Отчёта по лабораторной работе №6

Мандатное разграничение прав в Linux (SELinux) с использованием веб-сервера Apache

Акондзо Жордани Лади Гаэл

Содержание

# Цель работы

Цель данной лабораторной работы — развить навыки администрирования в операционной системе Linux с акцентом на использование технологии SELinux. Основное внимание уделено настройке SELinux для работы с веб-сервером Apache, что позволяет на практике проверить ограничения прав доступа.

# Теоретическое введение

## Подготовка стенда

Для выполнения лабораторной работы использовался дистрибутив Linux с включённой политикой SELinux targeted и режимом enforcing. В качестве веб-сервера использовался Apache, который был настроен для работы на портах 80 и 81. Важно было убедиться, что iptables настроен корректно и не блокирует доступ к данным портам.

# Выполнение лабораторной работы

1. Проверка статуса SELinux

* В первую очередь, я проверил, что SELinux работает в режиме enforcing с использованием следующей команды: (рис. 1)



Рис. 1: Проверка статуса SELinux

* Результат показал, что система настроена корректно для выполнения лабораторной работы.

1. Настройка веб-сервера Apache

* После этого я убедился, что сервер Apache работает, используя команду: (рис. 2)

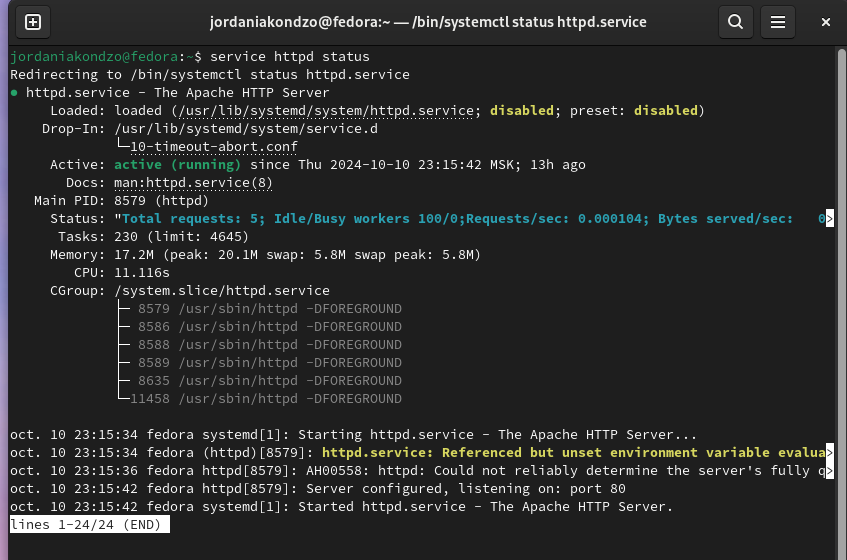


Рис. 2: Настройка веб-сервера ApacheПроверка статуса SELinux

1. Проверка контекста SELinux для Apache

* Я проверил контексты безопасности для процессов Apache, используя команду: (рис. 3)

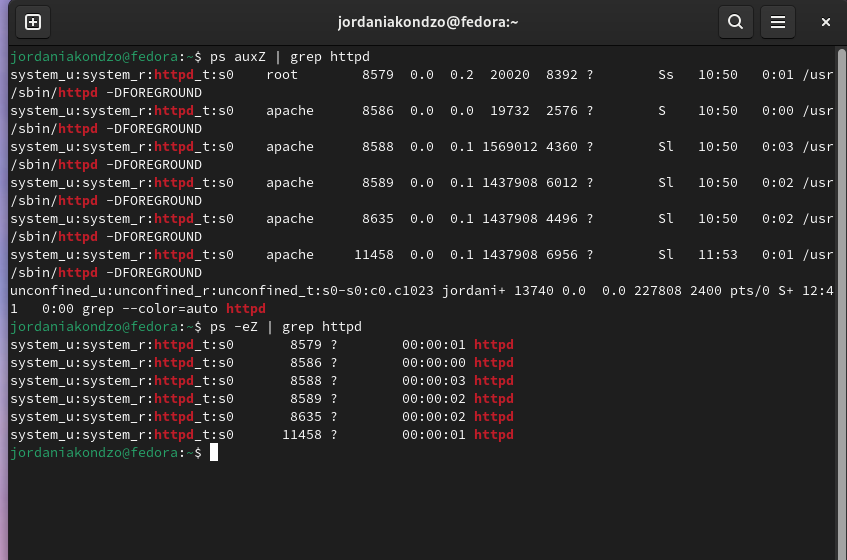


Рис. 3: Проверка контекста SELinux для Apache

* Эта команда показала, что процессы Apache работают с контекстом httpd\_t, что позволяет серверу правильно обрабатывать веб-запросы.

1. Проверка текущих настроек SELinux для Apache

* Далее, я использовал команду sestatus для проверки настроек SELinux, связанных с веб-сервером Apache: (рис. 4)

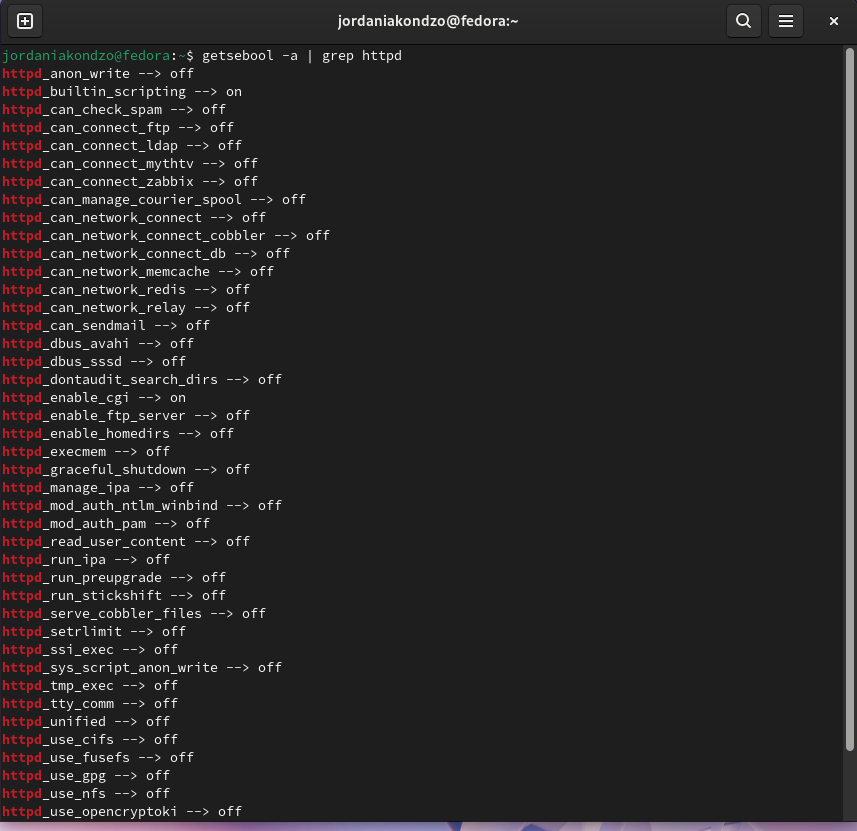


Рис. 4: Проверка текущих настроек SELinux для Apache

* Это позволило мне увидеть, какие функции включены для Apache.

1. Анализ политик и типов SELinux

* Я также изучил статистику политик и список пользователей, ролей и типов, используя команду: (рис. 5) и (рис. 6)

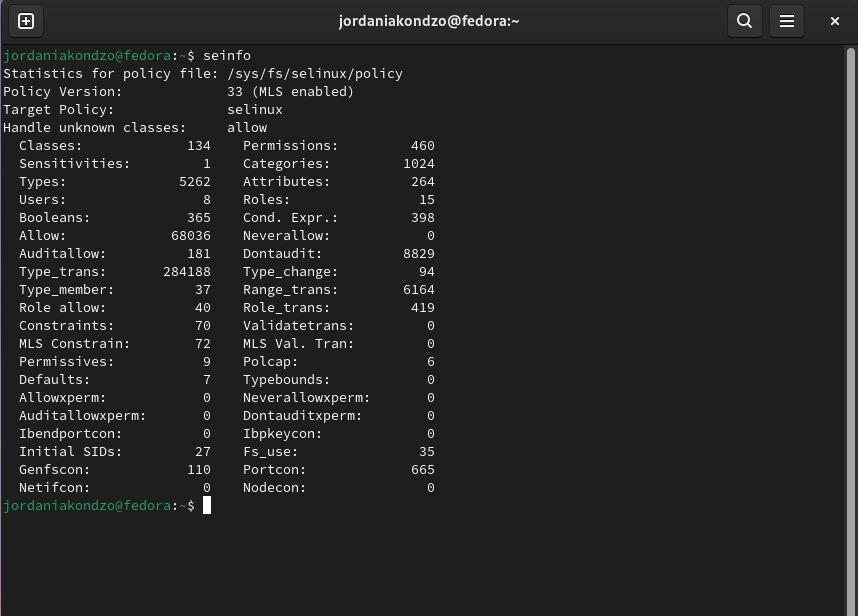


Рис. 5: Анализ политик и типов SELinux

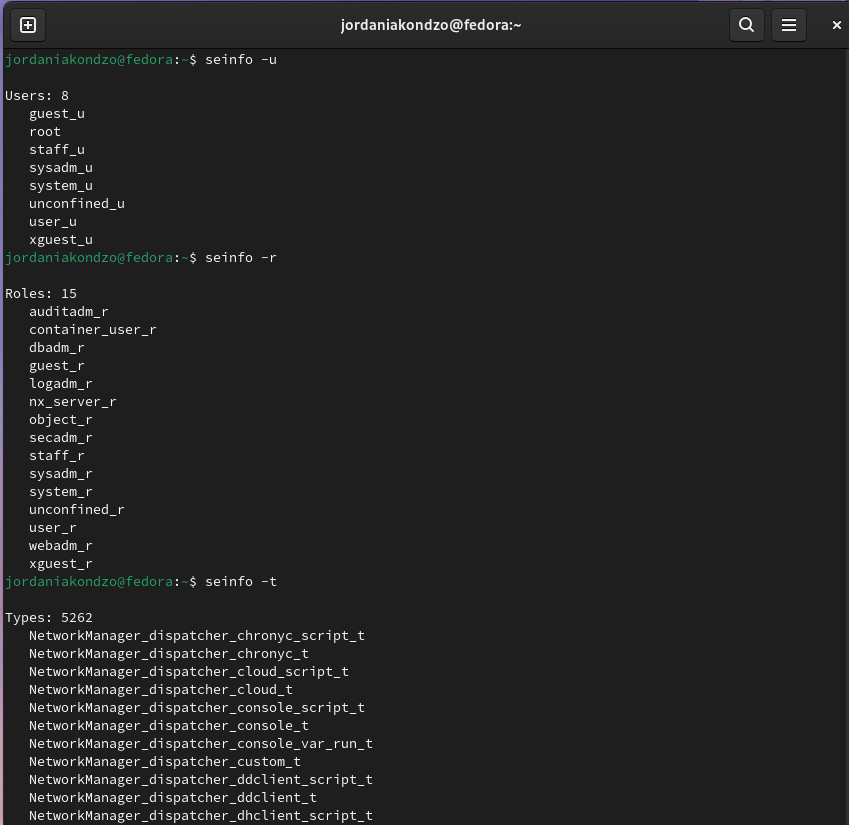


Рис. 6: Анализ политик и типов SELinux

* Это позволило мне получить полную картину того, как SELinux управляет доступом для Apache.

1. Определение типа файлов и поддиректорий, находящихся в директории /var/www, с помощью команды: (рис. 7)

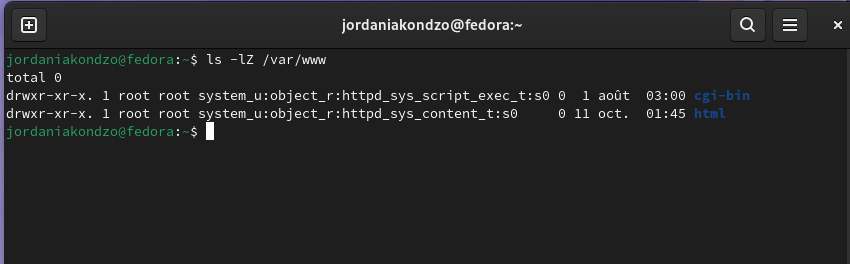


Рис. 7: Определение типа файлов и поддиректорий

1. Определение типа файлов, находящихся в директории `/var/www/html: (рис. 8)

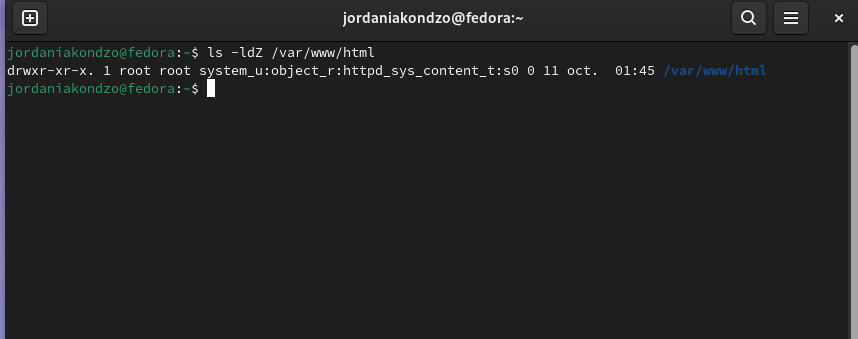


Рис. 8: Определение типа файлов и поддиректорий

1. Определение круга пользователей

* Я определил круг пользователей, которым разрешено создание файлов в директории **/var/www/html.** (рис. 9)

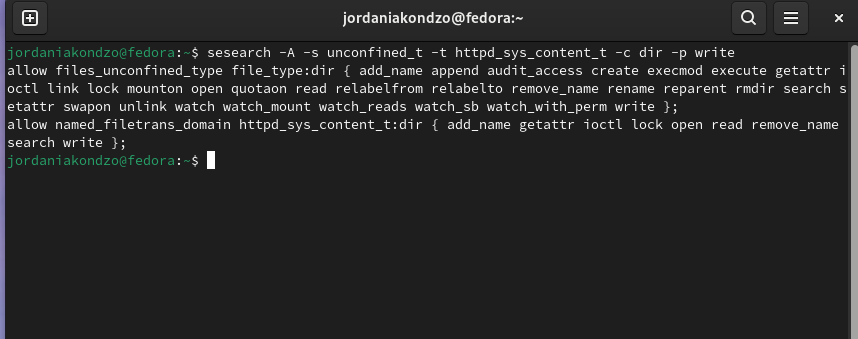


Рис. 9: Определение круга пользователей

1. Создание тестового файла и проверка доступа

* Я создал тестовый файл test.html в каталоге /var/www/html, используя команду: (рис. 10)

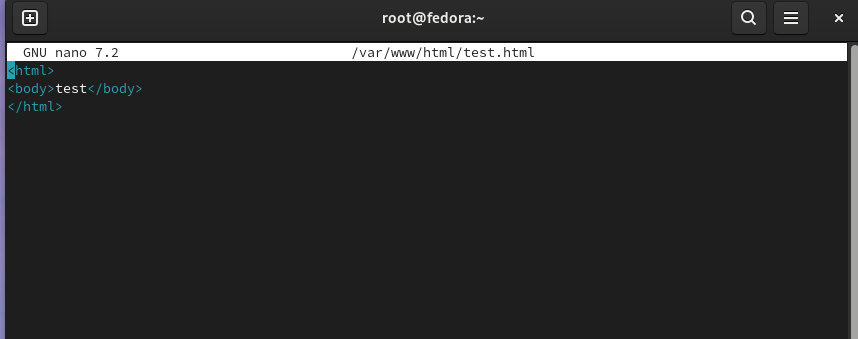


Рис. 10: Создание тестового файла и проверка доступа

* Содержание файла: (рис. 11)

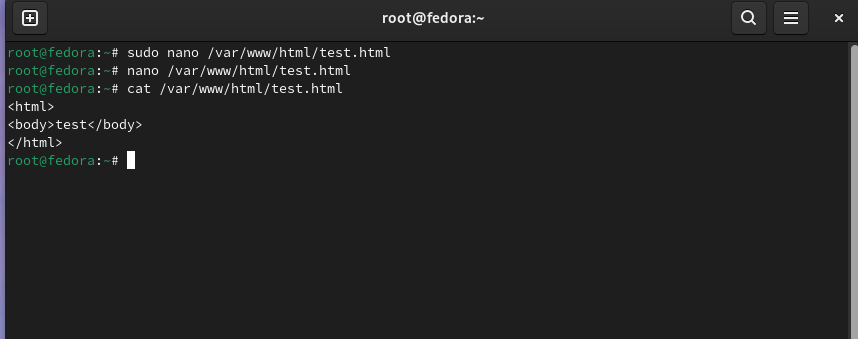


Рис. 11: Проверка статуса SELinux

1. Проверка контекста созданного файла

* Затем я проверил контекст этого файла командой: (рис. 12)

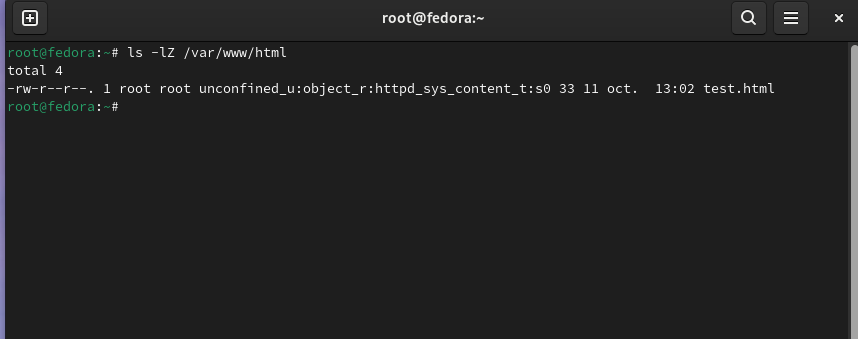


Рис. 12: Проверка контекста созданного файла

* Контекст был установлен как httpd\_sys\_content\_t, что позволило серверу Apache получить доступ к файлу через браузер.

1. Тестирование работы веб-сервера

* Я открыл браузер и перешел по адресу: (рис. 13)

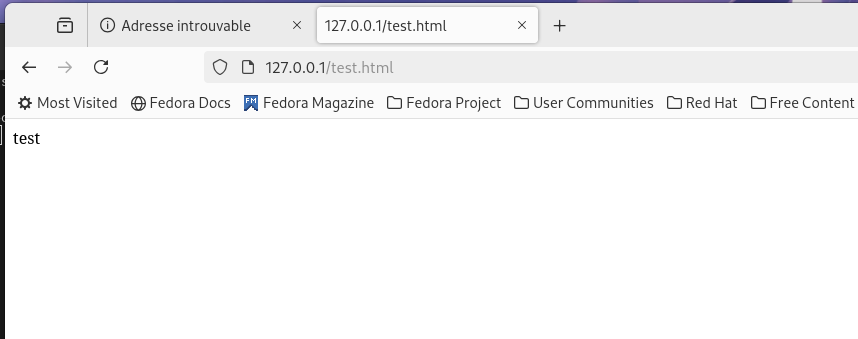


Рис. 13: Тестирование работы веб-сервера

* Файл успешно отобразился, что подтвердило правильность настроек SELinux для данного файла.

1. Анализ контекста файлов

* Я изучил контексты, определенные для файлов Apache. Контекст httpd\_sys\_content\_t позволяет серверу Apache получать доступ к файлам, но не выполнять их или модифицировать без специальных разрешений. (рис. 14)

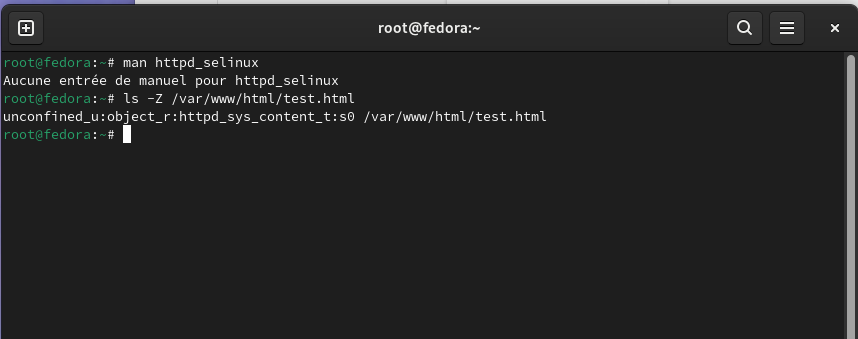


Рис. 14: Анализ контекста файлов

1. Изменение контекста безопасности

* Я изменил контекст безопасности файла test.html на samba\_share\_t, чтобы проверить, как изменятся права доступа: (рис. 15)

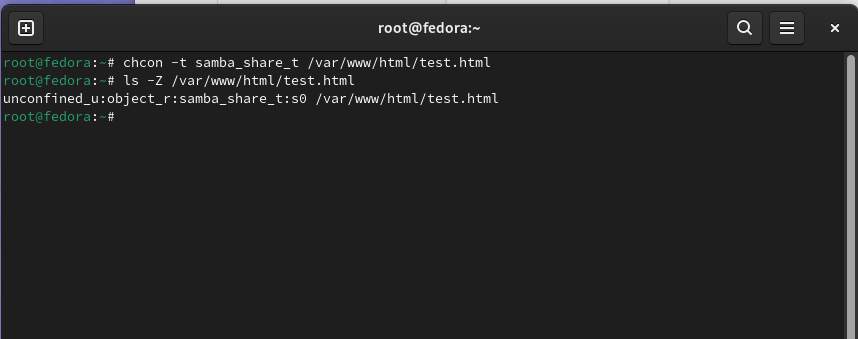


Рис. 15: Изменение контекста безопасности

1. Проверка блокировки доступа

* После изменения контекста на samba\_share\_t, я снова попытался открыть файл через браузер, но получил ошибку **403 Forbidden**. Это показало, что SELinux эффективно блокирует доступ к файлу с неподходящим контекстом. (рис. 16)



Рис. 16: Проверка блокировки доступа

* После этого проверил, что контекст поменялся.

1. Анализ ситуаций

* Я проанализировал ситуацию с помощью следующих команд: (рис. 17)

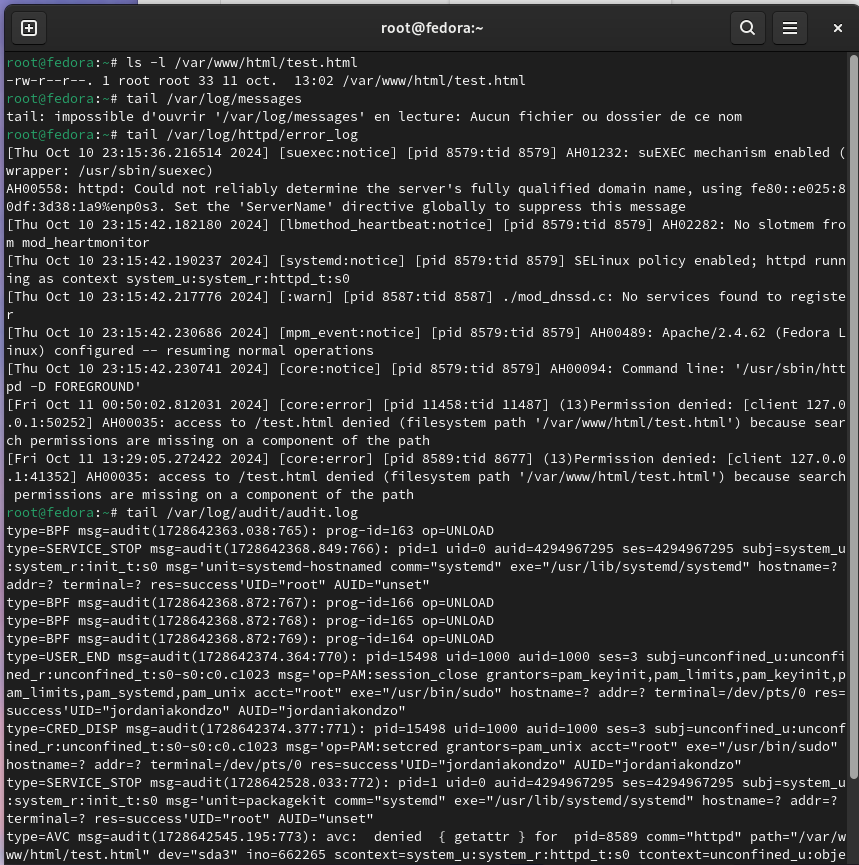


Рис. 17: Анализ ситуаций

* Анализ логов не выявил ошибок или предупреждений, связанных с перезапуском сервера.

1. Замение порта 80 на 81 для Apache в SELinux

* Попробовал запустить веб-сервер Apache на прослушивание ТСР-порта 81 (а не 80, как рекомендует IANA и прописано в /etc/services). Для этого в файле /etc/httpd/httpd.conf нашёл строчку Listen 80 и заменил её на Listen 81. (рис. 18)

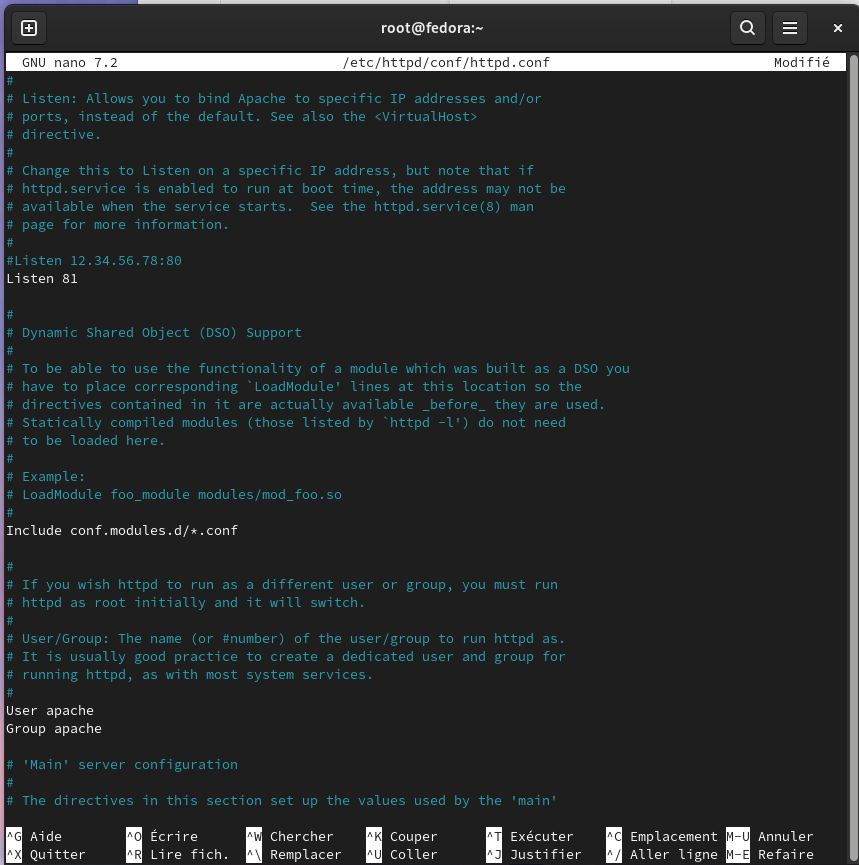


Рис. 18: Замение порта 80 на 81 для Apache в SELinux

1. Перезапуск веб-сервера Apache

* Я перезапустил сервер Apache с помощью команды: (рис. 19)

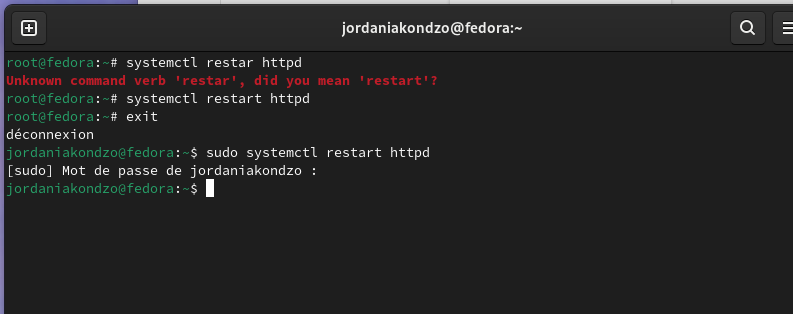


Рис. 19: Перезапуск веб-сервера Apache

* При перезапуске не возникло никаких сбоев, что подтвердило корректность конфигурации.

1. Анализ лог-файлов

* Я проанализировал лог-файлы с помощью следующих команд: (рис. 20)

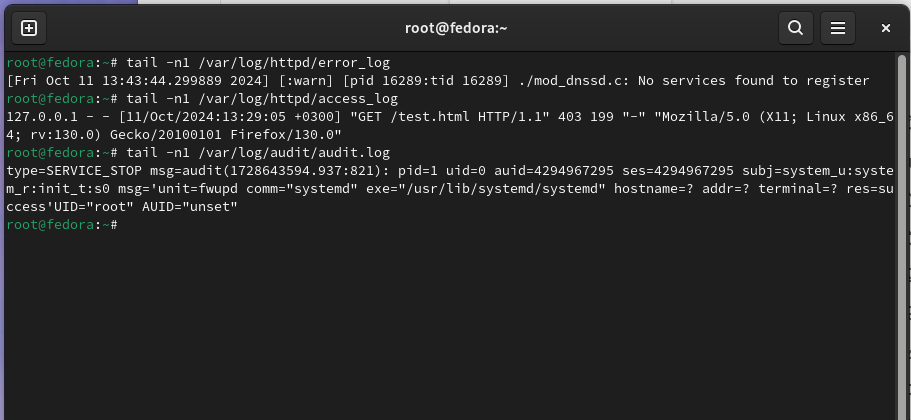


Рис. 20: Анализ лог-файлов

1. Добавление порта 81 для Apache в SELinux

* Для добавления порта 81 я использовал команду: (рис. 21)

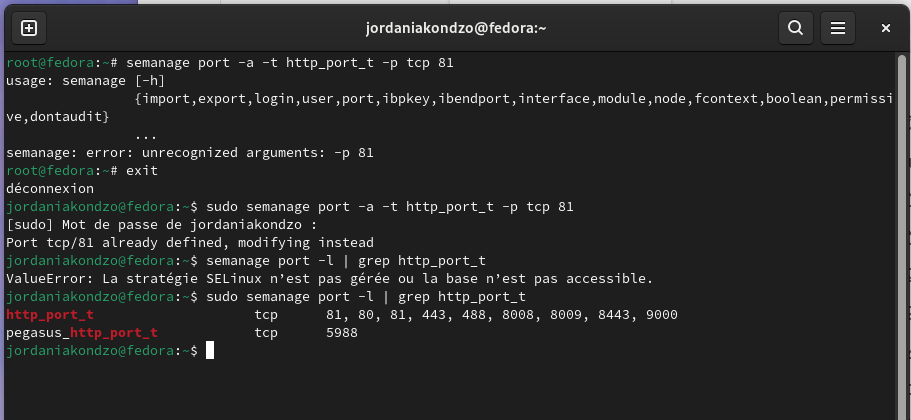


Рис. 21: Добавление порта 81 для Apache в SELinux

* Это позволило серверу Apache работать на порту 81.

1. Проверка списка портов

* Я проверил, что порт 81 был успешно добавлен с помощью команды: (рис. 22)



Рис. 22: Проверка списка портов

* В выводе я увидел, что порты 80 и 81 доступны для Apache.

1. Перезапуск Apache

* Я перезапустил веб-сервер Apache, чтобы изменения вступили в силу: (рис. 23)
* Файл **test.html** успешно отобразился, что подтвердило работу Apache на новом порту 81.

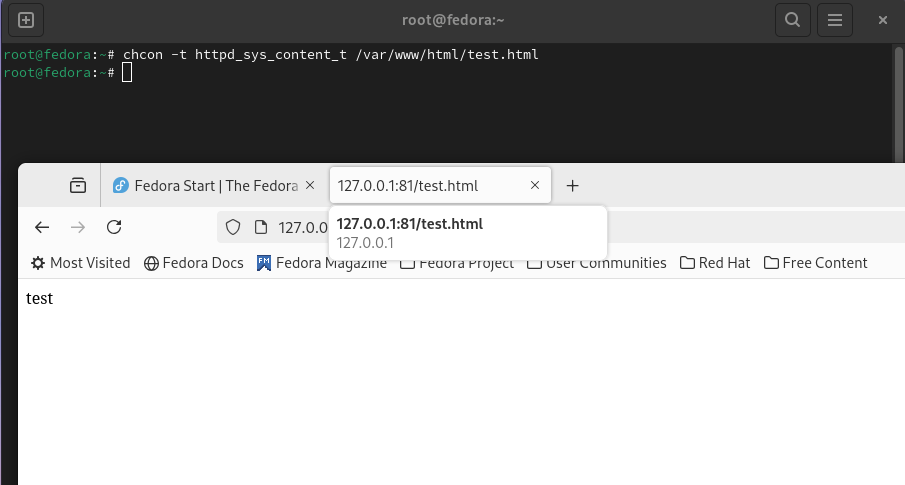


Рис. 23: Перезапуск Apache

1. Вернул контекст httpd\_sys\_cоntent\_\_t к файлу **/var/www/html/ test.html**: (рис. 24)

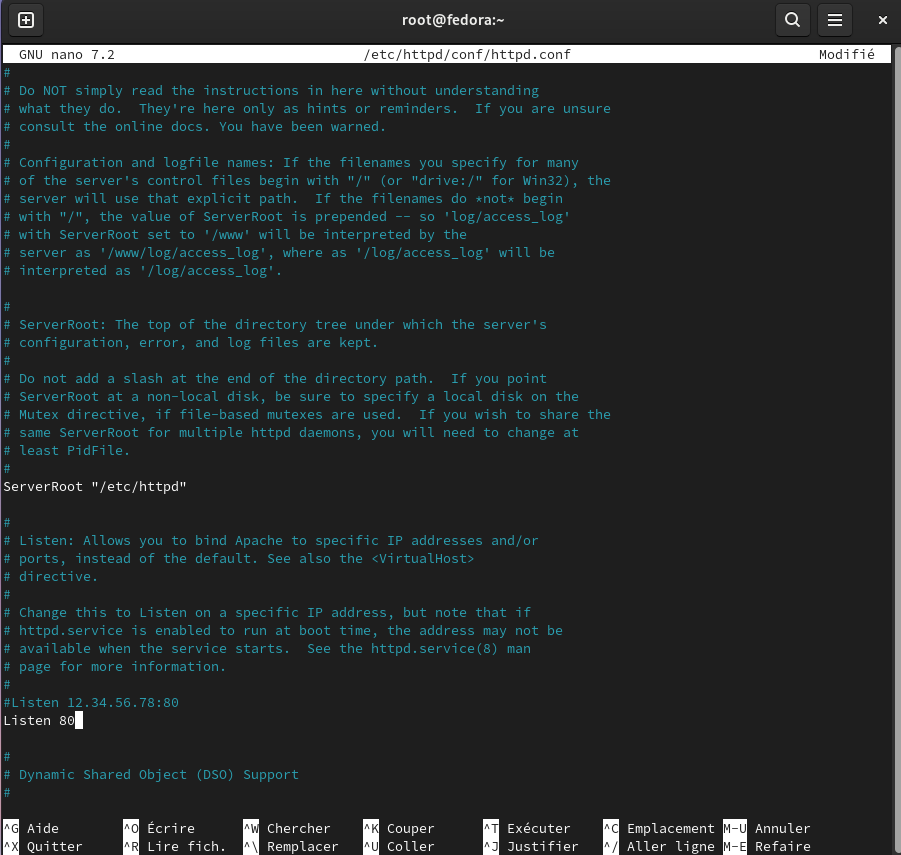


Рис. 24: Вернул контекст httpd\_sys\_cоntent\_\_t

1. Удаление привязки порта 81 к Apache

* После успешного тестирования я удалил порт 81 из списка разрешенных для SELinux с помощью команды: (рис. 25)

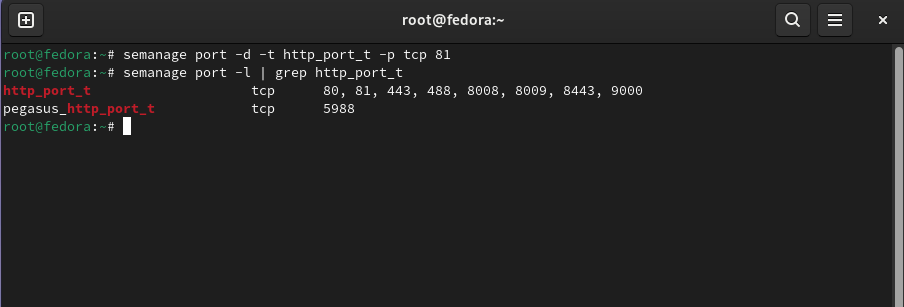


Рис. 25: Удаление привязки порта 81 к Apache

1. Удаление тестового файла

* Я удалил тестовый файл test.html с помощью команды: (рис. 26)



Рис. 26: Удаление тестового файла

# Выводы

В ходе лабораторной работы я приобрел практические навыки работы с SELinux в связке с веб-сервером Apache. Я научился настраивать контексты безопасности для файлов и управлять портами, используя SELinux для обеспечения мандатного контроля доступа. Работая с различными контекстами, такими как httpd\_sys\_content\_t и samba\_share\_t, я увидел, как SELinux блокирует несанкционированный доступ, что помогает значительно повысить безопасность системы.