Filter content in Linux

In this reading, you'll continue exploring Linux commands, which can help you filter for the information you need. You'll learn a new Linux command, **`find`**, which can help you search files and directories for specific information.

Filtering for information

You previously explored how filtering for information is an important skill for security analysts. **Filtering** is selecting data that match a certain condition. For example, if you had a virus in your system that only affected the **`.txt`** files, you could use filtering to find these files quickly. Filtering allows you to search based on specific criteria, such as file extension or a string of text.

grep

The **`grep`** command searches a specified file and returns all lines in the file containing a specified string. The **`grep`** command commonly takes two arguments: a specific string to search for and a specific file to search through.

For example, entering **`grep OS updates.txt`** returns all lines containing OS in the **`updates.txt`** file. In this example, OS is the specific string to search for, and **`updates.txt`** is the specific file to search through.

Piping

The pipe command is accessed using the pipe character (`|`). **Piping** sends the standard output of one command as standard input to another command for further processing. As a reminder, **standard output** is information returned by the OS through the shell, and **standard input** is information received by the OS via the command line.

The pipe character (`|`) is located in various places on a keyboard. On many keyboards, it's located on the same key as the backslash character (`\`). On some keyboards, the `|` can look different and have a small space through the middle of the line. If you can't find the `|`, search online for its location on your particular keyboard.

When used with **`grep`**, the pipe can help you find directories and files containing a specific word in their names. For example, **`ls /home/analyst/reports | grep users`** returns the file and directory names in the **`reports`** directory that contain **`users`**. Before the pipe, **`ls`** indicates to list the names of the files and directories in **`reports`**. Then, it sends this output to the command after the pipe. In this case, **`grep users`** returns all of the file or directory names containing **`users`** from the input it received.

Note: Piping is a general form of redirection in Linux and can be used for multiple tasks other than filtering. You can think of piping as a general tool that you can use whenever you want the output of one command to become the input of another command.

find

The **find** command searches for directories and files that meet specified criteria. There's a wide range of criteria that can be specified with **find**. For example, you can search for files and directories that

- Contain a specific string in the name,
- Are a certain file size, or
- Were last modified within a certain time frame.

When using **find**, the first argument after **find** indicates where to start searching. For example, entering **find /home/analyst/projects** searches for everything starting at the **projects** directory.

After this first argument, you need to indicate your criteria for the search. If you don't include a specific search criteria with your second argument, your search will likely return a lot of directories and files.

Specifying criteria involves options. **Options** modify the behavior of a command and commonly begin with a hyphen (**-**).

-name and -iname

One key criteria analysts might use with **find** is to find file or directory names that contain a specific string. The specific string you're searching for must be entered in quotes after the **-name** or **-iname** options. The difference between these two options is that **-name** is case-sensitive, and **-iname** is not.

For example, you might want to find all files in the **projects** directory that contain the word "log" in the file name. To do this, you'd enter **find /home/analyst/projects -name "*log*"**. You could also enter **find /home/analyst/projects -iname "*log*"**.

In these examples, the output would be all files in the **projects** directory that contain **log** surrounded by zero or more characters. The **"*log*"** portion of the command is the search criteria that indicates to search for the string "log". When **-name** is the option, files with names that include **Log** or **LOG**, for example, wouldn't be returned because this option is case-sensitive. However, they would be returned when **-iname** is the option.

Note: An asterisk (*****) is used as a wildcard to represent zero or more unknown characters.

-mtime

Security analysts might also use **find** to find files or directories last modified within a certain time frame. The **-mtime** option can be used for this search. For example, entering **find /home/analyst/projects -mtime -3** returns all files and directories in the **projects** directory that have been modified within the past three days.

The **-mtime** option search is based on days, so entering **-mtime +1** indicates all files or directories last modified more than one day ago, and entering **-mtime -1** indicates all files or directories last modified less than one day ago.

Note: The option `-mmin` can be used instead of `-mtime` if you want to base the search on minutes rather than days.

Key takeaways

Filtering for information using Linux commands is an important skill for security analysts so that they can customize data to fit their needs. Three key Linux commands for this are `grep`, piping (`|`), and `find`. These commands can be used to navigate and filter for information in the file system.