Санкт-Петербургский государственный университет

Факультет прикладной математики – процессов управления

**Лабораторная работа №2**

**Отчет**

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

**Создание демона для регулярного резервного копирования данных.**

Автор работы: Дацык Р.В.

Группа: 22.Б15-пу

Преподаватель: Дик А.Г.

Санкт-Петербург, 2023

**Оглавление**

[**1.** **Цель работы** 3](#_Toc151824592)

[**2.** **Задачи** 3](#_Toc151824593)

[**3.** **Введение** 3](#_Toc151824594)

[**4.** **Теоретическая часть** 3](#_Toc151824595)

[**5.** **Алгоритм метода** 5](#_Toc151824596)

[**6.** **Описание программы** 5](#_Toc151824597)

[**7.** **Рекомендации пользователю** 7](#_Toc151824598)

[**8.** **Рекомендации программисту** 7](#_Toc151824599)

[**9.** **Контрольный пример** 8](#_Toc151824600)

[**10.** **Вывод** 9](#_Toc151824601)

[**11.** **Список использованной литературы** 9](#_Toc151824602)

# **Цель работы**

Создание демона для автоматизированного регулярного резервного копирования данных в среде Linux. Работа включала в себя разработку конфигурационного файла, написание кода демона на языке C++, интеграцию с системой и обеспечение безопасности функционирования, а также создание документации пользователя.

# **Задачи**

* 1. **Настройка конфигурации:**

1. Разработка конфигурационного файла для указания параметров, таких как исходный каталог, каталог для резервных копий, частота резервного копирования и другие.
   1. **Создание демона:**
2. Написание кода демона на языке C++ для:
3. Считывания конфигурации из конфигурационного файла.
4. Ожидания наступления времени для следующей резервной копии.
5. Создания резервных копий файлов с добавлением временной метки.
6. Журналирования выполнения операций в системный журнал.
   1. **Интеграция с системой:**
7. Внедрение демона в систему так, чтобы он запускался при старте операционной системы и регулярно выполнял резервное копирование согласно настройкам.
   1. **Управление и мониторинг:**
8. Создание механизма для управления демоном через командную строку, включая возможность остановки, перезапуска и изменения настроек.
9. Реализация механизма мониторинга для отслеживания статуса демона и успешности резервного копирования.
   1. **Документация:**
10. Создание документации для пользователя, включая описание настроек, команд управления и процесса резервного копирования.
    1. **Тестирование и оптимизация:**
11. Тестирование демона на различных сценариях и оптимизация для эффективной работы с минимальной нагрузкой на систему.
    1. **Безопасность:**
12. Обеспечение безопасности демона, включая ограничение доступа к конфигурационным файлам и резервным копиям.

# **Введение**

Цель данной лабораторной работы заключается в разработке и внедрении системного демона на языке C++ для регулярного резервного копирования данных в среде Linux. Проект включает в себя создание конфигурационного файла, интеграцию демона в систему с использованием systemd, а также реализацию управления и мониторинга. В этом контексте были выполнены задачи по обеспечению безопасности, оптимизации и документированию процесса работы демона.

# **Теоретическая часть**

Системное программирование в среде Linux предполагает использование демонов и системы инициализации systemd. Демон – это фоновый процесс, работающий независимо от пользовательской сессии, управление которым обеспечивается systemd.

Создание демона для регулярного резервного копирования данных включает в себя работу с файлами, организацию фонового выполнения, обеспечение безопасности и интеграцию с системой инициализации.

Использование конфигурационных файлов позволяет гибко настраивать параметры работы демона, такие как каталоги для резервных копий и интервалы времени.

Вопросы безопасности решаются ограничением доступа к конфигурационным файлам и резервным копиям. Оптимизация направлена на эффективное использование ресурсов системы.

Документация играет ключевую роль в облегчении взаимодействия пользователя с демоном, предоставляя понятные инструкции по настройке, управлению и мониторингу. Таким образом, создание и внедрение демона для регулярного резервного копирования данных требует комплексных навыков в системном программировании.

# **Алгоритм метода**

**Алгоритм работы демона для регулярного резервного копирования данных:**

1. **Инициализация:**
   * Загрузка конфигурационных параметров из файла.
   * Установка рабочей директории.
   * Настройка журналирования для записи в системный журнал.
2. **Бесконечный цикл:**
   * Ожидание наступления времени для следующей резервной копии с использованием таймера.
   * Считывание конфигурации перед каждой операцией резервного копирования.
3. **Резервное копирование:**
   * Создание временной метки для отличия каждой резервной копии.
   * Копирование файлов из исходного каталога в каталог для резервных копий с учетом временной метки.
   * Журналирование успешного выполнения операции в системный журнал.
4. **Управление и мониторинг:**
   * Возможность управления демоном через командную строку (остановка, перезапуск, изменение настроек).
   * Механизм мониторинга статуса демона и успешности резервного копирования.
5. **Безопасность:**
   * Ограничение доступа к конфигурационным файлам и каталогам с резервными копиями.
   * Установка минимально необходимых привилегий для работы демона.
6. **Документация:**
   * Создание подробной документации для пользователя, описывающей настройки, команды управления и процесс резервного копирования.
7. **Тестирование и оптимизация:**
   * Тестирование демона на различных сценариях работы.
   * Оптимизация кода для эффективной работы и минимизации нагрузки на систему.
8. **Интеграция с системой:**
   * Внедрение демона в систему, чтобы он запускался при старте операционной системы и регулярно выполнял резервное копирование согласно настройкам.
9. **Завершение:**
   * Обработка завершения работы демона при выходе из бесконечного цикла.
   * Завершение работы демона при получении соответствующей команды.
10. **Выход:**

* Завершение выполнения демона с уведомлением в системный журнал.

# **Описание программы**

Алгоритм реализован на языке С++ с использованием следующих пакетов:<iostream>, <vector>, <cstdlib>, <ctime>, <string>

Ниже представлена таблица описывающая функции, которые были использованы в программе

Таблица 6.1. Описание функций

|  |  |
| --- | --- |
| Название функции | Назначение функции |
| **readConfigFile()** | Считывает конфигурационный файл, извлекает необходимые параметры (например, исходный каталог, каталог для резервных копий, частоту резервного копирования) и сохраняет их для дальнейшего использования. |
| **createBackup()** | Создает резервную копию файлов из исходного каталога в каталог для резервных копий. Каждая резервная копия должна быть уникальной и может включать временную метку для идентификации. |
| **startDaemon()** | Запускает демона в фоновом режиме. Включает цикл ожидания наступления времени для следующей резервной копии. |
| **logMessage()** | Записывает сообщение в системный журнал, чтобы отслеживать выполнение операций и статус демона. |
| **stopDaemon()** | Останавливает выполнение демона. Это может быть вызвано, например, при завершении работы операционной системы. |
| **restartDaemon()** | Перезапускает демона. Может быть полезно после изменения конфигурации или других параметров. |
| **reloadConfig()** | Перезагружает конфигурацию без остановки демона. Позволяет внесение изменений в конфигурацию без прерывания работы демона. |
| **monitorDaemon()** | Реализует механизм мониторинга статуса демона. Может включать проверку успешности резервного копирования и других ключевых параметров. |
| **controlDaemon(command)** | Обрабатывает команды управления демоном. Например, может включать обработку команд остановки, перезапуска или изменения конфигурации. |
| **Main()** | Инициализирует и читает конфигурацию, запускает демона в фоновом режиме и входит в бесконечный цикл регулярного резервного копирования. |

# **Рекомендации пользователю**

Для успешного запуска программы необходимо устройство с операционной системой Linux, macOS или Windows, а также среда разработки, поддерживающей запуск компилятора С++

Для установки Backup Daemon скомпилируйте исходный код с помощью команды g++ -o backup\_daemon main.cpp, разместите бинарный файл в /home/user/daemon/. Настройте демона в файле конфигурации /home/user/daemon/config.txt, указав исходный и целевой каталоги, частоту резервного копирования и прочие параметры.

Для запуска используйте sudo systemctl start backup\_daemon.service, остановка - sudo systemctl stop backup\_daemon.service, перезапуск - sudo systemctl restart backup\_daemon.service. После изменения настроек перезагрузите демона командой sudo systemctl restart backup\_daemon.service.

Для мониторинга статуса выполнения используйте sudo systemctl status backup\_daemon.service. Детали операций доступны в системном журнале: journalctl -u backup\_daemon.service. Полные настройки и параметры прописаны в файле конфигурации /home/user/daemon/config.txt.

# **Рекомендации программисту**

Для запуска программы необходима 64-битная операционная система Windows, Linux или macOS. Для работы с кодом необходима среда разработки, совместимая с компилятором С++ и библиотеки <iostream>, <ctime>, <cstdlib> и другие.

Исходный код программы и доступен по ссылке:

<https://github.com/CapTopGrade/Linux/tree/main/Chess>

# **Контрольный пример**

В данном разделе представлен контрольный пример, демонстрирующий работу программы.

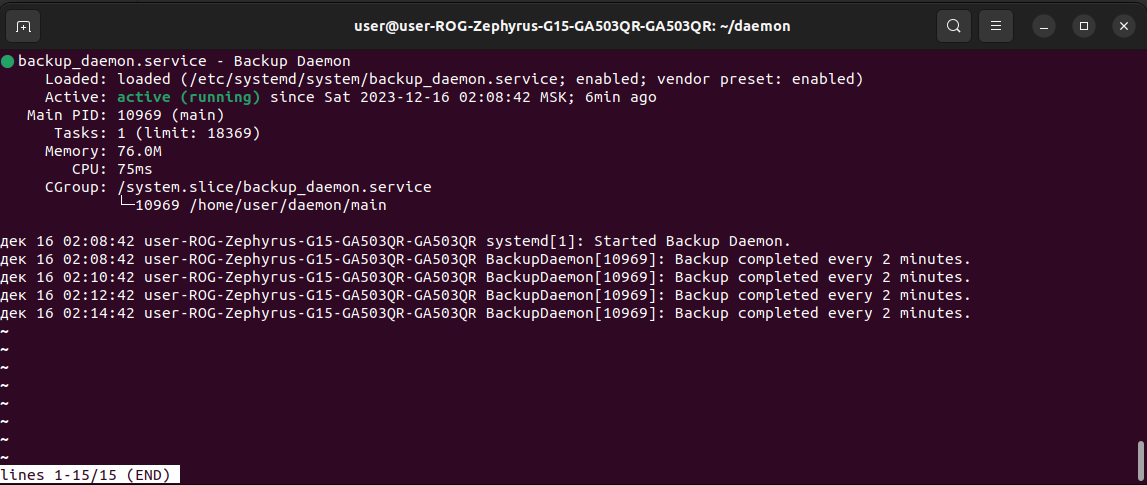


Рис. 9.1 Демонстрация работы службы

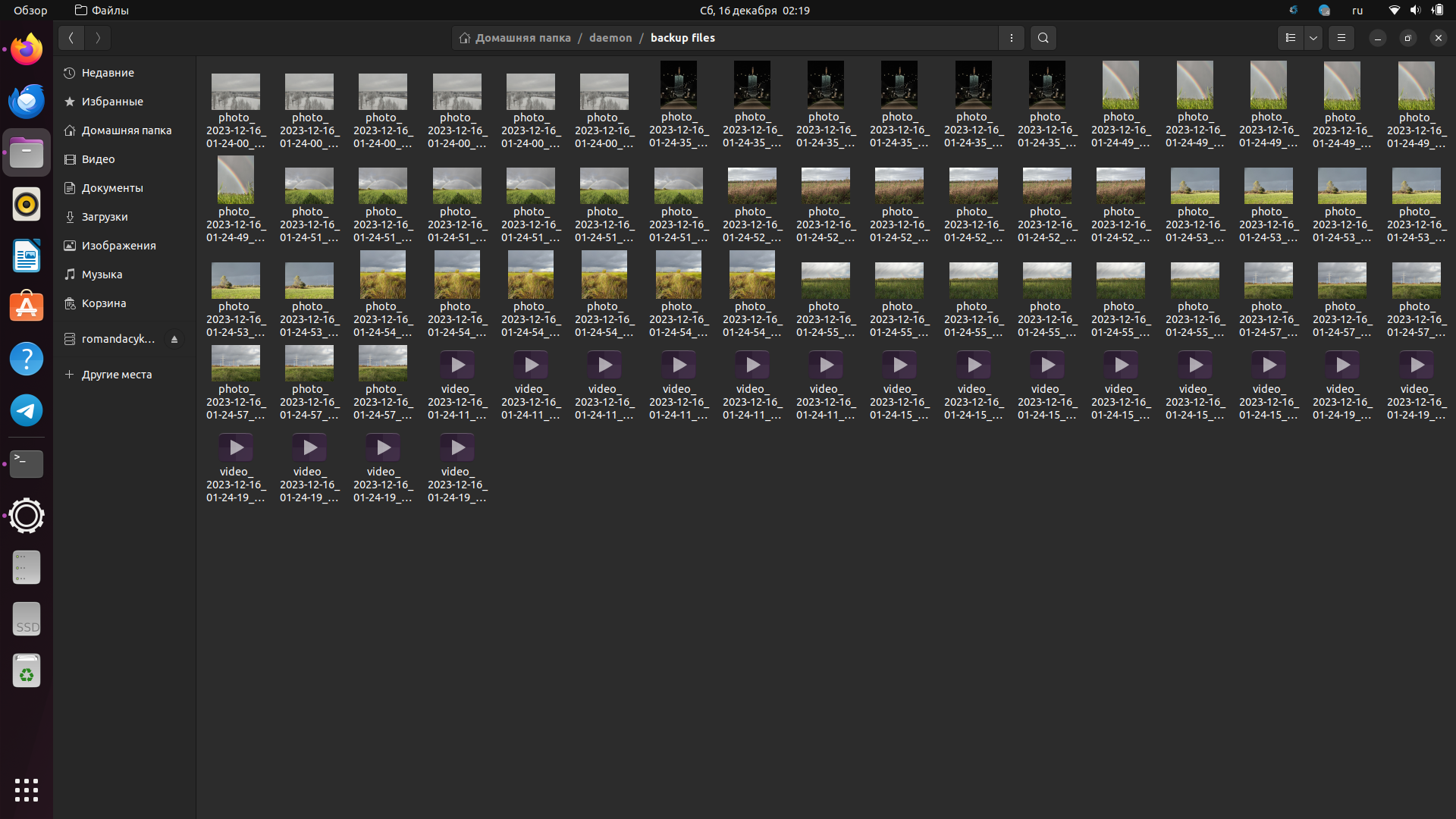


Рис. 9.2 Пример работы программы

# **Вывод**

В данной лабораторной работе был создан демон для системы Linux с целью автоматизированного регулярного резервного копирования данных. Реализовано чтение конфигурации из файла, создание демона на языке C++, интеграция с systemd для запуска при старте и периодического выполнения задач. Также предусмотрено управление демоном через командную строку и мониторинг его статуса. Документация пользователя содержит информацию о настройках, командах управления и процессе резервного копирования. Важным аспектом является обеспечение безопасности демона, включая ограничение доступа к конфигурационным файлам и резервным копиям.

# **Список использованной литературы**

[1] Документация по работе с С++: <https://ru.cppreference.com/w/>

[2] Рекомендации по улучшению кода: <https://github.com/isocpp/CppCoreGuidelines/blob/master/CppCoreGuidelines.md>