LESSON NOTES

Intro to Linux

Security

2.4.2 Executing Commands as Another User

Lesson Overview:

Students will:

- Understand how access files and execute commands as another user, including the root user, to perform administrative tasks in a Linux based system

Guiding Question: How can users securely access files or run commands that they normally would not have access to without compromising the security and integrity of the Linux system?

Suggested Grade Levels: 9 - 12

Technology Needed: None

CompTIA Linux+ XK0-005 Objective:

2.4 - Given a scenario, configure and execute remote connectivity for system management

- Executing commands as another user
  - /etc/sudoers
  - PolicyKit rules
  - Commands
    - sudo
    - visudo
    - su –
    - pkexec

This content is based upon work supported by the US Department of Homeland Security's Cybersecurity & Infrastructure Security Agency under the Cybersecurity Education Training and Assistance Program (CETAP).

Executing commands as another user

Occasionally when using a Linux system, the need for accessing a file or running a particular command may be limited or impossible based on the user that is currently logged in. In most cases, these situations involve running processes or performing administrative tasks that are only allowed to be executed by the root user. Since the root user account has full access to everything in the system, it presents a possible security threat to be logged in as that root user while performing day to day tasks. In other cases, there may be a file or program that is saved under another user and needs to be accessed by the current user. In Linux, there are different methods to run a command as a different user.

Using the **`su -`** command allows you to switch to a different user and execute a command as that user. You will be prompted to enter the password of the target user. An example of using this is **`su other-user -c "<command>"`**. This process can be run by any user if the user knows the password of the account they are attempting to use, including the root user. This again poses a potential problem in that multiple people would know the password for the root user. To remedy this, the **`sudo`** command allows a user to run a command as a different user with administrative privileges. Typically, using the **`sudo`** command would require the user to enter their password. When a user initiates a process with the **`sudo`** command, that action is also logged in the system. Allowing a user to perform an administrative action just by using their own password does not sound very secure; however, Linux systems have additional features to secure this process.
Linux systems contain the **`/etc/sudoers`** file that lists or defines which users or groups are allowed to use the **`sudo`** command. If a user attempts to use the **`sudo`** command and they are not in the file or in one of the groups listed, then they will be denied when attempting to run the command. This file is also unique in that it has a separate command for opening and editing the file, such as when a new user or group should be added to the **`sudo`** list.

The **`visudo`** command will first check to make sure the file is not currently open by another user as multiple edits occurring at the same time can lead to issues or corruption to the file. When opened, it will appear as a standards text editor; however, it also has a built-in syntax checker to assist in making sure mistakes are not made to the file that made lead to issues. Once opened, additions or alterations to the file can be made, then it can be saved and closed like a normal file.

An additional utility, PolicyKit (also referred to PolKit), can be used to manage who can perform operations that require administrative privileges. It provides a way to fine tune access control for various system and/or user specific tasks such as mounting drives, changing systems, installing software, and more. The security of the system can be enhanced by categorizing which tasks are considered administrative and which are considered regular tasks. When a user tries to perform an action requiring administrative privileges, such as modifying network settings, PolicyKit checks the rules to determine whether the user has the necessary privileges. One of the common commands used is **`pkexec`** which allows a user to execute a program as another user. When using the command, the other user can be specified; however, if not specified it will default to run the program as the root user.