

# **Отчет по лабораторной работе №6**

**Основы информационной безопасности**

ФЕДОРОВ Андрей, НБИбд-01-22

# Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	8
4	Выводы	18
	Список литературы	19

## Список иллюстраций

3.1	проверка режима работы SELinux . . . . .	8
3.2	Проверка работы Apache . . . . .	9
3.3	Контекст безопасности Apache . . . . .	9
3.4	Состояние переключателей SELinux . . . . .	10
3.5	Статистика по политике . . . . .	10
3.6	Типы поддиректорий . . . . .	11
3.7	Типы файлов . . . . .	11
3.8	Создание файла . . . . .	11
3.9	Контекст файла . . . . .	11
3.10	Отображение файла . . . . .	12
3.11	Изучение справки по команде . . . . .	12
3.12	Изменение контекста . . . . .	13
3.13	Отображение файла . . . . .	13
3.14	Попытка прочесть лог-файл . . . . .	13
3.15	Изменение файла . . . . .	14
3.16	Изменение порта . . . . .	14
3.17	Попытка прослушивания другого порта . . . . .	14
3.18	Проверка лог-файлов . . . . .	15
3.19	Проверка лог-файлов . . . . .	15
3.20	Проверка портов . . . . .	16
3.21	Перезапуск сервера . . . . .	16
3.22	Проверка сервера . . . . .	16
3.23	Проверка порта 81 . . . . .	17
3.24	Удаление файла . . . . .	17

## **Список таблиц**

# 1 Цель работы

Развить навыки администрирования ОС Linux. Получить первое практическое знакомство с технологией SELinux<sup>1</sup>. 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?].

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.

Более подробно см. в [s?].

### 3 Выполнение лабораторной работы

Вошёл в систему под своей учетной записью. Убедился, что SELinux работает в режиме enforcing политики targeted с помощью команд `getenforce` и `sestatus` (рис. 3.1).

```
[afedorov@localhost ~]$ getenforce
Permissive
[afedorov@localhost ~]$ setstatus
bash: setstatus: команда не найдена...
[afedorov@localhost ~]$ sestatus
SELinux status:                enabled
SELinuxfs mount:                /sys/fs/selinux
SELinux root directory:         /etc/selinux
Loaded policy name:              targeted
Current mode:                    permissive
Mode from config file:           enforcing
Policy MLS status:               enabled
Policy deny_unknown status:      allowed
Memory protection checking:      actual (secure)
Max kernel policy version:       33
```

Рис. 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. 3.2).



```
the HTTP Server
lib/systemd/system/httpd.service; enabled; preset: c
ng) since Tue 2024-06-18 00:42:09 MSK; 40s ago
vice(8)
Main PID: 47142 (httpd)
Status: "Total requests: 0; Idle/Busy workers 100/0;Requests/sec: 0; Byte
Tasks: 177 (limit: 10964)
Memory: 26.4M
CPU: 91ms
CGroup: /system.slice/httpd.service
├─47142 /usr/sbin/httpd -DFOREGROUND
├─47143 /usr/sbin/httpd -DFOREGROUND
├─47144 /usr/sbin/httpd -DFOREGROUND
├─47145 /usr/sbin/httpd -DFOREGROUND
└─47146 /usr/sbin/httpd -DFOREGROUND

июн 18 00:42:09 localhost.localdomain systemd[1]: Starting The Apache HTTP Ser
июн 18 00:42:09 localhost.localdomain httpd[47142]: AH00558: httpd: Could not
июн 18 00:42:09 localhost.localdomain httpd[47142]: Server configured, listen
июн 18 00:42:09 localhost.localdomain systemd[1]: Started The Apache HTTP Ser
```

Рис. 3.2: Проверка работы Apache

С помощью команды `ps auxZ | grep httpd` нашёл веб-сервер Apache в списке процессов. Его контекст безопасности - `httpd_t` (рис. 3.3).

```
[afedorov@localhost ~]$ ps auxZ | grep httpd
system_u:system_r:httpd_t:s0 root 47142 0.0 0.6 20148 11424 ?
Ss 00:42 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 47143 0.0 0.3 22028 7100 ?
S 00:42 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 47144 0.0 0.7 2095624 13044 ?
Sl 00:42 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 47145 0.0 0.6 1964488 10948 ?
Sl 00:42 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 47146 0.0 0.5 1964488 10848 ?
Sl 00:42 0:00 /usr/sbin/httpd -DFOREGROUND
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 afedorov 47390 0.0 0.1 22
1688 2432 pts/0 S+ 00:43 0:00 grep --color=auto httpd
```

Рис. 3.3: Контекст безопасности Apache

Просмотрел текущее состояние переключателей SELinux для Apache с помощью команды `sestatus -bigrep httpd` (рис. 3.4).

```

xdm_manage_bootloader      on
xdm_sysadm_login           off
xdm_write_home             off
xen_use_nfs                off
xend_run_blkmap            on
xend_run_qemu              on
xguest_connect_network     on
xguest_exec_content        on
xguest_mount_media         on
xguest_use_bluetooth       on
xserver_clients_write_xshm off
xserver_execmem            off
xserver_object_manager     off
zabbix_can_network         off
zabbix_run_sudo            off
zarafa_setrlimit           off
zebra_write_config         off
zoneminder_anon_write     off
zoneminder_run_sudo       off
[afedorov@localhost ~]$

```

Рис. 3.4: Состояние переключателей SELinux

Просмотрел статистику по политике с помощью команды `seinfo`. Множество пользователей - 8, ролей - 39, типов - 5135. (рис. 3.5).

```

Booleans:      356      Cond. Expr.:      388
Allow:         65500    Neverallow:       0
Auditallow:    176     Dontaudit:        8682
Type_trans:    271770  Type_change:      94
Type_member:   37      Range_trans:      5931
Role allow:    40      Role_trans:       417
Constraints:   70     Validatetrans:    0
MLS Constrai: 72      MLS Val. Tran:    0
Permissives:   4      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:      109    Portcon:          665
Netifcon:      0      Nodecon:          0
[afedorov@localhost ~]$ ls -lZ

```

Рис. 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. 3.6).

```
[afedorov@localhost ~]$ ls -lZ /var/www
итого 0
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_script_exec_t:s0 6 апр 22 04
:04 cgi-bin
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0 6 апр 22 04
:04 html
```

Рис. 3.6: Типы поддиректорий

В директории /var/www/html нет файлов. (рис. 3.7).

```
[afedorov@localhost ~]$ ls -lZ /var/www/html
итого 0
```

Рис. 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>
```

(рис. 3.8).

```
[afedorov@localhost ~]$ sudo touch /var/www/html/test.html
[afedorov@localhost ~]$ sudo nano /var/www/html/test.html
[afedorov@localhost ~]$ sudo cat /var/www/html/test.html
<html>
<body>test</body>
</html>
```

Рис. 3.8: Создание файла

Проверяю контекст созданного файла. По умолчанию это httpd\_sys\_content\_t (рис. 3.9).

```
[afedorov@localhost ~]$ ls -lZ /var/www/html/
итого 4
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_content_t:s0 33 июн 18 0
0:48 test.html
[afedorov@localhost ~]$
```

Рис. 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. 3.10).

```
[afedorov@localhost ~]$ ls -lZ /var/www/html/
итого 4
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_content_t:s0 33 июн 18 0
0:48 test.html
[afedorov@localhost ~]$
```

Рис. 3.10: Отображение файла

Изучил 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 `/rpgos` 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. 3.11).

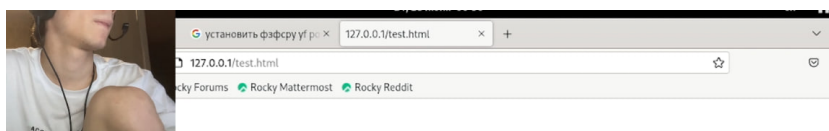


Рис. 3.11: Изучение справки по команде

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,

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. 3.12).

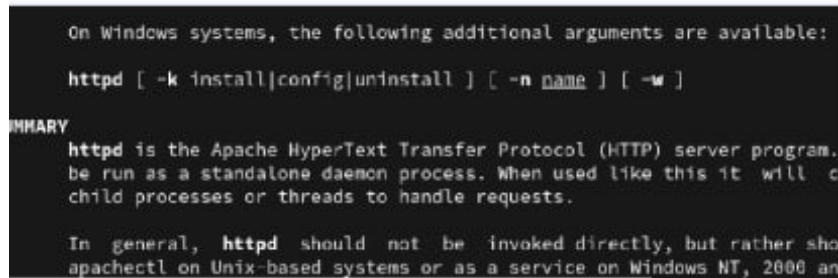


Рис. 3.12: Изменение контекста

При попытке отображения файла в браузере получаем сообщение об ошибке (рис. 3.13).



Рис. 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. 3.14).

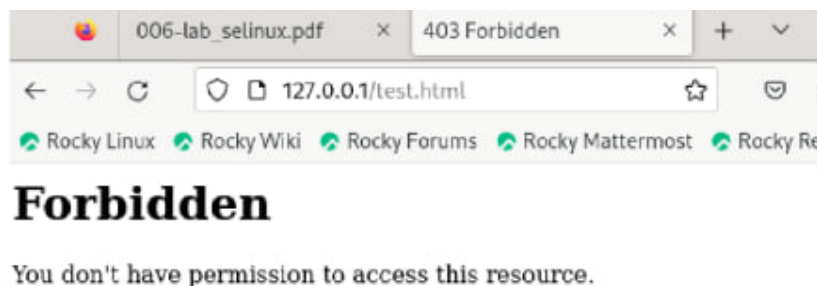


Рис. 3.14: Попытка прочесть лог-файл

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. 3.15).

```
systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success" UID="root" AUID="unset"
type=USER_ACCT msg=audit(1718661302.783:1502): pid=47899 uid=1000 auid=1000 ses=
13 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:accounting grantors=pam_unix,pam_localuser acct="afedorov" exe="/usr/bin/sudo" hostname=? addr=? terminal=/dev/pts/0 res=success' UID="afedorov" AUID="afedorov"
type=USER_CMD msg=audit(1718661302.783:1503): pid=47899 uid=1000 auid=1000 ses=13 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='cwd="/home/afedorov" cmd=7461696C202F7661722F6C6F672F61756469742F61756469742E6C6F67 exe="/usr/bin/sudo" terminal=pts/0 res=success' UID="afedorov" AUID="afedorov"
type=CRED_REFR msg=audit(1718661302.783:1504): pid=47899 uid=1000 auid=1000 ses=13 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:setcred grantors=pam_env,pam_fprintd acct="root" exe="/usr/bin/sudo" hostname=? addr=? terminal=/dev/pts/0 res=success' UID="afedorov" AUID="afedorov"
type=USER_START msg=audit(1718661302.785:1505): pid=47899 uid=1000 auid=1000 ses=13 subj=unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 msg='op=PAM:session_open grantors=pam_keyinit,pam_limits,pam_systemd,pam_unix acct="root" exe="/usr/bin/sudo" hostname=? addr=? terminal=/dev/pts/0 res=success' UID="afedorov" AUID="afedorov"
```

Рис. 3.15: Изменение файла

Нахожу строчку Listen 80 и заменяю её на Listen 81. (рис. 3.16).

```
-- --
[afedorov@localhost ~]$ sudo nano /etc/httpd/conf/httpd.conf
```

Рис. 3.16: Изменение порта

Выполняю перезапуск веб-сервера Apache. Произошёл сбой, потому что порт 80 для локальной сети, а 81 нет (рис. 3.17).

```
# page for more information.
#
#Listen 12.34.56.78:80
Listen 81

#
# Dynamic Shared Object (DSO) Support
#
# To be able to use the functionality of a module which was built as a DSO you
# have to place corresponding 'LoadModule' lines at this location so the
# directives contained in it are actually available _before_ they are used.
# Statically compiled modules (those listed by 'httpd -l') do not need
```

Рис. 3.17: Попытка прослушивания другого порта

Проанализируйте лог-файлы: tail -nl /var/log/messages (рис. 3.18).



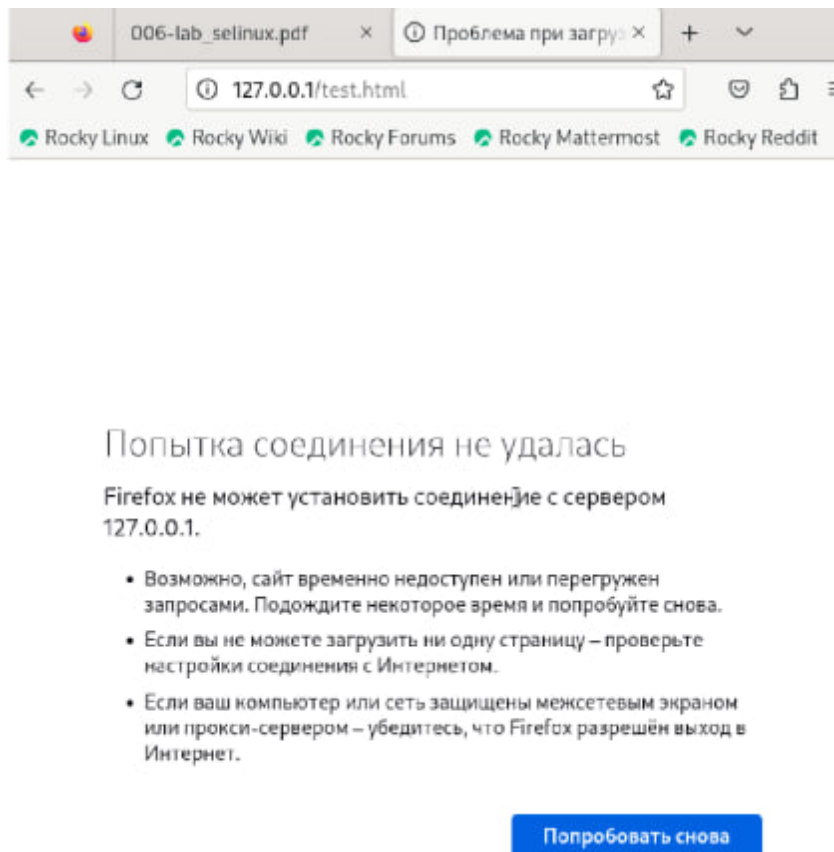


Рис. 3.18: Проверка лог-файлов

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 файле `error_log` (рис. 3.19).

```
[afedorov@localhost ~]$ sudo tail -n1 /var/log/messages
Jun 18 01:03:25 localhost systemd[1]: setroubleshootd.service: Deactivated successfully.
```

Рис. 3.19: Проверка лог-файлов

Выполняю команду `semanage port -a -t http_port_t -p tcp 81` После этого проверяю список портов командой `semanage port -l | grep http_port_t` Порт 81 появился в списке (рис. 3.20).

```
[Tue Jun 18 01:03:10.021512 2024] [core:error] [pid 47145:tid 47279] (13)Permission denied: [client 127.0.0.1:40368] AH00035: access to /test.html denied (filesystem path '/var/www/html/test.html') because search permissions are missing on a component of the path
[Tue Jun 18 01:03:11.047275 2024] [core:error] [pid 47145:tid 47280] (13)Permission denied: [client 127.0.0.1:40368] AH00035: access to /test.html denied (filesystem path '/var/www/html/test.html') because search permissions are missing on a component of the path
[Tue Jun 18 01:03:11.855907 2024] [core:error] [pid 47145:tid 47281] (13)Permission denied: [client 127.0.0.1:40368] AH00035: access to /test.html denied (filesystem path '/var/www/html/test.html') because search permissions are missing on a component of the path
[Tue Jun 18 01:03:14.669982 2024] [core:error] [pid 47145:tid 47282] (13)Permission denied: [client 127.0.0.1:40368] AH00035: access to /test.html denied (filesystem path '/var/www/html/test.html') because search permissions are missing on a component of the path
[afedorov@localhost ~]$ sudo semanage port -a
```

Рис. 3.20: Проверка портов

Перезапускаю сервер Apache (рис. 3.21).

```
[afedorov@localhost ~]$ sudo semanage port -a -t http_port_t -p tcp 81
Port tcp/81 already defined, modifying instead
[afedorov@localhost ~]$ semanage port -l | grep http_port_t
ValueError: Политика SELinux не задана, или нет доступа к хранилищу.
[afedorov@localhost ~]$ sudo semanage port -l | grep http_port_t
http_port_t      tcp      81, 80, 81, 443, 488, 8008, 8009, 8443, 9000
nagios_http_port_t tcp      5088
```

Рис. 3.21: Перезапуск сервера

Теперь он работает, ведь мы внесли порт 81 в список портов httpd\_port\_t (рис. 3.22).

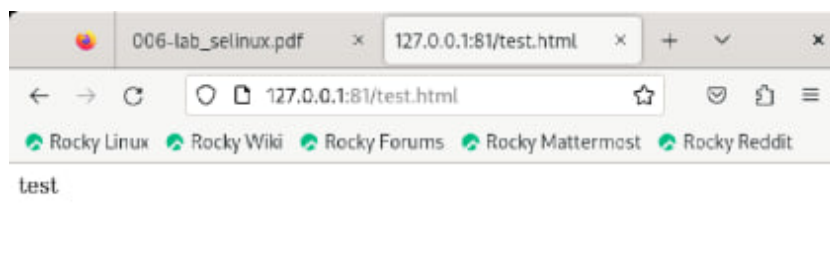


Рис. 3.22: Проверка сервера

Возвращаю в файле /etc/httpd/httpd.conf порт 80, вместо 81. Проверяю, что порт 81 удален, это правда. (рис. 3.23).



```

[afedorov@localhost ~]$ sudo systemctl restart httpd
[afedorov@localhost ~]$ sudo chcon -t httpd_sys_content_t /var/www/html/test.h
[afedorov@localhost ~]$ sudo systemctl restart httpd
[afedorov@localhost ~]$ sudo systemctl restart httpd
[afedorov@localhost ~]$ sudo nano /etc/httpd/conf/ht

```

Рис. 3.23: Проверка порта 81

Далее удаляю файл test.html, проверяю, что он удален(рис. 3.24).

```

[afedorov@localhost ~]$ ls /var/www/html
test.html
[afedorov@localhost ~]$ cd /var/www/html
[afedorov@localhost html]$ rm test.html
rm: удалить защищенный от записи обычный файл 'test.html'? y
[afedorov@localhost html]$ sudo rm test.html
[afedorov@localhost html]$ cd
[afedorov@localhost ~]$ cd /var/www/html

```

Рис. 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.

## **Список литературы**