Мандатное разграничение прав в Linux

Основы информационной безопасности

Симонова Полина Игоревна

Содержание

Список иллюстраций

Список таблиц

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**].

2. **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.

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**]).

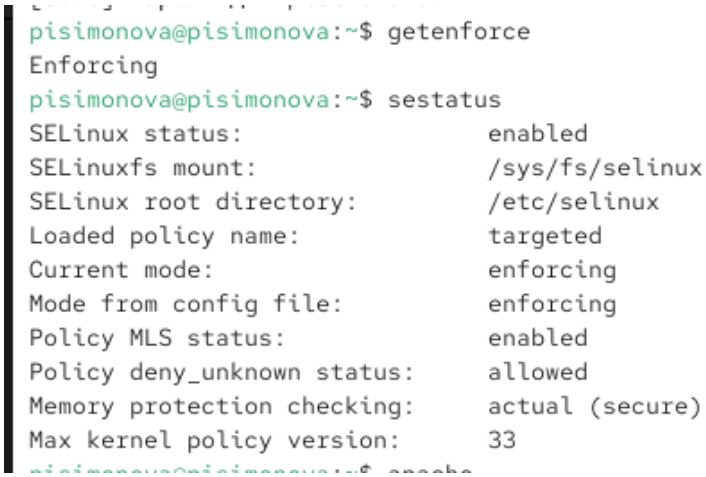

Рисунок 3.1: проверка режима работы SELinux

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**]).

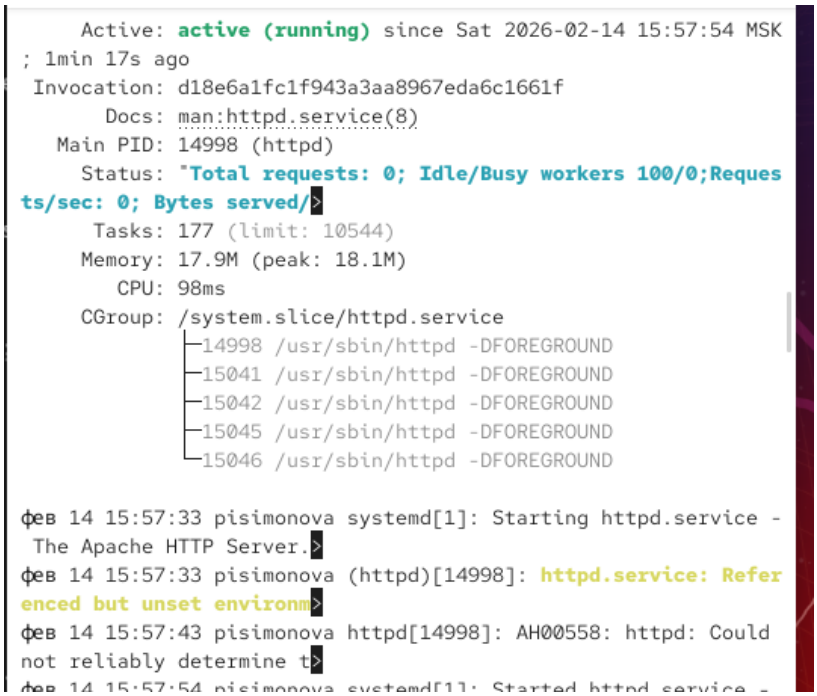

Рисунок 3.2: Проверка работы Apache

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

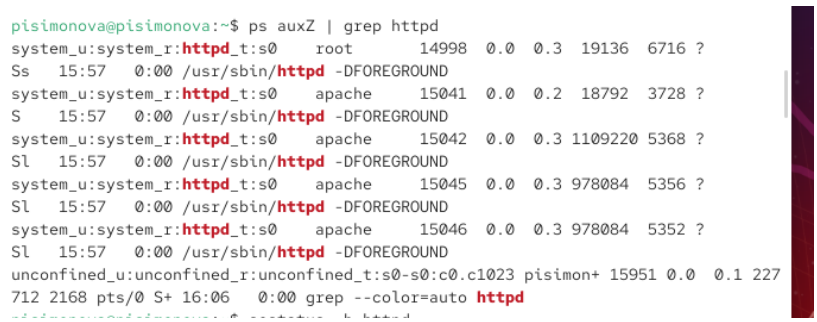

Рисунок 3.3: Контекст безопасности Apache

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

Рисунок 3.4: Состояние переключателей SELinux

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

```
Statistics for policy file: /sys/fs/selinux/policy
Policy Version:             33 (MLS enabled)
Target Policy:              selinux
Handle unknown classes:     allow
  Classes:             133    Permissions:         458
  Sensitivities:         1    Categories:         1024
  Types:              4699    Attributes:          251
  Users:                 8    Roles:                15
  Booleans:            325    Cond. Expr.:         353
  Allow:             59647    Neverallow:            0
  Auditallow:          158    Dontaudit:          7974
  Type_trans:       246125    Type_change:          66
  Type_member:          34    Range_trans:        3893
  Role allow:           39    Role_trans:          318
  Constraints:          70    Validatetrans:         0
  MLS Constrain:        72    MLS Val. Tran:         0
  Permissives:          24    Polcap:                6
  Defaults:              7    Typebounds:            0
  Allowxperm:            0    Neverallowxperm:       0
  Auditallowxperm:       0    Dontauditxperm:        0
  Ibendportcon:          0    Ibpkeycon:             0
  Initial SIDs:         27    Fs_use:               35
  Genfscon:            112    Portcon:             667
  Netifcon:              0    Nodecon:               0
```

Рисунок 3.5: Статистика 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**]).

```
По команде «ls --help» можно получить дополнительную информацию.
pisimonova@pisimonova:~$ ls -lZ /var/www
итого 0
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_script_exec_t:s0 6 дек 10 03:
00 cgi-bin
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0     6 дек 10 03:
00 html
```

Рисунок 3.6: Типы поддиректорий

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

```
pisimonova@pisimonova:~$ ls -lZ /var/www/html
итого 0
```

Рисунок 3.7: Типы файлов

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

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

Листинг 1 (рис. [**fig:008**]).

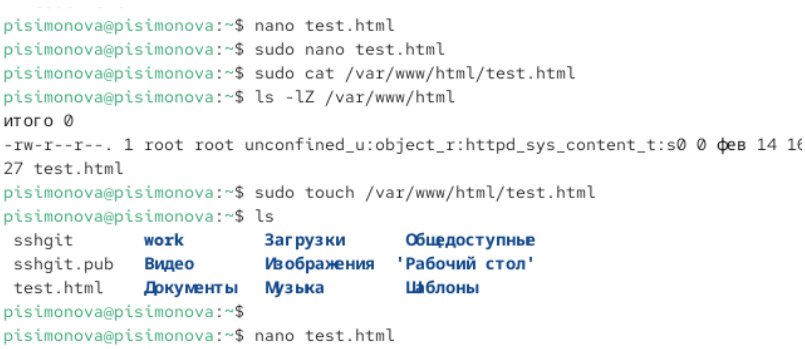

Рисунок 3.8: Создание файла

Проверяю контекст созданного файла. По умолчанию это httpd_sys_content_t (рис. [**fig:009**]).

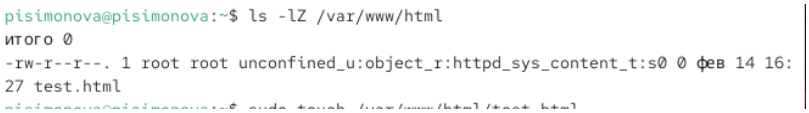

Рисунок 3.9: Контекст файла

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

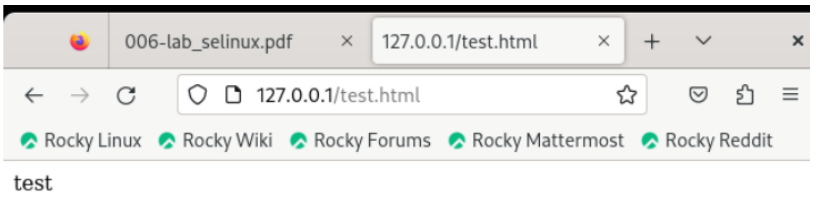

Рисунок 3.10: Отображение файла

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 /proc 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**]).

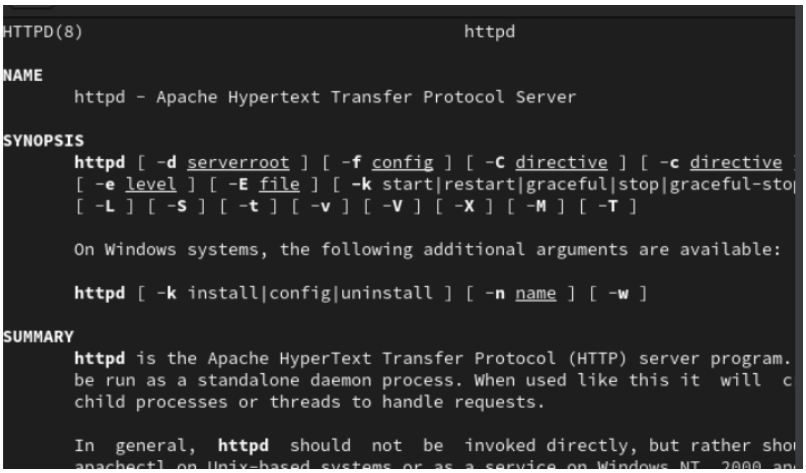

Рисунок 3.11: Изучение справки по команде

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, ⌧⌧ `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**]).

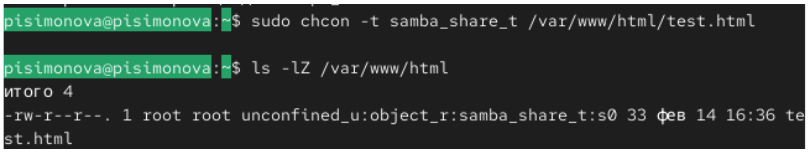

Рисунок 3.12: Изменение контекста

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

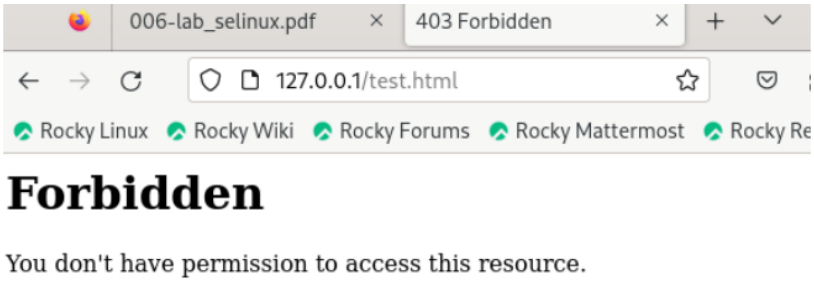

Рисунок 3.13: Отображение файла

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.

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**]).

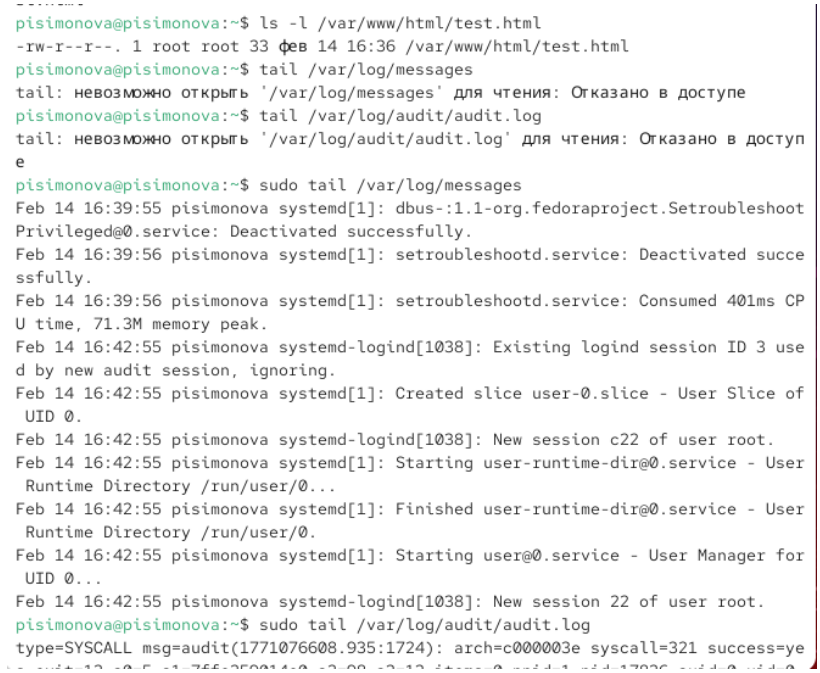

Рисунок 3.14: Попытка прочесть лог-файл

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

Рисунок 3.15: Изменение файла

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

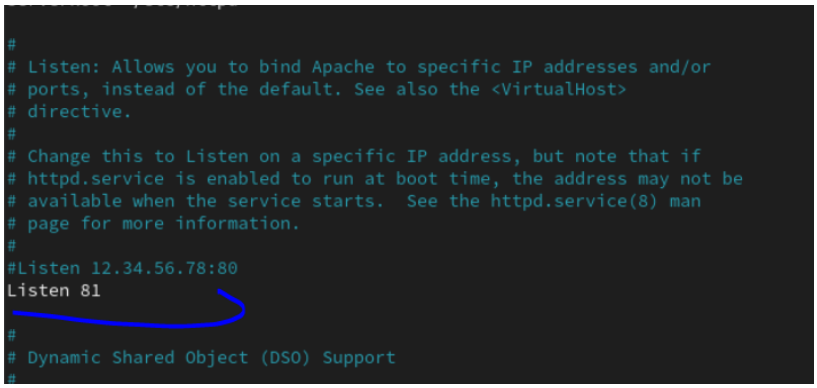

Рисунок 3.16: Изменение порта

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

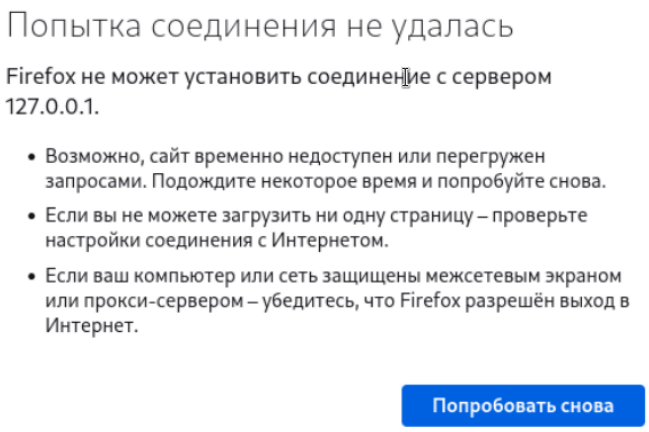

Рисунок 3.17: Попытка прослушивания другого порта

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

Рисунок 3.18: 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**]).

```
pisimonova@pisimonova:~$ semanage port -l | grep http_port_t
ValueError: Политика SELinux не задана, или нет доступа к хранилищу.
pisimonova@pisimonova:~$ sudo nano /etc/httpd/httpd.conf
pisimonova@pisimonova:~$ sudo semanage port -a -t http_port_t -p tcp 81

usage: semanage [-h]
                {import,export,login,user,port,ibpkey,ibendport,interface,module,
node,fcontext,boolean,permissive,dontaudit}
                ...
semanage: error: unrecognized arguments: -p 81
pisimonova@pisimonova:~$ sudo semanage port -l | grep http_port_t
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
http_port_t                    udp      80, 443
pegasus_http_port_t            tcp      5988
pisimonova@pisimonova:~$ sudo systemctl resart httpd
Unknown command verb 'resart', did you mean 'restart'?
pisimonova@pisimonova:~$ sudo systemctl restart httpd
pisimonova@pisimonova:~$ sudo chcon -t httpd_sys_content_t /var/www/html/test.htm
l
```

Рисунок 3.19: Проверка лог-файлов

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

```
pisimonova@pisimonova:~$ sudo semanage port -a -t http_port_t -p tcp 81

usage: semanage [-h]
                {import,export,login,user,port,ibpkey,ibendport,interface,module,
node,fcontext,boolean,permissive,dontaudit}
                ...
semanage: error: unrecognized arguments: -p 81
pisimonova@pisimonova:~$ sudo semanage port -l | grep http_port_t
http_port_t                    tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
http_port_t                    udp      80, 443
pegasus_http_port_t            tcp      5988
pisimonova@pisimonova:~$ sudo systemctl resart httpd
Unknown command verb 'resart', did you mean 'restart'?
pisimonova@pisimonova:~$ sudo systemctl restart httpd
```

Рисунок 3.20: Проверка портов

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

```
pisimonova@pisimonova:~$ sudo systemctl restart httpd
pisimonova@pisimonova:~$ sudo chcon -t httpd_sys_content_t /var/www/html/test.htm
l
pisimonova@pisimonova:~$ sudo systemctl restart httpd
```

Рисунок 3.21: Перезапуск сервера

Теперь он работает, ведь мы внесли порт 81 в список портов `httpd_port_t` (рис. [**fig:022**]).

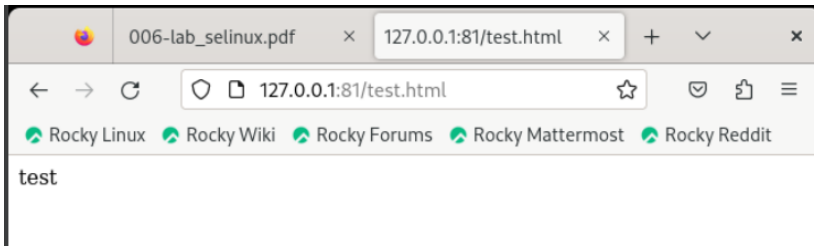

Рисунок 3.22: Проверка сервера

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

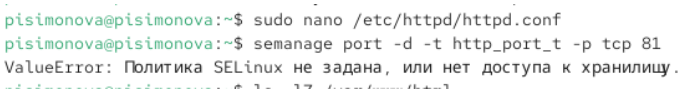

Рисунок 3.23: Проверка порта 81

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

Рисунок 3.24: 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.

5 Список литературы

[0] Методические рекомендации курса