

Лабораторная работа №6

Мандатное разграничение прав в Linux

Парфенова Елизавета Евгеньевна

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Выводы	21
	Список литературы	22

Список иллюстраций

4.1	Подготовка лабораторного стенда	9
4.2	Просмотр режима и политики SELinux	10
4.3	Запуск и проверка работы сервера Apache	10
4.4	Контекст безопасности сервера Apache	11
4.5	Текущее состояние переключателей SELinux для Apache	12
4.6	Статистика по политике	13
4.7	Множество пользователей, ролей, типов	14
4.8	Файлы и поддиректори в /var/www	15
4.9	Создание файла test.html и его контекст	15
4.10	Отображение файла через веб-сервер	15
4.11	Изменение контекста файла	16
4.12	Сообщение об ошибке после изменения контекста файла	16
4.13	Файл /var/log/messages*	17
4.14	Файл /var/log/audit/audit.log	17
4.15	Замена порта в config файле	18
4.16	Сбой сервера при запуске с 81 порта	18
4.17	Файл /var/log/messages	19
4.18	Добавление 81 порта	19
4.19	Изменение контекста обратно	20
4.20	Содержимое файла по новому адресу	20
4.21	Возвращение всех изменений	20

Список таблиц

1 Цель работы

Развить навыки администрирования ОС Linux. Получить первое практическое знакомство с технологией SELinux

2 Задание

Проверить работу SELinux на практике совместно с веб-сервером Apache.

3 Теоретическое введение

SELinux — это система принудительного контроля доступа, реализованная на уровне ядра. Впервые эта система появилась в четвертой версии CentOS, а в 5 и 6 версии реализация была существенно дополнена и улучшена. Эти улучшения позволили SELinux стать универсальной системой, способной эффективно решать массу актуальных задач. Стоит помнить, что классическая система прав Unix применяется первой, и управление перейдет к SELinux только в том случае, если эта первичная проверка будет успешно пройдена.

Основные термины, использующиеся в SELinux:

- *Домен* — список действий, которые может выполнять процесс. Обычно в качестве домена определяется минимально-возможный набор действий, при помощи которых процесс способен функционировать. Таким образом, если процесс дискредитирован, злоумышленнику не удастся нанести большого вреда.
- *Роль* — список доменов, которые могут быть применены. Если какого-то домена нет в списке доменов какой-то роли, то действия из этого домена не могут быть применены.
- *Тип* — набор действий, которые допустимы по отношению к объекту. Тип отличается от домена тем, что он может применяться к пайпам, каталогам и файлам, в то время как домен применяется к процессам.
- *Контекст безопасности* — все атрибуты SELinux — роли, типы и домены.

SELinux имеет три основных режим работы, при этом по умолчанию установлен режим Enforcing.

Режимы работы SELinux:

1. 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.
2. 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.
3. Disabled: Полное отключение системы принудительного контроля доступа [1].

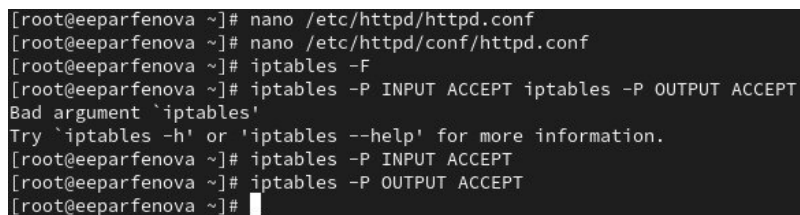
Для просмотра контекста безопасности используется команда `ls -Z` с указанием адреса. Например: `ls -Z /var/www/html/index.html`

4 Выполнение лабораторной работы

Подготовим лабораторный стенд. Для этого проверим установку сервера Apache. У меня он установлен не был, поэтому я заранее провела его установку. Затем в конфигурационном файле `/etc/httpd/conf/httpd.conf` необходимо зададим параметр `ServerName: ServerName test.ru`. Также было необходимо проследить, чтобы пакетный фильтр был либо отключён, либо в своей рабочей конфигурации позволял подключаться к 80-у и 81-у портам протокола tcp. Мной было решено отключить фильтр следующими командами

```
iptables -F
iptables -P INPUT ACCEPT
iptables -P OUTPUT ACCEPT
```

(рис. 4.1).



```
[root@eeparfenova ~]# nano /etc/httpd/httpd.conf
[root@eeparfenova ~]# nano /etc/httpd/conf/httpd.conf
[root@eeparfenova ~]# iptables -F
[root@eeparfenova ~]# iptables -P INPUT ACCEPT iptables -P OUTPUT ACCEPT
Bad argument `iptables'
Try `iptables -h' or `iptables --help' for more information.
[root@eeparfenova ~]# iptables -P INPUT ACCEPT
[root@eeparfenova ~]# iptables -P OUTPUT ACCEPT
[root@eeparfenova ~]#
```

Рис. 4.1: Подготовка лабораторного стенда

Убедимся, что SELinux работает в режиме enforcing политики targeted с помощью команд `getenforce` и `sestatus`. Видим, что этой действительно так (рис. 4.2)

```

[eeeparfenova@eeeparfenova ~]$ getenforce
Enforcing
[eeeparfenova@eeeparfenova ~]$ sestatus
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

```

Рис. 4.2: Просмотр режима и политики SELinux

Запустим сервер Apache командой *sudo systemctl start httpd* и с помощью браузера обратимся к нему, убедившись, что сервер работает командой *sydo systemctl status httpd* (рис. 4.3).

```

[eeeparfenova@eeeparfenova ~]$ su -
Password:
[root@eeeparfenova ~]# service httpd status
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: inactive (dead)
     Docs: man:httpd.service(8)
[root@eeeparfenova ~]# sudo systemctl restart httpd
[root@eeeparfenova ~]# sudo systemctl start httpd
[root@eeeparfenova ~]# sudo systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; preset: disabled)
   Active: active (running) since Wed 2024-10-09 17:56:00 MSK; 24s ago
     Docs: man:httpd.service(8)
  Main PID: 42967 (httpd)
    Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec:  0 B/sec"
    Tasks: 177 (limit: 12207)
   Memory: 24.0M
      CPU: 125ms
  CGroup: /system.slice/httpd.service
          └─42967 /usr/sbin/httpd -DFOREGROUND
            └─42968 /usr/sbin/httpd -DFOREGROUND
              └─42969 /usr/sbin/httpd -DFOREGROUND
                └─42970 /usr/sbin/httpd -DFOREGROUND
                  └─42971 /usr/sbin/httpd -DFOREGROUND

Oct 09 17:56:00 eeeparfenova.localdomain systemd[1]: Starting The Apache HTTP Server...
Oct 09 17:56:00 eeeparfenova.localdomain systemd[1]: Started The Apache HTTP Server.
Oct 09 17:56:00 eeeparfenova.localdomain httpd[42967]: Server configured, listening on: port 80
[root@eeeparfenova ~]#

```

Рис. 4.3: Запуск и проверка работы сервера Apache

С помощью команд *ps auxZ | grep httpd* и *ps -eZ | grep httpd* найдем веб-сервер Apache в списке процессов, определив его контекст безопасности. Его контекст безопасности: *system_u:system_r:httpd_t:s0*. (рис. 4.4)

```

[root@eeparfenova ~]# ps auxZ | grep httpd
system_u:system_r:httpd_t:s0 root 42967 0.0 0.5 20152 11544 ? Ss 17:55 0:00 /usr/sbin/httpd
# -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 42968 0.0 0.3 22032 7484 ? S 17:56 0:00 /usr/sbin/httpd
# -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 42969 0.0 0.6 1112588 13600 ? Sl 17:56 0:00 /usr/sbin/httpd
# -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 42970 0.0 0.6 981452 13412 ? Sl 17:56 0:00 /usr/sbin/httpd
# -DFOREGROUND
system_u:system_r:httpd_t:s0 apache 42971 0.0 0.5 981452 11284 ? Sl 17:56 0:00 /usr/sbin/httpd
# -DFOREGROUND
unconfined_u:unconfined_r:unconfined_t:s0-s0:c0.c1023 root 43203 0.0 0.1 221664 2304 pts/0 S+ 17:57 0:00 grep
--color=auto httpd
[root@eeparfenova ~]# ps -eZ | grep httpd
system_u:system_r:httpd_t:s0 42967 ? 00:00:00 httpd
system_u:system_r:httpd_t:s0 42968 ? 00:00:00 httpd
system_u:system_r:httpd_t:s0 42969 ? 00:00:00 httpd
system_u:system_r:httpd_t:s0 42970 ? 00:00:00 httpd
system_u:system_r:httpd_t:s0 42971 ? 00:00:00 httpd
[root@eeparfenova ~]#

```

Рис. 4.4: Контекст безопасности сервера Apache

Посмотрим текущее состояние переключателей SELinux для Apache с помощью команды *sestatus -b | grep httpd*. Обратим внимание, что многие из них находятся в положении «off». (рис. 4.5)

```
[root@eeparfenova ~]# sestatus -b | grep httpd
httpd_anon_write off
httpd_builtin_scripting on
httpd_can_check_spam off
httpd_can_connect_ftp off
httpd_can_connect_ldap off
httpd_can_connect_mythtv off
httpd_can_connect_zabbix off
httpd_can_manage_courier_spool off
httpd_can_network_connect off
httpd_can_network_connect_cobbler off
httpd_can_network_connect_db off
httpd_can_network_memcache off
httpd_can_network_relay off
httpd_can_sendmail off
httpd_dbus_avaahi off
httpd_dbus_sssd off
httpd_dontaudit_search_dirs off
httpd_enable_cgi on
httpd_enable_ftp_server off
httpd_enable_homedirs off
httpd_execmem off
httpd_graceful_shutdown off
httpd_manage_ipa off
httpd_mod_auth_ntlm_winbind off
httpd_mod_auth_pam off
httpd_read_user_content off
httpd_run_ipa off
httpd_run_preupgrade off
httpd_run_stickshift off
httpd_serve_cobbler_files off
httpd_setrlimit off
httpd_ssi_exec off
httpd_sys_script_anon_write off
httpd_tmp_exec off
httpd_tty_comm off
httpd_unified off
httpd_use_cifs off
httpd_use_fusefs off
httpd_use_gpg off
httpd_use_nfs off
httpd_use_openscryptoki off
httpd_use_openstack off
httpd_use_sasl off
httpd_verify_dns off
[root@eeparfenova ~]#
```

Рис. 4.5: Текущее состояние переключателей SELinux для Apache

Посмотрим статистику по политике с помощью команды *seinfo* (рис. 4.6), также определим множество пользователей, ролей, типов с помощью команд *seinfo -u*,

seinfo -r, *seinfo -t* соответственно (рис. 4.7). Оно соответственно равно 8, 15, 5145.

```
[root@eeparfenova ~]# seinfo
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:                   5145  Attributes:            259
Users:                   8     Roles:                 15
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 Constrains:          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
```

Рис. 4.6: Статистика по политике

```

[root@eeparfenova ~]# seinfo -u

Users: 8
  guest_u
  root
  staff_u
  sysadm_u
  system_u
  unconfined_u
  user_u
  xguest_u
[root@eeparfenova ~]# seinfo -r

Roles: 15
  auditadm_r
  container_user_r
  dbadm_r
  guest_r
  logadm_r
  nx_server_r
  object_r
  secadm_r
  staff_r
  sysadm_r
  system_r
  unconfined_r
  user_r
  webadm_r
  xguest_r
[root@eeparfenova ~]# seinfo -t

Types: 5145
  NetworkManager_dispatcher_chronyc_script_t
  NetworkManager_dispatcher_chronyc_t
  NetworkManager_dispatcher_cloud_script_t
  NetworkManager_dispatcher_cloud_t
  NetworkManager_dispatcher_console_script_t
  NetworkManager_dispatcher_console_t
  NetworkManager_dispatcher_console_var_run_t
  NetworkManager_dispatcher_custom_t
  NetworkManager_dispatcher_ddclient_script_t
  NetworkManager_dispatcher_ddclient_t
  NetworkManager_dispatcher_dhclient_script_t
  NetworkManager_dispatcher_dhclient_t
  NetworkManager_dispatcher_dnsssec_script_t
  NetworkManager_dispatcher_dnsssec_t

```

Рис. 4.7: Множество пользователей, ролей, типов

Определим тип файлов и поддиректорий, находящихся в директории `/var/www`, с помощью команды `ls -lZ /var/www`. Видим, что все файлы и поддиректории принадлежат пользователю `root`, а также имеют некоторый контекст безопасности. Определим тип файлов, находящихся в директории `/var/www/html` командой `ls -lZ /var/www/html`. Она оказалась пуста. (рис. 4.8)

```
[root@eeparfenova ~]# ls -lZ /var/www
total 0
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_script_exec_t:s0 6 Aug 8 19:30 cgi-bin
drwxr-xr-x. 2 root root system_u:object_r:httpd_sys_content_t:s0 6 Aug 8 19:30 html
[root@eeparfenova ~]# ls -lZ /var/www/html
total 0
[root@eeparfenova ~]#
```

Рис. 4.8: Файлы и поддиректори в /var/www

Как уже было указано, создание файлов в директории /var/www/html доступно только владельцу, то есть пользователю root.

Создадим от имени суперпользователя html-файл /var/www/html/test.html следующего содержания:

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

Проверим его контекст, присвоенный по умолчанию созданным файлам в директории /var/www/html: unconfined_u:object_r:httpd_sys_content_r:s0 (рис. 4.9).

```
[root@eeparfenova ~]# nano /var/www/html/test.html
[root@eeparfenova ~]# ls -lZ /var/www/html
total 4
-rw-r--r--. 1 root root unconfined_u:object_r:httpd_sys_content_t:s0 33 Oct 9 18:07 test.html
[root@eeparfenova ~]# ls -Z /var/www/html
unconfined_u:object_r:httpd_sys_content_t:s0 test.html
[root@eeparfenova ~]#
```

Рис. 4.9: Создание файла test.html и его контекст

Обратимся к файлу через веб-сервер, введя в браузере адрес <http://127.0.0.1/test.html> убедимся, что файл был успешно отображён (рис. 4.10).

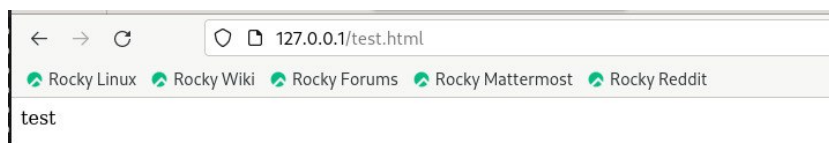


Рис. 4.10: Отображение файла через веб-сервер

Изучив справку `man httpd_selinux`, увидим, что для файла test.html был определен один из контекстов безопасности httpd. Проверить контекст файла можно командой `ls -Z /var/www/html/test.html` (мы уже делали это ранее, на рис. 9)

Рассмотрим полученный контекст детально. - Обратим внимание, что так как по умолчанию пользователи являются свободными от типа, 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. - 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.

Изменим 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`. Сделаем это с помощью команды `chcon -t samba_share_t /var/www/html/test.html`. Затем проверим успешность действия командой `ls -Z /var/www/html/test.html` (рис. 4.11).

```
[root@eeparfenova ~]# chcon -t samba_share_t /var/www/html/test.html
[root@eeparfenova ~]# ls -Z /var/www/html/test.html
unconfined_u:object_r:samba_share_t:s0 /var/www/html/test.html
[root@eeparfenova ~]#
```

Рис. 4.11: Изменение контекста файла

Попробуем ещё раз получить доступ к файлу через веб-сервер, введя в браузере адрес `http://127.0.0.1/test.html`. Но получаем следующее сообщение об ошибке (рис. 4.12):

Forbidden

You don't have permission to access /test.html on this server.

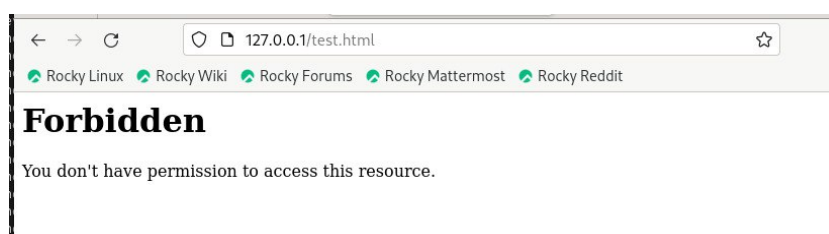


Рис. 4.12: Сообщение об ошибке после изменения контекста файла

Данная ситуация произошла, так как контекст безопасности был изменен на то, к которому httpd не имеет доступа, то есть читать файл по-прежнему могут все, но отображать его через браузер запрещено.

Посмотрим log-файлы веб-сервера Apache. Посмотрим системный лог-файл командой `tail /var/log/messages` (рис. 4.13) и файл `/var/log/audit/audit.log` с помощью команды `tail /var/log/audit/audit.log` (рис. 4.14). Видим в системе запущены процессы `setroubleshootd` и `auditd`, которые свидетельствуют об ошибке доступа - пишется, что SELinux препятствует серверу в получении полного доступа к вызываемому файлу (это и является причиной, почему мы не можем прочесть его через браузер).

```
[root@eeparfenova ~]# tail /var/log/messages
Oct 9 18:16:13 eeparfenova systemd[1]: Started dbus-1.1-org.fedoraproject.SetroubleshootPrivileged@0.service.
Oct 9 18:16:14 eeparfenova setroubleshoot[44465]: SELinux is preventing /usr/sbin/httpd from getattr access on the file /var/www/html/test.html. For complete SELinux messages run: sealert -l 0d941f33-d423-4a9c-86d4-881632fdb2a3
Oct 9 18:16:14 eeparfenova setroubleshoot[44465]: SELinux is preventing /usr/sbin/httpd from getattr access on the file /var/www/html/test.html. #012#012***** Plugin restorecon (92.2 confidence) suggests *****
*****#012#012If you want to fix the label. #012/var/www/html/test.html default label should be httpd_sys_content_t. #012Then you can run restorecon. The access attempt may have been stopped due to insufficient permissions to access a parent directory in which case try to change the following command accordingly. #012Do#012# /sbin/restorecon -v /var/www/html/test.html#012#012***** Plugin public_content (7.83 confidence) suggests *****
*****#012#012If you want to treat test.html as public content#012Then you need to change the label on test.html to public_content_t or public_content_rw_t. #012Do#012# semanage fcontext -a -t public_content_t '/var/www/html/test.html' #012# restorecon -v '/var/www/html/test.html' #012#012***** Plugin catchall (1.41 confidence) suggests *****
*****#012#012If you believe that httpd should be allowed getattr access on the test.html file by default. #012Then you should report this as a bug. #012You can generate a local policy module to allow this access. #012Do#012# allow this access for now by executing: #012# ausearch -c 'httpd' --raw | audit2allow -M my-httpd#012# semodule -X 300 -i my-httpd.pp#012
Oct 9 18:16:14 eeparfenova setroubleshoot[44465]: SELinux is preventing /usr/sbin/httpd from getattr access on the file /var/www/html/test.html. For complete SELinux messages run: sealert -l 0d941f33-d423-4a9c-86d4-881632fdb2a3
```

Рис. 4.13: Файл `/var/log/messages`*

```
[root@eeparfenova ~]# tail /var/log/audit/audit.log
==> /var/log/audit/audit.log <==
type=AVC msg=audit(1728486971.845:410): avc: denied { getattr } for pid=42970 comm="httpd" path="/var/www/html/test.html" dev="dm-0" ino=837653 scontext=system_u:system_r:httpd_t:s0 tcontext=unconfined_u:object_r:samba_share_t:s0 tclass=file permission=0
type=SYSCALL msg=audit(1728486971.845:410): arch=c000003e syscall=262 success=no exit=-13 a0=ffffff9c a1=7f472800 b160 a2=7f4734ff8b0 a3=100 items=0 ppid=42967 pid=42970 auid=4294967295 uid=48 gid=48 euid=48 suid=48 fsuid=48 egid=48 sgid=48 fsgid=48 tty=(none) ses=4294967295 comm="httpd" exe="/usr/sbin/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="apache" FSUID="apache" EGID="apache" SGID="apache" FSGID="apache"
type=PROCTITLE msg=audit(1728486971.845:410): proctitle=2F7573722F736269652F6874747064002044464F524547524F554E44
type=SERVICE_START msg=audit(1728486972.891:411): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=setroubleshootd comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
type=SERVICE_START msg=audit(1728486973.323:412): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=dbus-1.1-org.fedoraproject.SetroubleshootPrivileged@0 comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
type=SERVICE_STOP msg=audit(1728486985.188:413): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=dbus-1.1-org.fedoraproject.SetroubleshootPrivileged@0 comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
type=SERVICE_STOP msg=audit(1728486985.303:414): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=setroubleshootd comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
type=SERVICE_STOP msg=audit(1728487010.929:415): pid=1 uid=0 auid=4294967295 ses=4294967295 subj=system_u:system_r:init_t:s0 msg='unit=systemd-hostnamed comm="systemd" exe="/usr/lib/systemd/systemd" hostname=? addr=? terminal=? res=success' UID="root" AUID="unset"
type=BPF msg=audit(1728487010.944:416): prog-id=99 op=UNLOAD
type=BPF msg=audit(1728487010.944:417): prog-id=98 op=UNLOAD
```

Рис. 4.14: Файл `/var/log/audit/audit.log`

Попробуем запустить веб-сервер Apache на прослушивание TCP-порта 81 (а не

80). Для этого в файле `/etc/httpd/httpd.conf` найдем строчку `Listen 80` и заменим её на `Listen 81` (рис. 4.15).

```
#  
# 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
```

Рис. 4.15: Замена порта в config файле

Выполним перезапуск веб-сервера Apache. У нас произошел сбой, так как порт настроен на 80 порт, а мы пытаемся запустить его с 81 (рис. 4.16).

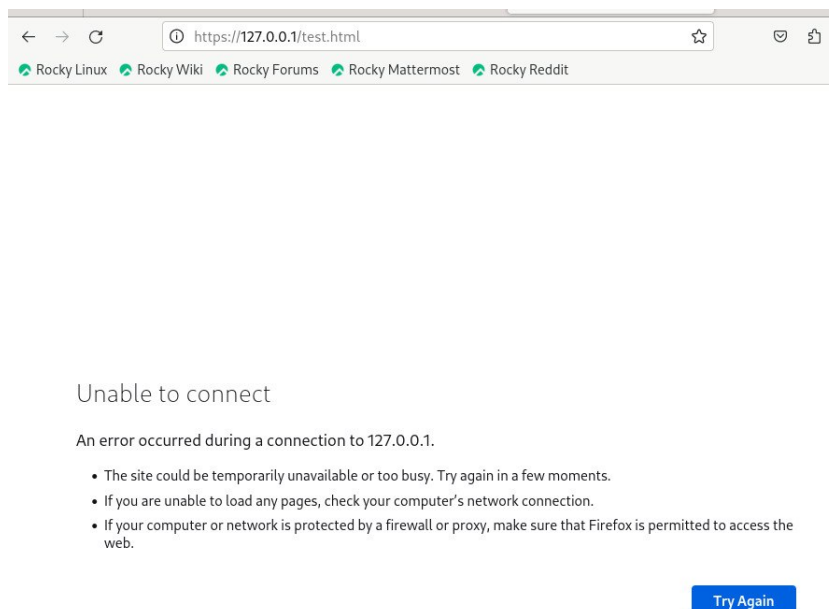


Рис. 4.16: Сбой сервера при запуске с 81 порта

Проанализируйте лог-файлы с помощью команд `tail -l /var/log/messages` (рис. 4.17), `tail -l /var/log/http/error_log` и `tail -l /var/log/http/access_log..`

```

[root@eeparfenova ~]# tail -l /var/log/messages
oct 9 18:32:48 eeparfenova systemd[1]: Stopped The Apache HTTP Server.
oct 9 18:32:48 eeparfenova systemd[1]: httpd.service: Consumed 1.581s CPU time.
oct 9 18:32:48 eeparfenova systemd[1]: Starting The Apache HTTP Server...
oct 9 18:32:48 eeparfenova systemd[1]: Started The Apache HTTP Server.
oct 9 18:32:48 eeparfenova httpd[45113]: Server configured, listening on: port 81
oct 9 18:34:34 eeparfenova systemd[1646]: Started dbus-1.2-org.gnome.Screenshot@18.service.
oct 9 18:34:41 eeparfenova systemd[1]: Starting Hostname Service...
oct 9 18:34:41 eeparfenova systemd[1]: Started Hostname Service.
oct 9 18:35:01 eeparfenova systemd[1]: packageKit.service: Deactivated successfully.
oct 9 18:35:01 eeparfenova systemd[1]: packageKit.service: Consumed 1.208s CPU time.

```

Рис. 4.17: Файл /var/log/messages

Видим, что есть сообщение, что файл сконфигурирован и прослушивается с 81 порта. Также у нас появилась новая запись в лог-файла ошибок о невозможности загрузки, и не появилось новой записи в лог-файле доступа.

Выполним команду *semanage port -a -t http_port_t -p tcp 81*, добавив 81 порт в список подключенных портов. После этого проверим список портов командой *semanage port -l | grep http_port_t* Убедимся, что порт 81 появился в списке (он у нас еще и повторяется дважды) (рис. 4.18).

```

[root@eeparfenova ~]# semanage port -l | grep http_port_t
http_port_t          tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t  tcp      5988
[root@eeparfenova ~]# semanage port -d -t http_port_t -p tcp 81
ValueError: Port tcp/81 is defined in policy, cannot be deleted
[root@eeparfenova ~]# semanage port -a -t http_port_t -p tcp 81
Port tcp/81 already defined, modifying instead
[root@eeparfenova ~]# semanage port -l | grep http_port_t
http_port_t          tcp      81, 80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t  tcp      5988
[root@eeparfenova ~]#

```

Рис. 4.18: Добавление 81 порта

Попробуем запустить веб-сервер Apache ещё раз, но и в этот раз у меня произошел сбой, так как указан не тот контекст. Вернем контекст *httpd_sys_content_t* к файлу */var/www/html/ test.html* командой *chcon -t httpd_sys_content_t /var/www/html/test.html* (рис. 4.19). После этого попробуем получить доступ к файлу через веб-сервер, введя в браузере адрес *http://127.0.0.1:81/test.html*, и на этот раз мы действительно смогли прочитать файл через браузер, увидев содержимое файла — слово «test». (рис. 4.20)

```
[root@eeparfenova ~]# chcon -t httpd_sys_content_t /var/www/html/test.html
[root@eeparfenova ~]# ls -Z /var/www/html/test.html
unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/test.html
[root@eeparfenova ~]#
```

Рис. 4.19: Изменение контекста обратно

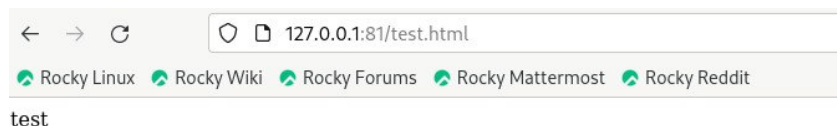


Рис. 4.20: Содержимое файла по новому адресу

В конце вернем все, что изменили на место: исправим обратно конфигурационный файл `apache`, вернув `Listen 80`; удалим привязку `http_port_t` к 81 порту командой `semanage port -d -t http_port_t -p tcp 81`; проверим, что порт 81 удалён; удалим файл `/var/www/html/test.html` командой `rm /var/www/html/test.html` (рис. 4.21)

```
[root@eeparfenova ~]# nano /etc/httpd/conf/httpd.conf
[root@eeparfenova ~]# semanage port -d -t http_port_t -p tcp 81
[root@eeparfenova ~]# ls -Z /var/www/html/test.html
unconfined_u:object_r:httpd_sys_content_t:s0 /var/www/html/test.html
[root@eeparfenova ~]# semanage port -l | grep http_port_t
http_port_t      tcp      80, 81, 443, 488, 8008, 8009, 8443, 9000
pegasus_http_port_t tcp      5988
[root@eeparfenova ~]# rm /var/www/html/test.html
rm: remove regular file '/var/www/html/test.html'? y
[root@eeparfenova ~]#
```

Рис. 4.21: Возвращение всех изменений

5 Выводы

Мы развили навыки администрирования ОС Linux и получили первое практическое знакомство с технологией SELinux. Для этого мы проверили работу SELinux на практике совместно с веб-сервером Apache.

Список литературы

1. SELinux – описание и особенности работы с системой. Часть 1 [Электронный ресурс]. © 2006–2024, Habr, 2014. URL: <https://habr.com/ru/companies/kingservers/articles/209644/>.