15.9.1 Application Armor

Click one of the buttons to take you to that part of the video.

Application Armor 0:00-0:55

When you work with computers, security is always a concern.

To secure a Linux system, the first thing you should do is protect the hardware. You should keep servers in a secure room and control who has access to the desktop computers in your workspace.

You also need to make sure that only authorized individuals can log on to each computer using usernames and passwords, and use permissions to limit each user's access to files, directories, and applications.

Applications are becoming more complex and are often intertwined, sharing libraries and files scattered all over your hard disk. In addition, when a user runs an application, Discretionary Access Control gives the app all of the user account's permissions. Although this can be helpful, it also introduces allows a hacker to exploit a defect in an application and gain superuser or root privileges to a system.

In this video, we're going to discuss Application Armor, or AppArmor, which can mitigate or eliminate these threats.

We'll also cover commands you can use to manage AppArmor.

AppArmor 0:56-2:23

AppArmor is a Mandatory Access Control method used to protect your Linux systems from untrusted or unsecure processes. It locks down an application and the files it will access with absolute path names followed by the common read and write access modes.

In other words, the kernel queries AppArmor before each system call to determine whether each requested process is authorized.

To accomplish all of this, AppArmor applies a set of rules called a profile on each protected program. These plaintext profiles are stored in the /etc/apparmor.d directory and contain a list of access control rules that each program can make use of.

Most distributions that support AppArmor, such as Ubuntu and Fedora, are installed with AppArmor enabled and include a few profiles for common applications. Although creating these profiles is beyond the scope of this lesson, you should be aware that you can usually download addition profiles or create your own using the aa-genprof utility.

To see how all this works, let's look at one AppArmor implementation on an Ubuntu distribution.

Ubuntu typically includes a PDF document viewer calledEvince. When you log on to your system, you can use Evince to view your PDF files. But with the traditional Linux security model, Evince would also have access to everything else. So, if a hacker tricked you into opening a malicious PDF document, they could exploit a vulnerability in Evince and damage your system. With AppArmor, Evince only has the permissions to work with PDF documents.

AppArmor Modes 2:24-2:49

Once installed, AppArmor can operate in two modes. This first is called Enforce. The second is called Complain, or Learning.

In Enforce mode, the settings in the profiles prevent applications from taking restricted actions.

In Complain mode, the profiles loaded aren't enforced, but policy violation attempts are stored in a log file. This is useful when you want to test an AppArmor profile because it lets you see any errors that would occur in Enforce mode.

Verify AppArmor Status 2:50-3:34

To use AppArmor, you should first ensure that it's installed and enabled by running the systemctl status apparmor command. This command checks the status of the AppArmor service and tells you if it is enabled on boot.

In this example, you see that AppArmor is not running and will not run at boot.

If, for some reason, the AppArmor service isn't started, you can start it by running ‘sudo systemctl start apparmor'. Likewise, running ‘sudo systemctl enable apparmor' ensures that AppArmor is started each time your system is booted.

aa-status is another very useful command.

If you type ‘sudo aa-status' from the terminal window, you'll get a list of the loaded AppArmor modules that tells you which ones are running in Enforce mode and which ones are running in Complain mode.

Change the AppArmor Profile to Disable 3:35-4:11

Now, if a program being monitored by AppArmor isn't working, you can disable its profile using the aa-disable command.

For example, I can disable the Firefox profile by running aa-disable usr.bin.firefox as the root user or using sudo.

When this command is run, AppArmor unloads the profile from the kernel, meaning it's no longer monitoring that program, and will also prevent the profile from being loaded the next time AppArmor starts up. This command also moves the profile from the /etc/apparmor.d directory into the /etc/apparmor.d/disable directory.

Change the AppArmor Profile to Complain 4:12-4:36

However, before disabling a profile, you may want to change the AppArmor mode to complain. This lets the application run without enforcing the policy, but all the access violations the profile defines are written to the system log.

So, to change Firefox to the Complain mode, you'd type ‘aa-complain usr.bin.firefox'.

The application will run for a while, and then you can look at your log files and remove or adjust the restrictions.

Change the AppArmor Profile to Enforce 4:37-4:42

When you're finished editing, you can use the aa-enforce usr.bin.firefox command to enable the profile.

Verify the Protected Ports 4:43-5:49

aa-unconfined is another useful command. It displays a list of processes with TCP or UDP ports that don't have AppArmor profiles loaded.

This command includes three options.

First, we have --with-ss.

This is the default option. It runs the unconfined command using the ss command to find and list the processes listening on the network sockets.

Next, we have --with-netstat. This option does the same thing. But instead of using the ss command, it uses the older netstat command to find and list the processes listening on the network sockets.

And finally, we have the --paranoid option. This is the most thorough option. It displays all of the processes from the /proc filesystem with TCP or UDP ports that are not protected by AppArmor profiles.

It's important to note, however, that the aa-unconfined commands are not 100 percent accurate and, therefore, should not be used for forensics. Instead, this command should only be used as an aid to profiling all network-accessible processes in the lab.

Tunables 5:50-6:33

AppArmor also provides a way for you to tune your configuration without adjusting profiles.

Tunable items are stored as files in the /etc/apparmor.d/tunables directory.

The location of user's home directory is often tuned.

By default, many implementations of AppArmor list /home as the location for all users' home directories. But if you have some users who are using a different location for home directors, such as /exports/home, you can edit the tunable file named home to include the additional location.

For example, by default, the home file may only include

@{HOMEDIRS}=/home/.

To include the alternative path, you would simply edit the file named home and add a space and the path to the alternative home directory on the @home line.

Summary 6:34-6:59

That's it for this lesson.

In this lesson, we covered the basics of AppArmor and explained how it can help secure your system by protecting you from hackers.

We discussed how to verify the installation and status of AppArmor and how to start and enable AppArmor.

We looked at several commands that help manage AppArmor. We discussed how to disable and enable an AppArmor profile and how to switch AppArmor modes, such as changing from Enforce mode to Complain mode.

And finally, we covered how to make small adjustment to AppArmor by editing a tunable file.
