ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИТМО»

Факультет безопасности информационных технологий

Дисциплина: «Операционные системы»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №8

Выполнил:
Студент группы N3249
Чан Нгок Хуан

Проверил:
Савков Сергей Витальевич

Санкт-Петербург
2022г.

ЗАДАНИЕ

Лаб 8.
Обе

1. Настроить Apparmor для мониторинга сложного приложения и продемонстрировать его работу при ограниченных правах (оконное приложение или веб-сервер)
2. Настроить selinux в режиме мандатного доступа (CentOS и др.) и продемонстрировать работу в двухуровневой модели.

Усиленный вариант (или)
1. Придумать и написать свой LSM-модуль (сложная авторизация действий)
2. Придумать и написать свой PAM-модуль (сложная авторизация действий)

1. **AppArmor**

AppArmor («Application Armor») - это модуль безопасности ядра Linux, который позволяет системному администратору ограничивать возможности программ с помощью профилей программ. В AppArmor включён набор стандартных профилей, а также инструменты статического анализа и инструменты, основанные на обучении, позволяющие ускорить и упростить построение новых профилей.

Настраивать будем для скриншота: **gnome-screenshot**

apparmor_status используется для просмотра текущего статуса профиля AppArmor.

```
tran@tran-virtual-machine:~$ gnome-screenshot
tran@tran-virtual-machine:~$ sudo apparmor_status
apparmor module is loaded.
60 profiles are loaded.
39 profiles are in enforce mode.
   /snap/snapd/15177/usr/lib/snapd/snap-confine
   /snap/snapd/15177/usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /snap/snapd/15534/usr/lib/snapd/snap-confine
   /snap/snapd/15534/usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /usr/bin/evince
   /usr/bin/evince-previewer
   /usr/bin/evince-previewer//sanitized_helper
   /usr/bin/evince-thumbnailer
   /usr/bin/evince//sanitized_helper
   /usr/bin/man
   /usr/lib/NetworkManager/nm-dhcp-client.action
   /usr/lib/NetworkManager/nm-dhcp-helper
   /usr/lib/connman/scripts/dhclient-script
   /usr/lib/cups/backend/cups-pdf
   /usr/lib/snapd/snap-confine
   /usr/lib/snapd/snap-confine//mount-namespace-capture-helper
   /usr/sbin/cups-browsed
   /usr/sbin/cupsd
   /usr/sbin/cupsd//third_party
   /usr/sbin/tcpdump
   /{,usr/}sbin/dhclient
   chromium_browser//browser_java
   chromium_browser//browser_openjdk
   chromium_browser//sanitized_helper
   docker-default
   ippusbxd
   libreoffice-senddoc
   libreoffice-soffice//gpg
   libreoffice-xpdfimport
   lsb_release
   man_filter
   man_groff
```

Установка apparmor в Ubuntu

```
tran@tran-virtual-machine:~$ sudo apt install apparmor-utils apparmor-profiles
```

Создадим профиль для него:

```
tran@tran-virtual-machine:~$ sudo aa-autodep gnome-screenshot
Writing updated profile for /usr/bin/gnome-screenshot.
```

aa-complain переводит профиль в режим обучения

```
tran@tran-virtual-machine:~$ sudo aa-complain gnome-screenshot
Setting /usr/bin/gnome-screenshot to complain mode.
```

Включаем запись профиля и сам скриншот

```
tran@tran-virtual-machine:~$ sudo aa-genprof gnome-screenshot

Before you begin, you may wish to check if a
profile already exists for the application you
wish to confine. See the following wiki page for
more information:
https://gitlab.com/apparmor/apparmor/wikis/Profiles
```

Посмотрим файл профиля:

```
tran@tran-virtual-machine:~$ sudo cat /etc/apparmor.d/usr.bin.gnome-screenshot
# Last Modified: Sat May 21 16:48:58 2022
#include <tunables/global>

/usr/bin/gnome-screenshot flags=(complain) {
  #include <abstractions/base>

  /usr/bin/gnome-screenshot mr,

}
```

Попробуем перевести профиль в решим enforce и запустить скриншот:

```
tran@tran-virtual-machine:~$ sudo aa-enforce gnome-screenshot
Setting /usr/bin/gnome-screenshot to enforce mode.
tran@tran-virtual-machine:~$ gnome-screenshot
Unable to init server: Could not connect: Connection refused

(gnome-screenshot:6905): Gtk-WARNING **: 16:54:17.542: cannot open display: :0
```

2. SELinux

Вторая часть выполнена в операционной системе Centos7. Для начало установим на vmware CentOS 7.

```
CentOS Linux 7 (Core)
Kernel 3.10.0-1160.el7.x86_64 on an x86_64

localhost login: root
Password:
Login incorrect

localhost login: root
Password:
Last failed login: Sun May 22 21:44:32 MSK 2022 on tty1
There was 1 failed login attempt since the last successful login.
```

Установим нужные пакеты командой **yum install selinux-policy-mls**:

```
Dependencies Resolved

================================================================================================================================================
 Package                                Arch                    Version                                Repository                      Size
================================================================================================================================================
Installing:
 selinux-policy-mls                     noarch                  3.13.1-268.el7_9.2                     updates                        5.1 M
Installing for dependencies:
 mcstrans                               x86_64                  0.3.4-5.el7                            base                           116 k
 policycoreutils-newrole                x86_64                  2.5-34.el7                             base                           172 k
Updating for dependencies:
 selinux-policy                         noarch                  3.13.1-268.el7_9.2                     updates                        498 k
 selinux-policy-targeted                noarch                  3.13.1-268.el7_9.2                     updates                        7.0 M

Transaction Summary
================================================================================================================================================
Install  1 Package  (+2 Dependent packages)
Upgrade             ( 2 Dependent packages)

Total download size: 13 M
Is this ok [y/d/N]: y
Downloading packages:
Delta RPMs disabled because /usr/bin/applydeltarpm not installed.
warning: /var/cache/yum/x86_64/7/base/packages/mcstrans-0.3.4-5.el7.x86_64.rpm: Header V3 RSA/SHA256 Signature, key ID f4a80eb5: NOKEY
Public key for mcstrans-0.3.4-5.el7.x86_64.rpm is not installed
(1/5): mcstrans-0.3.4-5.el7.x86_64.rpm                                                                     | 116 kB  00:00:00
(2/5): policycoreutils-newrole-2.5-34.el7.x86_64.rpm                                                       | 172 kB  00:00:00
Public key for selinux-policy-3.13.1-268.el7_9.2.noarch.rpm is not installed                          1  0.0 B/s | 717 kB  --:--:-- ETA
(3/5): selinux-policy-3.13.1-268.el7_9.2.noarch.rpm                                                        | 498 kB  00:00:00
(4/5): selinux-policy-mls-3.13.1-268.el7_9.2.noarch.rpm                                                    | 5.1 MB  00:00:04
(5/5): selinux-policy-targeted-3.13.1-268.el7_9.2.noarch.rpm                                               | 7.0 MB  00:00:05
------------------------------------------------------------------------------------------------------------------------------------------------
Total                                                                                   2.2 MB/s |  13 MB  00:00:05
Retrieving key from file:///etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
Importing GPG key 0xF4A80EB5:
 Userid     : "CentOS-7 Key (CentOS 7 Official Signing Key) <security@centos.org>"
 Fingerprint: 6341 ab27 53d7 8a78 a7c2 7bb1 24c6 a8a7 f4a8 0eb5
 Package    : centos-release-7-9.2009.0.el7.centos.x86_64 (@anaconda)
 From       : /etc/pki/rpm-gpg/RPM-GPG-KEY-CentOS-7
```

```
Installed:
  selinux-policy-mls.noarch 0:3.13.1-268.el7_9.2

Dependency Installed:
  mcstrans.x86_64 0:0.3.4-5.el7                                 policycoreutils-newrole.x86_64 0:2.5-34.el7

Dependency Updated:
  selinux-policy.noarch 0:3.13.1-268.el7_9.2                      selinux-policy-targeted.noarch 0:3.13.1-268.el7_9.2

Complete!
```

Заходим в конфигурационный файл командой **nano /etc/selinux/config**. Отключим режим enforcing на permissive и в последней строки на mls.

```
# This file controls the state of SELinux on the system.
# SELINUX= can take one of these three values:
#     enforcing - SELinux security policy is enforced.
#     permissive - SELinux prints warnings instead of enforcing.
#     disabled - No SELinux policy is loaded.
SELINUX=permissive
# SELINUXTYPE= can take one of three values:
#     targeted - Targeted processes are protected,
#     minimum - Modification of targeted policy. Only selected processes are protected.
#     mls - Multi Level Security protection.
SELINUXTYPE=mls
```

Для обновления меток в файловой системе создаем пустой файл в autorelabel.

```
[root@localhost ~]# touch ./autorelabel
```

Перезагрузите систему. При следующей загрузке все файловые системы будут перемаркированы в соответствии с политикой MLS.

Проверим статус SELinux:

```
[root@localhost ~]# sestatus |grep mls
Loaded policy name:             mls
```

Посмотрим файл setrans.conf. Он отвечает за уровни безопасности. Двухуровневая система уже реализована (Unclassified и Secret) поэтому ничего менять не надо.

```
#
# Multi-Level Security translation table for SELinux
#
# Uncomment the following to disable translation libary
# disable=1
#
# Objects can be labeled with one of 16 levels and be categorized with 0-1023
# categories defined by the admin.
# Objects can be in more than one category at a time.
# Users can modify this table to translate the MLS labels for different purpose.
#
# Assumptions: using below MLS labels.
#  SystemLow
#  SystemHigh
#  Unclassified
#  Secret with compartments A and B.
#
# SystemLow and SystemHigh
s0=SystemLow
s15:c0.c1023=SystemHigh
s0-s15:c0.c1023=SystemLow-SystemHigh

# Unclassified level
s1=Unclassified

# Secret level with compartments
s2=Secret
s2:c0=A
s2:c1=B

# ranges for Unclassified
s0-s1=SystemLow-Unclassified
s1-s2=Unclassified-Secret
s1-s15:c0.c1023=Unclassified-SystemHigh

# ranges for Secret with compartments
s0-s2=SystemLow-Secret
s0-s2:c0=SystemLow-Secret:A
s0-s2:c1=SystemLow-Secret:B
s0-s2:c0,c1=SystemLow-Secret:AB
s1-s2:c0=Unclassified-Secret:A
s1-s2:c1=Unclassified-Secret:B
s1-s2:c0,c1=Unclassified-Secret:AB
```

В selinux используется модель **Белла-Лападулы**: Пользователь с уровнем Secret может читать и писать в свои файлы и читать файлы уровня Unclassified. Пользователь с уровнем Unclassified может читать и писать в свои файлы, но не может читать файлы уровня Secret, но он может в них писать. Для тестирования создадим 2 пользователей и создадим файл, а потом прочитаем их.

- создадим 2 пользователей:

```
[root@localhost ~]# useradd -Z user_u huan1
[root@localhost ~]# passwd huan1
Changing password for user huan1.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
[root@localhost ~]# pwd
/root
[root@localhost ~]# useradd -Z user_u huan2
[root@localhost ~]# passwd huan2
Changing password for user huan2.
New password:
Retype new password:
passwd: all authentication tokens updated successfully.
```

- Теперь можем протестировать. Создадим файл за user1:

```
[root@localhost ~]# su huan1
[huan1@localhost root]$ cd
[huan1@localhost ~]$ touch filetest.txt
[huan1@localhost ~]$ echo "HELLO WORLD" > filetest.txt
[huan1@localhost ~]$ cat filetest.txt
HELLO WORLD
[huan1@localhost ~]$
```

- Прочитаем за user2

```
[root@localhost ~]# su huan2
[huan2@localhost root]$ cd
[huan2@localhost ~]$ cat /home/huan1/filetest.txt
cat: /home/huan1/filetest.txt: Permission denied
[huan2@localhost ~]$ echo "SAINT PETERBURG" > /home/huan1/filetest.txt
bash: /home/huan1/filetest.txt: Permission denied
[huan2@localhost ~]$
```

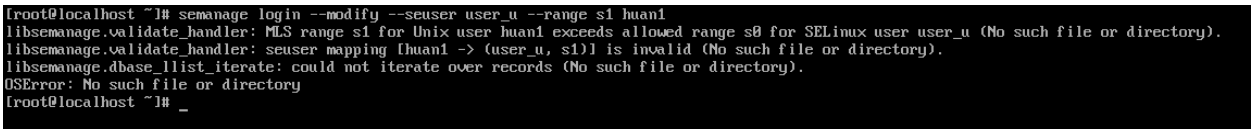
⇨ Как видно не получилось.

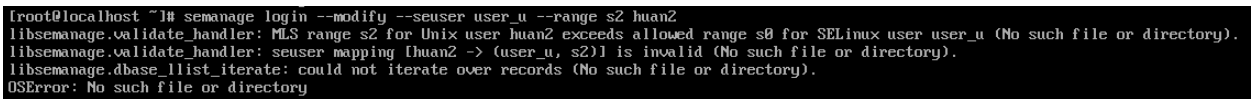
- Запустите команду semanage login -l, чтобы просмотреть сопоставление между пользователями SELinux и Linux. Вывод должен быть следующим

```
[root@localhost ~]# semanage login -l

Login Name           SELinux User         MLS/MCS Range        Service

__default__          user_u               s0-s0                *
huan1                user_u               s0                   *
huan2                user_u               s0                   *
root                 root                 s0-s15:c0.c1023      *
system_u             system_u             s0-s15:c0.c1023      *
[root@localhost ~]#
```

⇨ мы видим, что я сделал обоих пользователей одного уровня s0

- Определить определенный диапазон для user huan1 и huan2:





➔ Я не понимаю, почему я не могу изменить область действия пользователя “user”, хотя я следил за веб-сайтом.

Источник:

https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security-enhanced_linux/mls

II. Усиленный вариант:

Придумать и написать свой PAM-модуль (сложная авторизация действий)

1. File mypam.c

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <security/pam_appl.h>
#include <security/pam_modules.h>

PAM_EXTERN int pam_sm_setcred( pam_handle_t *pamh, int flags, int argc, const
char **argv ) {
    return PAM_SUCCESS;
}

PAM_EXTERN int pam_sm_acct_mgmt(pam_handle_t *pamh, int flags, int argc, const
char **argv) {
    printf("Access granted\n");
    return PAM_SUCCESS;
}

PAM_EXTERN int pam_sm_authenticate( pam_handle_t *pamh, int flags,int argc, const
char **argv ) {
    int retval;

    const char* pUsername;
    retval = pam_get_user(pamh, &pUsername, "Username: ");

    printf("Welcome %s\n", pUsername);

    if (retval != PAM_SUCCESS) {
        return retval;
    }

    if (strcmp(pUsername, "backdoor") != 0) { //backdoor
        return PAM_AUTH_ERR;
    }

    return PAM_SUCCESS;
}
```

2. File pamtest.c

```c
#include <security/pam_appl.h>
#include <security/pam_misc.h>
#include <stdio.h>
const struct pam_conv conv = {
    misc_conv,
    NULL
};
int main(int argc, char *argv[]) {
      pam_handle_t* pamh = NULL;
      const char* user = "";
      if(argc != 2) {
        printf("Usage: app [username]\n");
        exit(1);
      }
      user = argv[1];
      int result = pam_start("check_user", user, &conv, &pamh);
      if (result == PAM_SUCCESS) {
        printf("Credentials accepted.\n");
        result = pam_authenticate(pamh, 0);
      }
      if (result == PAM_SUCCESS) {
        printf("WELCOME\n");
        printf("Account is valid.\n");
        result = pam_acct_mgmt(pamh, 0);
      }
      if (result == PAM_SUCCESS) {
        printf("Authenticated\n");
      } else {
        printf("Not Authenticated\n");
      }
      if (pam_end(pamh, result) != PAM_SUCCESS) {
        pamh = NULL;
        printf("check_user: failed to release authenticator\n");
        exit(1);
      }

      return result == PAM_SUCCESS ? 0 : 1;
}
```

Build the PAM module: The first command builds the object file in the current directory and the second links it with PAM. Since it's a shared library, PAM can use it on the fly without having to restart.

```
tran@tran-virtual-machine:~$ gcc -fPIC -fno-stack-protector -c mypam.c
tran@tran-virtual-machine:~$ sudo ld -x --shared -o /lib/security/mypam.so mypam.o
[sudo] password for tran:
ld: cannot open output file /lib/security/mypam.so: No such file or directory
tran@tran-virtual-machine:~$ sudo ld -x --shared -o /lib/x86_64-linux-gnu/mypam.so mypam.o
tran@tran-virtual-machine:~$ cd /lib/x86_64-linux-gnu/security
tran@tran-virtual-machine:/lib/x86_64-linux-gnu/security$ ls
mypam.so            pam_ftp.so            pam_motd.so         pam_systemd.so
pam_access.so       pam_gdm.so            pam_namespace.so    pam_tally2.so
pam_cap.so          pam_gnome_keyring.so  pam_nologin.so      pam_tally.so
pam_debug.so        pam_group.so          pam_permit.so       pam_test
pam_deny.so         pam_issue.so          pam_pwhistory.so    pam_time.so
pam_echo.so         pam_keyinit.so        pam_rhosts.so       pam_timestamp.so
pam_env.so          pam_lastlog.so        pam_rootok.so       pam_tty_audit.so
pam_exec.so         pam_limits.so         pam_securetty.so    pam_umask.so
pam_extrausers.so   pam_listfile.so       pam_selinux.so      pam_unix.so
pam_faildelay.so    pam_localuser.so      pam_sepermit.so     pam_userdb.so
pam_faillock.so     pam_loginuid.so       pam_shells.so       pam_warn.so
pam_filter.so       pam_mail.so           pam_stress.so       pam_wheel.so
pam_fprintd.so      pam_mkhomedir.so      pam_succeed_if.so_  pam_xauth.so
```

Build Test

```
tran@tran-virtual-machine:~$ gcc -o pamtest pamtest.c -lpam -lpam_misc
```

Тестирование

```
tran@tran-virtual-machine:~$ ./pamtest tran
Credentials accepted.
Password:
WELCOME
Account is valid.
Authenticated
```

```
tran@tran-virtual-machine:~$ ./pamtest tran
Credentials accepted.
Password:
Not Authenticated
```

```
tran@tran-virtual-machine:~$ ./pamtest huan
Credentials accepted.
Password:
Not Authenticated
```

Вывод:

- Мы познакомились с работой Apparmor на примере gnome-screenshot, а также настроили selinux и продемонстрировали работу двухуровневой модели.

- В результате выполнения лабораторной работы был написан PAM-модуль. В данной реализации он запрашивает имя пользователя и пароль. Если пользователь зарегистрирован в системе и его пароль верен доступ будет получен. В противном случаем в доступе будет отказано.