F20FO/F21FO - Digital Forensics

Lab 03: Linux Commands and Artefacts

Lab 03 Objectives

1. To use Linux commands to view digital artefacts and metadata
2. To examine different artefacts on Linux Ubuntu

The purpose of this lab is for you to explore different digital artefacts that could be used as part of a digital forensics investigation. You will focus on using standard OS commands for this purpose so that you can see the information as it's available through the OS. With this experience, you should be better able to understand and interpret data that is recovered with a DF tools.

Notes

There is nothing to submit at the end of this lab, however students are encouraged to log and record their activities and results in a document. There are some questions coloured in blue that you can use as an indication to record in your logbook (e.g., lab03_logbook). There are some answers to the questions at the end of this lab sheet.

Often the lab instructions are intentionally open ended, and you will have to figure some details out on your own. There are also some new topics in this lab, that may not be covered till future lectures. To answer some of the questions, you should do your own research to find the answer.

This lab assumes that you're working on Linux Ubuntu, using a command line interface, though the commands should also work in other flavours of Linux (possibly with some minor variations depending on the command shell used). You can perform this lab directly on a host machine as well as from Ubuntu running as a guest OS in a VM (such as the one you would have installed during Lab 01). You will need root access in to run some of the commands in this lab.

Lab 3.1: Listing commands, files, and related data

The ls command will list the files contained in a directory, including hidden files by using the "ls -a" option. For each file it can also list properties such as the owner and permissions, such as when the "long format" option is chosen as in "ls -l". Such information about a file is referred to as its *metadata*.

An important piece of metadata for files is the recording of *times* related to activities performed on files (time information is typically displayed as the date and/or time, depending on the command used). These times are important when analysing file artefacts and they can allow a digital forensics practitioner to reconstruct a timeline of activities related to a file.

Linux records the following times for each file:

- Access time. The *atime* is the last time that the content of a file was read.
- Modification time. The *mtime* is the last time that the contents of a file were modified.
- Change time. The *ctime* is the last time that the file's metadata was modified. This includes information such as the file name and file permissions.
- Creation time. The *crtime* is the time at which the file was first created sometimes referred to as the file's time of birth. The crtime is only available in more recent Linux filesystem versions.

If you run the command, "ls -l fileA", to list the file properties of fileA, which time is shown: access, modification, change, or creation?

What commands can you use to show the other times for a file?

You should create and modify (contents and metadata) of a few files to experiment with different file times. You should also reflect on the usefulness of the different times for creating a file's timeline.

Lab 3.2: Files for logs and processes

Linux logs several activities in a variety of files. In some cases, this is for administrative purposes for a network administrator, e.g., by identifying which users have successfully (or unsuccessfully) logged into a device.

In Linux, many log files are stored in the "/var/log" directory. This includes log files such as those related to login attempts, file printing, scheduled tasks, and server logs. For example, the file /var/log/auth.log (or /var/log/secure) file records login/authentication attempts, including use of the sudo command. If you are on a machine where you have root access, try entering a few commands with "sudo" in front (you may be prompted for the root password), and then "tail /var/log/auth.log" to view the last 10 authentication attempts to see your sudo command. What information are you shown in the log file that might be relevant to a digital forensics investigation?

There are other files in which Linux records other events. For example, in the home directory of each user, the file "~/.bash_history" records a history of the commands entered by the user. For the normal day-to-day use of the computer by the user, this

information can be useful, such as when a user wants to repeat earlier commands. It can also be useful to a digital forensics practitioner. Explain whether a user can delete or modify their "~/.bash_history" file (answer this without attempting to delete or modify the file).

Process information is also useful for digital forensics investigations. Every entered command starts a new process, including the opening of an application. Create a new text file with the command "echo abcd > file.txt".

Now open the file with gedit (or another editor of your choice) to edit the file from the command line as "gedit file.txt &". The "&" causes the process that is started to run in the background so that we can still use the command line at the same terminal. From the command line, enter the command "ps -a" to view processes started from the terminal and find the process identifier (PID) for the process that started the gedit application. Suppose this PID is 1234. You can view the files associated with this process at the directory "/proc/1234/".

Go to this directory and list the files. View the contents of the file cmdline with the command "xxd cmdline" (we use xxd in order to view the contents of files that may contain non-readable characters). This should show you the command that was used to start the gedit application. Look closely at the size of the cmdline file as well as all other files. What do you notice? Are the files in the /proc/ directory useful for digital forensics purposes?

Lab 3.3: File blocks and fragmentation

Move to a directory that contains a few files of varying sizes and enter "ls -l". For each file, the numeric value that is in the column to the left of the modification date is the file size. In what unit of size is the file size given? In other words, if there is a number 5 shown for a file size, what number is associated with the 5?

There are other options for showing a file's size. The command "ls -lh" will show a more human-readable file size with bytes represented as Kilobytes (K), Megabytes (M), or Gigabytes (G) where the unit used smartly depends on the magnitude of the file size. For example, suppose that with "ls -l" a file's size is shown as 246779 bytes. With "ls -lh", the size is shown as 241K, meaning 241 Kilobytes. Can you explain the calculation that was used to obtain 241K from 246779?

You can also view the file size in blocks, where a block is formed of multiple sectors on the physical disk. The first thing that you need to do is determine the block size of your file system. The command "stat -fc %s ." should do this (don't forget to include the period!).

This will likely give you a value of 4096 bytes, at least for most modern Linux systems. For example, if the sector size is 512 bytes, then a block is a grouping of 8 sectors. To see the number of blocks in a particular file, you can use the command “filefrag -b4096 –v filename”, where the second parameter is the block size. For the example file above, there would be 61 blocks. Explain why.

END OF LAB SHEET (Answers on next page)

Lab 03: Some Task Answers

Lab 3.1: Listing commands, files, and related data

- The “ls –l fileA” command shows the file’s modification time of the file named fileA.
- “ls –lu fileA” displays the access time, “ls –lc fileA” displays the change time, and “ls --time=birth fileA” displays the creation time (the “--time=birth” is not be supported on all systems). The command “stat fileA” will display all times associated with a file.

Lab 3.2: Files for logs and processes

- When you examine the auth.log file, you should see information such as the date of the authentication attempt, the fact that a “sudo session” was initiated, the user that initiated the authentication attempt, and the duration of the attempt. If you had some unsuccessful authentication attempts, you should also see these. Such information is useful for a digital forensics investigation by identifying the time that a particular user accessed a device, and that a user had root access on the device.
- Yes, a user can delete their bash_history file. We know this because the file owner is the user, and user permissions include the “w” (write) permission.
- The files in the /proc/ directory all have size 0, according to the ls command. This is because these files are part of a Linux “pseudo-filesystem” whereby they are not stored on disk storage but are only stored in memory (this is different from log files that are written to disk storage). For this reason, these files would not be readable in a static image of the storage drive and wouldn’t be available for forensic analysis. However, “live forensic analysis” could be used, as will be discussed in a later lecture.

Lab 3.3: File blocks and fragmentation

- By default, the “ls -l” command displays the file size for each file in bytes. In other words, a file with the number 5 shown for its size is 5 bytes long.
- In this case 1 Kilobyte is 1024 (2 to the power 10) bytes, not 1000 bytes.
- In this case 61 blocks of 4096 bytes is 249856 bytes of space. The file size is 246779 bytes. Thus, there are 249856-246779=3077 bytes of free space in the last block. Since this is smaller than the block size, fewer blocks (e.g., 60 blocks) would be insufficient to store the file.