Отчет по лабораторной работе №6

Основы информационной безопасности

Петрова Алевтина Александровна НКА-бд-04-23

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. [**course?**]

# 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:*

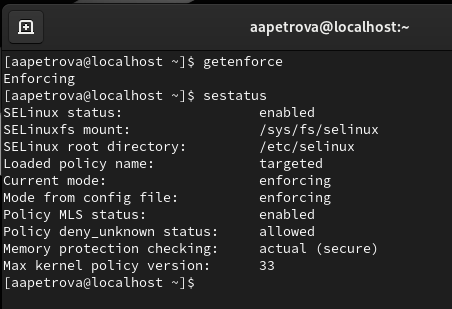
* чтобы открывать динамические PHP-страницы,
* для распределения поступающей на сервер нагрузки,
* для обеспечения отказоустойчивости сервера,
* чтобы потренироваться в настройке сервера и запуске PHP-скриптов.

Apache является кроссплатформенным ПО и поддерживает такие операционные системы, как Linux, BSD, MacOS, Microsoft, BeOS и другие.

Более подробно см. в [**s?**].

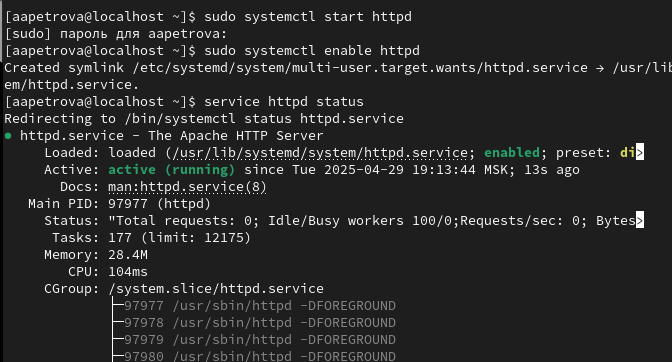
# 3 Выполнение лабораторной работы

Вошла в систему под своей учетной записью. Убедилась, что SELinux работает в режиме enforcing политики targeted с помощью команд getenforce и sestatus (рис. [**fig:001?**]).



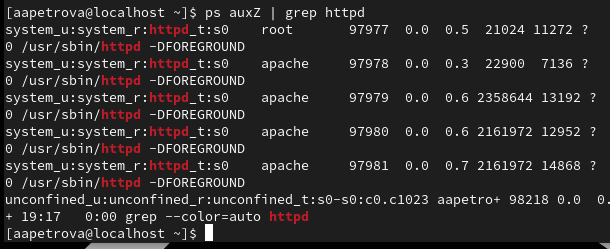
проверка режима работы SELinux

Запускаю сервер apache, далее обращаюсь с помощью браузера к веб-серверу, запущенному на компьютере, он работает, что видно из вывода команды service httpd status (рис. [**fig:002?**]).



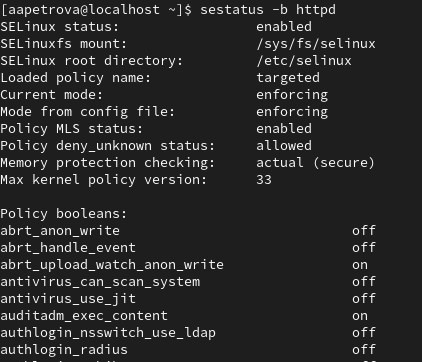
Проверка работы Apache

С помощью команды ps auxZ | grep httpd нашла веб-сервер Apache в списке процессов. Его контекст безопасности - httpd\_t (рис. [**fig:003?**]).



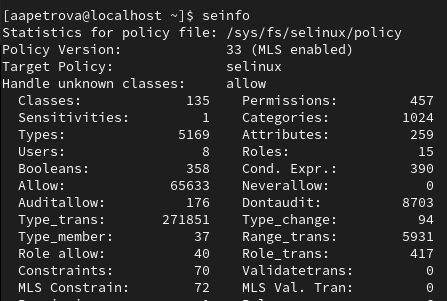
Контекст безопасности Apache

Просмотрела текущее состояние переключателей SELinux для Apache с помощью команды sestatus -bigrep httpd (рис. [**fig:004?**]).



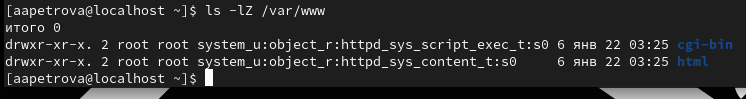
Состояние переключателей SELinux

Просмотрела статистику по политике с помощью команды seinfo. Множество пользователей - 8, ролей - 39, типов - 5135. (рис. [**fig:005?**]).



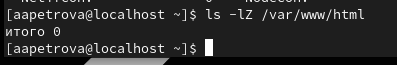
Cтатистика по политике

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. [**fig:006?**]).



Типы поддиректорий

В директории /var/www/html нет файлов. (рис. [**fig:007?**]).

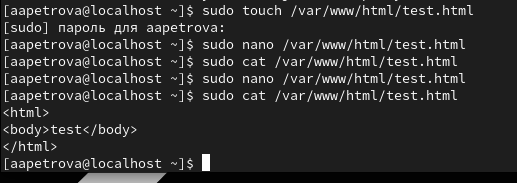


Типы файлов

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

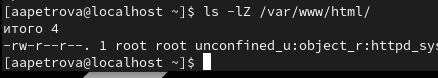
<html>  
<body>test</body>  
</html>

(рис. [**fig:008?**]).



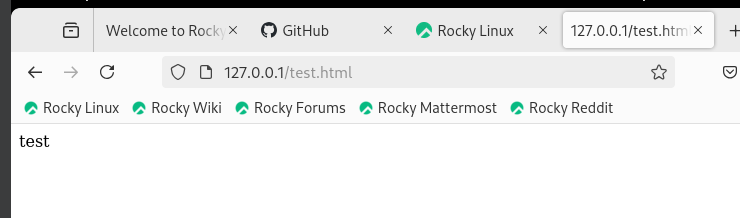
Создание файла

Проверяю контекст созданного файла. По умолчанию это httpd\_sys\_content\_t (рис. [**fig:009?**]).



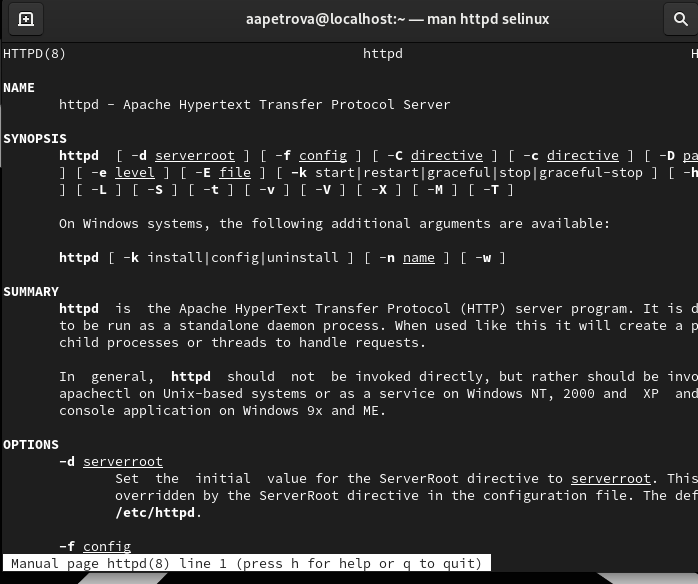
Контекст файла

Обращаюсь к файлу через веб-сервер, введя в браузере адрес http://127.0.0.1/test.html. Файл был успешно отображён (рис. [**fig:010?**]).



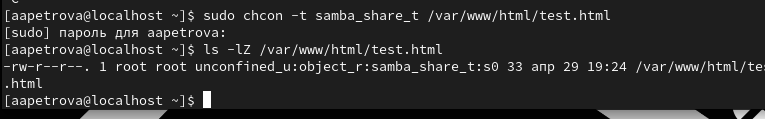
Отображение файла

Изучила 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. [**fig:011?**]).



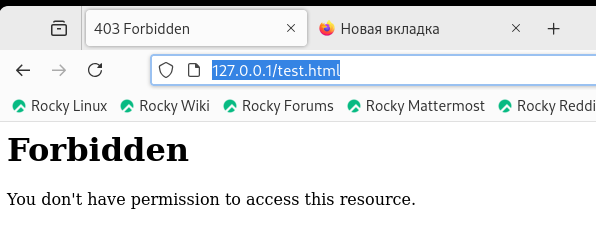
Изучение справки по команде

Изменяю 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. [**fig:012?**]).



Изменение контекста

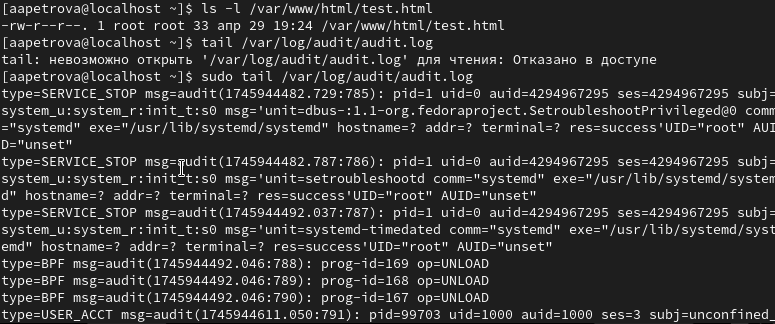
При попытке отображения файла в браузере получаем сообщение об ошибке (рис. [**fig:013?**]).



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.

Просматриваю 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. [**fig:014?**]).



Попытка прочесть лог-файл

Чтобы запустить веб-сервер Apache на прослушивание ТСР-порта 81 (а не 80, как рекомендует IANA и прописано в /etc/services) открываю файл /etc/httpd/httpd.conf для изменения. (рис. [**fig:015?**]).

Изменение файла

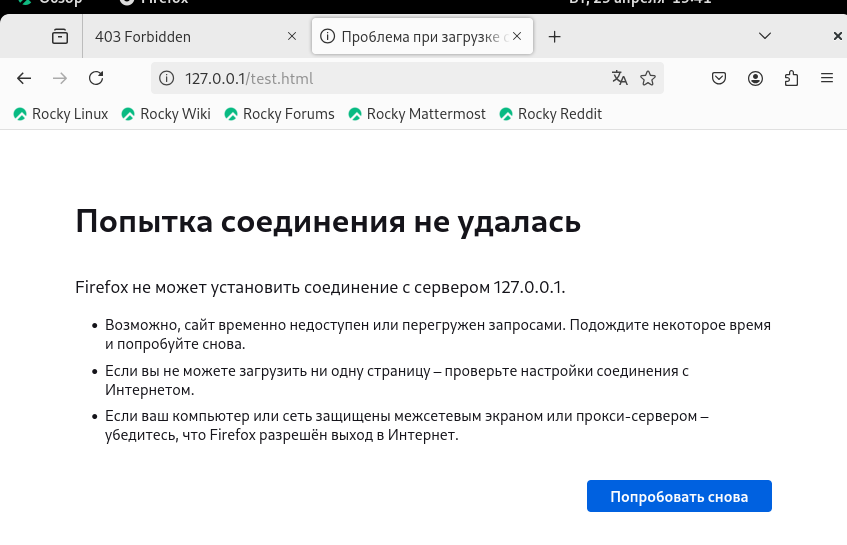
Изменение файла

Нахожу строчку Listen 80 и заменяю её на Listen 81. (рис. [**fig:016?**]).



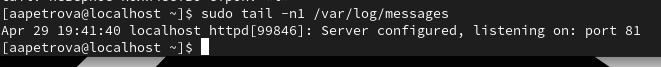
Изменение порта

Выполняю перезапуск веб-сервера Apache. Произошёл сбой, потому что порт 80 для локальной сети, а 81 нет (рис. [**fig:017?**]).



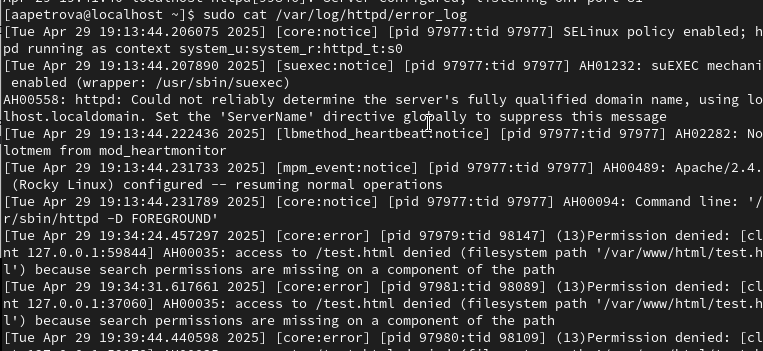
Попытка прослушивания другого порта

Проанализируйте лог-файлы: tail -nl /var/log/messages (рис. [**fig:018?**]).



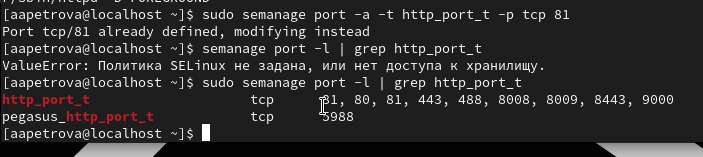
Проверка лог-файлов

Просмотрите файлы /var/log/http/error\_log, /var/log/http/access\_log и /var/log/audit/audit.log и выясните, в каких файлах появились записи. Запись появилась в файлу error\_log (рис. [**fig:019?**]).



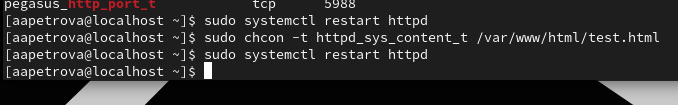
Проверка лог-файлов

Выполняю команду semanage port -a -t http\_port\_t -р tcp 81 После этого проверяю список портов командой semanage port -l | grep http\_port\_t Порт 81 появился в списке (рис. [**fig:020?**]).



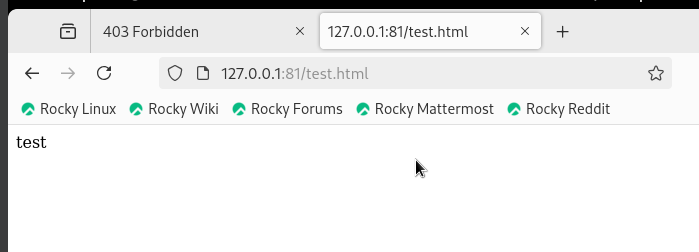
Проверка портов

Перезапускаю сервер Apache (рис. [**fig:021?**]).



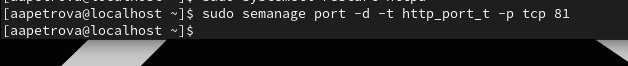
Перезапуск сервера

Теперь он работает, ведь мы внесли порт 81 в список портов htttpd\_port\_t (рис. [**fig:022?**]).



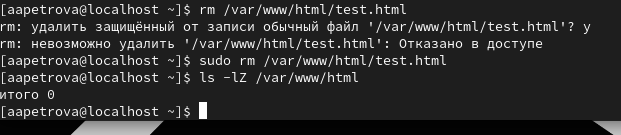
Проверка сервера

Возвращаю в файле /etc/httpd/httpd.conf порт 80, вместо 81. Проверяю, что порт 81 удален, это правда. (рис. [**fig:023?**]).



Проверка порта 81

Далее удаляю файл test.html, проверяю, что он удален(рис. [**fig:024?**]).



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.

# Список литературы