Отчёт по лабораторной работе №6  
Информационная безопасность

Мандатное разграничение прав в Linux

Выполнила: Сингх Ааруши ,  
НКАбд-02-23, 132215095

Содержание

# Цель работы

Развить навыки администрирования ОС Linux. Получить первое практическое знакомство с технологией SELinux1 . Проверить работу SELinx 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].

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:*

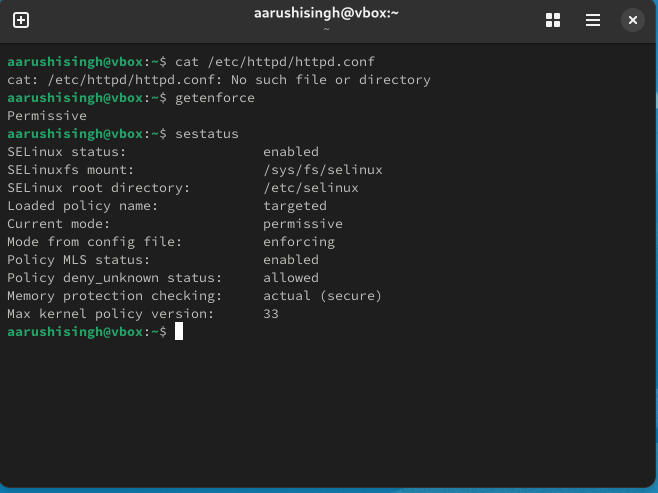
* чтобы открывать динамические PHP-страницы,
* для распределения поступающей на сервер нагрузки,
* для обеспечения отказоустойчивости сервера,
* чтобы потренироваться в настройке сервера и запуске PHP-скриптов.

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

Более подробно см. в [2].

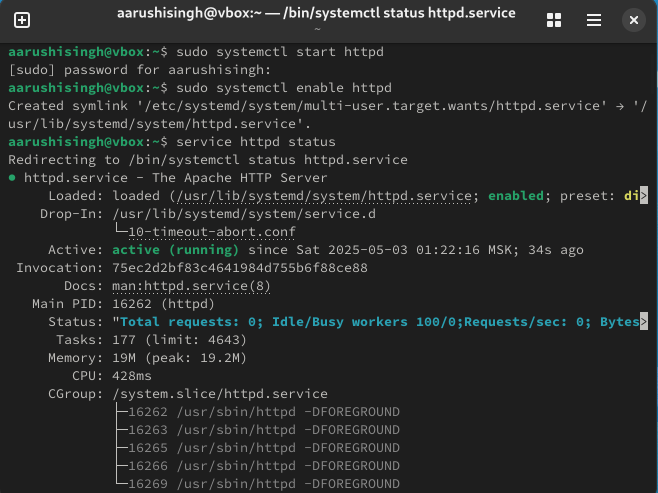
# Выполнение лабораторной работы

Вошли в систему под своей учетной записью и убедились, что SELinux работает в режиме enforcing политики targeted с помощью команд “getenforce” и “sestatus”



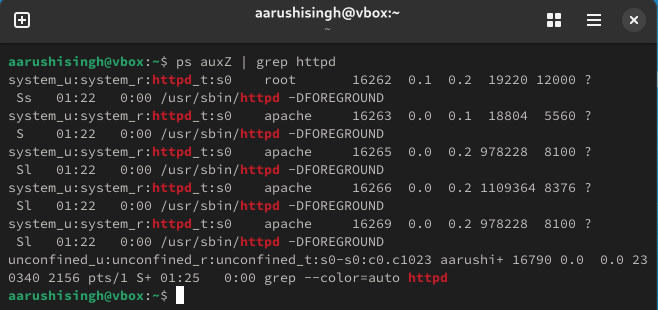
(рис. 1. Проверка режима enforcing политики targeted)

Обратились с помощью браузера к веб-серверу, запущенному на компьютере, и убедились, что последний работает с помощью команды “service httpd status”



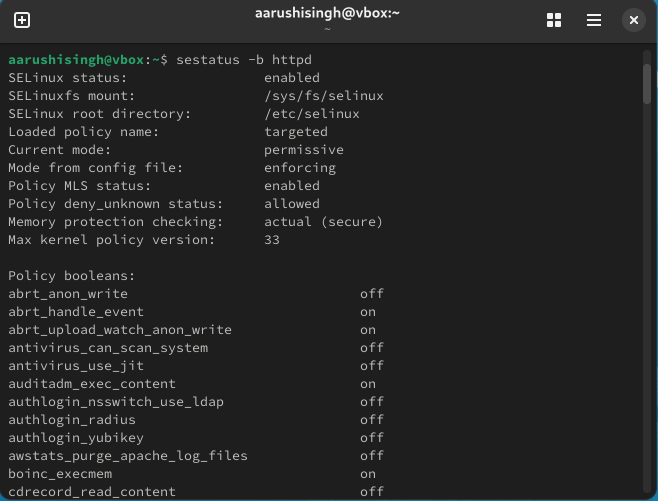
(рис. 2. Проверка работы веб-сервера)

С помощью команды “ps auxZ | grep httpd” определили контекст безопасности веб-сервера Apache - httpd\_t

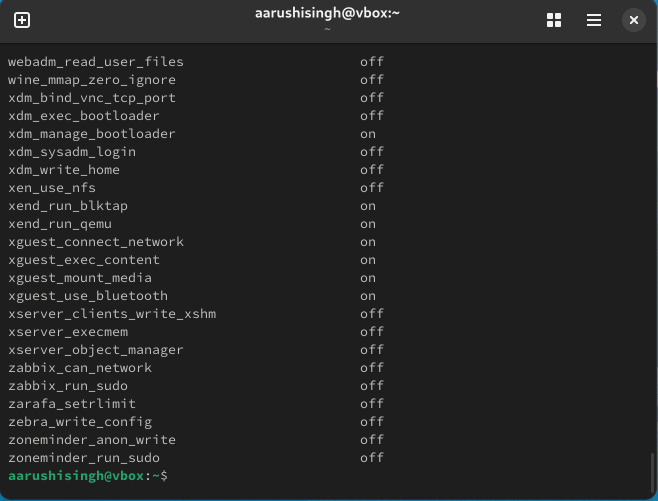


(рис. 3. Контекст безопасности веб-сервера Apache)

Посмотрели текущее состояние переключателей SELinux для Apache с помощью команды “sestatus -bigrep httpd”, многие из переключателей находятся в положении “off”

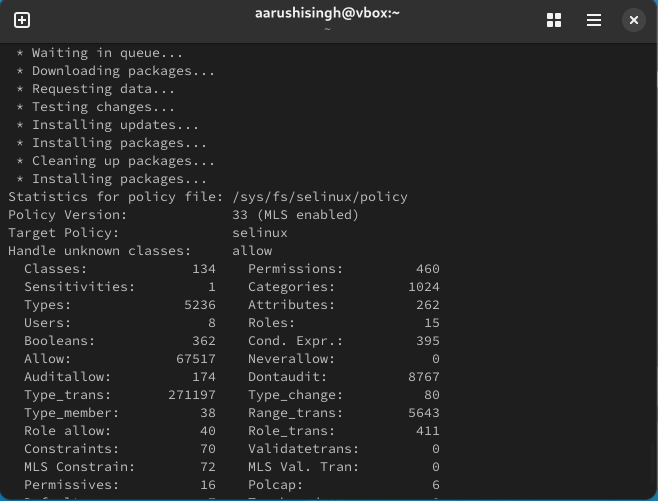


(рис. 4.1. Текущее состояние переключателей SELinux)



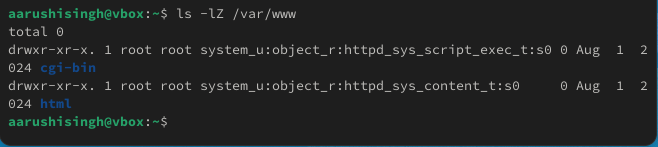
(рис. 4.2. Текущее состояние переключателей SELinux)

Посмотрели статистику по политике с помощью команды “seinfo”. Множество пользователей - 8, ролей - 14, типов 5100



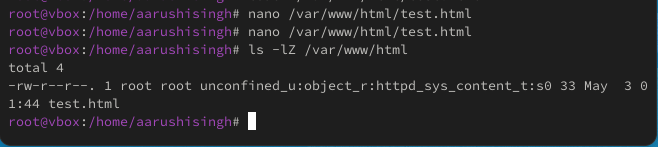
(рис. 5. Статистика по политике)

С помощью команды “ls -lZ /var/www” посмотрели файлы и поддиректории, находящиеся в директории /var/www. Используя команду “ls -lZ /var/www/html”, определили, что в данной директории файлов нет. Только владелец/суперпользователь может создавать файлы в директории /var/www/html



(рис. 6. Просмотр файлов и поддиректориий в директории /var/www)

От имени суперпользователя создали html-файл /var/www/html/test.html. Контекст созданного файла - httpd\_sys\_content\_t



(рис. 7. Создание файла /var/www/html/test.html)

Обратились к файлу через веб-сервер, введя в браузере адрес “http://127.0.0.1/test.html”. Файл был успешно отображен

|  |
| --- |
| (рис. 8. Обращение к файлу через веб-сервер) |

(рис. 8. Обращение к файлу через веб-сервер)

Изучив справку man httpd\_selinux, выяснили, 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). Изменили контекст файла на samba\_share\_t командой “sudo chcon -t samba\_share\_t/var/www/html/test.html” и проверили, что контекст поменялся



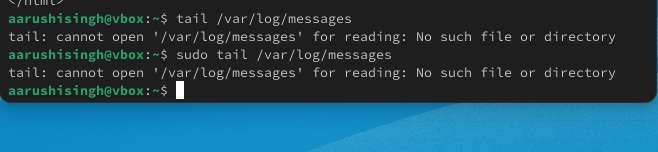
(рис. 9. Изменение контекста)

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

|  |
| --- |
| (рис. 10. Обращение к файлу через веб-сервер) |

(рис. 10. Обращение к файлу через веб-сервер)

Командой “ls -l /var/www/html/test.html” убедились, что читать данный файл может любой пользователь. Просмотрели системный лог-файл веб-сервера Apache командой “sudo tail /var/log/messages”, отображающий ошибки



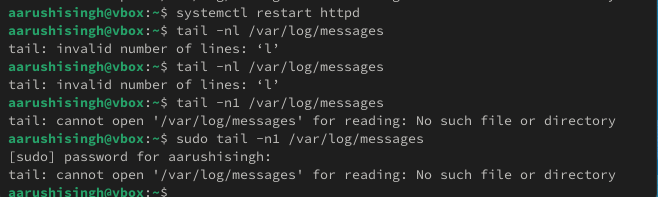
(рис. 11. Просмотр log-файла)

В файле /etc/httpd/conf/httpd.conf заменили строчку “Listen 80” на “Listen 81”, чтобы установить веб-сервер Apache на прослушивание TCP-порта 81

|  |
| --- |
| (рис. 12. Установка веб-сервера Apache на прослушивание TCP-порта 81) |

(рис. 12. Установка веб-сервера Apache на прослушивание TCP-порта 81)

Перезапускаем веб-сервер Apache и анализируем лог-файлы командой “tail -nl /var/log/messages”



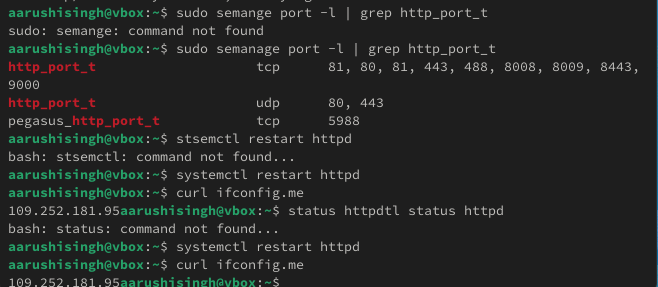
(рис. 13. Перезапуск веб-сервера и анализ лог-файлов)

Просмотрели файлы “var/log/http/error\_log”, “/var/log/http/access\_log” и “/var/log/audit/audit.log” и выяснили, что запись появилась в последнем файле

|  |
| --- |
| (рис. 14. Содержание файла var/log/audit/audit.log) |

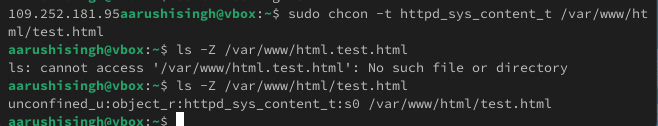
(рис. 14. Содержание файла var/log/audit/audit.log)

Выполнили команду “semanage port -a -t http\_port\_t -р tcp 81” и убедились, что порт TCP-81 установлен. Проверили список портов командой “semanage port -l | grep http\_port\_t”, убедились, что порт 81 есть в списке и запускаем веб-сервер Apache снова



(рис. 15. Проверка установки порта 81)

Вернули контекст “httpd\_sys\_cоntent\_t” файлу “/var/www/html/test.html” командой “chcon -t httpd\_sys\_content\_t /var/www/html/test.html” и после этого попробовали получить доступ к файлу через веб-сервер, введя адрес “http://127.0.0.1:81/test.html”, в результате чего увидели содежимое файла - слово “test”

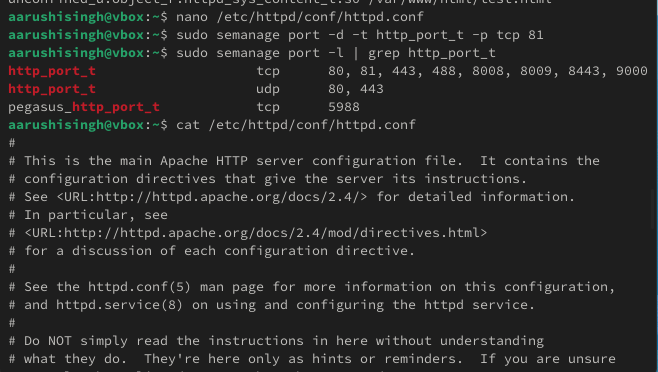


(рис. 16. Возвращение исходного контекста файлу)

|  |
| --- |
| (рис. 17. Обращение к файлу через веб-сервер) |

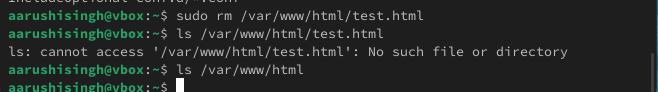
(рис. 17. Обращение к файлу через веб-сервер)

Исправили обратно конфигурационный файл apache, вернув “Listen 80”. Попытались 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



(рис. 18. Возвращение Listen 80 и попытка удалить порт 81)

Удалили файл “/var/www/html/test.html” командой “rm /var/www/html/test.html”



(рис. 19. Удаление файла test.html)

# Вывод

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. Библиография

[0] Методические материалы курса

[1] SELinux: https://habr.com/ru/companies/kingservers/articles/209644/

[2] Apache: https://2domains.ru/support/vps-i-servery/shto-takoye-apache