Отчёта по лабораторной работе №6

Мандатное разграничение прав в Linux

Кармацкий Никита Сергеевич

Содержание

# 1 Цель работы

Развить навыки администрирования ОС Linux. Получить первое практическое знакомство с технологией SELinux1. Проверить работу SELinx на практике совместно с веб-сервером Apache

# 2 Теоретическое введение

1. **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 [**f?**].

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 [**s?**].

# 3 Выполнение лабораторной работы

1. Вошли в систему под своей учетной записью. Убедились, что SELinux работает в режиме enforcing политики targeted с помощью команд getenforce и sestatus (рис. 1).

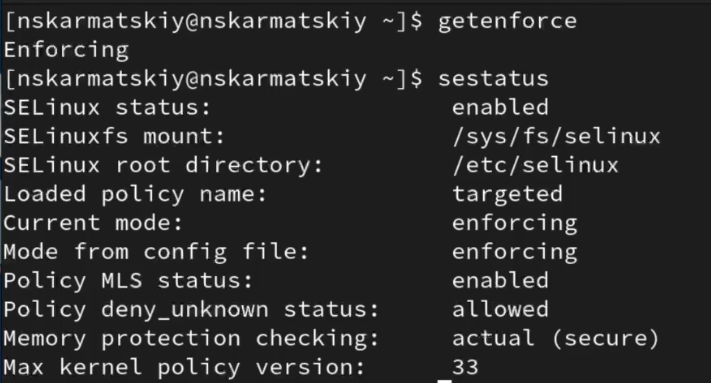


Рис. 1: проверка режима работы SELinux

1. Запускаем сервер apache, далее обращаемся с помощью браузера к веб-серверу, запущенному на компьютере, он работает, что видно из вывода команды service httpd status (рис. 2).

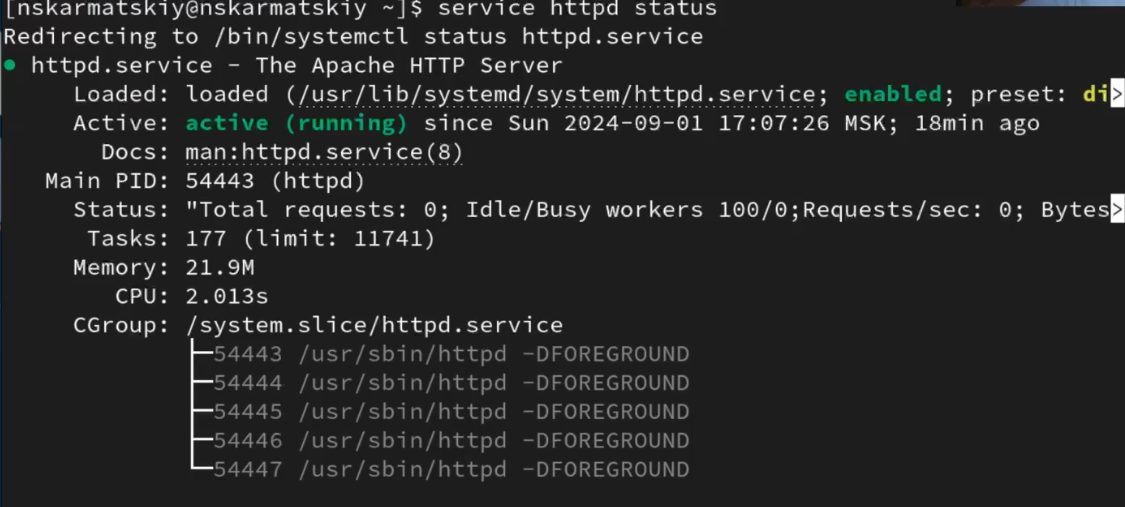


Рис. 2: Проверка работы Apache

1. С помощью команды ps auxZ | grep httpd найдем веб-сервер Apache в списке процессов. Его контекст безопасности - httpd\_t (рис. 3).

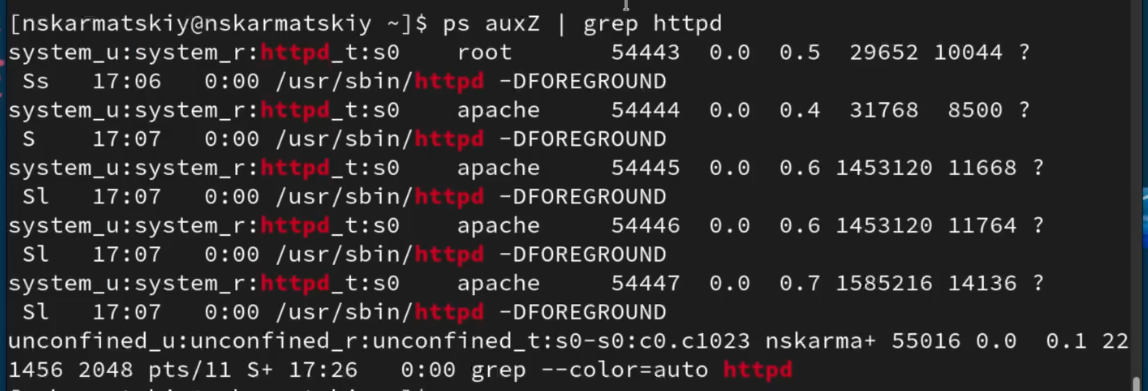


Рис. 3: Контекст безопасности Apache

1. Просмотрим текущее состояние переключателей SELinux для Apache с помощью команды sestatus -bigrep httpd (рис. 4).

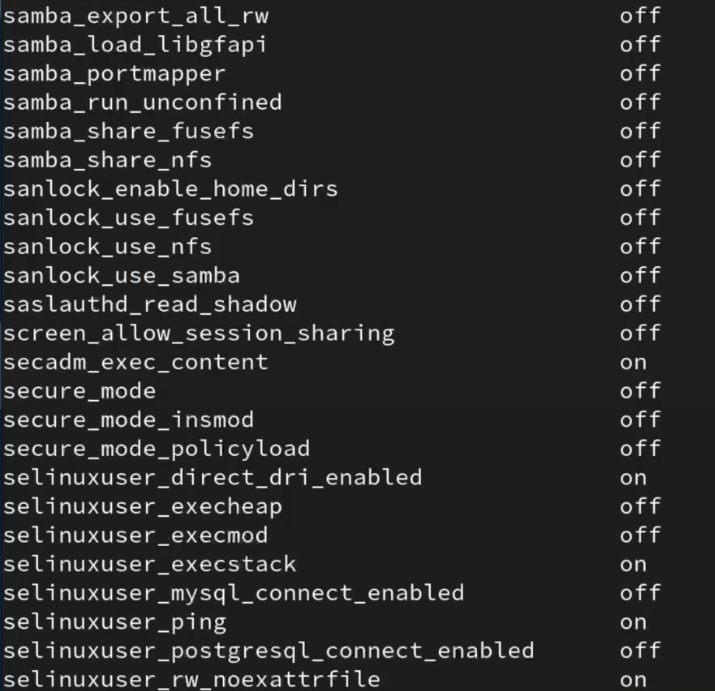


Рис. 4: Состояние переключателей SELinux

1. Просмотрим статистику по политике с помощью команды seinfo. Множество пользователей - 8, ролей - 39, типов - 5135. (рис. 5).

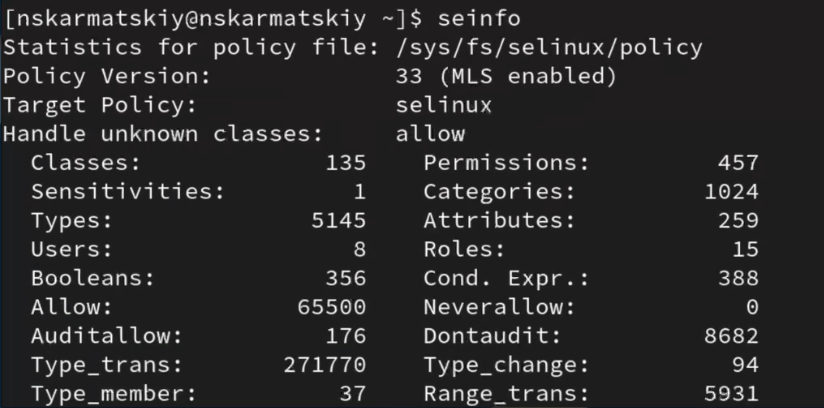


Рис. 5: Cтатистика по политике

1. Типы поддиректорий, находящихся в директории /var/www, с помощью команды ls -lZ /var/www следующие: владелец - root, права на изменения только у владельца. Файлов в директории нет (рис. 6).

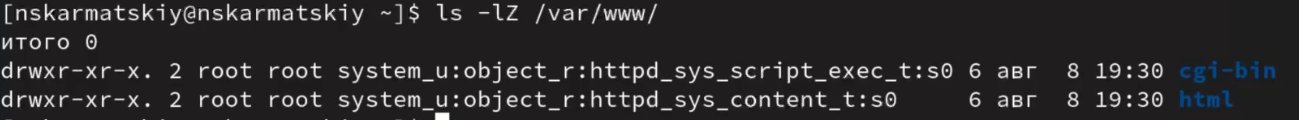


Рис. 6: Типы поддиректорий

1. В директории /var/www/html нет файлов. (рис. 7).

Рис. 7: Типы файлов

Рис. 7: Типы файлов

1. Создать файл может только суперпользователь, поэтому от его имени создаем файл touch.html cо следующим содержанием:

<html>  
<body>test</body>  
</html>

(рис. 8).

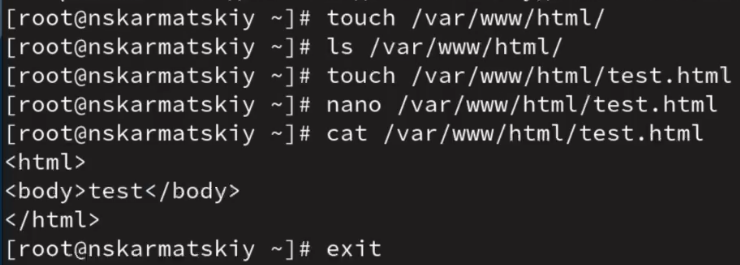


Рис. 8: Создание файла

1. Проверяем контекст созданного файла. По умолчанию это httpd\_sys\_content\_t (рис. 9).

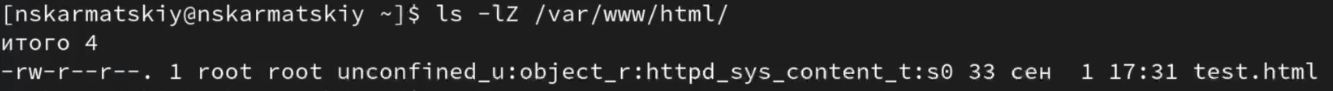


Рис. 9: Контекст файла

1. Обращемся к файлу через веб-сервер, введя в браузере адрес http://127.0.0.1/test.html. Файл был успешно отображён (рис. 10).

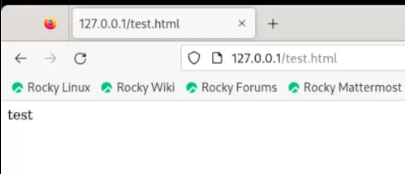


Рис. 10: Отображение файла

1. Изучим справку man httpd\_selinux. Рассмотрим полученный контекст детально. Так как по умолчанию пользователи CentOS являются свободными от типа (unconfined в переводе с англ. означает свободный), созданному нами файлу test.html был сопоставлен SELinux, пользователь unconfined\_u. Это первая часть контекста. Далее политика ролевого разделения доступа RBAC используется процессами, но не файлами, поэтому роли не имеют никакого значения для файлов. Роль object\_r используется по умолчанию для файлов на «постоянных» носителях и на сетевых файловых системах. (В директории /ргос файлы, относящиеся к процессам, могут иметь роль system\_r. Если активна политика MLS, то могут использоваться и другие роли, например, secadm\_r. Данный случай мы рассматривать не будем, как и предназначение :s0). Тип httpd\_sys\_content\_t позволяет процессу httpd получить доступ к файлу. Благодаря наличию последнего типа мы получили доступ к файлу при обращении к нему через браузер. (рис. 11).

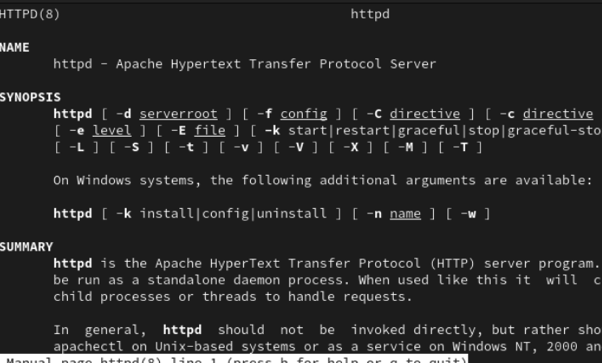


Рис. 11: Изучение справки по команде

1. Изменяем контекст файла /var/www/html/test.html с httpd\_sys\_content\_t на любой другой, к которому процесс httpd не должен иметь доступа, например, на samba\_share\_t: chcon -t samba\_share\_t /var/www/html/test.html ls -Z /var/www/html/test.html Контекст действительно поменялся (рис. 12).

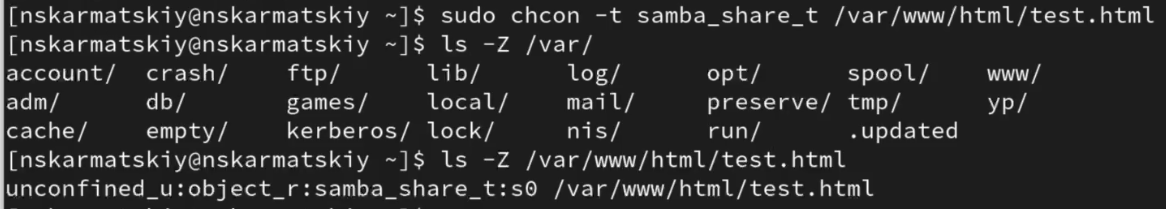


Рис. 12: Изменение контекста

1. При попытке отображения файла в браузере получаем сообщение об ошибке (рис. 13).

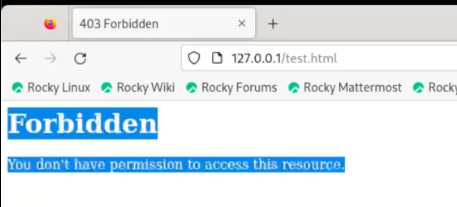


Рис. 13: Отображение файла

файл не был отображён, хотя права доступа позволяют читать этот файл любому пользователю, потому что установлен контекст, к которому процесс httpd не должен иметь доступа.

1. Просматриваем log-файлы веб-сервера Apache и системный лог-файл: tail /var/log/messages. Если в системе окажутся запущенными процессы setroubleshootd и audtd, то вы также сможете увидеть ошибки, аналогичные указанным выше, в файле /var/log/audit/audit.log. (рис. 14).

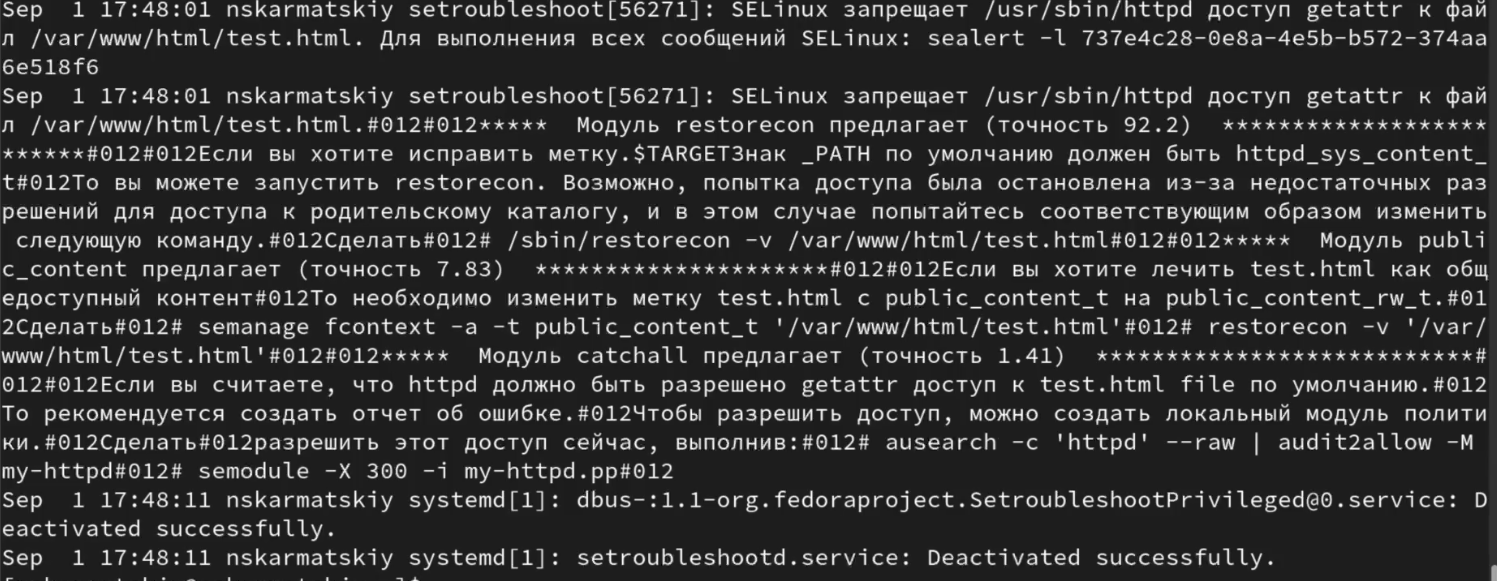


Рис. 14: Попытка прочесть лог-файл

1. Чтобы запустить веб-сервер Apache на прослушивание ТСР-порта 81 (а не 80, как рекомендует IANA и прописано в /etc/services) открываем файл /etc/httpd/httpd.conf для изменения. (рис. 15).

Рис. 15: Изменение файла

Рис. 15: Изменение файла

1. Находим строчку Listen 80 и заменяем её на Listen 81. (рис. 16).

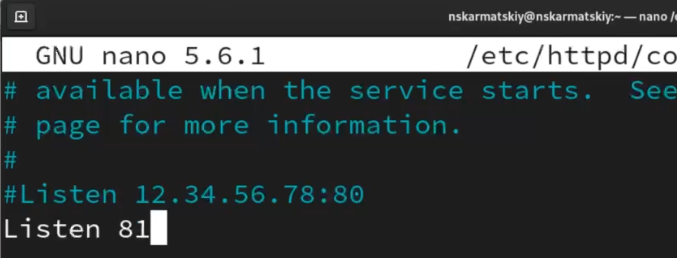


Рис. 16: Изменение порта

1. Выполняем перезапуск веб-сервера Apache. Произошёл сбой, потому что порт 80 для локальной сети, а 81 нет (рис. 17).

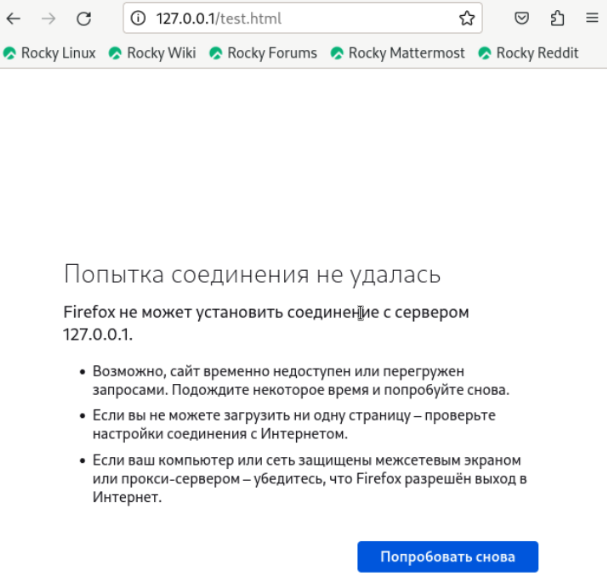


Рис. 17: Попытка прослушивания другого порта

1. Проанализируем лог-файлы: tail -nl /var/log/messages (рис. 18).

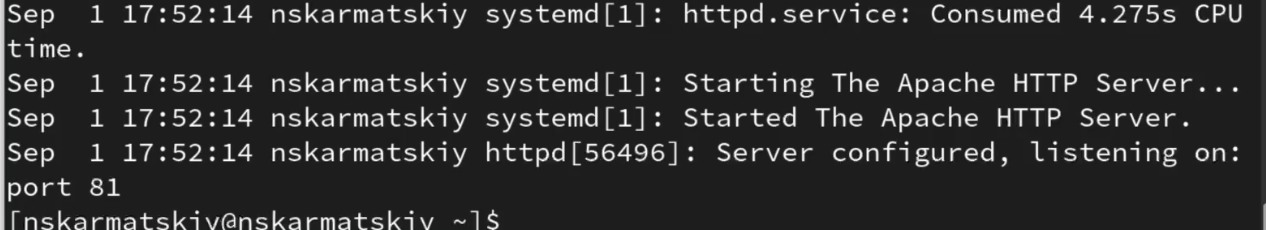


Рис. 18: Проверка лог-файлов

1. Просмотрим файлы /var/log/http/error\_log, /var/log/http/access\_log и /var/log/audit/audit.log и выясним, в каких файлах появились записи. Запись появилась в файлу error\_log (рис. 19).

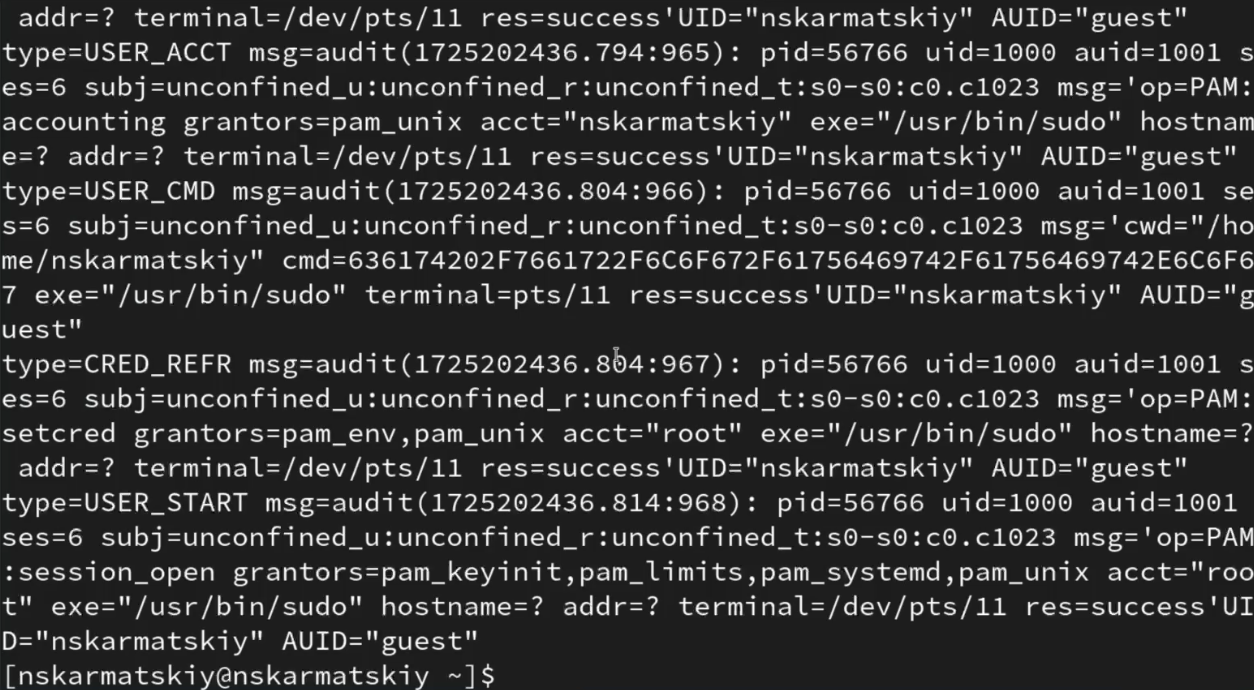


Рис. 19: Проверка лог-файлов

1. Выполняем команду semanage port -a -t http\_port\_t -р tcp 81 После этого проверяем список портов командой semanage port -l | grep http\_port\_t Порт 81 появился в списке (рис. 20).

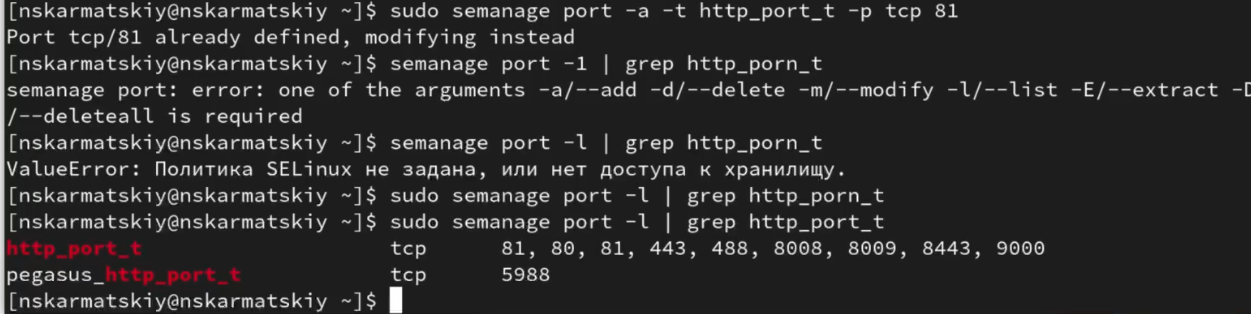


Рис. 20: Проверка портов

1. Перезапускаем сервер Apache (рис. 21).

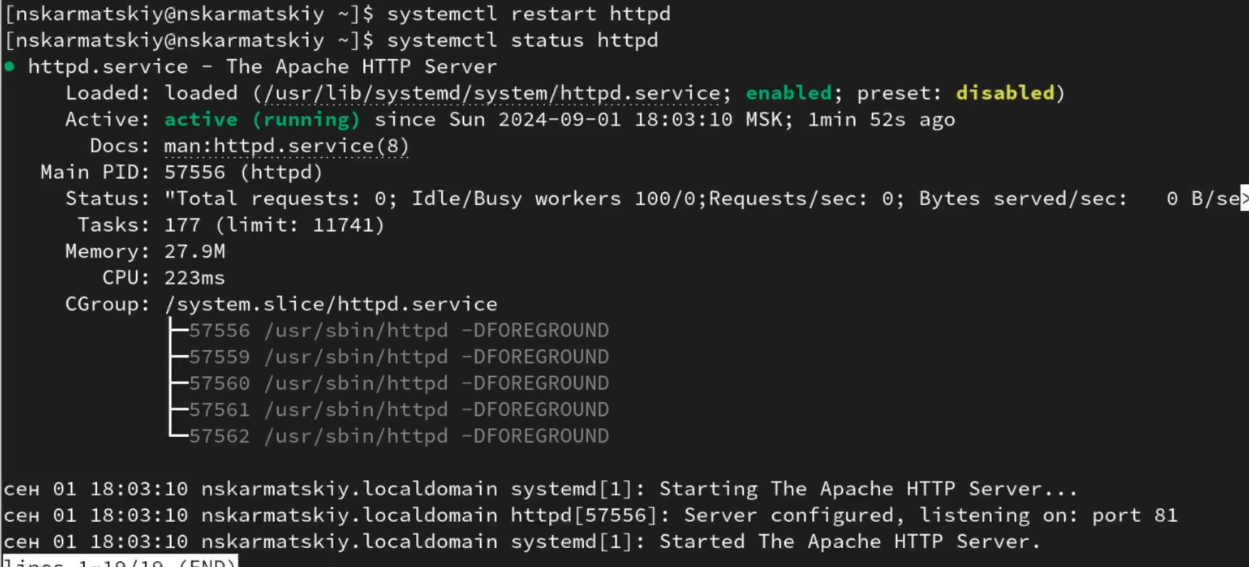


Рис. 21: Перезапуск сервера

1. Теперь он работает, ведь мы внесли порт 81 в список портов httpd\_port\_t (рис. 22).

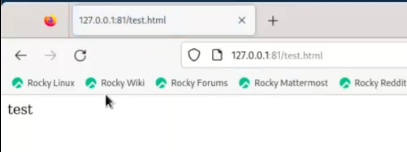


Рис. 22: Проверка сервера

1. Возвращаем в файл /etc/httpd/httpd.conf порт 80, вместо 81. Проверяем, что порт 81 удален, это правда. (рис. 23).

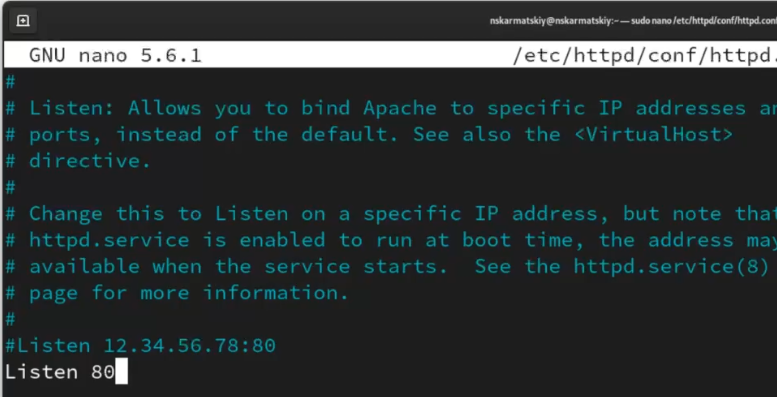


Рис. 23: Проверка порта 81

1. Далее удаляем файл test.html, проверяем, что он удален(рис. 24).

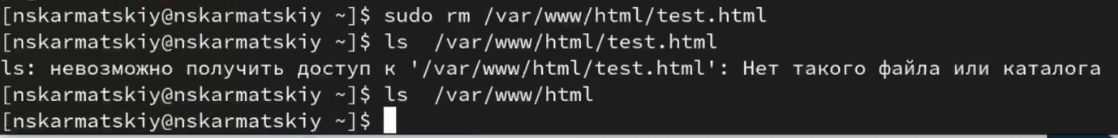


Рис. 24: Удаление файла

# 4 Выводы

В ходе выполнения данной лабораторной работы были развиты навыки администрирования ОС Linux, получено первое практическое знакомство с технологией SELinux и проверена работа SELinux на практике совместно с веб-сервером Apache.