Exercise 41.2: Explore the apparmor security

Please Note

This exercise can only be performed on a system (such as **Ubuntu**) where **AppArmor** is installed.

The below was tested on **Ubuntu 17.04**, but should work on other **AppArmor**-enabled systems, such as **OpenSUSE**, where the **apt** commands should be replaced by **zypper**.

On **Ubuntu**, the /**bin**/**ping** utility runs with SUID enabled. For this exercise, we will copy **ping** to **ping-x** and adjust the capabilities so the program functions.

Then we will build an **AppArmor** profile, install and verify that nothing has changed. Modifying the **AppArmor** profile and adding capabilities will allow the program more functionality.

1. Make sure all necessary packages are installed:

```
student@ubuntu:~$ sudo apt install apparm*
```

2. Create a copy of **ping** (called **ping-x**) and verify it has no initial special permissions or capabilities. Furthermore, it cannot work when executed by **student**, a normal user:

```
student@ubuntu:~$ sudo cp /bin/ping /bin/ping-x
student@ubuntu:~$ sudo ls -l /bin/ping-x

-rwxr-xr-x 1 root root 64424 Oct 17 10:12 /bin/ping-x

student@ubuntu:~$ sudo getcap /bin/ping-x
student@ubuntu:~$

student@ubuntu:~$ ping-x -c3 -4 127.0.0.1
ping: socket: Operation not permitted
```

3. Set the **capabilities** and re-try **ping-x**:

```
student@ubuntu:~$ sudo setcap cap_net_raw+ep  /bin/ping-x

student@ubuntu:~$ ping-x  -c3 -4 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.092 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.093 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.086 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2034ms
rtt min/avg/max/mdev = 0.086/0.090/0.093/0.008 ms
```

   The modified **ping-x** program now functions normally.

4. Verify there is no pre-existing **AppArmor** profile for **ping-x**, but there is a profile for **ping**. Determine the status of the current **ping** program:

```
student@ubuntu:~$ sudo aa-status
```

   The output from **aa-status** is long, so we can **grep** for the interesting lines:


```
student@ubuntu:~$ sudo aa-status | grep -e "^[[:alnum:]]" -e ping

apparmor module is loaded.
87 profiles are loaded.
51 profiles are in enforce mode.
   ping
36 profiles are in complain mode.
17 processes have profiles defined.
6 processes are in enforce mode.
11 processes are in complain mode.
0 processes are unconfined but have a profile defined.
```

We can see **ping** has a profile that is loaded and enabled for enforcement.

5. Next we will construct a new profile for **ping-x**. This step requires two terminal windows.

   The first window (**window1**)will be running the **aa-genprof** command. This will generate a **AppArmor** profile by scanning `/var/log/syslog` for **AppArmor** errors.

   The second window (**window2**) will be used to run **ping-x**. (See the **man** page for **aa-genprof** for additional information.)

   In **window1**:

```
student@ubuntu:~$ sudo aa-genprof /bin/ping-x
Writing updated profile for /bin/ping-x.
Setting /bin/ping-x to complain mode.

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:
http://wiki.apparmor.net/index.php/Profiles

Please start the application to be profiled in
another window and exercise its functionality now.

Once completed, select the "Scan" option below in
order to scan the system logs for AppArmor events.

For each AppArmor event, you will be given the
opportunity to choose whether the access should be
allowed or denied.

Profiling: /bin/ping-x

[(S)can system log for AppArmor events] / (F)inish
```

   In **window2**:

```
student@ubuntu:~$ ping-x  -c3 -4 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.099 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.120 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.114 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2041ms
rtt min/avg/max/mdev = 0.099/0.111/0.120/0.008 ms
```

   In **window1**:

   The command `ping-x` has completed, we must now instruct **aa-genprof** to scan for the required information to be added to the profile. It may require several **scans** to collect all of the information for the profile.

   Enter `S` to scan:


```
Reading log entries from /var/log/syslog.
Updating AppArmor profiles in /etc/apparmor.d.
Complain-mode changes:

Profile:    /bin/ping-x
Capability: net_raw
Severity:   8

 [1 - capability net_raw,]
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish
```

Enter `A` to allow the capability:

```
Adding capability net_raw, to profile.

Profile:        /bin/ping-x
Network Family: inet
Socket Type:    raw

 [1 - network inet raw,]
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish
```

Enter `A` to allow the network family:

```
Adding network inet raw, to profile.

Profile:        /bin/ping-x
Network Family: inet
Socket Type:    dgram

 [1 - #include <abstractions/nameservice>]
  2 - network inet dgram,
(A)llow / [(D)eny] / (I)gnore / Audi(t) / Abo(r)t / (F)inish
```

Enter `A` to add the socket type datagram to the profile:

```
Adding #include <abstractions/nameservice> to profile.

= Changed Local Profiles =

The following local profiles were changed. Would you like to save them?

 [1 - /bin/ping-x]
(S)ave Changes / Save Selec(t)ed Profile / [(V)iew Changes] / View Changes b/w (C)lean profiles / Abo(r)t
```

Enter `S` to save the new profile:

```
Writing updated profile for /bin/ping-x.

Profiling: /bin/ping-x

[(S)can system log for AppArmor events] / (F)inish
```

Enter `F` to finish:

```
Setting /bin/ping-x to enforce mode.

Reloaded AppArmor profiles in enforce mode.

               Please consider contributing your new profile!
See the following wiki page for more information:
http://wiki.apparmor.net/index.php/Profiles

Finished generating profile for /bin/ping-x.
```

6. View the created profile, which has been stored in `/etc/appamor.d/bin.ping-x`.


```
student@ubuntu:~$ sudo cat /etc/apparmor.d/bin.ping-x
# Last Modified: Tue Oct 17 11:30:47 2017
#include <tunables/global>

/bin/ping-x {
  #include <abstractions/base>
  #include <abstractions/nameservice>

  capability net_raw,

  network inet raw,

  /bin/ping-x mr,
  /lib/x86_64-linux-gnu/ld-*.so mr,

}
```

7. The **aa-genproc** utility installs and activates the new policy so it should be ready to use, and the policies can be reloaded on demand with the `systemctl reload apparmor` command. To avoid any potential issues, and verify the changes will survive, reboot the system.

   Once the system has restarted, as the user `student`, verify **ping-x** still functions with the new profile enabled. Ping the localhost by ip address:

```
student@ubuntu:~$ ping-x -c3 -4 127.0.0.1
PING 127.0.0.1 (127.0.0.1) 56(84) bytes of data.
64 bytes from 127.0.0.1: icmp_seq=1 ttl=64 time=0.057 ms
64 bytes from 127.0.0.1: icmp_seq=2 ttl=64 time=0.043 ms
64 bytes from 127.0.0.1: icmp_seq=3 ttl=64 time=0.095 ms

--- 127.0.0.1 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2027ms
rtt min/avg/max/mdev = 0.043/0.065/0.095/0.021 ms
```

8. This should work as expected. The profile is very specific, and **AppArmor** will not allow functionality outside of the specified parameters. To verify **AppArmor** is protecting this application, try to ping the **IPV6** localhost address.

   This should fail:

```
student@ubuntu:~$ ping-x -c3  -6 ::1
ping: socket: Permission denied
```

   (Note, the `-6` option means use only **IPv6** and `::1` is the local host in **IPv6**.)

   The output indicates there is a socket issue. If the system log is examined it will be discovered that our **ping-x** program has no access to **IPv6** within **AppArmor**:

```
766:104): apparmor="DENIED" operation="create" profile="/bin/ping-x"
pid=2709 comm="ping-x" family="inet6" sock_type="raw" protocol=58
requested_mask="create" denied_mask="create
```

9. To correct this deficiency, re-run **aa-genprof** as we did earlier, and in **window2**, ping the **IPv6** loopback and append the additional options

