

Отчет по лабораторной работе №7

Основы информационной безопасности

Чванова Ангелина Дмитриевна

Содержание

Цель работы	5
Теоретическое введение	6
SELinux (Security-Enhanced Linux)	6
Apache	7
Выполнение лабораторной работы	8
Выводы	19
Список литературы	20

Список иллюстраций

1	проверка режима работы SELinux	8
2	Проверка работы Apache	8
3	Контекст безопасности Apache	9
4	Состояние переключателей SELinux	9
5	Статистика по политике	10
6	Типы поддиректорий	11
7	Типы файлов	11
8	Создание файла	11
9	Отображение файла	12
10	Изучение справки по команде	13
11	Изменение контекста	13
12	Отображение файла	13
13	Попытка прочесть лог-файл	14
14	Изменение файла	14
15	Изменение порта	15
16	Попытка прослушивания другого порта	16
17	Проверка лог-файлов	16
18	Проверка лог-файлов	17
19	Проверка портов	17
20	Перезапуск сервера	17
21	Проверка сервера	18
22	Проверка порта 81	18
23	Удаление файла	18

Список таблиц

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 выполнен 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]).

```
[root@localhost ~]# getenforce
Permissive
[root@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
[root@localhost ~]# _
```

Рис. 1: проверка режима работы SELinux

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

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

Рис. 2: 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]).


```
Redirecting to /bin/systemctl status httpd.service
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: active (running) since Sat 2024-04-20 04:52:10 MSK; 31s ago
     Docs: man:httpd.service(8)
  Main PID: 30093 (httpd)
    Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0"
    Tasks: 213 (limit: 10899)
   Memory: 37.9M
      CPU: 301ms
   CGroup: /system.slice/httpd.service
           └─30093 /usr/sbin/httpd -DFOREGROUND
             └─30133 /usr/sbin/httpd -DFOREGROUND
               └─30134 /usr/sbin/httpd -DFOREGROUND
                 └─30135 /usr/sbin/httpd -DFOREGROUND
                   └─30136 /usr/sbin/httpd -DFOREGROUND
```

Рис. 3: Контекст безопасности Apache

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

```
system_u:system_r:httpd_t:s0 root 30093 0.1 0.6 20340 11624 ?
Ss 04:52 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 30133 0.0 0.4 21676 7436 ?
S 04:52 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 30134 0.0 1.0 2193664 19320 ?
Sl 04:52 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 30135 0.0 0.8 2062528 15228 ?
Sl 04:52 0:00 /usr/sbin/httpd -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 30136 0.0 0.8 2062528 15228 ?
Sl 04:52 0:00 /usr/sbin/httpd -DFOREGROUND
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 evdwork+ 42224 0.0 0.1 22
1688 2388 pts/0 S+ 04:53 0:00 grep --color=auto httpd
```

Рис. 4: Состояние переключателей SELinux

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

```

SELinux status:                enabled
SELinuxfs mount:              /sys/fs/selinux
SELinux root directory:      /etc/selinux
Loaded policy name:          targeted
Current mode:                enforcing
Mode from config file:      enforcing
Policy MLS status:          enabled
Policy deny_unknown status:  allowed
Memory protection checking:  actual (secure)
Max kernel policy version:   33

Policy booleans:
abrt_anon_write                off
abrt_handle_event              off
abrt_upload_watch_anon_write  on
antivirus_can_scan_system     off
antivirus_use_jit              off
auditadm_exec_content          on
authlogin_nsswitch_use_ldap    off
authlogin_radius               off
authlogin_yubikey              off
awstats_purge_apache_log_files off
boinc_execmem                  on
cdrecord_read_content           off
cluster_can_network_connect    off

```

Рис. 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]).

```
Statistics for policy file: /sys/fs/selinux/policy
```

Policy Version:	33 (MLS enabled)
Target Policy:	selinux
Handle unknown classes:	allow

Classes:	135	Permissions:	457
Sensitivities:	1	Categories:	1024
Types:	5135	Attributes:	259
Users:	8	Roles:	15
Booleans:	357	Cond. Expr.:	390
Allow:	65409	Neverallow:	0
Auditallow:	172	Dontaudit:	8647
Type_trans:	267813	Type_change:	94
Type_member:	37	Range_trans:	6164
Role allow:	39	Role_trans:	419
Constraints:	70	Validatetrans:	0
MLS Constrains:	72	MLS Val. Tran:	0
Permissives:	2	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

Рис. 6: Типы поддиректорий

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

```
итого 0
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_script_exec_t:s0 6 окт 28 12:35 cgi-bin
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0 6 окт 28 12:35 html
```

Рис. 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. [-@fig:008]).

```
sudo touch /var/www/html/test.html
```

Рис. 8: Создание файла

Проверяем контекст созданного файла. По умолчанию это `httpd_sys_content_t` (рис. [-@fig:009]), (рис. [-@fig:010]).

```
sudo nano /var/www/html/test.html
sudo cat /var/www/html/test.html
```

итого 4
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_con

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

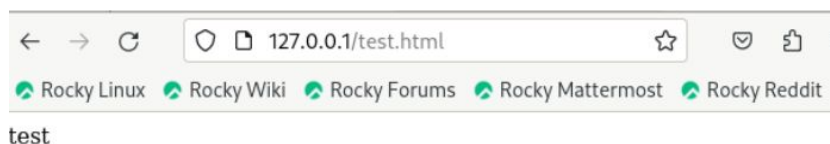


Рис. 9: Отображение файла

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 `/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. [-@fig:012]).

```
HTTPD(8)                                httpd

NAME
    httpd - Apache Hypertext Transfer Protocol Server

SYNOPSIS
    httpd [ -d serverroot ] [ -f config ] [ -C directive ] [ -c directive ]
    [ -e level ] [ -E file ] [ -k start|restart|graceful|stop|graceful-stop ]
    [ -L ] [ -S ] [ -t ] [ -v ] [ -V ] [ -X ] [ -M ] [ -T ]

    On Windows systems, the following additional arguments are available:

    httpd [ -k install|config|uninstall ] [ -n name ] [ -w ]

SUMMARY
    httpd is the Apache HyperText Transfer Protocol (HTTP) server program.
    It can be run as a standalone daemon process. When used like this it will create
    child processes or threads to handle requests.
```

Рис. 10: 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]).

```
итого 4
-rw-r--r--. 1 root root unconfined_u:object_r:samba_share_t:s0 33 апр 20 05:01 test.html
```

Рис. 11: Изменение контекста

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

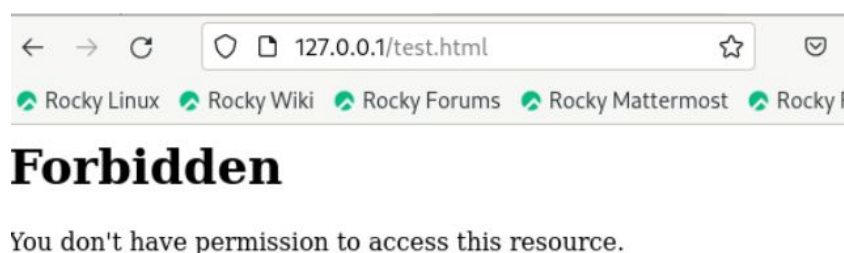


Рис. 12: Отображение файла

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]).

```
type=SYSCALL msg=audit(1713578987.972:279): arch=c000003e syscall=262 success=no exit=-13 a0=ffffff9c a1=7f97cc004c10 a2=7f97c2ffc8b0 a3=100 items=0 p
pid=30093 ppid=30136 uid=4294967295 uid=48 gid=48 euid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4294967295 comm="httpd" exe="/usr/s
bin/httpd" subj=system_u:system_r:httpd_t:s0 key=(null)ARCH=x86_64 SYSCALL=newfstatat AUID="unset" UID="apache" GID="apache" EUID="apache" SUID="apach
e" FSUID="apache" EGID="apache" SGID="apache" FSGID="apache"
type=PROCTITLE msg=audit(1713578987.972:279): proctitle=2F7573722F7362696E2F6874747066400D44464F524547524F554E44
type=SERVICE_START msg=audit(1713578989.341:280): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:unit:setroubleshoot
d comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success'UID="root" AUID="unset"
type=SERVICE_START msg=audit(1713578990.334:281): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:unit:dbus-1.1-org.
fedoraproject.SetroubleshootPrivileged1 comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success'UID="root" AUID="unset"
```

Рис. 13: Попытка прочесть лог-файл

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

```
sudo nano /etc/httpd/conf/httpd.conf
```

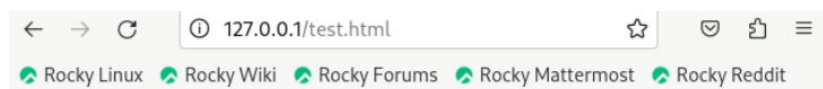
Рис. 14: Изменение файла

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

```
# Do not add a slash at the end of the directory path. If you point
# ServerRoot at a non-local disk, be sure to specify a local disk on the
# Mutex directive, if file-based mutexes are used. If you wish to share the
# same ServerRoot for multiple httpd daemons, you will need to change at
# least PidFile.
#
ServerRoot "/etc/httpd"
#
# Listen: Allows you to bind Apache to specific IP addresses and/or
# ports, instead of the default. See also the <VirtualHost>
# directive.
#
# Change this to Listen on a specific IP address, but note that if
# httpd.service is enabled to run at boot time, the address may not be
# available when the service starts. See the httpd.service(8) man
# 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 yo
# 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. 15: 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 соединения не удалась

Firefox не может установить соединение с сервером 127.0.0.1.

- Возможно, сайт временно недоступен или перегружен запросами. Подождите некоторое время и попробуйте снова.
- Если вы не можете загрузить ни одну страницу – проверьте настройки соединения с Интернетом.
- Если ваш компьютер или сеть защищены межсетевым экраном или прокси-сервером – убедитесь, что Firefox разрешён выход в Интернет.

Рис. 16: Попытка прослушивания другого порта

Проанализируем лог-файлы:

`tail -n1 /var/log/messages` (рис. [-@fig:019]).

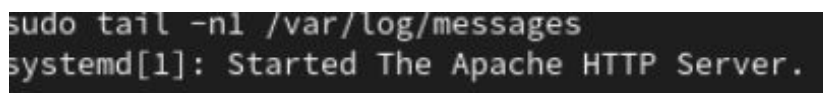


Рис. 17: Проверка лог-файлов

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]).


```
[Sat Apr 20 04:52:10.304359 2024] [core:notice] [pid 30093:tid 30093] SELinux
policy enabled; httpd running as context system_u:system_r:httpd_t:s0
[Sat Apr 20 04:52:10.307330 2024] [suexec:notice] [pid 30093:tid 30093] AH012
2: suEXEC mechanism enabled (wrapper: /usr/sbin/suexec)
AH00558: httpd: Could not reliably determine the server's fully qualified dom
in name, using fe80::a00:27ff:fe98:bdea%enp0s3. Set the 'ServerName' directiv
globally to suppress this message
[Sat Apr 20 04:52:10.371973 2024] [lbmethod_heartbeat:notice] [pid 30093:tid
30093] AH02282: No slotmem from mod_heartbeat
[Sat Apr 20 04:52:10.389422 2024] [mpm_event:notice] [pid 30093:tid 30093] AH
0489: Apache/2.4.57 (Rocky Linux) configured -- resuming normal operations
[Sat Apr 20 04:52:10.389524 2024] [core:notice] [pid 30093:tid 30093] AH00094
: Command line: '/usr/sbin/httpd -D FOREGROUND'
[Sat Apr 20 05:09:47.974451 2024] [core:error] [pid 30136:tid 30312] (13)Perm
ission denied: [client 127.0.0.1:44098] AH00035: access to /test.html denied (
filesystem path '/var/www/html/test.html') because search permissions are miss
ing on a component of the path
[Sat Apr 20 05:15:41.743945 2024] [core:error] [pid 30134:tid 30322] (13)Perm
ission denied: [client 127.0.0.1:58006] AH00035: access to /test.html denied (
filesystem path '/var/www/html/test.html') because search permissions are miss
ing on a component of the path
[Sat Apr 20 05:16:30.614988 2024] [mpm_event:notice] [pid 30093:tid 30093] AH
```

Рис. 18: Проверка лог-файлов

Выполняем команду `semanage port -a -t http_port_t -p tcp 81` После этого проверяем список портов командой `semanage port -l | grep http_port_t` Порт 81 появился в списке (рис. [-@fig:021]).

```
sudo semanage port -l | grep http_port_t
tcp      80, 81, 443, 488, 8008, 8009, 8443, 90
```

Рис. 19: Проверка портов

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

```
sudo systemctl restart httpd
sudo chcon -t httpd_sys_content_t /var/www/html/test.html
sudo systemctl restart httpd
```

Рис. 20: Перезапуск сервера

Теперь он работает, ведь порт 81 вненен в список портов `httpd_port_t` (рис. [-@fig:023]).

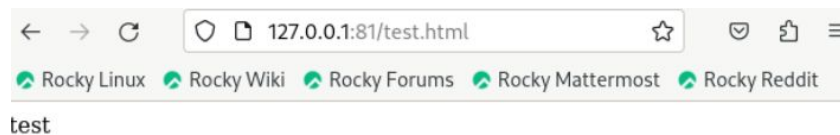


Рис. 21: Проверка сервера

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

```
sudo nano /etc/httpd/conf/httpd.conf
semanage port -d -t http_port_t -p tcp 81
x не задана, или нет доступа к хранилищу.
sudo semanage port -d -t http_port_t -p tcp 81
defined in policy, cannot be deleted
```

Рис. 22: Проверка порта 81

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



Рис. 23: 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>