Computer Forensics

Week 6: Linux and Macintosh File Systems

Kumi University
Lecturer: Lemi Agrey Oliver
codingissweet@gmail.com

Content

- Introduction to File systems
- Examining Linux File Structures
- File Structures in Ext4
- Hard Links and Symbolic Links
- Understanding Macintosh File Structures
- Acquisition Methods in macOS
- Using Linux Forensics Tools

Recap of last week

- Let's recap the key points on what was covered last week:
- We defined digital forensic tools and provided a brief overview of their importance in investigations.
- We also discussed the criteria that should be considered when selecting a digital forensic tool, emphasizing the importance of choosing the right tool for the specific requirements of an investigation.
- We categorized forensic tools into hardware and software tools and provided examples of each. For example, we learned that software tools include forensic imaging tools, data recovery tools, and analysis tools, while hardware tools include Write blockers and forensic workstations.

Recap of last week

- We then introduced the concept of a write blocker, which can be either in software or hardware form. We learned that a write blocker is used during data acquisition to prevent the alteration of forensic evidence, ensuring its integrity.
- Furthermore, we discussed the tasks performed by digital tools, which can be summarized as follows:
  - ✓ Acquisition: Collecting digital evidence in a forensically sound manner.
  - ✓ Validation and Verification: Ensuring that the collected evidence is valid and reliable.

Recap of last week+

- ✓ Extraction: Extracting relevant information from the collected evidence.
- ✓ Reconstruction: Reconstructing events based on the extracted information.
- ✓ Reporting: Documenting findings and presenting them clearly and concisely.

- ➢ Finally, we delved into the importance of validating and testing forensic tools to ensure their reliability and accuracy in forensic investigations.
- ➢ We concluded that understanding digital forensic tools and their proper use is essential for conducting effective digital investigations and ensuring that evidence is admissible in court.
- ➢ This week we are going to discuss Linux and Macintosh File Systems

Introduction to Linux File systems

- A Linux file system is a structured collection of files on a disk drive or a partition. A partition is a segment of memory and contains some specific data[1]
- Linux file system is generally a built-in layer of a Linux Operating system used to handle the data management of the storage.
- It helps to arrange the file on the disk storage. It manages the file name, file size, creation date, and much more information about a file.

Introduction to Linux File systems

- Popular Linux distributions include Ubuntu, CentOS, Mint, Fedora, and Gentoo, with Linux being the core maintained by Linus Torvalds, while other tools and interfaces are developed by different groups.
- The term "kernel" is often associated with Linux, but all UNIX-like and Windows OSs also have kernels.
- The kernel is a core component of an operating system (OS) which acts as a bridge between the software and the hardware of a computer, managing resources and providing services for all other parts of the operating system and applications

Examining Linux File Structures

- UNIX was developed in the early 1970s to be a secure, multiuser, and multithreaded operating system, leading to various UNIX-based OSs.
- The Open Group was established as a neutral standards body certifying UNIX compliance in different distributions.
- Notable UNIX distributions included SGI IRIX, SCO UnixWare, Sun Solaris, IBM AIX, and HP-UX, though many are no longer available.
- Linux, while not UNIX certified, is available in more varieties than UNIX and typically includes additional software components for a complete working environment.

Examining Linux File Structures+

- Linux file system has a hierarchal file structure as it contains a root directory and its subdirectories.
- All other directories can be accessed from the root directory.
- A partition usually has only one file system, but it may have more than one file system
- File systems manage and provide space for non-volatile storage data.
- They require a namespace for naming and organizing files, and defining naming processes and file structures.
- Metadata describes files, including size, creation date, and location.

Examining Linux File Structures ++

- File systems support a hierarchical directory structure for organizing files.
- They store advanced information about disk sections, such as partitions and volumes.

System file	Contents
`/etc/exports`	File systems exported to remote hosts; might include remote drive mappings
`/etc/fstab`	File system table of devices and mount points
`/var/log/lastlog`	User's last logon
`/var/log/wtmp`	Logon and logoff history information
`/var/run/utmp`	Current user's logon information
`/var/log/dmesg`	System messages log
`/var/log/syslog`	System log, occasionally called `system.log` or `kernel.log`
`/etc/shadow`	Master password file, containing hashed passwords for the local system
`/etc/group`	Group memberships for the local system
`/etc/passwd`	Account information for the local system

- Figure 1: Linux File systems[2]

Directory Structure

- Directories store and help locate files when needed.
- They are analogous to physical folders on a desktop.
- Directories can be organized in a tree-like hierarchy.
- The Linux directory structure is well-documented in the Linux FHS.
- Accessing directories is done via sequentially deeper names linked by '/' (forward slash), forming paths like /var/spool/mail and /var/log.

Directory Structure+

Directory	Description
/ (root filesystem)	The root filesystem is the top-level directory of the filesystem. It must contain all of the files required to boot the Linux system before other filesystems are mounted. It must include all of the required executables and libraries required to boot the remaining filesystems. After the system is booted, all other filesystems are mounted on standard, well-defined mount points as subdirectories of the root filesystem.
/bin	The /bin directory contains user executable files.
/boot	Contains the static bootloader and kernel executable and configuration files required to boot a Linux computer.
/dev	This directory contains the device files for every hardware device attached to the system. These are not device drivers, rather they are files that represent each device on the computer and facilitate access to those devices.
/etc	Contains the local system configuration files for the host computer.
/home	Home directory storage for user files. Each user has a subdirectory in /home.
/lib	Contains shared library files that are required to boot the system.
/media	A place to mount external removable media devices such as USB thumb drives that may be connected to the host.
/mnt	A temporary mountpoint for regular filesystems (as in not removable media) that can be used while the administrator is repairing or working on a filesystem.

- Figure 2: The top level of the Linux filesystem hierarchy. [3] continue…

Directory Structure++

/opt	Optional files such as vendor supplied application programs should be located here.
/root	This is not the root (/) filesystem. It is the home directory for the root user.
/sbin	System binary files. These are executables used for system administration.
/tmp	Temporary directory. Used by the operating system and many programs to store temporary files. Users may also store files here temporarily. Note that files stored here may be deleted at any time without prior notice.
/usr	These are shareable, read-only files, including executable binaries and libraries, man files, and other types of documentation.
/var	Variable data files are stored here. This can include things like log files, MySQL, and other database files, web server data files, email inboxes, and much more.

❖Figure 2: The top level of the Linux filesystem hierarchy. [3]

Installing Ubuntu on Virtual Machine[2]

- Start VirtualBox, and click the **New** icon at the upper left to start the Create Virtual Machine Wizard.
- In the Name and Operating System window, type **Ubuntu -20.4.04** for the virtual machine name. Accept the default settings, and click **Next**.
- In the Memory size window, increase the setting to **1024**, and then click **Next**.
- In the Hard drive window, click **Create a virtual hard drive now**, and then click **Create**. In the Hard drive file type window, click **Virtual Machine Disk (VMDK)**, and then click **Next**. In the "Storage on physical hard drive" window, click the **Dynamically allocated** option button, and then click **Next**.
- In the File location and size window, increase the setting to **20 GB**, and then click **Create**. Leave VirtualBox open.

Installing Ubuntu on Virtual Machine[2]+

- Start a Web browser, go to **www.ubuntu.com/download/desktop**, and download the ISO image for Ubuntu **Ubuntu -20.4.04**
- In the Oracle VM VirtualBox Manager, click the **Settings** icon.
- Click **Storage** in the left pane. In the Storage Tree section, click **Empty** under Controller: IDE. In the Attributes section on the right, click the CD icon. Click **Choose Virtual Optical Disk File**. Navigate to the folder where the ISO file is stored, double-click the ISO file, and then click **OK**.
- In the Oracle VM VirtualBox Manager, click the **Ubuntu -20.4.04** virtual machine, and then click the **Start** icon. The VM should follow a standard OS installation. Accept the default settings. Leave the virtual machine running for the next activity

Understanding some basic Linux commands

- Before delving into Linux forensics tools, an opportunity is provided to review certain commands.
- For instance, the uname command is used to identify a machine's name.
- Displaying file listings and permissions is valuable for investigators.
- The > character redirects command output to a specified file, overwriting it if it exists or creating a new one if it doesn't.
- Using >> adds output to the end of an existing file.
- Command outputs can be viewed in the terminal window or appended to a log file using >> ~/my.log.

Understanding some basic Linux commands+

- The echo command can add notes or headings to the log file, and blank lines can enhance readability.
- Caution is warranted with the > character as it replaces the entire file without warning.
- Linux commands utilize options to modify their behavior.
- Letter arguments can be grouped or entered separately without distinction.
- Arguments comprising multiple letters require two -- characters and cannot be grouped, e.g., ls --all.

File Structures in Ext4

- Linux supports various file systems, with Ext2 and Ext3 being early standards. Ext3 introduced journaling for file recovery after crashes.
- Ext4, introduced later, supports larger partitions, better large file management, and flexible feature addition. It's now the standard file system for most Linux distributions.
- In UNIX/Linux, everything, including disk drives and directories, is considered a file with properties and methods.
- The file system is defined by four components: boot block (startup instructions), superblock (system information and inode management), inode block (allocation unit control), and data block (file and directory storage).

File Structures in Ext4+

- Inodes are assigned to each file allocation unit, managing file and directory access.
- Data blocks store directories and files, linked directly to inodes, similar to clusters in Microsoft file systems.
- Block sizes in Linux volumes range from 1024 to 4096 bytes, with a data block equivalent to a cluster of disk sectors in FAT or NTFS volumes.

Inodes

- **Inodes** contain file and directory metadata and provide a mechanism for linking data stored in data blocks. [2]
- An inode contains 13 pointers that link to data blocks and other pointers where files are stored.
- Pointers 1 through 10 are direct pointers that link directly to data storage blocks on the disk, each associated with one block of data storage.
- As a file grows, up to three layers of additional inode pointers can be provided. The first 10 pointers are indirect pointers, the second layer has double-indirect pointers, and the third layer has triple-indirect pointers.

Inodes+

- To expand storage allocation, the OS uses the 11th pointer in the original inode, which links to 128 pointer inodes, each pointing to 128 blocks in the drive's data block.
- If all 10 pointers in the original inode are used, the 11th pointer links to another 128 pointers, creating a second layer of pointers.
- The 12th pointer of the original inode can be used to link to another 128 inode pointers, creating a second layer of pointers that are linked directly to blocks in the drive's data block.
- The 13th pointer links to 128 pointer inodes, each pointing to another 128 pointers, creating a third layer of pointers.

Inodes++

- Disks typically have more storage capacity than stated due to the presence of bad sectors. For instance, a 240 GB disk might actually have 240.5 GB of usable space.
- Windows does not track bad sectors, but Linux does so in an inode known as the bad block inode, with inode 1 designated for this purpose.
- Some forensic tools overlook inode 1, potentially missing valuable data in their recovery efforts.
- To manipulate bad block information, a person could access the bad block inode, list good sectors within it, and then conceal information in these "bad" sectors.
- Linux provides tools like badblocks, mke2fs, and e2fsck for managing bad blocks, with precautions in place to prevent accidental data loss. Accessing these tools typically requires root permissions.

Hard Links and Symbolic Links

- A hard link is a pointer that allows accessing the same file by different filenames, referring to the same inode and physical location on a drive.
- Originally, hard links were used for different users to access the same file, ensuring changes made by one user would be visible to others.
- Hard links are created using the ln command and require all linked files to be on the same physical drive.
- Each file has an inode containing a link count field specifying the number of hard links pointing to it.
- When a file is deleted, the link count decreases by one, and when it reaches zero, the file is effectively deleted.

Hard Links and Symbolic Links +

- Directories have at least two hard links, representing the directory itself (.) and its parent (..).
- Subdirectories add to the parent directory's link count due to the dot-dot reference in each subdirectory.

Hard Links and Symbolic Links ++

- Symbolic links, also known as "soft links" or "symlinks," are pointers to other files that are not included in the link count.
- Unlike hard links, symbolic links can point to items on other drives or locations on the network by using an absolute path.
- Symbolic links have their own inode, separate from the inode of the item they point to.
- They depend on the continued existence of the destination they point to, and if the destination is removed, the symbolic link stops working.

Hard Links and Symbolic Links +++

- Symbolic links are easier to identify on a running Linux system compared to hard links.
- Symbolic links identify their destination by name and path, unlike hard links which use inode numbers.
- To create symbolic links, the ln -s command is used.

Understanding Macintosh File Structures

- The current Macintosh operating system is macOS, version 10.13, called High Sierra, with other versions including Sierra (10.12.5), El Capitan (10.11), Yosemite (10.9), Snow Leopard (10.6), Lion (10.7), and Mountain Lion (10.8).
- macOS uses the Apple File System (APFS), providing improved security, encryption, and performance, while still supporting HFS+ drives.
- Apple's operating systems have evolved since 1984, with OS X marking a shift to Intel processors and UNIX-based architecture.
- Before OS X, Macintosh used the Hierarchical File System (HFS), later upgrading to HFS+ with Mac OS 8.1, allowing for more efficient disk use with larger volumes.

Understanding Macintosh File Structures+

- macOS High Sierra also supports the Unix File System (UFS) and introduces APFS, which copies metadata when writing data to a device for crash protection.
- Updated information on macOS can be found at www.macworld.co.uk/news/mac-software/macos-sierra-latest-version-updates-beta-features-3630374.

An Overview of Mac File Structures

- Older versions of macOS used a file structure with two parts: a data fork for storing user-created data and a resource fork for metadata and application information.
- Both forks contain essential information such as resource maps, headers, window locations, and icons for each file.
- The data fork typically stores user-created data like text or spreadsheets, while the resource fork contains additional information for application files, such as menus, icons, and executable code.
- A volume in macOS is any storage medium used for storing files, with allocation blocks and logical blocks used for organizing data.

An Overview of Mac File Structures+

- Logical blocks are collections of data that cannot exceed 512 bytes, and when a file is saved, it's assigned to an allocation block, which is a group of consecutive logical blocks.
- HFS and HFS+ file systems in macOS have two descriptors for the end of a file: logical EOF, the actual ending of a file's data, and physical EOF, the number of bytes allotted on the volume for a file.
- macOS reduces file fragmentation by using clumps, which are groups of contiguous allocation blocks, and volume fragmentation is minimized by adding more clumps to larger files.

Forensics Procedures in Mac

- Linux and macOS file structures have some differences. For instance, Linux uses /home/username and /root directories, while macOS uses /users/username and /private/var/root. The /home directory exists in macOS but is empty. Additionally, macOS users have limited access to other user accounts' files, and the guest account is disabled by default, with guest files deleted at logout.
- For forensics procedures in macOS, it's important to know where file system components are located and how files and file components are stored. Application settings are stored in plaintext, plist files (plain XML plists and binary plists), and the SQLite database.

Forensics Procedures in Mac +

- Plist files, located in /Library/Preferences, are preference files for installed applications and can be viewed using special editors like the one available at the Apple Developer website or PlistEdit Pro.
- The SQLite database, used for application data storage, can be viewed using the SQLite Database Browser.
- The new macOS feature called unified logging, located in /var/db/diagnostics, includes utilities like log, log collect, and log show, which are useful for forensics examination.
- Other files that may contain useful information for investigations include SystemVersion.plist, NetworkInterfaces.plist, flatfile.db, users plist files, and the hash file containing account passwords.

Forensics Procedures in Mac ++

- FileVault, introduced with macOS 10.3, encrypts and decrypts a user's /users directory, but vulnerabilities were found in earlier versions. FileVault2 was introduced to address these vulnerabilities and encrypts the whole disk with 128-bit AES encryption.
- Keychains have been used since macOS 8.6 to manage passwords for applications, websites, and system files, with keychain files located in various places like /System/Library/Keychains and /Library/Keychains. The Mac application Keychain Access enables users to restore passwords.
- Deleted files in macOS are moved to the Trash folder, but files deleted at the command line do not appear in the Trash.

Forensics Procedures in Mac +++

- Several vendors offer software for examining the macOS file system, including BlackBag Technologies, SubRosaSoft MacForensicsLab, ProDiscover Forensic Edition, and freeware tools like Sleuth Kit and Autopsy.

Acquisition Methods in macOS

- To examine a macOS computer, you need to create an image of the drive, which can be challenging due to Apple's design. For example, opening a Mac Mini or MacBook Air requires special tools.
- You need a macOS-compatible forensic boot CD/DVD to create an image, which can then be written to an external drive like FireWire or USB.
- BlackBag Technologies offers acquisition tools for macOS and a forensic boot CD called MacQuisition for making drive images. They also provide some free tools for forensic examiners.
- After acquiring the image, you can examine the file system using a forensic tool. The tool depends on the image file's format. For example, if you used EnCase, FTK, or X-Ways Forensics to create an Expert Witness (.e0l) image, you must use one of these tools to analyze it.

Acquisition Methods in macOS+

- If you made a raw format image, you can use various tools like BlackBag Technologies Macintosh Forensic Software, SubRosaSoft MacForensicsLab, X-Ways Forensics, or AccessData FTK.
- BlackBag Technologies Macintosh Forensic Software and SubRosaSoft MacForensicsLab have a function for disabling and enabling Disk Arbitration, a macOS feature for automatic mounting when a drive is connected via USB or FireWire. Disabling this function allows you to connect a suspect drive to a Mac without a write-blocking device.

Using Linux Forensics Tools

- Linux forensics tools are useful when Windows tools fail or you have trouble booting a Windows machine. These tools can analyze UNIX and Linux file systems.
- Commercial tools like OSForensics, X-Ways Forensics, Guidance Software EnCase, and AccessData FTK can analyze Linux Ext2, Ext3, Ext4, ReiserFS, and Reiser4 file systems.
- Freeware tools include Sleuth Kit with its Web browser interface, Autopsy Forensic Browser, and Foremost.
- Sleuth Kit, previously called TASK, is based on The Coroner's Toolset (TCT) and designed for network analysis to investigate attackers.

Using Linux Forensics Tools+

- Foremost, developed by the U.S. Air Force Office of Special Investigations and the Center for Information Systems Security Studies and Research, is a carving tool that can read many image file formats. It has a configuration file, foremost.conf, for adding new file formats.
- To update the foremost.conf file, use a hex editor to determine the new format's header and footer values, and a text editor to update the file. The file is typically in the /usr/local/etc directory.
- A tarball is a highly compressed data file containing one or more files or directories and their contents. It's similar to Windows zip utilities and typically has a .tar or .gz extension.

Summary

- Introduction to File Systems: File systems organize and manage data on storage devices. They define how data is stored, accessed, and managed.
- Examining Linux File Structures: Linux uses a hierarchical file system where files are organized in directories (folders) that can contain other directories or files. Understanding this structure is crucial for forensic analysis.
- File Structures in Ext4: Ext4 is a popular file system used in Linux. It improves upon earlier versions like Ext3, offering better performance and scalability.
- Hard Links and Symbolic Links: Hard links and symbolic links are ways to create shortcuts to files in Linux. Hard links directly point to the file's inode, while symbolic links are pointers to the file's path.

Summary+

- Understanding Macintosh File Structures: macOS uses the Apple File System (APFS), which offers better security, encryption, and performance. It also supports older file systems like HFS+.
- Acquisition Methods in macOS: To examine a macOS computer, you need to create an image of the drive, which can be challenging due to Apple's design. Special tools and a forensic boot CD are often required.
- Using Linux Forensics Tools: Linux forensics tools are useful for analyzing UNIX and Linux file systems. Commercial tools like OSForensics and freeware tools like Sleuth Kit can be used to analyze file systems like Ext2, Ext3, Ext4, ReiserFS, and Reiser4.

Reference

[1] *Linux file system* (n.d.). https://www.javatpoint.com/linux-file-system

[2]] Nelson, B., Phillips, A., & Steuart, C. (2019). *Guide to computer forensics and investigations* Course Technology Cengage Learning.

[3] Both, D. (n.d.). *An introduction to linux filesystems*. Opensource.com. https://opensource.com/life/16/10/introduction-linux-filesystems