

**САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**Факультет прикладной математики-процессов управления**

**Программа бакалавриата**

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

**ОТЧЕТ**

**по лабораторной работе №1**

**по дисциплине «Программирование в Linux»**

**на тему**

**«Разработка сервиса Demon»**

**Студент гр. 23Б15-пу  
Трофимов И.А.**

**Преподаватель  
Киямов Ж. У.**

**Санкт-Петербург  
2024 г.**

## Оглавление

1. Цель работы	3
2. Описание задачи	3
3. Теоретическая часть	4
4. Основные шаги программы	5
5. Описание программы	5
6. Рекомендации пользователя	7
7. Рекомендации программиста	7
8. Исходный код программы	7
9. Контрольный пример	8
10. Вывод	8
11. Источники	9

## **Цель работы**

Разработать системный демон для автоматического регулярного резервного копирования данных. Демон должен выполнять задачи по считыванию конфигурации, созданию резервных копий, журналированию операций и обеспечению безопасности. Работа включает интеграцию с системой, управление и мониторинг процесса, а также тестирование и оптимизацию.

## **Описание задачи**

В рамках лабораторной работы необходимо разработать системный демон, который будет выполнять автоматическое регулярное резервное копирование данных. Демон должен работать в фоновом режиме, обеспечивая надёжное копирование файлов и минимальную нагрузку на систему.

Основные этапы реализации:

1. **Настройка конфигурации:** Создать конфигурационный файл для указания исходного каталога, каталога резервного копирования, частоты выполнения задач и других параметров.
2. **Реализация демона:** Написать программу, которая:
  - Считывает настройки из конфигурационного файла.
  - Ожидает наступления времени для очередного резервного копирования.
  - Копирует файлы с временной меткой в указанный каталог.
  - Журналирует выполнение операций.
3. **Интеграция с системой:** Настроить запуск демона при старте операционной системы.

## **Теоретическая часть**

Демон — это системный процесс, работающий в фоновом режиме без прямого взаимодействия с пользователем. Демоны используются для выполнения задач, требующих периодического или постоянного контроля, таких как резервное копирование, мониторинг состояния системы или обслуживание сетевых соединений.

### **1. Архитектура демона**

Демон запускается как самостоятельный процесс и работает в фоне. Он обычно отсоединяется от управляющего терминала, создаёт собственный сессионный идентификатор и взаимодействует с системой через механизмы сигналов и логирования.

### **2. Конфигурационные файлы:**

Конфигурационные файлы необходимы для хранения параметров работы демона, таких как пути к исходным и целевым каталогам, частота выполнения задач, и другие настройки. Используются форматы, удобные для чтения и редактирования, например TOML, INI или JSON.

### **3. Журналирование**

Демоны часто записывают свои действия в системный журнал (например, через syslog в Linux). Журналирование помогает отслеживать успешное выполнение задач и обнаруживать ошибки.

### **4. Алгоритмы резервного копирования:**

Резервное копирование предполагает копирование файлов из исходного каталога в каталог для резервных копий. Временная метка добавляется для

различия между копиями. Для оптимизации могут использоваться методы инкрементального копирования (копирование только изменённых данных).

## **Основные шаги программы**

1. **Создание сервиса:**
  - Скрипт `mydemon.service`.
  - Загрузка сервиса.
2. **Запуск сервиса:**
3. **Включение сервиса**
4. **Просмотр статуса**
5. **Закрытие сервера:**

## **Описание программы**

Программа представляет собой демон, который автоматически выполняет регулярное резервное копирование данных с одного каталога в другой. Конфигурация, включая исходный каталог, каталог для резервных копий, частоту копирования и путь к лог-файлу, загружается из JSON-файла. В программе используется функция для создания резервных копий с временной меткой, что позволяет легко различать каждый набор данных.

Основной цикл программы выполняет резервное копирование с заданной периодичностью, записывая информацию о процессе в лог-файл. Программа использует модуль ``logging`` для журналирования и ``shutil.copytree()`` для копирования данных. После выполнения копирования она приостанавливается на заданное время и затем повторяет процесс.

Программа работает в фоновом режиме и продолжает выполнять резервное копирование до ее остановки.

Таблица 1. main.py

Функция	Описание	Результат
load_config	Загружает конфигурацию	None
backup_files	Создает backup	None
run_daemon	Подгружает конфиг и запускает систему в бесконечный цикл	None

## **Рекомендации пользователя**

**Конфигурация:** Перед использованием программы убедитесь, что правильно настроили конфигурационный файл `config.json`. Укажите корректные пути к исходному каталогу и каталогу для резервных копий, а также задайте частоту копирования в секундах. Также не забудьте указать путь к файлу журнала, в который будет записываться информация о выполненных операциях.

**Права доступа:** Убедитесь, что у программы есть необходимые права для чтения данных в исходном каталоге и записи в каталог для резервных копий. Также настройте права на файл журнала, чтобы программа могла записывать ошибки и успешные операции.

## **Рекомендации программиста**

**Обработка ошибок:** При необходимости доработать программу, уделите внимание обработке ошибок, чтобы она могла корректно реагировать на непредвиденные ситуации (например, проблемы с доступом к каталогам или нехватку свободного места).

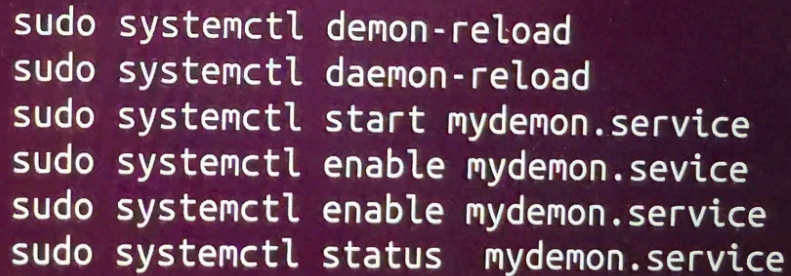
**Поддержка дополнительных форматов резервного копирования:** Можно добавить поддержку различных методов резервного копирования, например, создание архивов или использование системных инструментов резервного копирования для улучшения эффективности работы программы.

**Исходный код программы:**

<https://github.com/hanglider/Jasur-Labs/tree/main/linux>

## Контрольный пример

Впишите команды (рис. 1)



```
sudo systemctl demon-reload
sudo systemctl daemon-reload
sudo systemctl start mydemon.service
sudo systemctl enable mydemon.sevice
sudo systemctl enable mydemon.service
sudo systemctl status mydemon.service
```

Рис.1 Рабочая область

## Вывод

В ходе выполнения лабораторной работы был разработан демон для регулярного резервного копирования данных, который работает в фоновом режиме, автоматически выполняет копирование файлов и логирует свои действия. В ходе реализации были изучены ключевые аспекты системного программирования, такие как работа с конфигурационными файлами, интеграция с операционной системой, механизмы управления и мониторинга процессов, а также обеспечение безопасности.

Демон эффективно выполняет задачи резервного копирования, поддерживает возможность настройки через конфигурационный файл и может быть управляем через командную строку. Тестирование показало стабильную работу программы, и в процессе работы были реализованы необходимые механизмы защиты данных.

В целом, разработанный демон является надёжным инструментом для автоматизации процесса резервного копирования и может быть использован для различных задач, требующих регулярного и безопасного сохранения данных.



## **Источники**

Нет