Отчет по лабораторной работе №7

Основы информационной безопасности

Чванова Ангелина Дмитриевна

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 3 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.

## 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.

# Выполнение лабораторной работы

Для начала был выполен вход в систему под своей учетной записью. После чего необходимо было проверить, что SELinux работает в режиме enforcing политики targeted с помощью команд getenforce и sestatus (рис. [-@fig:001]).



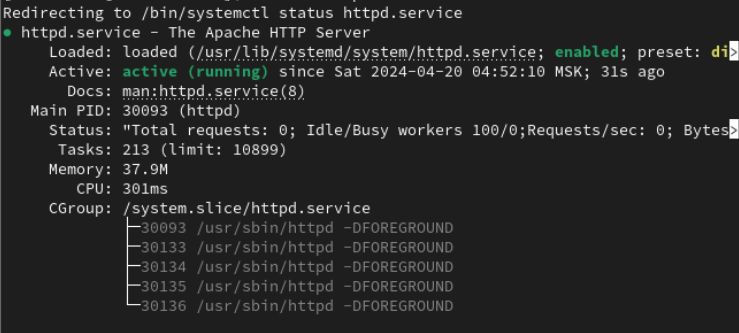
проверка режима работы SELinux

Запускаем сервер apache, далее обращаемсь 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

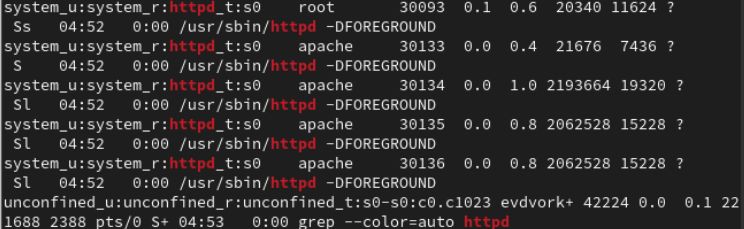
Проверка работы Apache

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



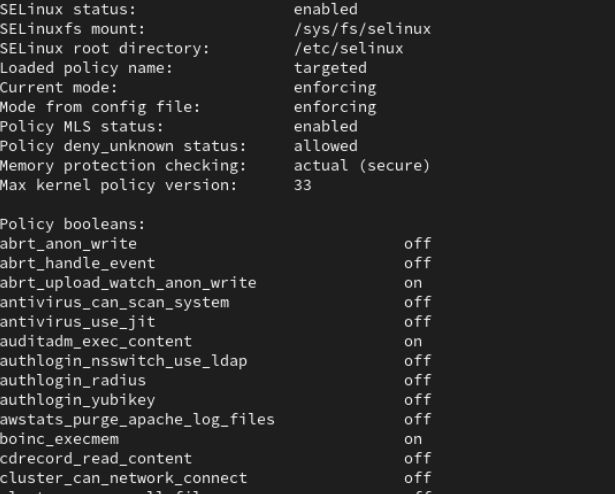
Контекст безопасности Apache

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



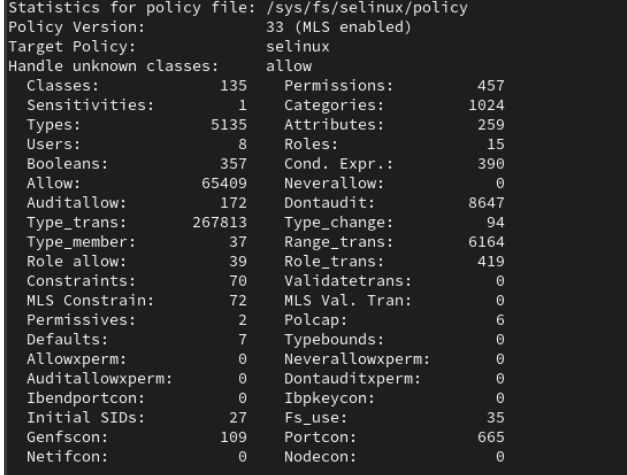
Состояние переключателей SELinux

Просмотрим 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

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]).

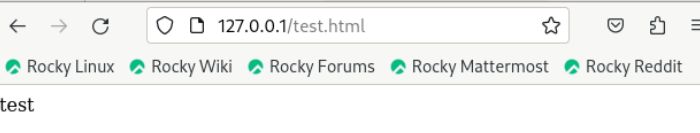
Создание файла

Создание файла

Проверяем контекст созданного файла. По умолчанию это httpd\_sys\_content\_t (рис. [-@fig:009]),(рис. [-@fig:010]).

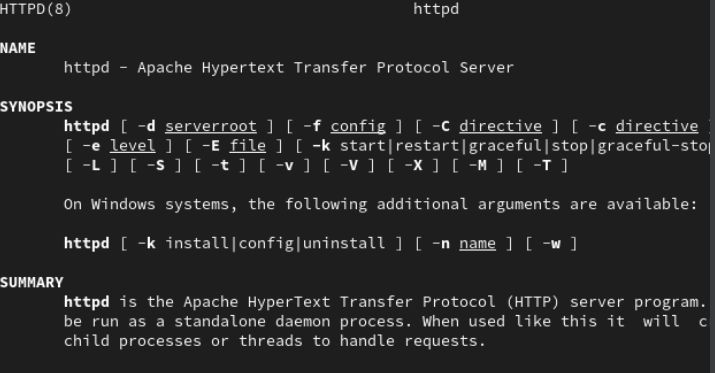
Контекст файла Контекст файла

Обращаемся к файлу через веб-сервер, введя в браузере адрес http://127.0.0.1/test.html. Файл успешно отображается (рис. [-@fig:011]).



Отображение файла

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:012]).



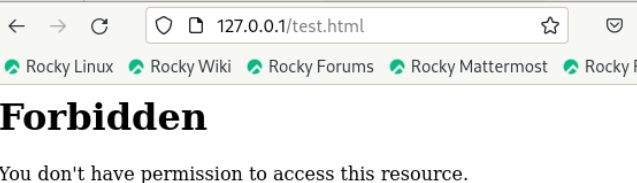
Изучение справки по команде

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. [-@fig:013]).

Изменение контекста

Изменение контекста

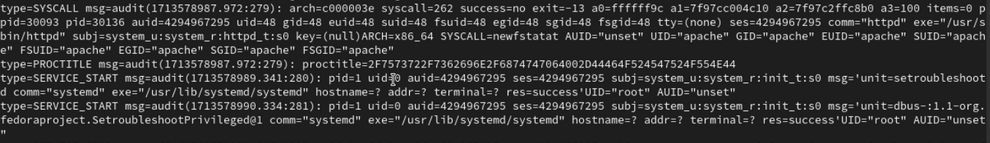
При попытке отображения файла в браузере получаем сообщение об ошибке (рис. [-@fig:014]).



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.

Просматриваем log-файлы веб-сервера Apache и системный лог-файл: tail /var/log/messages. Если в системе окажутся запущенными процессы setroubleshootd и audtd, то также можно увидеть ошибки, аналогичные указанным выше, в файле /var/log/audit/audit.log. (рис. [-@fig:015]).



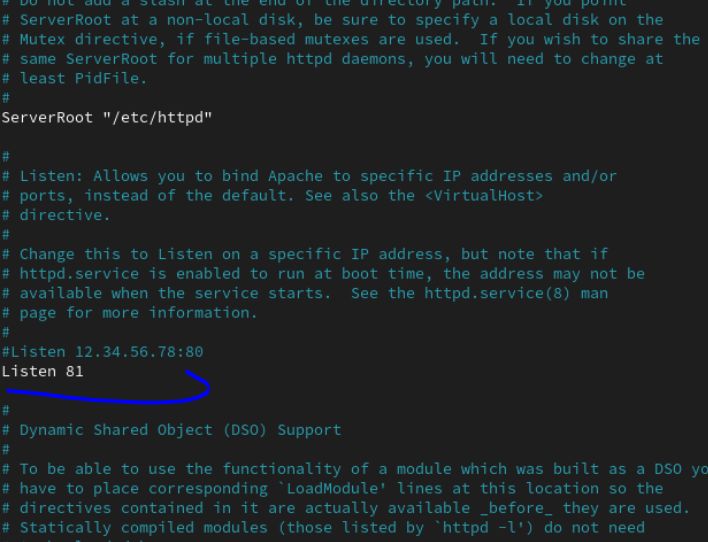
Попытка прочесть лог-файл

Чтобы запустить веб-сервер Apache на прослушивание ТСР-порта 81 (а не 80) открываем файл /etc/httpd/httpd.conf для изменения. (рис. [-@fig:016]).

Изменение файла

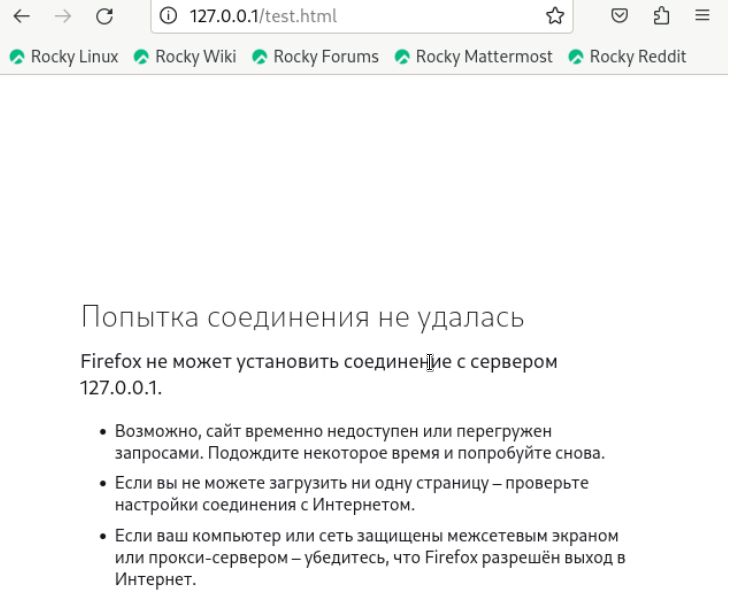
Изменение файла

Находим строчку Listen 80 и заменяем её на Listen 81. (рис. [-@fig:017]).



Изменение порта

После чего выполняем перезапуск веб-сервера Apache. Произошёл сбой, потому что порт 80 для локальной сети, а 81 нет (рис. [-@fig:018]).



Попытка прослушивания другого порта

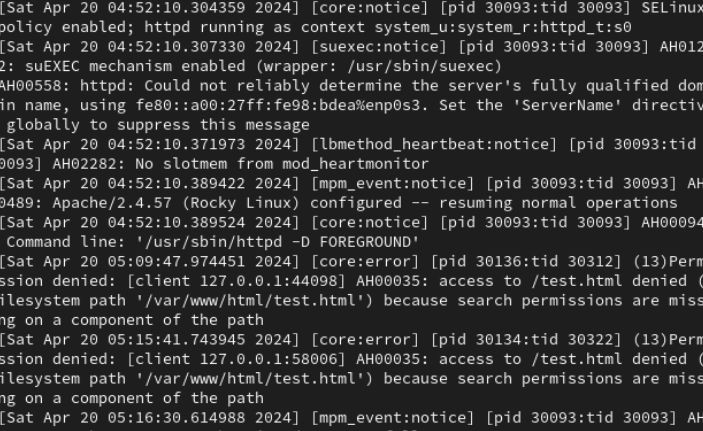
Проанализируем лог-файлы:

tail -nl /var/log/messages (рис. [-@fig:019]).

Проверка лог-файлов

Проверка лог-файлов

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. [-@fig:020]).



Проверка лог-файлов

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. [-@fig:021]).

Проверка портов

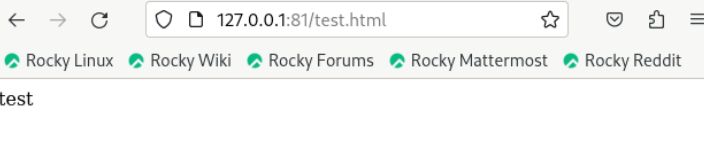
Проверка портов

Перезапускаем сервер Apache (рис. [-@fig:022]).

Перезапуск сервера

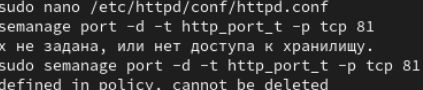
Перезапуск сервера

Теперь он работает, ведь порт 81 внечен в список портов htttpd\_port\_t (рис. [-@fig:023]).



Проверка сервера

Возвращаем в файле /etc/httpd/httpd.conf порт 80, вместо 81. Проверяем, что порт 81 удален (рис. [-@fig:024]).



Проверка порта 81

Далее чего удаляем файл test.html, проверяем, что он удален(рис. [-@fig:025]).

Удаление файла

Удаление файла

# 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

[1] Документация по Virtual Box: https://www.virtualbox.org/wiki/Documentation

[2] Документация по Git: https://git-scm.com/book/ru/v2

[3] Документация по Markdown: https://learn.microsoft.com/ru-ru/contribute/markdown-reference