны записывают все ключевые события и ошибки в системный журнал.

В контексте резервного копирования демоны играют важную роль, так как они позволяют непрерывно и надежно создавать резервные копии данных, что критически важно для обеспечения сохранности информации и восстановления в случае сбоев системы или утраты данных.

Автоматизация процесса резервного копирования

Резервное копирование данных представляет собой копирование файлов и каталогов из исходного местоположения в другой каталог для защиты от потерь. Этот процесс должен происходить регулярно, и автоматизация с использованием демонов позволяет обеспечить бесперебойное выполнение резервного копирования с минимальным вмешательством пользователя.

Основные задачи, решаемые при разработке демона для резервного копирования:

* Чтение конфигурационных файлов: Демон должен уметь загружать параметры, такие как исходные и целевые каталоги, периодичность копирования и список исключений, из конфигурационного файла. Это позволяет гибко управлять поведением программы без необходимости ее перекомпиляции или ручного изменения параметров в коде.
* Регулярное копирование данных: Демон автоматически выполняет копирование файлов по расписанию, сохраняя временные метки для всех резервных копий. Это обеспечивает возможность возврата к различным версиям данных и защиты от случайных изменений или удаления файлов.
* Журналирование операций: Каждая операция резервного копирования должна быть зарегистрирована в системном журнале. Логи позволяют отслеживать успешность резервного копирования, а также диагностировать возможные ошибки или сбои в работе демона.

Преимущества автоматического резервного копирования

Автоматизация процесса резервного копирования обладает рядом преимуществ по сравнению с ручными методами:

* Минимизация человеческого фактора: Автоматический процесс исключает риск того, что пользователь забудет выполнить резервное копирование или сделает это некорректно.
* Экономия времени: Демон работает в фоновом режиме и не требует от пользователя каких-либо действий, кроме начальной настройки.
* Непрерывность работы: Демон запускается при старте системы и обеспечивает регулярное выполнение резервных копий, даже если пользователь не находится за компьютером.

Механизм работы демона

Алгоритм работы демона включает несколько ключевых этапов:

* Инициализация и запуск в фоновом режиме: Демон запускается при старте системы или вручную, переходит в фоновый режим, отделяется от управляющего терминала и начинает выполнение своих задач.
* Чтение и обработка конфигурационного файла: Демон загружает конфигурацию, содержащую параметры для копирования, такие как исходный каталог, каталог назначения, частота копирования, а также исключения (каталоги и файлы, которые не нужно копировать).
* Выполнение резервного копирования: По наступлению времени, указанного в конфигурации, демон запускает процесс копирования файлов. Каждая резервная копия сохраняется с уникальной временной меткой, что позволяет хранить несколько версий данных.
* Журналирование результатов: После завершения каждого копирования демон записывает результаты операции в системный журнал — это может быть информация об успешном копировании, а также данные о возможных ошибках или сбоях.

Обеспечение безопасности

Одним из важных аспектов при разработке демона является обеспечение безопасности. Важно защитить конфигурационные файлы и каталоги с резервными копиями от несанкционированного доступа, чтобы злоумышленники или обычные пользователи не могли их изменять или удалять. Для этого должны использоваться механизмы управления доступом, такие как назначение правильных прав на файлы и каталоги, а также ограничение круга пользователей, которые могут управлять демоном.

# Основные шаги программы

1. Запуск программы Программа запускается из командной строки с параметром, который указывает действие: запуск демона, его остановка или проверка статуса. В зависимости от переданного аргумента вызывается соответствующая функция: startDaemon(), stopDaemon() или getStatus().
2. Чтение конфигурационного файла Функция readConfig() загружает параметры из конфигурационного файла. Эти параметры включают исходный каталог для резервного копирования, каталог для хранения резервных копий и частоту создания резервных копий (в минутах). Если файл не удается открыть, демон выводит сообщение об ошибке в системный журнал.
3. Запуск демона Функция startDaemon() запускает процесс в фоновом режиме:

* Процесс форкается, и родительский процесс завершается.
* Создается новая сессия с помощью setsid().
* Демон закрывает стандартные потоки ввода, вывода и ошибок.
* PID демона сохраняется в файл /var/run/backup\_daemon.pid для последующего управления процессом.
* После запуска демон начинает основную работу — выполнение резервного копирования с заданной периодичностью.

1. Цикл работы демона В функции daemonLoop() демон бесконечно выполняет цикл, в котором:

* Загружает параметры из конфигурационного файла для обновления конфигурации.
* Запускает процесс резервного копирования с помощью функции createBackup().
* Ждет интервал времени, указанный в конфигурации, перед выполнением следующего резервного копирования (используется функция std::this\_thread::sleep\_for()).

1. Создание резервной копии В функции createBackup() выполняется процесс резервного копирования:

* Создается уникальная директория для копии с временной меткой.
* Используется команда rsync для копирования данных из исходного каталога в директорию для резервных копий, исключая системные каталоги (например, /proc, /dev).
* Процесс rsync запускается через функцию popen(), а его результат записывается в лог-файл.
* Если копирование проходит успешно, в системный журнал записывается сообщение о завершении резервного копирования.
* • Остановка демона Функция stopDaemon():
* Считывает PID демона из файла и отправляет сигнал SIGTERM для завершения процесса.
* Если процесс успешно завершен, файл с PID удаляется, и в системный журнал записывается сообщение о завершении работы демона.

1. Проверка статуса демона Функция getStatus() проверяет, работает ли демон:

* Считывается PID из файла.
* Если процесс с таким PID существует, в журнал записывается сообщение о том, что демон работает.
* Если процесс не найден, в журнал выводится сообщение о том, что демон не запущен.

1. Журналирование Во всех ключевых точках программы ведется журналирование с помощью функции syslog(). В журнал записываются ошибки, результаты резервного копирования, информация о запуске и остановке демона.

# Описание программы

Программа написана на языке C++ и предназначена для автоматического регулярного резервного копирования данных с помощью демона, работающего в фоновом режиме. Основные системные библиотеки, используемые в программе: <chrono>, <csignal>, <cstdio>, <cstdlib>, <ctime>, <errno.h>, <fstream>, <iostream>, <map>, <string>, <syslog.h>, <sys/stat.h>, <sys/types.h>, <thread>, <unistd.h>. Программа организована в единый модуль backup\_daemon.cpp, который включает в себя основной процесс демона и несколько вспомогательных функций.

Таблица 1. main.cpp

|  |  |  |
| --- | --- | --- |
| Функция | Описание | Возвращаемое значение |
| createBackup | Выполняет резервное копирование файлов с помощью команды rsync. Создает папку с временной меткой для каждой копии. | int (код возврата) |
| daemonLoop | Основной цикл работы демона: загружает конфигурацию, выполняет резервное копирование и ждет до следующего цикла. | void |
| getStatus | Проверяет, запущен ли демон, и выводит информацию в системный журнал. | void |
| readConfig | Считывает параметры конфигурации из файла и сохраняет их в глобальную переменную config. | int (PID процесса) |
| readPIDFromFile | Читает PID демона из файла для дальнейшего управления (остановка или проверка статуса). | void |
| savePIDToFile | Сохраняет PID текущего процесса демона в файл для последующего использования при остановке или проверке статуса. | void |
| startDaemon | Запускает процесс демона в фоновом режиме, создает новую сессию, сохраняет PID и запускает основной цикл работы. | void |
| stopDaemon | Останавливает процесс демона, отправляя ему сигнал SIGTERM, и удаляет файл с PID. | void |

# Рекомендации пользователя

После установки и настройки программы, пользователю рекомендуется следовать следующей инструкции для эффективного использования демона резервного копирования:

1. Настройка конфигурационного файла:

* Откройте файл backup\_config.ini, который находится в директории /opt/backup\_daemon/.
* Убедитесь, что указаны правильные пути к исходному каталогу и каталогу для резервных копий.
* Задайте частоту резервного копирования (в минутах), указав соответствующее значение.

Пример содержания конфигурационного файла:

source\_dir=/path/to/source\_directory

backup\_dir=/path/to/backup\_directory

backup\_frequency\_minutes=30

* Убедитесь, что указанные директории действительно существуют.

1. Запуск демона:

* После настройки конфигурации, выполните команду запуска демона:

sudo systemctl start backup\_daemon

* Демон будет работать в фоновом режиме, автоматически создавая резервные копии в соответствии с заданной конфигурацией.

1. Проверка статуса:

* Для проверки статуса демона выполните команду:

sudo systemctl status backup\_daemon

* Это покажет, работает ли демон и его текущий статус.

1. **Остановка демона**:

* Чтобы остановить выполнение демона, используйте команду:

sudo systemctl stop backup\_daemon

# Рекомендации программиста

Если вы хотите внести изменения в программу или собрать её самостоятельно, следуйте этим рекомендациям:

1. Клонирование репозитория:

* Сначала клонируйте репозиторий с кодом на вашу локальную машину:

git clone https://github.com/FasterXaos/System\_programming\_in\_Linux.git

cd your-repository-directory

1. Сборка программы:

* Убедитесь, что у вас установлен CMake и необходимые библиотеки для сборки. Затем выполните:

./build.sh

* Этот скрипт создаст необходимые файлы для сборки.

1. Перемещение файлов:

После успешной сборки:

* Переместите файл backup\_daemon.service в директорию /etc/systemd/system/:

sudo mv backup\_daemon.service /etc/systemd/system/

* Переместите файл backup\_config.ini в директорию /opt/backup\_daemon/:

sudo mv backup\_config.ini /opt/backup\_daemon/

* Переместите файл backup\_daemon в директорию /opt/backup\_daemon/:

sudo mv build\_ninja /opt/backup\_daemon/

1. Настройка службы:

* После перемещения файла службы, выполните следующую команду, чтобы перезагрузить систему управления службами:

sudo systemctl daemon-reload

1. Запуск и тестирование:

* Теперь вы можете запустить демона с помощью команды:

sudo systemctl start backup\_daemon

* Проверьте, работает ли он корректно, используя команду:

sudo systemctl status backup\_daemon

* Выполните следующую команду для автоматического запуска демона:

sudo systemctl enable backup\_daemon

1. Отладка:

* Если возникают ошибки, проверьте журналы с помощью:

journalctl -u backup\_daemon

1. Безопасность:

Обеспечьте безопасность демона, ограничив доступ к конфигурационным файлам и резервным копиям. Это поможет предотвратить несанкционированный доступ и изменение важных данных.

* Установите права доступа на файл конфигурации backup\_config.ini, чтобы только пользователь root или соответствующий пользователь службы мог его читать и изменять:

sudo chmod 600 /opt/backup\_daemon/backup\_config.ini

sudo chown root:root /opt/backup\_daemon/backup\_config.ini

* Аналогично, ограничьте доступ к директории с резервными копиями, чтобы предотвратить их случайное удаление или модификацию:

sudo chmod 700 /path/to/backup/directory

sudo chown root:root /path/to/backup/directory

* Это гарантирует, что только авторизованные пользователи или процессы смогут управлять конфигурациями и данными резервного копирования, что снижает риск атак и случайных ошибок.

# Исходный код программы

[https://github.com/FasterXaos/System\_programming\_in\_Linux.git](https://github.com/FasterXaos/System_programming_in_Linux.git%20)

# Контрольный пример

1. Запуск программы

Для запуска программы необходимо запустить службу резервного копирования с помощью командной строки. Используйте следующую команду для старта демона:

sudo systemctl start backup\_daemon

После запуска демон начнет свою работу в фоновом режиме, создавая резервные копии через заданные промежутки времени, указанные в конфигурационном файле backup\_config.ini.

* 1. Просмотр процесса выполнения резервного копирования

Демон будет автоматически создавать резервные копии через указанные промежутки времени. Чтобы проверить статус работы демона в любой момент, выполните команду:

sudo systemctl status backup\_daemon

Вывод демонстрирует успешное выполнение и создание резервных копий, если демон работает без ошибок (Рис. 1).

* 1. Остановка программы

Для остановки демона выполните команду:

sudo systemctl stop backup\_daemon

Это остановит резервное копирование.

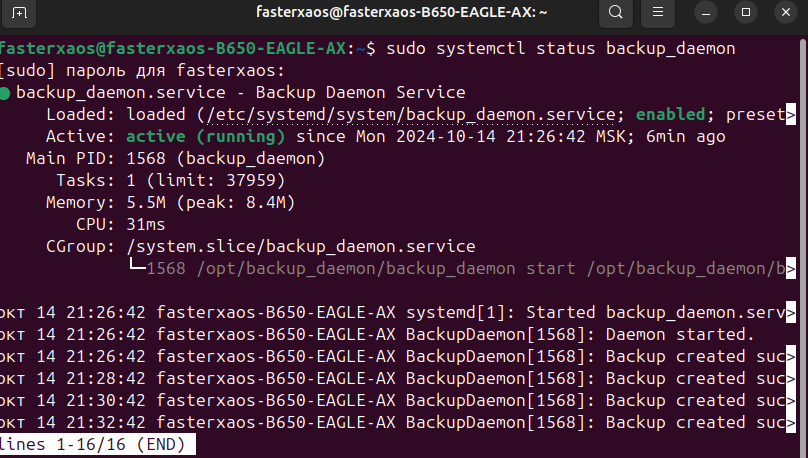


Рис 1. пример вывода статуса программы

* 1. Просмотр результатов

Все резервные копии будут сохраняться в директории, указанной в конфигурационном файле backup\_config.ini (Рис. 2).

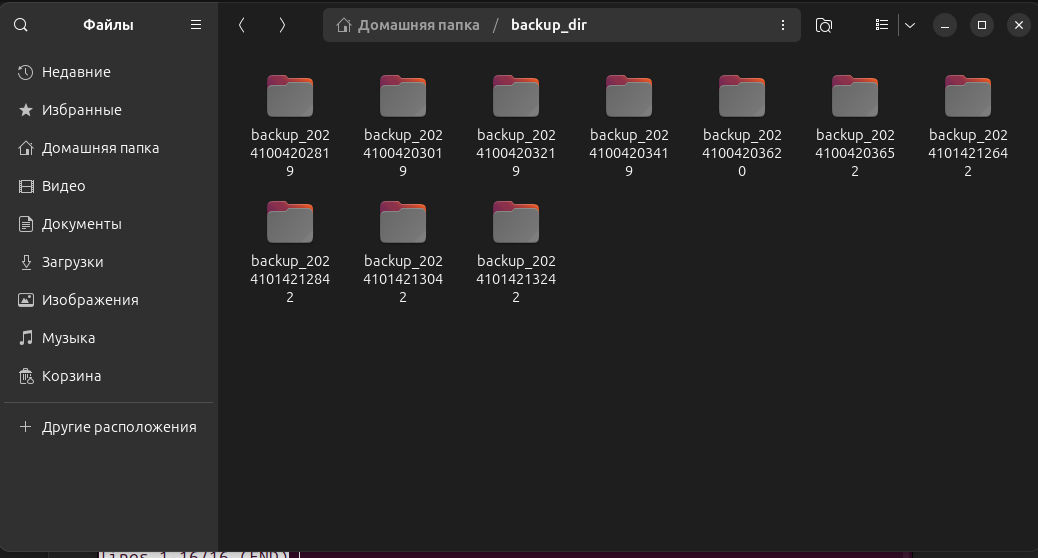


Рис. 2 пример структуры каталога с резервными копиями

# Вывод

В ходе выполнения данной работы была разработана программа-демон для автоматизированного создания резервных копий с использованием системных средств Linux. Основной задачей было создать эффективную и надежную систему для регулярного резервного копирования данных, работающую в фоновом режиме и требующую минимального вмешательства со стороны пользователя.

Программа реализована на языке C++ с использованием многопоточной обработки и системных вызовов для создания резервных копий с помощью утилиты rsync. Основные функции программы включают в себя создание резервных копий по заданному расписанию, мониторинг состояния демона, а также ведение логов для отслеживания статуса и возможных ошибок.

Был реализован механизм автоматического запуска и остановки демона через систему systemd, что позволило интегрировать решение в стандартную инфраструктуру Linux для управления службами. Программа поддерживает параллельную обработку данных и гарантирует надежное сохранение копий, исключая из копирования системные директории, которые могут содержать динамические или временные файлы.

Разработанное решение предоставляет пользователю удобный инструмент для автоматизации процессов резервного копирования, обеспечивает сохранность данных и повышает надежность системного администрирования.

# Литература

1. "man pages" для команд и системных вызовов:

man 2 fork

man 2 setsid

man 3 syslog

man 1 rsync

man systemd.service  
Linux man pages: <https://man7.org/linux/man-pages/>

1. Lennart Poettering. *systemd for Administrators*. Документация по systemd, <https://www.freedesktop.org/wiki/Software/systemd/>
2. "The Linux Documentation Project".  
   Большой сборник документации по различным аспектам Linux,  
   <https://www.tldp.org/>
3. *rsync Documentation*.   
   <https://download.samba.org/pub/rsync/rsync.html>
4. Michael Kerrisk. *Daemons Writing Guide* на LWN.net.  
   Практическое руководство по созданию демонов в Linux с использованием системных вызовов и управления процессами:  
   <https://lwn.net/Articles/573090/>