**МИНОБРНАУКИ РОССИИ**

**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. 24Б16-пу |  | Бублик И.О. |
| Преподаватель |  | Киямов Ж. У. |

**Санкт-Петербург**

**2025 г**

Оглавление

[Цель работы 3](#_Toc185234978)

[Формализация задачи 3](#_Toc185234979)

[Основные шаги программы 5](#_Toc185234980)

[Описание программы 6](#_Toc185234981)

***Блок схема ………………………………………………………………………..9***

[Контрольный пример 11](#_Toc185234982)

[Рекомендации пользователя 12](#_Toc185234983)

[Рекомендации для программиста 13](#_Toc185234984)

[Вывод 15](#_Toc185234985)

Цель работы

Целью данной работы является разработка демона — системного процесса, работающего в фоновом режиме, для автоматического регулярного резервного копирования данных. Основная задача демона заключается в автоматизации процесса создания резервных копий, что позволяет минимизировать риски потери данных в результате сбоев системы, ошибок пользователей или внешних факторов. Демон должен быть настроен на регулярное копирование данных с одного каталога в другой в соответствии с параметрами, указанными в конфигурационном файле.

Формализация задачи

Задача заключается в создании демона резервного копирования с реализацией следующих функциональных требований:

1. **Чтение конфигурации**: Демон должен загружать параметры из конфигурационного файла, где указываются исходный каталог, каталог для резервных копий, частота копирования и путь для ведения журнала.
2. **Фоновый режим работы**: Программа должна работать в фоновом режиме, периодически выполняя резервное копирование на основе заданного интервала времени.
3. **Создание резервных копий**: Каждый цикл резервного копирования должен включать в себя копирование файлов из исходного каталога в каталог для резервных копий с добавлением временной метки.
4. **Журналирование**: Все операции резервного копирования должны быть зафиксированы в системном журнале с указанием успешности выполнения или возникающих ошибок.
5. **Интеграция с операционной системой**: Демон должен запускаться при старте операционной системы и работать согласно заданным настройкам.
6. **Управление демоном**: Необходимо предусмотреть механизм управления процессом через командную строку, позволяющий останавливать, перезапускать и изменять настройки демона.
7. **Безопасность**: Демон должен быть настроен таким образом, чтобы обеспечить безопасное хранение конфигурационных файлов и резервных копий с ограничением доступа к ним.

Теоретическая часть

Демон — это системный процесс, который выполняется в фоновом режиме без прямого взаимодействия с пользователем. В операционных системах Linux демоны часто используются для выполнения длительных задач, таких как мониторинг состояния системы, обслуживание серверов и автоматизация различных процессов. Демон запускается при старте операционной системы и продолжает работать до её выключения или перезапуска. В рамках данной работы демон используется для автоматизации процесса резервного копирования файлов с одного каталога в другой, что позволяет гарантировать сохранность данных.

Основные концепции, применяемые при разработке демона резервного копирования:

1. **Резервное копирование** — это процесс создания копий данных, которые можно использовать для восстановления в случае потери или повреждения оригинальных данных. В программном обеспечении для резервного копирования часто применяются техники создания временных меток, инкрементального копирования и ведения журнала для отслеживания успешности выполнения операций.
2. **Фоновая работа** демона подразумевает его непрерывное выполнение в рамках операционной системы без активного вмешательства пользователя. Демон должен эффективно управлять системными ресурсами, чтобы не создавать чрезмерную нагрузку на систему.
3. **Конфигурационные файлы** позволяют гибко настраивать работу демона, задавая исходный каталог для копирования, директорию для хранения резервных копий, частоту выполнения копирования и другие параметры.
4. **Журналирование** — это процесс записи событий и ошибок, происходящих во время работы демона. Ведение журнала важно для отслеживания проблем, которые могут возникнуть в процессе резервного копирования, и для обеспечения прозрачности работы системы.
5. **Управление процессами** — ключевая часть создания демона, которая включает возможность его остановки, перезапуска и изменения конфигурации без необходимости завершения работы системы или выполнения резервных копий вручную. Это достигается через сигналы или встроенные механизмы управления в коде

Основные шаги программы

1. Загрузка конфигурации. При запуске демона первым шагом является считывание параметров из конфигурационного файла. В конфигурации указываются исходный каталог для данных, директория для хранения резервных копий, частота выполнения резервного копирования (в секундах) и путь к файлу журнала.
2. Настройка журналирования. После загрузки конфигурации демон настраивает журналирование с помощью модуля logging. Все операции (успешные или неуспешные) будут записываться в файл, указанный в конфигурационном файле.
3. Резервное копирование. На каждом цикле работы демона выполняется функция резервного копирования, которая копирует все файлы и папки из исходного каталога в директорию для резервных копий. Каждая резервная копия получает временную метку, чтобы избежать перезаписи предыдущих копий.
4. Фоновое выполнение. Демон работает в фоновом режиме, ожидая наступления следующего времени для резервного копирования. Для этого он использует цикл с задержкой, основанной на указанном интервале времени.
5. Остановка и управление демоном. Программа поддерживает возможность остановки демона. Управление процессом возможно через скрипты, которые позволяют запускать, останавливать демона и изменять его конфигурацию.
6. Мониторинг и безопасность. С помощью механизма журналирования пользователь может отслеживать успешность операций резервного копирования. Для обеспечения безопасности конфигурационный файл и директория с резервными копиями должны быть защищены от несанкционированного доступа, что может быть дополнительно настроено через права доступа на уровне файловой системы Linux.

Описание программы

Программа написана на языке Python и реализует демон для автоматического резервного копирования данных с одного каталога в другой. Программа включает в себя работу с конфигурационными файлами, журналирование, обработку сигналов и создание графического интерфейса для управления процессом. Важной особенностью является возможность управления демоном с помощью скриптов.

**Использованные библиотеки**

**os**: Библиотека для работы с файловой системой и выполнения операций над каталогами и файлами.

**time**: Используется для организации задержек между циклами резервного копирования, а также для работы с временными метками.

**shutil**: Предоставляет инструменты для копирования файлов и каталогов. В частности, используется функция copytree для создания резервных копий целого каталога.

**logging**: Библиотека для ведения журнала событий. Используется для записи информации о процессе резервного копирования, включая успешные и неуспешные операции.

**datetime**: Используется для генерации временных меток, которые добавляются к именам резервных копий.

**Основные функции программы**

|  |  |  |
| --- | --- | --- |
| **Название функции** | **Входные переменные** | **описание** |
| **load\_config** | **-** | Загрузка конфигурационного файла. Функция открывает файл по указанному пути и возвращает данные конфигурации в виде словаря |
| **setup\_logging** | **log\_path** | Настройка журналирования для записи операций в лог-файл. Используется logging.basicConfig, чтобы определить формат сообщений и путь для сохранения журнала. |
| **create\_backup** | **source, distination** | Создание резервной копии файлов из исходного каталога в каталог для резервных копий. К каждой копии добавляется временная метка для избежания перезаписи предыдущих данных. В случае успешного копирования в лог записывается сообщение об успехе, в противном случае фиксируется ошибка. |
| **main** |  | Основная функция демона, запускающаяся в фоновом режиме. Она выполняет загрузку конфигурации, настройку журналирования и периодическое копирование файлов с заданным интервалом времени, который указывается в конфигурации. |

**Основные переменные**

|  |  |
| --- | --- |
| **Имя переменной** | **функция** |
| **CONFIG\_PATH** | Путь к конфигурационному файлу, который используется для хранения настроек программы. |
| **log\_file** | Путь к файлу журнала, в который записываются все события резервного копирования |

Блок Схема

Изображение выглядит как текст, снимок экрана, Шрифт, белый

Содержимое, созданное искусственным интеллектом, может быть неверным.

Контрольный пример

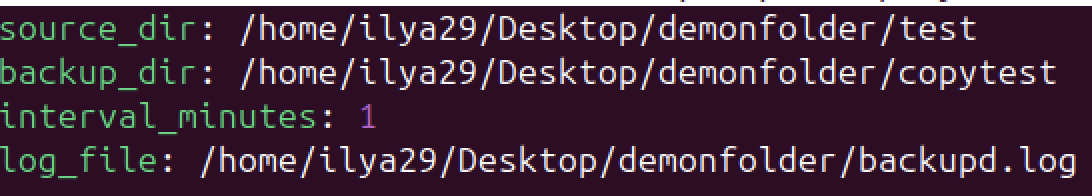


Рисунок 1. Конфигурация в формате yaml

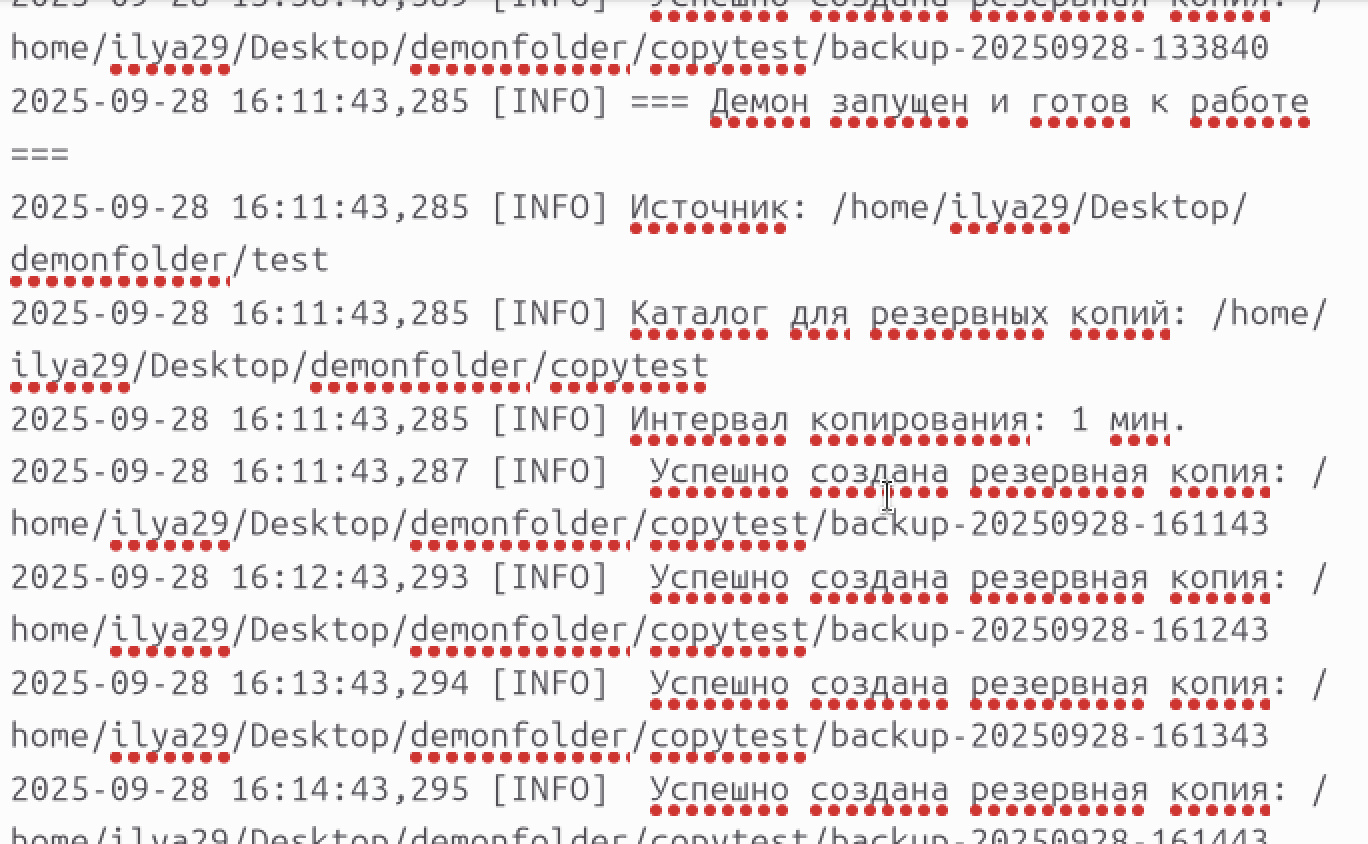


Рисунок 2. log файл

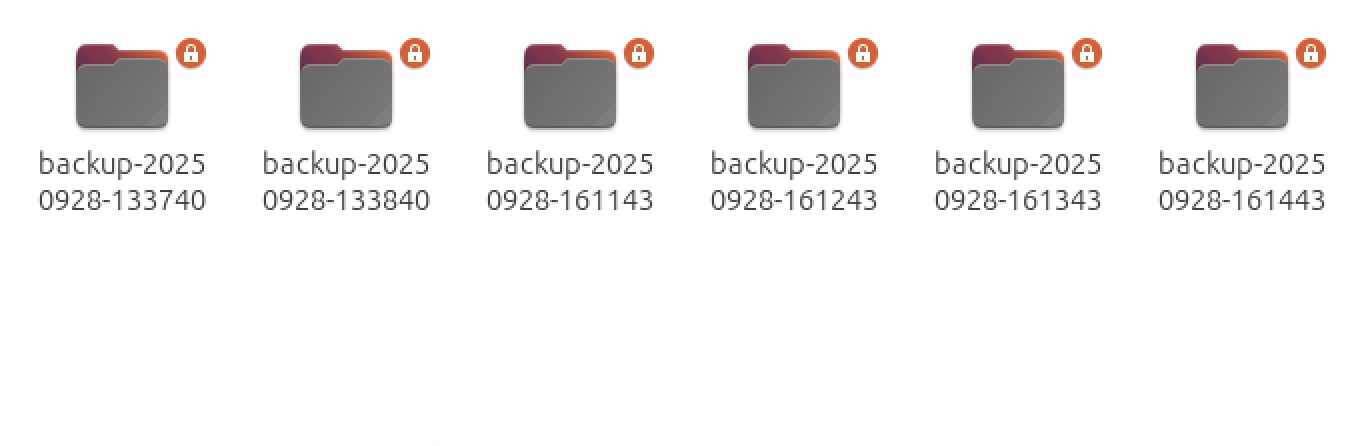


Рисунок 3. Папки с резервными копиями

Рекомендации пользователя

1. **Запуск и остановка демона:** С помощью скриптов пользователь может легко через терминал запускать и останавливать демона
2. **Проверка логов**: Пользователь должен регулярно проверять логи работы демона, чтобы убедиться, что резервные копии создаются корректно. В случае возникновения ошибок в процессе копирования это будет отражено в лог-файле. Лог-файл можно найти по пути, указанному в конфигурационном файле.
3. **Остановка и перезапуск демона**: Если необходимо изменить настройки (например, изменить интервалы или каталоги), рекомендуется сначала остановить демон, внести изменения через интерфейс или конфигурационный файл, а затем снова запустить демон.
4. **Мониторинг доступного пространства**: Пользователю стоит контролировать объем свободного места в каталоге для резервных копий, чтобы предотвратить заполнение диска, которое может привести к сбоям в работе системы.

Рекомендации для программиста

1. **Оптимизация производительности**: Если объём данных велик или частота резервного копирования высока, стоит рассмотреть возможность инкрементного копирования, а не полного. Это уменьшит нагрузку на систему и сэкономит место на диске.
2. **Настройка каталогов**: Перед запуском демона убедитесь, что указаны правильные пути к исходному каталогу и директории для резервного копирования.
3. **Обработка ошибок и исключений**: Программа уже ведет логи ошибок, но можно расширить функциональность, добавив более детализированную обработку исключений. Например, обработку ситуаций, связанных с недостаточным количеством места на диске илиотсутствием доступа к каталогу.
4. **Ограничение доступа**: Для повышения безопасности важно установить правильные права доступа к каталогу с резервными копиями и конфигурационному файлу. Это предотвратит несанкционированный доступ к важным данным и изменениям настроек демона.
5. **Модульность**: Для улучшения поддержки и масштабирования кода рекомендуется разбить программу на отдельные модули, такие как управление конфигурацией, процесс копирования, работа с логами и управление интерфейсом. Это позволит проще расширять функциональность и поддерживать код.
6. **Реализация безопасности**: Важно внедрить дополнительные меры безопасности, такие как шифрование резервных копий или использование проверок целостности данных, чтобы защитить файлы от повреждений или несанкционированного доступа.

# **Вывод**

В рамках данной работы был создан демон для регулярного резервного копирования данных на языке Python. Программа позволяет автоматически создавать резервные копии заданного каталога через определенные интервалы времени, используя конфигурационный файл для задания параметров. Благодаря использованию системных возможностей, таких как фоновый режим и журналирование, демон эффективно управляет процессом резервного копирования без необходимости постоянного вмешательства пользователя.

Дополнительно были учтены ключевые аспекты безопасности, такие как ограничение доступа к конфигурационным файлам и каталогам с резервными копиями.

Программа успешно протестирована в различных сценариях, демонстрируя устойчивую работу в фоновых процессах. Для дальнейшего развития проекта можно рассмотреть возможности расширения функциональности, такие как инкрементное копирование, улучшение обработки ошибок и внедрение асинхронности для повышения производительности.

Создание данного демона позволяет автоматизировать процесс резервного копирования, что важно для обеспечения безопасности и сохранности данных, особенно в условиях динамично изменяющихся файловых систем.

Исходный код:

https://github.com/Ilya2914/system\_and\_functionalprogramy