Лабораторная работа №6

Дисциплина: Основы информационной безопасности

Дарижапов Тимур Андреевич

Содержание

# Цель работы

Развить навыки администрирования ОС Linux. Получить первое практическое знакомство с технологией SELinux. Проверить работу SELinux на практике совместно с веб-сервером Apache.

# Теоретическое введение

SELinux (Security-Enhanced Linux) обеспечивает усиление защиты путем внесения изменений как на уровне ядра, так и на уровне пространства пользователя, что превращает ее в действительно «непробиваемую» операционную систему. Впервые эта система появилась в четвертой версии CentOS, а в 5 и 6 версии реализация была существенно дополнена и улучшена. SELinux имеет три основных режим работы: • Enforcing: Режим по-умолчанию. При выборе этого режима все действия, которые каким-то образом нарушают текущую политику безопасности, будут блокироваться, а попытка нарушения будет зафиксирована в журнале. • Permissive: В случае использования этого режима, информация о всех действиях, которые нарушают текущую политику безопасности, будут зафиксированы в журнале, но сами действия не будут заблокированы. • Disabled: Полное отключение системы принудительного контроля доступа. Политика SELinux определяет доступ пользователей к ролям, доступ ролей к доменам и доступ доменов к типам. Контекст безопасности — все атрибуты SELinux — роли, типы и домены. Более подробно см. в [1].

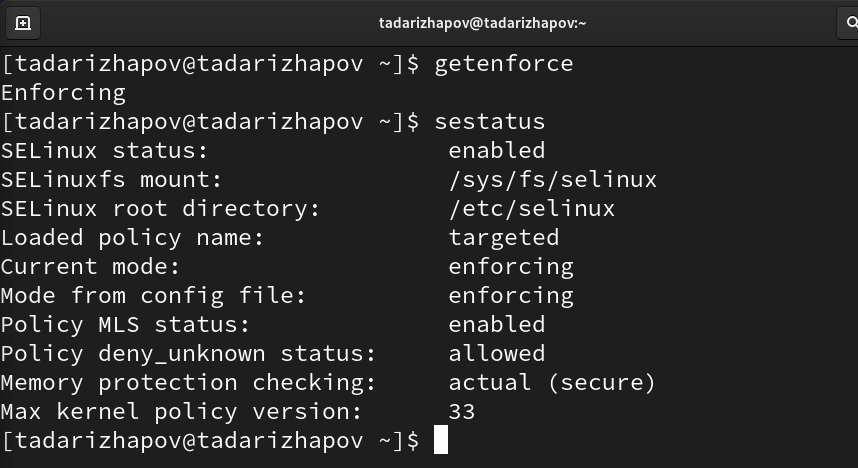
Apache — это свободное программное обеспечение, с помощью которого можно создать веб-сервер. Данный продукт возник как доработанная версия другого HTTP-клиента от национального центра суперкомпьютерных приложений (NCSA).

Для чего нужен Apache сервер: • чтобы открывать динамические PHP-страницы, • для распределения поступающей на сервер нагрузки, • для обеспечения отказоустойчивости сервера, • чтобы потренироваться в настройке сервера и запуске PHP-скриптов.

Apache является кроссплатформенным ПО и поддерживает такие операционные системы, как Linux, BSD, MacOS, Microsoft, BeOS и другие. Более подробно см. в [2].

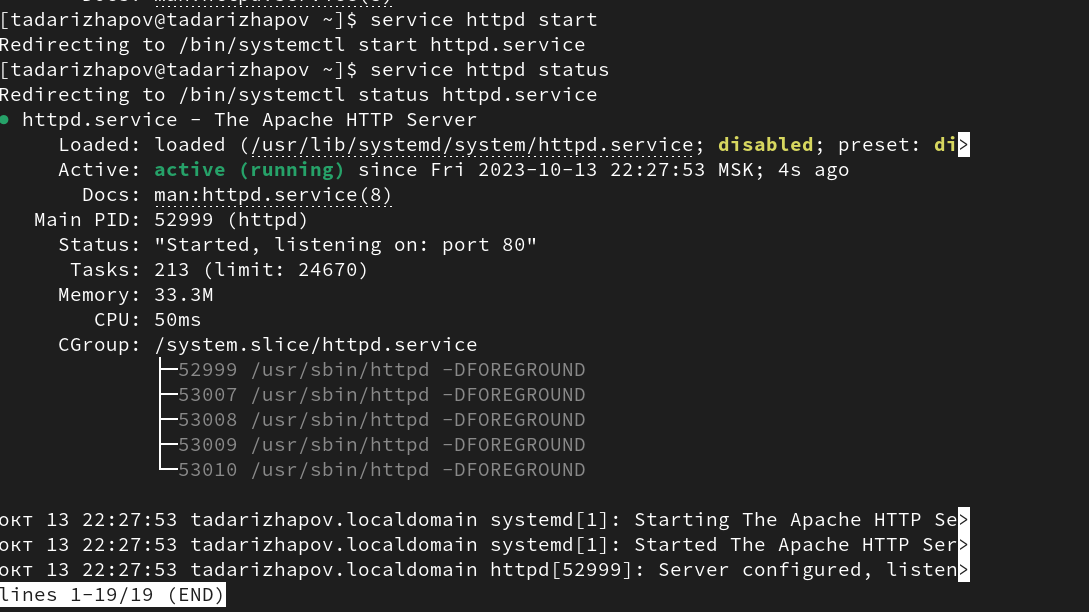
# Выполнение лабораторной работы

1)Входим в систему под своей учетной записью и убеждаемся, что SELinux работает в режиме enforcing политики targeted с помощью команд “getenforce” и “sestatus”(Рисунок 3.1).



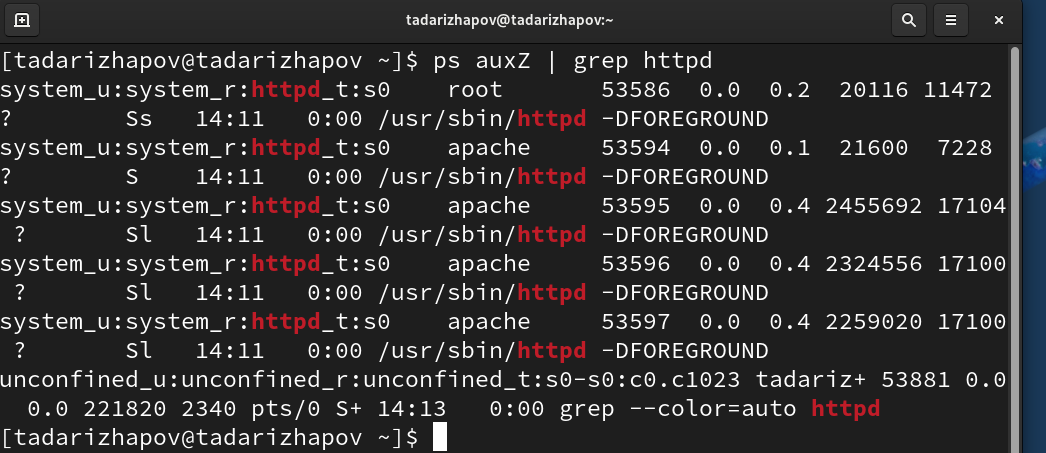
Проверка режима enforcing политики targeted

2)Обращаемся с помощью браузера к веб-серверу, запущенному на моем компьютере, и убеждаемся, что последний работает с помощью команды “service httpd status” (Рисунок 3.2).



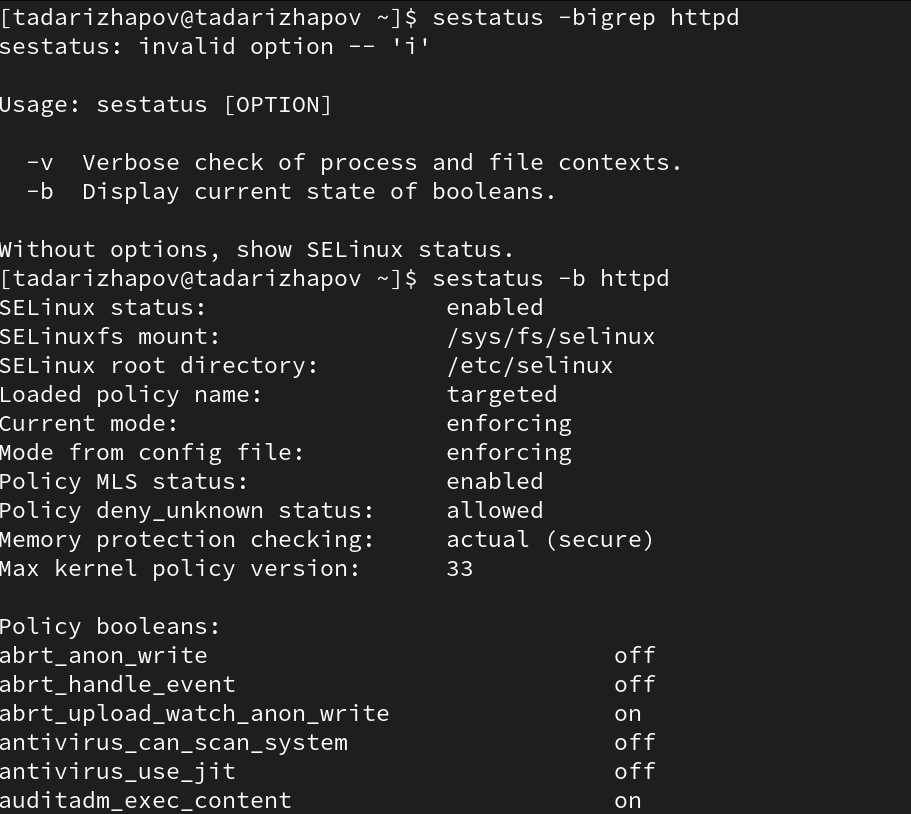
Проверка работы веб-сервера

3)С помощью команды “ps auxZ | grep httpd” определяем контекст безопасности веб-сервера Apache - httpd\_t(Рисунок 3.3).



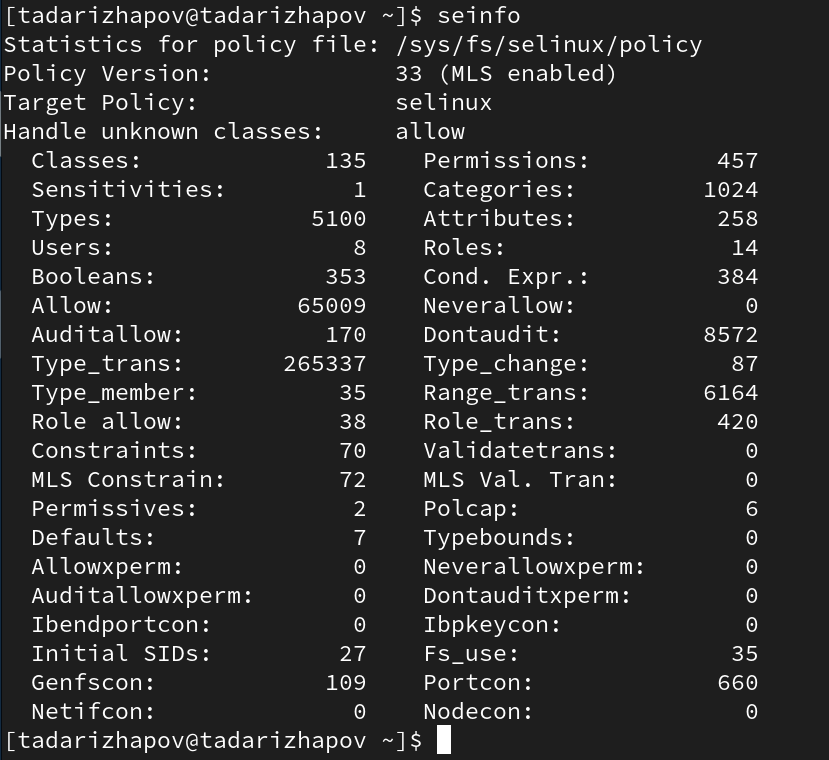
Контекст безопасности веб-сервера Apache

4)Посмотрим текущее состояние переключателей SELinux для Apache с помощью команды “sestatus -bigrep httpd”, многие из переключателей находятся в положении “off”(Рисунок 3.4).



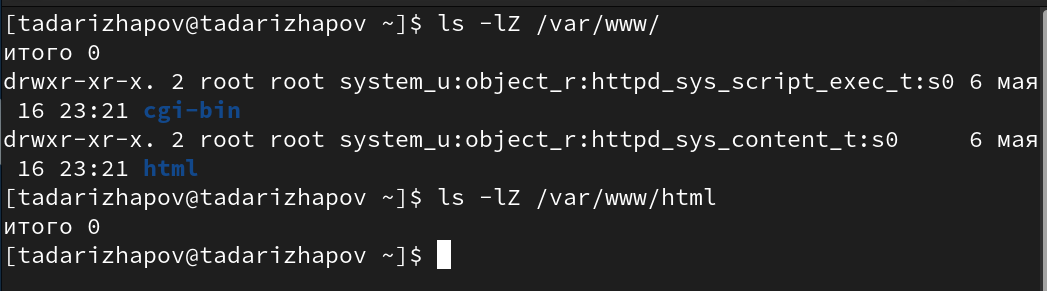
Текущее состояние переключателей SELinux

5)Посмотрим статистику по политике с помощью команды “seinfo”. Множество пользователей - 8, ролей - 14, типов 5100 (Рисунок 3.5).



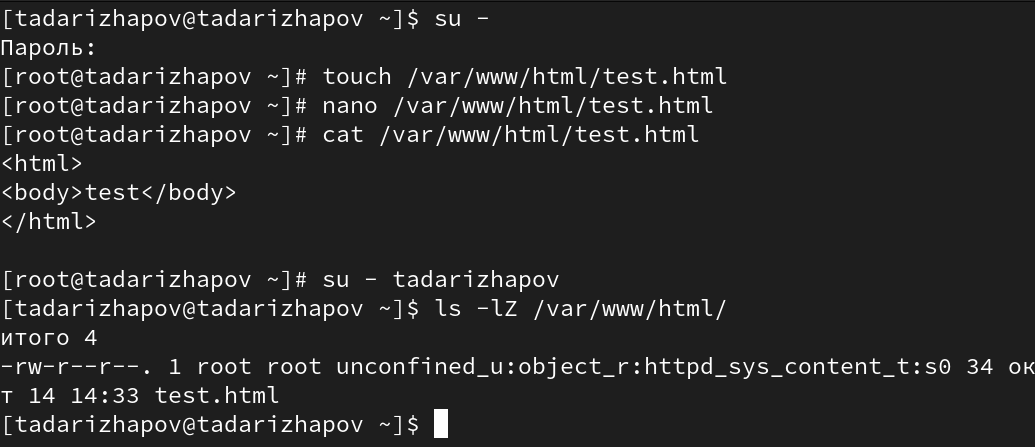
Статистика по политике

6)С помощью команды “ls -lZ /var/www” посмотрим файлы и поддиректории, находящиеся в директории /var/www. Используя команду “ls -lZ /var/www/html”, определяем, что в данной директории файлов нет. Только владелец или суперпользователь может создавать файлы в директории /var/www/html(Рисунок 3.6).



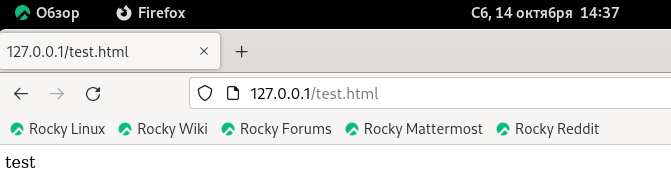
Просмотр файлов и поддиректориий в директории /var/www

7)От имени суперпользователя создаём html-файл /var/www/html/test.html. Контекст созданного файла - httpd\_sys\_content\_t (Рисунок 3.7).



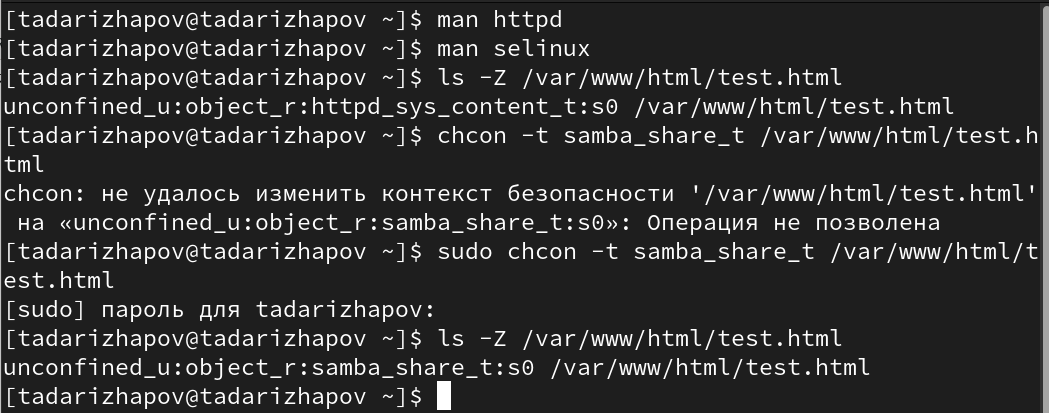
Создание файла /var/www/html/test.html

8)Обращаемся к файлу через веб-сервер, введя в браузере адрес “http://127.0.0.1/test.html”. Файл был успешно отображен (Рисунок 3.8).



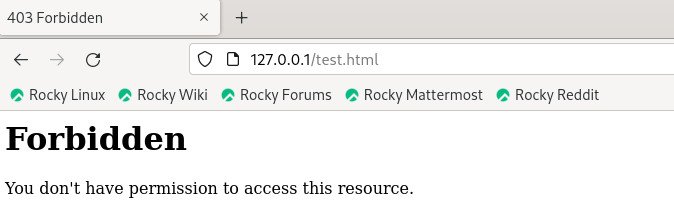
Обращение к файлу через веб-сервер

9)Изучив справку man httpd\_selinux, выясняем, что для httpd определены следующие контексты файлов: httpd\_sys\_content\_t, httpd\_sys\_script\_exec\_t, httpd\_sys\_script\_ro\_t, httpd\_sys\_script\_rw\_t, httpd\_sys\_script\_ra\_t, httpd\_unconfined\_script\_exec\_t. Контекст моего файла - httpd\_sys\_content\_t (в таком случае содержимое должно быть доступно для всех скриптов httpd и для самого демона). Изменяем контекст файла на samba\_share\_t командой “sudo chcon -t samba\_share\_t /var/www/html/test.html” и проверяем, что контекст поменялся(Рисунок 3.9).



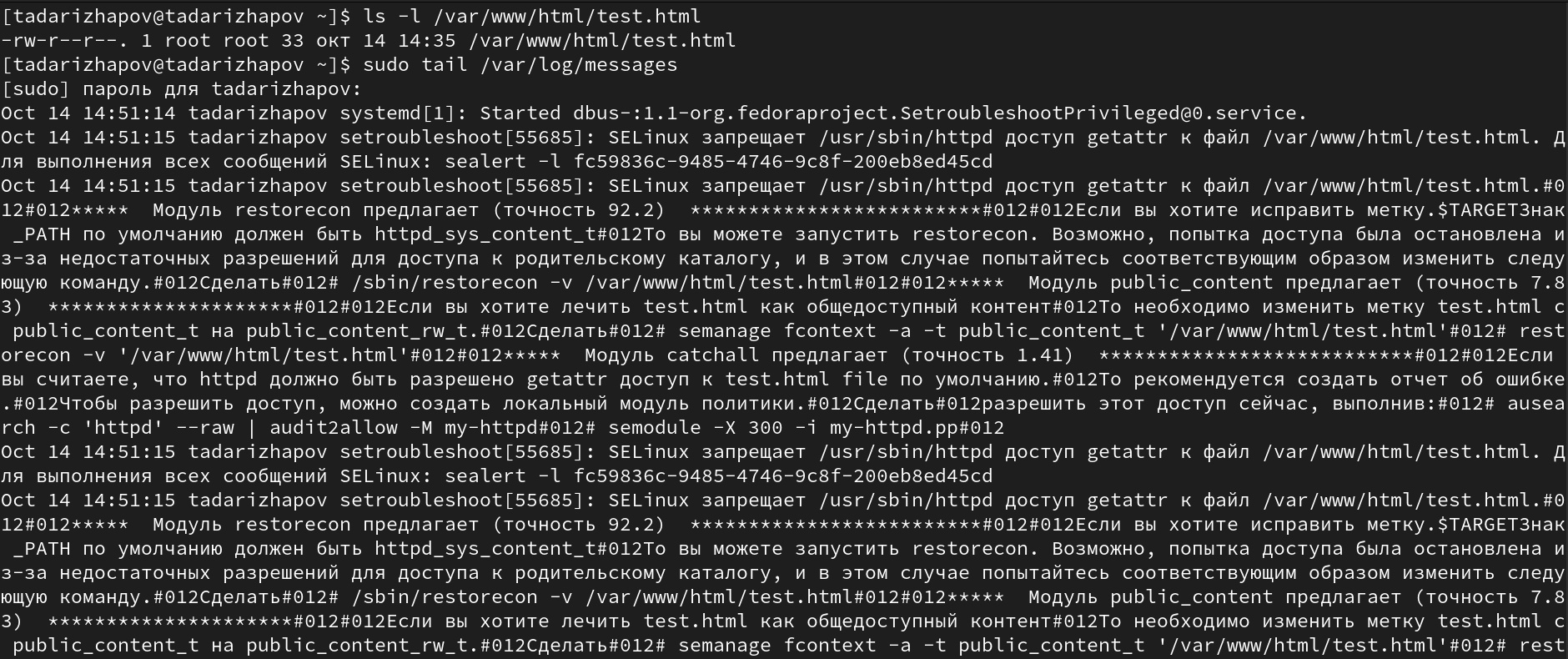
Изменение контекста

10)Попробуем еще раз получить доступ к файлу через веб-сервер, введя в браузере адрес “http://127.0.0.1/test.html” и получаем сообщение об ошибке(т.к. к установленному ранее контексту процесс httpd не имеет доступа)(Рисунок 3.10).



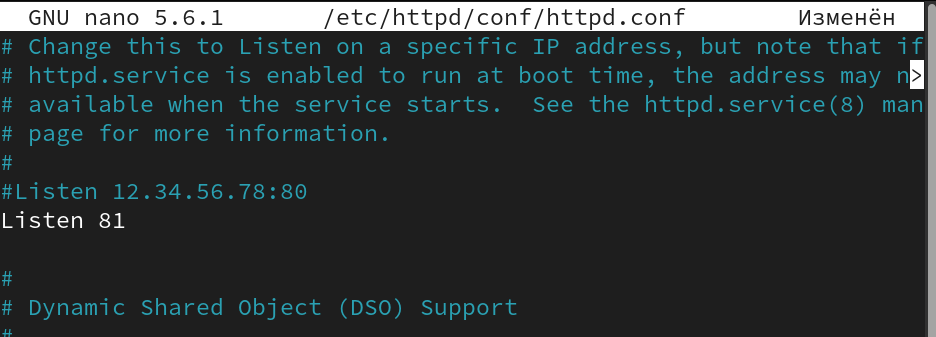
Обращение к файлу через веб-сервер

11)Командой “ls -l /var/www/html/test.html” убеждаемся, что читать данный файл может любой пользователь. Просматриваем системный лог-файл веб-сервера Apache командой “sudo tail /var/log/messages”, отображающий ошибки(Рисунок 3.11).



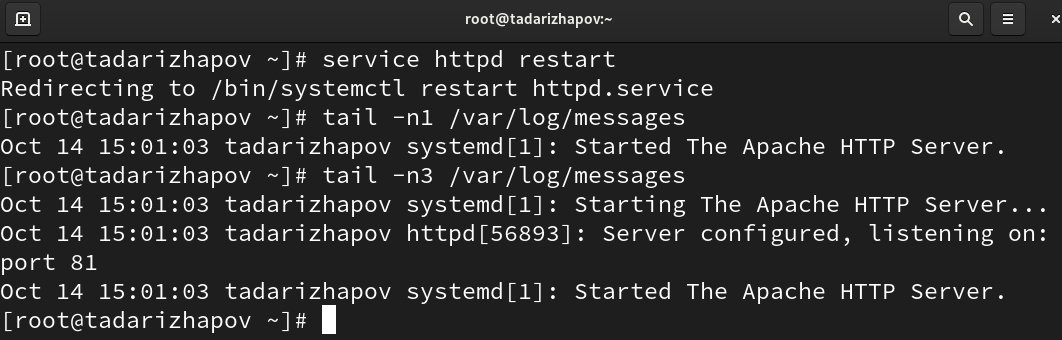
Просмотр log-файла

12)В файле /etc/httpd/conf/httpd.conf заменяем строчку “Listen 80” на “Listen 81”, чтобы установить веб-сервер Apache на прослушивание TCP-порта 81 (Рисунок 3.12).



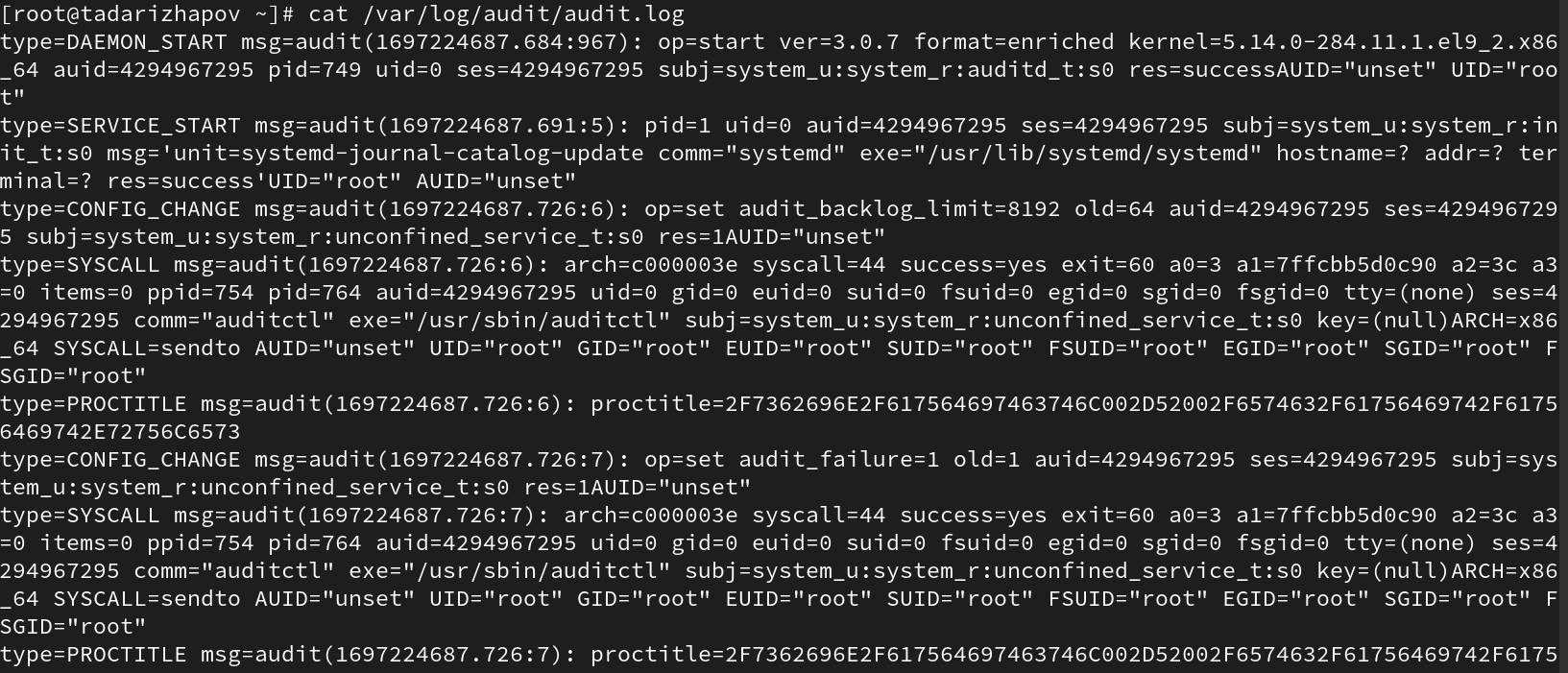
Установка веб-сервера Apache на прослушивание TCP-порта 81

13)Перезапускаем веб-сервер Apache и анализируем лог-файлы командой “tail -nl /var/log/messages” (Рисунок 3.13).



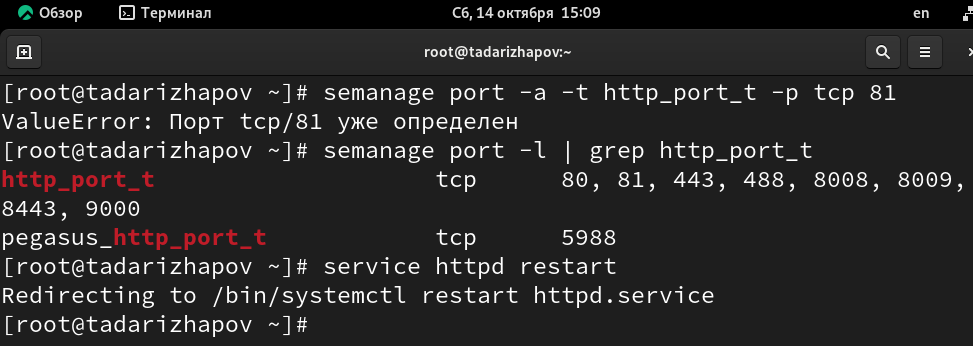
Перезапуск веб-сервера и анализ лог-файлов

14)Просматриваем файлы “var/log/http/error\_log”, “/var/log/http/access\_log” и “/var/log/audit/audit.log” и выясняем, что запись появилась в последнем файле(Рисунок 3.14).



Содержание файла var/log/audit/audit.log

15)Выполняем команду “semanage port -a -t http\_port\_t -р tcp 81” и убеждаемся, что порт TCP-81 установлен. Проверяем список портов командой “semanage port -l | grep http\_port\_t”, убеждаемся, что порт 81 есть в списке и запускаем веб-сервер Apache снова (Рисунок 3.15).

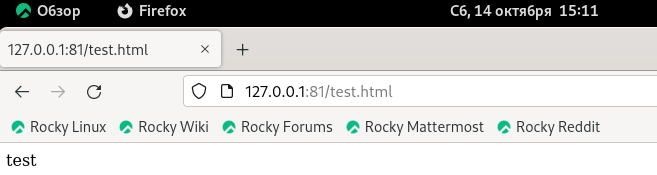


Проверка установки порта 81

16)Вернём контекст “httpd\_sys\_cоntent\_t” файлу “/var/www/html/test.html” командой “chcon -t httpd\_sys\_content\_t /var/www/html/test.html” (Рисунок 3.16) и после этого пробуем получить доступ к файлу через веб-сервер, введя адрес “http://127.0.0.1:81/test.html”, в результате чего увидим содежимое файла - слово “test” (Рисунок 3.17).

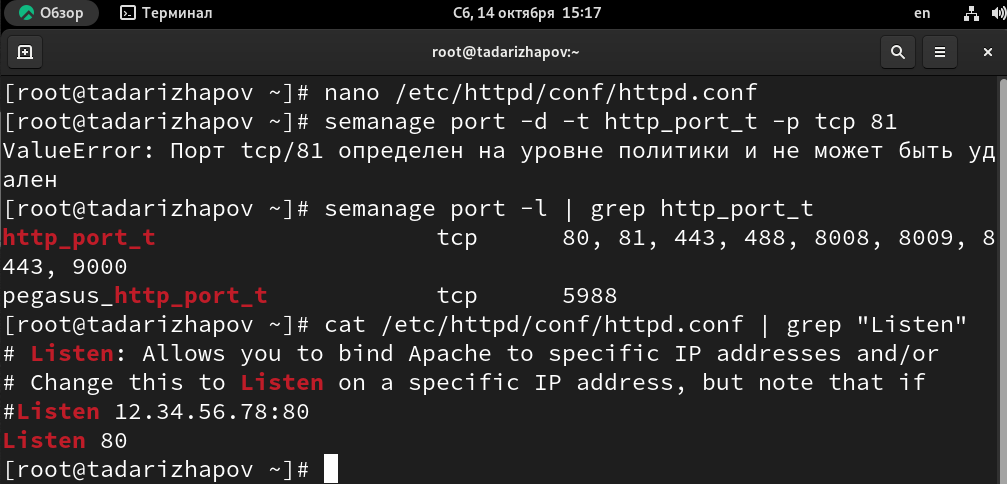


Возвращение исходного контекста файлу



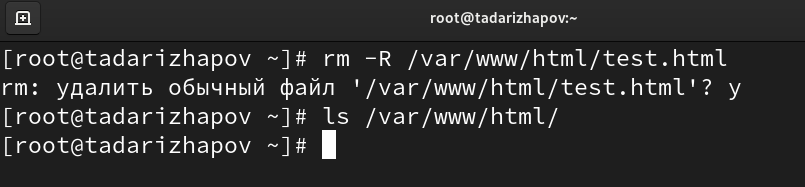
Обращение к файлу через веб-сервер

17)Исправим обратно конфигурационный файл apache, вернув “Listen 80”. Попытаемся удалить привязку http\_port к 81 порту командой “semanage port -d -t http\_port\_t -p tcp 81”, но этот порт определен на уровне политики, поэтому его нельзя удалить(Рисунок 3.18).



Возвращение Listen 80 и попытка удалить порт 81

18)Удаляем файл “/var/www/html/test.html” командой “rm /var/www/html/test.html”(Рисунок 3.19).



Удаление файла test.html

# Выводы

* В ходе выполнения данной лабораторной работы я развил навыки администрирования ОС Linux, получил первое практическое знакомство с технологией SELinux и проверил работу SELinux на практике совместно с веб-сервером Apache.

# Список литературы

* SELinux – описание и особенности работы с системой [Электронный ресурс]. URL: https://habr.com/ru/company/kingservers/blog/209644/.
* Что такое Apache и зачем он нужен? [Электронный ресурс]. URL: https://2domains.ru/support/vps-i-servery/shto-takoye-apache.