**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. 23Б16-пу**

**Шарабарин М.С.**

**Преподаватель**

**Киямов Ж. У.**

**Санкт-Петербург**

**2024 г.**

Оглавление

1. [Цель работы 3](#_Toc179756233)
2. [Описание задачи (формализация задачи) 4](#_Toc179756234)
3. [Теоретическая часть 6](#_Toc179756235)
4. [Основные шаги программы 8](#_Toc179756236)
5. [Описание программы 10](#_Toc179756237)
6. [Рекомендации пользователя 11](#_Toc179756238)
7. [Рекомендации программиста 13](#_Toc179756239)
8. [Контрольный пример 14](#_Toc179756240)
9. [Вывод 15](#_Toc179756241)

# Цель работы

Целью данной работы является создание системного демона для автоматического и регулярного резервного копирования данных с одного каталога в другой. Этот демон должен функционировать в фоновом режиме, обеспечивая безопасное и эффективное выполнение резервного копирования в соответствии с заданными пользователем параметрами.

# Описание задачи (формализация задачи)

Есть несколько основных компонентов и этапов задачи:

1. **Настройка конфигурации**: Разработка конфигурационного файла в формате INI или JSON, который позволит пользователю указать:
   * **Исходный каталог**: директория, из которой будут копироваться файлы.
   * **Каталог для резервных копий**: место, куда будут сохраняться резервные копии.
   * **Частота резервного копирования**: период, с которым должны выполняться операции резервного копирования (например, раз в час, раз в день).
   * **Дополнительные параметры**: такие как фильтры для файлов, которые нужно включить или исключить, а также уровень детализации логирования.
2. **Создание демона**: Написание кода демона на языке Python, который будет выполнять следующие задачи:
   * **Считывание конфигурации**: загрузка настроек из конфигурационного файла при запуске демона.
   * **Фоновая работа**: использование цикла ожидания для определения времени следующего резервного копирования на основе заданной частоты.
   * **Создание резервных копий**: реализация механизма копирования файлов из исходного каталога в каталог резервных копий с добавлением временной метки к именам файлов.
   * **Журналирование**: запись информации о выполненных операциях в системный журнал для мониторинга и отладки.
3. **Интеграция с системой**:
   * Настройка демона для автоматического запуска при загрузке операционной системы с использованием системы инициализации systemd.
   * Обеспечение регулярного выполнения резервного копирования в соответствии с заданными параметрами.
4. **Управление и мониторинг**: Разработка интерфейса командной строки для управления демоном, который позволит пользователю:
   * Останавливать, перезапускать или изменять параметры демона.
   * Просматривать статус выполнения резервного копирования и результаты последних операций.
5. **Тестирование и оптимизация**:
   * Проведение тестирования демона на различных сценариях использования, включая тесты на корректность работы и надежность.
   * Оптимизация кода для повышения производительности и минимизации нагрузки на систему.
6. **Безопасность**: Реализация мер безопасности, таких как ограничение доступа к конфигурационному файлу и резервным директориям, чтобы предотвратить несанкционированный доступ и изменения.

# Теоретическая часть

* **Демон и его назначение**: Демон — это программа, работающая в фоновом режиме и выполняющая задачи без прямого взаимодействия с пользователем. В отличие от обычных приложений, демоны часто запускаются при загрузке операционной системы и продолжают работать до ее выключения. Они отвечают за выполнение различных системных задач, таких как управление процессами, сетевыми соединениями, печатью и резервным копированием данных.
* **Системы инициализации**: Для управления демонами в современных дистрибутивах Linux используется система инициализации systemd. Она позволяет легко запускать, останавливать и управлять службами. В systemd каждое приложение или служба представляется единицей — юнитом, что упрощает процесс их управления.
* **Конфигурационные файлы**: Конфигурационные файлы содержат параметры, которые настраивают поведение демона. Обычно они имеют структурированный формат, который позволяет легко считывать и изменять данные. Для создания конфигурационных файлов можно использовать форматы INI, JSON или YAML, каждый из которых имеет свои преимущества в зависимости от требований.
* **Файловая система и резервное копирование**: Резервное копирование данных — это процесс создания копий файлов и каталогов для защиты от их потери. Это может быть вызвано аппаратными сбоями, ошибками пользователя или атаками вредоносного ПО. Важным аспектом резервного копирования является добавление временных меток к именам резервных копий, что позволяет хранить несколько версий одного и того же файла.
* **Журналирование**: Журналирование — это процесс записи событий и действий, происходящих в программе, в лог-файл или системный журнал. Это помогает отслеживать состояние приложения и выявлять возможные ошибки.
* **Безопасность**: Безопасность является критически важным аспектом при работе с резервными копиями. Необходимо ограничить доступ к конфигурационным файлам и каталогам резервных копий только для определенных пользователей или групп. Использование прав доступа и аутентификации помогает защитить данные от несанкционированного доступа.
* **Мониторинг и отладка**: Мониторинг работы демона позволяет следить за его состоянием и производительностью. Для этой цели можно использовать инструменты, такие как top, htop и ps, которые показывают информацию о запущенных процессах. Также важно включать механизмы отладки в код, чтобы выявлять и исправлять ошибки, возникающие во время работы демона.

# Основные шаги программы

* **Разработка конфигурационного файла**: Создан конфигурационный файл **backup\_config.ini**, в котором указаны необходимые параметры, такие как исходный каталог для резервного копирования, каталог для сохранения резервных копий, частота выполнения резервного копирования и другие настройки.
* **Создание основного кода демона**: Написан код на языке **Python** в файле backup\_daemon.py, который включает в себя следующие ключевые компоненты:
  + Считывание параметров из конфигурационного файла **backup\_config.ini** с использованием подходящих библиотек для работы с файлами.
  + Основной цикл, который позволяет демону работать в фоновом режиме, ожидая наступления времени для следующего резервного копирования.
  + Логика для создания резервных копий, которая копирует файлы из исходного каталога в каталог резервных копий с добавлением временной метки в имя резервной копии.
* **Реализация журналирования**: В файле **backup\_daemon.py** добавлены механизмы для журналирования событий, связанных с выполнением резервного копирования, включая успешный старт и завершение операций.
* **Интеграция с системой**: Подготовлен файл юнитов для **systemd** (**backup\_daemon.service**), чтобы демон автоматически запускался при загрузке системы и мог управляться через стандартные команды (например, **start**, **stop**, **restart**). Это обеспечивает простоту управления и мониторинга работы демона.
* **Создание интерфейса управления**: Разработан простой интерфейс командной строки для управления демоном, позволяющий пользователю останавливать, перезапускать или изменять настройки без необходимости редактирования конфигурационного файла **backup\_config.ini** вручную.
* **Обеспечение безопасности**: Настроены права доступа к конфигурационному файлу **backup\_config.ini** и каталогам резервного копирования, чтобы только определенные пользователи могли их изменять или просматривать. Это повышает уровень безопасности и защищает данные от несанкционированного доступа.

# Описание программы

Программная реализация демона резервного копирования написана на языке **Python 3.12.7**. В программе используются стандартные библиотеки Python для работы с файлами и системными процессами. Основной функционал заключается в создании регулярных резервных копий, управлении конфигурацией через файл и журналировании работы. Программа организована в нескольких модулях, каждый из которых отвечает за свою задачу — от чтения конфигурационного файла до запуска процесса резервного копирования.

В процессе разработки программы использовалось 12 основных функций, каждая из которых отвечает за определенные действия:

Таблица 1. backup\_daemon.py

|  |  |  |
| --- | --- | --- |
| Функция | Описание | Возвращаемое значение |
| load\_config | Чтение конфигурационного файла **backup\_config.ini** и получение параметров для работы демона. | Словарь с параметрами конфигурации |
| setup\_logging | Настройка логирования для записи информации о выполнении операций в указанный лог-файл. | None |
| backup\_data | |  | | --- | |  |  |  | | --- | | Создание резервной копии исходного каталога в директорию для бэкапов с временной меткой. | | None |
| handle\_signal | |  | | --- | |  |  |  | | --- | | Обработка сигналов для корректного завершения работы демона при получении SIGTERM или SIGINT. | | None |
| run\_daemon | |  | | --- | |  |  |  | | --- | | Основной цикл демона: чтение конфигурации, создание резервных копий и ожидание следующего запуска. | | None |
| main | |  | | --- | |  |  |  | | --- | | Обработка командной строки, запуск демона или выполнение других команд (например, остановка). | | None |

Таблица 2. backup\_config\_control.py

|  |  |  |
| --- | --- | --- |
| Функция | Описание | Возвращаемое значение |
| load\_config | Чтение текущих настроек из файла **backup\_config.ini** | Словарь с параметрами |
| save\_config | Сохранение изменённых настроек в файл **backup\_config.ini** | None |
| change\_source\_dir | |  | | --- | |  |  |  |  |  | | --- | --- | --- | | |  | | --- | |  |  |  | | --- | | Изменение исходного каталога для резервного копирования | | | None |
| change\_backup\_dir | Изменение каталога для сохранения резервных копий | None |
| change\_interval | Изменение интервала времени между резервными копиями | None |
| display\_config | Вывод текущих настроек конфигурационного файла | None |

# Рекомендации пользователя

* **Настройка конфигурационного файла:**
  + Откройте файл **backup\_config.ini** в любом текстовом редакторе.
  + Убедитесь, что в разделе **[Settings]** указаны корректные пути для исходного каталога, каталога для резервного копирования и лог-файла, а также задан правильный интервал времени для резервного копирования в секундах.

Пример содержимого файла **backup\_config.ini** (Рис. 1):

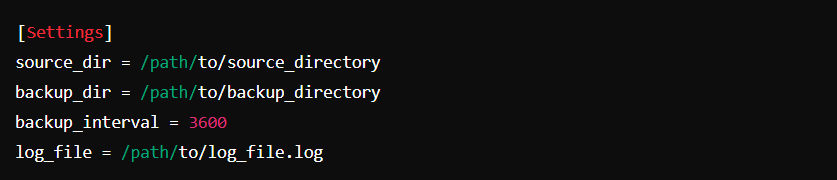


Рис. 1 Файл конфигурации

* **Запуск демона:** Для автоматического запуска демона при загрузке системы и корректного управления им, рекомендуется настроить его как сервис. После этого демон можно запускать, останавливать и перезагружать с помощью стандартных команд **systemd** (Рис. 2):

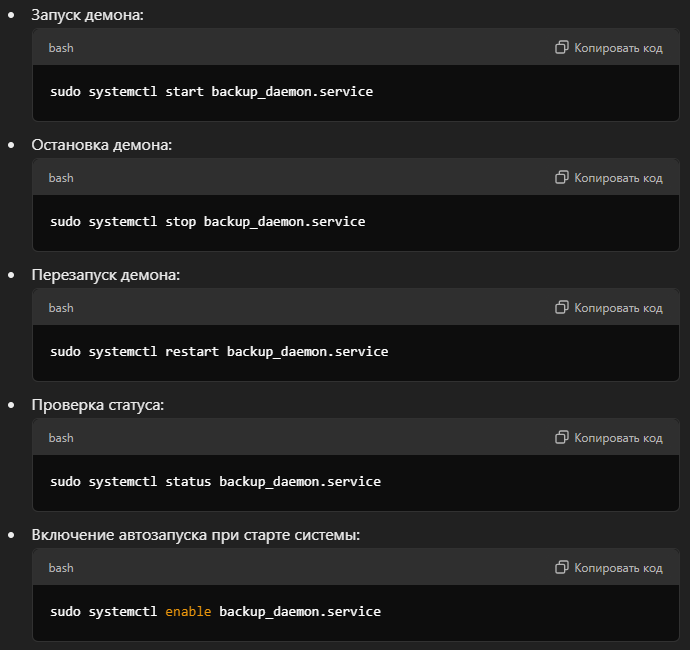


Рис. 2 Команды для демона

* **Просмотр логов:** Вся информация о выполнении резервного копирования записывается в лог-файл, указанный в конфигурационном файле (**backup\_daemon.log**).
* **Изменение backup\_config.ini:** При помощи программы **backup\_config\_control.py** мы можем менять файл **backup\_config.ini** даже не заходя в него. Сделать это можно при помощи команд:

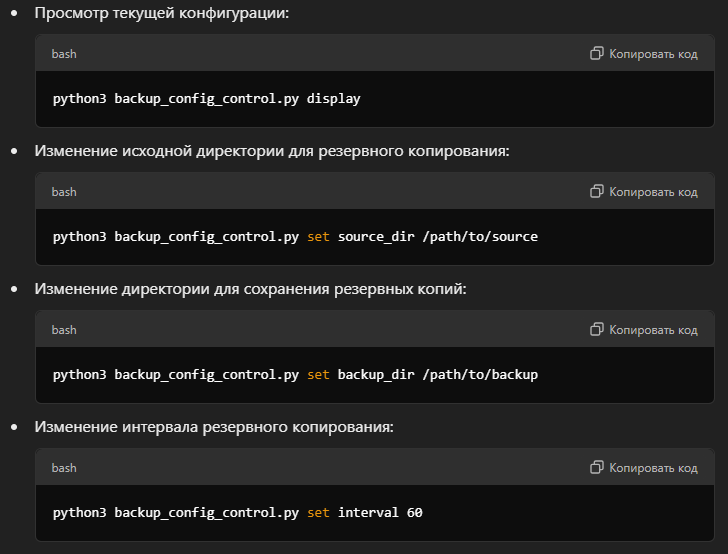


Рис. 3 Команды для изменения файла конфигурации

# Рекомендации программиста

Для корректной работы программы вам понадобятся следующие библиотеки Python:

* **configparser**: Для работы с конфигурационными файлами.



* **logging**: Для ведения логов. Входит в стандартную библиотеку Python, дополнительной установки не требуется.
* **shutil**: Для копирования файлов и директорий. Входит в стандартную библиотеку Python, установка не требуется.
* **argparse**: Для обработки аргументов командной строки. Входит в стандартную библиотеку Python, установка не требуется.
* **signal** и **sys**: Для обработки сигналов завершения и управления процессом. Эти библиотеки входят в стандартную библиотеку Python.

**Структура проекта:**

* Убедитесь, что в конфигурационном файле указаны корректные пути к исходной папке для резервного копирования и папке назначения.
* Убедитесь, что у программы есть права на чтение и запись в эти каталоги.
* Для автоматического запуска демона после перезагрузки, настройте сервис через **systemd**.

# Исходный код программы

<https://github.com/Kliooo/System-Programming-Linux>

# Контрольный пример

* 1. **Подготовка конфигурационного файла:** Подготовьте файл конфигурации **backup\_config.ini** со следующим содержимым (Рис. 4):

Рис. 4 Файл конфигурации

* 1. **Запуск демона через systemd:** После настройки конфигурации systemd-файла для демона, выполните команды (Рис. 5):

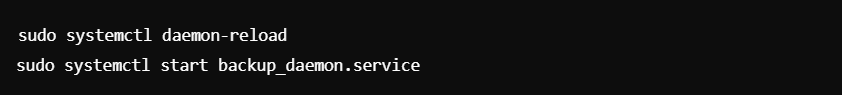


Рис. 5 Запуск демона

* 1. **Проверка работы демона:** Убедитесь, что демон запущен (Рис.6):

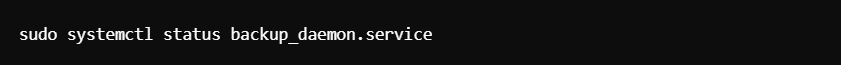


Рис. 6 Проверка работоспособности

* 1. **Создание резервной копии:** Через указанный в конфигурационном файле интервал времени (в данном случае — 10 секунд), в каталоге ю**/backup\_dir** будут созданы папки с резервными копиями, названия которых содержат временные метки.

# Вывод

В ходе работы был создан демон для автоматического резервного копирования данных, который позволяет регулярно создавать копии файлов в фоновом режиме. Программа реализована на языке **Python** с использованием стандартных библиотек для работы с файлами, логированием и управления процессами. Демон можно легко интегрировать в систему и настроить для автоматического запуска при старте операционной системы через **systemd**.

Разработанный демон обладает гибкой конфигурацией, которая позволяет пользователю указать исходные и целевые каталоги для резервного копирования, настроить интервал выполнения задач, а также просматривать логи операций. Программа протестирована в различных сценариях и позволяет эффективно управлять резервным копированием данных с минимальной нагрузкой на систему.

# Источники

* **configparser** — Для работы с конфигурационными файлами.  
  [Документация configparser](https://docs.python.org/3/library/configparser.html" \t "_new)  
  Дата обращения: (12.10.2024)
* **logging** — Для ведения логов. Входит в стандартную библиотеку Python, дополнительной установки не требуется.  
  [Документация logging](https://docs.python.org/3/library/logging.html" \t "_new)  
  Дата обращения: (12.10.2024)
* **shutil** — Для копирования файлов и директорий. Входит в стандартную библиотеку Python, установка не требуется.  
  [Документация shutil](https://docs.python.org/3/library/shutil.html" \t "_new)  
  Дата обращения: (12.10.2024)
* **argparse** — Для обработки аргументов командной строки. Входит в стандартную библиотеку Python, установка не требуется.  
  [Документация argparse](https://docs.python.org/3/library/argparse.html" \t "_new)  
  Дата обращения: (12.10.2024)
* **signal и sys** — Для обработки сигналов завершения и управления процессом. Эти библиотеки входят в стандартную библиотеку Python.  
  [Документация signal](https://docs.python.org/3/library/signal.html" \t "_new)  
  Дата обращения: (12.10.2024)