www.ijprems.com
editor@ijprems.com

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)
(Int Peer Reviewed Journal)
Vol. 04, Issue 09, September 2024, pp : 466-481

e-ISSN : 2583-1062

Impact Factor : 7.001

SECURE DELETION TECHNIQUES AND THEIR EFFICACY

Sheikh Masem Mandal[1]

[1]Lovely Professional University, India.

DOI: https://www.doi.org/10.58257/IJPREMS35989

ABSTRACT

As our reliance on digital devices grows, ensuring the safety and reliability of our data storage methods becomes increasingly critical. Hard drives (HDDs) and solid-state drives (SSDs) serve as repositories for our information, maintaining its integrity even during power outages. Every action we take on these devices—creating, modifying, or deleting files, installing programs, organizing folders—is a fundamental aspect of data storage. At the core of this infrastructure are file systems, which silently manage the whereabouts of our data.

This research seeks to explore the factors influencing the recoverability of deleted files, focusing on three main variables: the operating system (e.g., Windows, Ubuntu), the drive type (HDD or SSD), and user activity (such as device usage patterns, internet browsing, and file downloads).

In the realm of digital forensics, where thorough investigation is imperative, understanding the speed at which new data overwrites deleted files is paramount. This study aims to delve into how operating systems handle deleted data and how everyday user behaviours impact the potential recovery of deleted files. Ultimately, this research endeavours to enhance both data security and the efficacy of digital forensics inquiries.

1 INTRODUCTION

1.1 Background

In our digital age, the reliance on devices capable of retaining data even without power is paramount. Hard disk drives (HDDs) and solid-state drives (SSDs) serve as the primary storage mediums for files and data. Every action on our devices, from installing software to creating or modifying files, contributes to the data ecosystem. Operating silently in the background, file systems meticulously organize and manage these files.

Three key factors significantly impact the storage and deletion processes on these devices:

- File System: Various systems like ext4, NTFS, and FAT employ distinct strategies for managing files, affecting their accessibility and deletion protocols.
- Operating System: Operating systems like Windows, macOS, and Linux handle files differently, influencing how data is stored, accessed, and potentially deleted.
- Disk Type: Hard disk drives (HDDs) and solid-state drives (SSDs) exhibit differing behaviors in data storage, affecting the speed and efficiency of file operations and deletions.

When a file is deleted, the file system typically removes its reference and marks the space as available for new data. However, the actual data may persist on the disk until overwritten by new information. This residual data makes the recovery of deleted files possible, particularly on solid-state drives, where remnants may linger even after deletion. Advanced data recovery software can scan for these remnants and attempt to reconstruct deleted files.

For complete and irreversible data removal, techniques such as overwriting the deleted file location with random data or zeroes/ones are recommended. This ensures that the information becomes genuinely unrecoverable, safeguarding sensitive data from unauthorized access or exploitation.

Numerous variables affect the rate at which data from deleted files is overwritten, thereby impacting the potential for recovery:

- File System: The diverse management mechanisms of file systems like ext4, NTFS, and FAT result in varied speeds of overwriting deleted data. For instance, some file systems may prioritize efficient space allocation, leading to faster overwriting, while others may prioritize data integrity, potentially prolonging the persistence of deleted files.
- Disk Type: Solid State Drives (SSDs) and traditional Hard Disk Drives (HDDs) exhibit contrasting behaviors in data storage. SSDs, owing to their flash memory technology, typically overwrite deleted data more rapidly than HDDs due to their lack of physical moving parts and faster access times.
- Operating System: Windows, macOS, Linux, and other operating systems employ distinct file management strategies that influence the speed at which data is overwritten following file deletion. Operating systems with sophisticated file management algorithms may expedite overwriting deleted data, while less optimized systems may exhibit slower rates.

- User Activity: The frequency and intensity of user interactions, such as downloading files, browsing the web, and engaging in data-intensive tasks, contribute significantly to the rate of data overwriting. Continuous user activity increases the likelihood of swift overwriting of deleted files as new data is generated and stored on the disk.

Furthermore, this research delves into the impact of routine user actions, including system shut- downs, reboots, and the utilization of cleaning software, on the persistence or eradication of deleted files within storage devices. By examining these multifaceted factors, the study aims to provide comprehensive insights into the dynamics of data retention and deletion in digital environments, thereby informing strategies for data security and forensic investigations.

2 INVESGATIVE FOCUS

This study seeks to investigate the destiny of files that have been permanently deleted from digital devices. It aims to delve into the mechanisms employed by operating systems in managing these deleted files and to analyze the impact of everyday user behaviors on their longevity, specifically focusing on how long they remain recoverable.

The research will explore the intricate processes involved in handling deleted files across various operating systems, shedding light on differences in their approaches and efficiencies. Additionally, it will scrutinize the influence of routine user activities, such as system usage patterns, file downloads, and internet browsing, on the duration for which deleted files persist and remain susceptible to recovery.

The primary audience for the study’s findings is anticipated to be professionals engaged in forensic investigations. By gaining insights into the factors affecting the recoverability of deleted files, forensic experts can enhance their methodologies and tools for digital evidence collection and analysis, ultimately strengthening the integrity and reliability of forensic investigations in the digital realm.

3 LITERATURE REVIEW

This study examines the duration for which deleted files remain recoverable on Windows 10 PCs. The research adopts an empirical methodology to assess file persistence specifically within the NTFS file system. The results indicate that deleted files can persist on the system for extended periods, influenced by variables such as file format and storage medium. This enduring presence poses a potential security concern, as deleted data may still be accessible, highlighting the need for robust data management and security measures.

This research builds work by expanding the scope of investigation in several dimensions. While the research work was confined to examining hard disk drives within a Windows 10 environment and the Linux OS, this research broadens its focus to include solid-state drives (SSDs) and explores the dynamics within both Windows 10 and Linux (Ubuntu 24.04 LTS) operating systems in both SSD’s and HDD’s, each with its unique file system and filetype.

Moreover, this research encompasses a wider range of file formats, enhancing the comprehensiveness of the investigation. Additionally, this study delves into user actions not explored by Khan[6], further enriching the understanding of factors influencing file persistence.

The research design draws inspiration from prior studies that follow a similar methodology of creating test files, deleting them, and subsequently analyzing the device for any residual data. These earlier investigations spanned various storage devices and media, including HDDs, SSDs, and USB drives, other factors such as file system, file type, and available storage space that impact file persistence. These studies collectively revealed that deleted files can persist for prolonged durations, even after formatting or overwriting.

By assimilating these foundational insights and recognizing the contrasting functionalities of HDDs and SSDs, this research endeavors to compare both storage types comprehensively.

Data recovery presents a significant challenge for Internet of Things (IoT) devices due to their simplified storage solutions and constrained resources compared to traditional computers. With limited resources and less sophisticated file systems, recovering data from IoT devices becomes especially daunting in the face of power surges, software crashes, or physical damage. While certain high-end IoT devices may offer built-in protections, most resource-constrained devices necessitate specialized tools or the expertise of a data recovery professional. Ongoing research endeavors seek to develop lightweight data recovery methods tailored specifically to the unique storage configurations and capabilities of various IoT devices. This study serves as a foundational step in this pursuit by introducing a method to analyze data persistence, a critical aspect of file recovery.

The notion of data persistence bears resemblance to file recovery techniques employed in conventional disk systems, which have been extensively explored, particularly for file systems like Ext2/3. Recognizing the limitations of existing methods, researchers have proposed improved techniques for recovering deleted files from these systems. Their approach combines the strengths of both direct and indirect recovery strategies to enhance overall performance.

Through rigorous testing involving a predetermined set of deleted files, the research assessed recovery rates and processing times.

The results of their study indicate that the proposed technique surpasses existing methods in terms of both efficiency and effectiveness, providing promising insights for future advancements in data recovery methodologies.

For this research, comprehending file allocation across different file systems is paramount. A pertinent study conducted by other researchers delved into the chronology of the FAT32 file system utilized in Windows. Their objective centered on determining the age of file fragments, crucial for establishing legal admissibility as evidence. Through their analysis, they investigated the correlation between allocated clusters and file creation dates, as well as explored the feasibility of estimating a file fragment’s age based on the creation dates of nearby files.

The study revealed that the accuracy of age approximation is significantly influenced by the number of neighboring files considered. Interestingly, analyzing more than five adjacent files tended to reduce accuracy due to the tendency of drives to allocate files in contiguous clusters within a session. Consequently, the research concluded that a nuanced understanding of the allocation rules specific to each hard drive offers valuable guidance for investigators regarding which evidence to pursue and which to discard in legal proceedings. Their in-depth analysis of the FAT32 file system provides invaluable insights that align closely with the objectives of this research, offering a solid foundation for further exploration.

This research capitalizes on the Digital Body Farm (DBF) concept introduced by Agada[13] et al., which provides a controlled environment to simulate real-world conditions for studying the decay of deleted files on various storage devices over time. The insights gained from this endeavor are instrumental in advancing digital forensics methodologies. Agada[13] et al.’s DBF comprises a software suite designed to automate the creation, deletion, and tracking of changes to files and their metadata. Their experiments, conducted across hard drives, solid-state drives, and USB flash drives using the DBF, reveal that the rate of decay for deleted files is contingent upon the specific storage device and file system employed. This underscores the DBF’s significance as a valuable tool for digital forensics researchers seeking to investigate file persistence dynamics and devise enhanced data recovery techniques.

Furthermore, the book "File System Forensic Analysis[1]" serves as a cornerstone for this research by furnishing a comprehensive exploration of file system forensics. It covers fundamental aspects such as file system basics, structure, allocation, and analysis techniques, providing a robust understanding of the subject matter. The book’s discussion on forensic tools and techniques, including carving and data recovery methodologies, enriches the research by offering practical insights and methodologies for investigation and analysis. Drawing upon the DBF concept and the foundational knowledge provided by "File System Forensic Analysis," this research aims to advance the understanding of file persistence and improve data recovery practices in digital forensics.

Two notable studies have contributed valuable insights into file system behavior and data persistence, albeit employing slightly different methodologies compared to this research.

Fairbanks[14] et al. presented a technique for measuring data persistence utilizing the Ext4 file system journal. Their approach involves the creation of test data, subsequent deletion of the file, and continuous monitoring of the journal to track the duration for which the data remains present. The successful implementation of their method underscores its efficacy and its potential applications in the realm of digital forensics. This study underscores the significance of comprehending file system behavior in forensic analysis and suggests extending this technique to explore its applicability across other file systems.

Additionally, another pertinent study conducted by unnamed researchers addressed the persistence of deleted file information and its forensic recovery potential. By challenging the assumption of complete data erasure upon deletion, they elucidated various mechanisms through which file information can persist, including file slack space, unallocated clusters, and system files. The study also delved into the impact of fragmentation on data persistence, providing examples of successful deleted data recovery efforts. While the methodologies employed in this study differed slightly from those in the current research, the understanding garnered regarding data persistence mechanisms has proven beneficial in advancing forensic analysis techniques. By integrating insights from these studies with the methodologies employed in this research, a comprehensive understanding of file persistence dynamics can be attained, thereby enhancing data recovery practices in digital forensics. The authors delve into the intricate realm of file recovery techniques, tools, and the complexities associated with interpreting recovered data.

They underscore the ethical considerations and legal obligations inherent in forensic investigations, emphasizing the necessity for proper authorization prior to conducting such examinations. Their work illuminates the potential for deleted information to persist and underscores the importance of thorough examinations in digital forensics, where file recovery serves as a pivotal tool. Furthermore, another study by Garfinkel[11] delves into the challenges surrounding the representation of digital forensic data [16]. Given its inherent complexity and heterogeneity, representing such data in a manner that is both comprehensible to humans and machine-readable poses significant challenges.

The research underscores the critical need for a standardized format for exchanging digital forensic data. Such standardization not only streamlines data sharing among investigators but also facilitates the development of novel digital forensics tools and systems, ultimately advancing the field's capabilities and efficacy. Incorporating insights from these studies enriches the understanding of file recovery practices and data representation challenges, ultimately contributing to the enhancement of digital forensic methodologies and practices.

Garfinkel[17] explores the concept of a standardized format for digital forensics data through the investigation of DFXML (Digital Forensics XML) format in separate studies [16, 18]. DFXML, an XML-based approach, prioritizes both human readability and machine parsing, offering extensibility for the inclusion of new data types as necessary. Its versatility enables the representation of a broad spectrum of digital forensic data while ensuring comprehensibility for both humans and machines. Consequently, this research adopts DFXML as the primary tool for investigating deleted files.

In a complementary study, Garfinkel[18] provides a detailed account of the development of the DFXML project [18]. The primary objective was to establish a format conducive to data exchange among various forensic tools and systems, while ensuring readability across human and machine interfaces. Leveraging XML, DFXML facilitates extensibility for accommodating future data types. Moreover, this research introduces the DFXML toolset, a suite of open-source software tailored specifically for generating and analyzing DFXML files. By leveraging the insights gleaned from these studies, this research endeavors to enhance the efficiency and effectiveness of digital forensic investigations, particularly in the realm of file recovery and data analysis.

This research addresses a common challenge encountered in digital forensics: the arduous process of manually extracting metadata from diverse digital evidence pieces during investigations. In response, the DFXML (Digital Forensics XML) format was developed, offering a standardized approach to represent and exchange metadata across various forensic tools and platforms, thus enhancing workflow efficiency. The study delves comprehensively into the architecture of DFXML files, elucidating how they encompass a wide spectrum of metadata types. This encompasses intricate details on file system metadata, comprehensive insights into allocated and unallocated space, as well as granular information pertaining to individual files and directories. Additionally, the research extensively explores the capabilities of the DFXML toolset, showcasing its functionalities in generating DFXML files, facilitating seamless conversion between file formats, and enabling in-depth analysis of DFXML data for multifaceted investigative purposes. By providing these insights, the research offers substantial value to both digital forensics research and practitioners. It underscores the advantages of adopting a standardized, machine- readable format for the management and dissemination of metadata pertaining to digital evidence. Particularly noteworthy is the DFXML toolset, which emerges as a potent asset, demonstrating its potential to expedite investigations through its provision of practical tools meticulously designed for seamless interaction with DFXML files.

4 METHODOLOGY

4.1 Experimental Environment

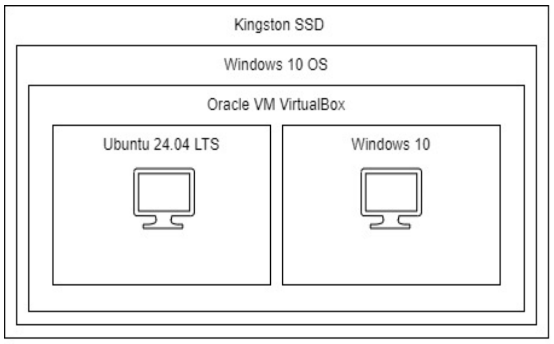
In this investigation, two drives were utilized: a 465GB solid-state drive (SSD) and a 495GB hard disk drive (HDD), both integrated into the Toshiba laptop operating Ubuntu 24.04 LTS in a dual- boot configuration with Windows 10. These machines serve as the foundational hosts for the testing environment.

To facilitate experimentation, VirtualBox software [19, 20] was installed on the Ubuntu host machines. Subsequently, the test operating systems were installed within virtual machines to leverage several advantages.

- Firstly, virtual machines enable the pausing of running operating systems and the capturing of snapshots at any point, a crucial feature for subsequent stages of the experiment.
- Secondly, unlike physical disks, virtual machines offer the capability to pre-allocate storage space.

For this study, Ubuntu 24.04 LTS [21] and Windows 10[24] were selected as the test operating systems. During the creation of virtual machines, a fixed size of 50GB was allocated to each. This fixed allocation was imperative because dynamically allocated disks, which adjust space usage based on data storage, would commence with zero allocated space. In contrast, fixed allocation assigns the entire 50GB chunk to the virtual machine upon creation.

A shared folder has been established within the virtual machines to enhance the efficiency of experimental setup procedures. This designated folder facilitates the seamless transfer of datasets between the host machine and the virtual environments. By eliminating the requirement for manual data transfer onto each test machine, this setup saves valuable time and ensures a high level of consistency across experiment



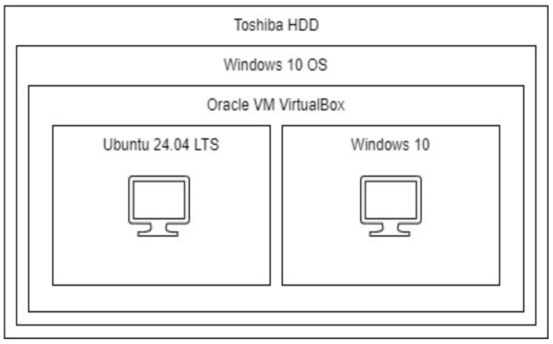


Figure 1: Oracle VM VirtualBox Layout

4.2 Experimental Data Setup

The experiments incorporated a standardized dataset totalling 5 GB, utilized across both Ubuntu and Windows operating systems under evaluation. This dataset was carefully curated to encompass a diverse array of files, varying in size, type, and content, ensuring a comprehensive analysis of system performance.

Complementing the dataset, a suite of programs was installed on the test machines. These programs underwent selection based on their compatibility with both Ubuntu and Windows environments, thereby enabling consistent evaluation across operating systems.

A pivotal aspect of the experimental setup involved the utilization of non-resident files. This decision was motivated by the potential skewing effect resident files, stored within the Master File Table (MFT) [25], could have on the results. By prioritizing non- resident files, the experiments aimed to maintain the integrity and accuracy of the performance metrics obtained.

4.3 Data Sources

A diverse array of sample files was gathered from multiple reputable sources to enrich the experimental dataset. A notable portion of the dataset was sourced from institutions such as:

1 Images

2 Bulkdata PDF

3 Text Files

4 Free Malware Samples and many more...

Expanding beyond traditional file types, image files from various operating systems were included to further diversify the dataset:

1 Windows 10 ISO

2 Ubuntu ISO

To facilitate comprehensive testing, a selection of programs was installed on both Ubuntu and Windows machines:

1 Mozilla Firefox

2 VLC Media Player

3 Steam

4 Google Chrome for Windows 10 OR Chromium for ubuntu 24.04 LTS

This array of installed programs ensured a thorough evaluation of system performance across different software environments.

Table 1: DataCollection

File type Extensions	Count
Text Files	5000
JPG	25
PNG	100
Word Document	20
Excel files	10
Windows Executables	5
ubuntu packages	3
Zip files	2
Videos	25

In both Ubuntu and Windows operating systems, a multitude of background services contribute to overall functionality, although only a subset is essential for system operation.

For Linux-based systems, the ‘systemctl list-unit-files –type=service’ command is invaluable for gaining insight into these services. It provides essential information in three key areas:

1 Service Name: This identifies each specific service currently running on the system.

2 Service State: This field indicates the current operational status of each service.

3 Vendor Preset: This information denotes whether the service is enabled or disabled by default, offering insight into system configuration.

Understanding these aspects is crucial for managing and optimizing system resources effectively.

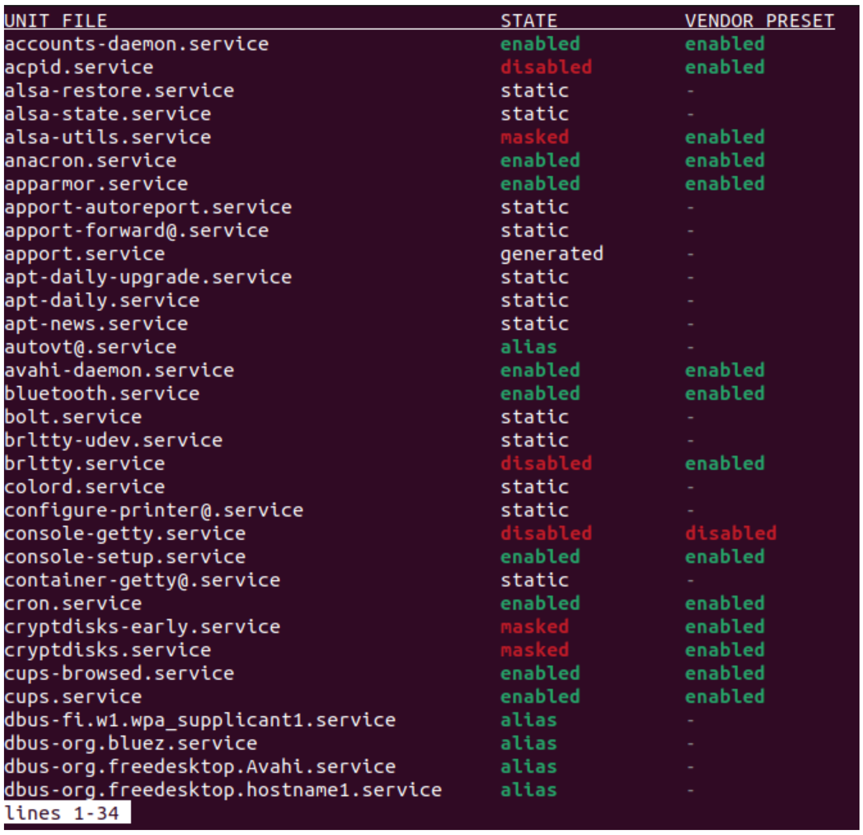
Table 2: Systemd service/unit states

Service status	Description
active (running) active (exited) active (waiting) inactive enabled disabled static masked alias linked	Service or daemon is running in the background. Service successfully started from the config file. Our service is running but waiting for an event such as CPUS/printing event. Service is not running. Service is enabled at boot time. Service is disbled and will not be started at Linux server boot time. Service cannot be enabled on Linux. Service is completely disabled and any start operation on it always fails. Service name is an alias. Made available through one or more symlinks to the unit file

To create a more controlled environment for your experiment and ensure that system resources are optimized, it’s advisable to disable any unnecessary background services. These services typically indicated as "active," "enabled," or "linked" in system settings, can inadvertently consume resources, and affect the performance of your experiment. Detailed lists of which services were deactivated, and which were deemed essential and kept running. By streamlining the operational services to only those that are crucial, you can achieve a cleaner and more dependable testing setup. This focused approach not only minimizes external variables but also enhances the overall efficiency and accuracy of your experimental results.

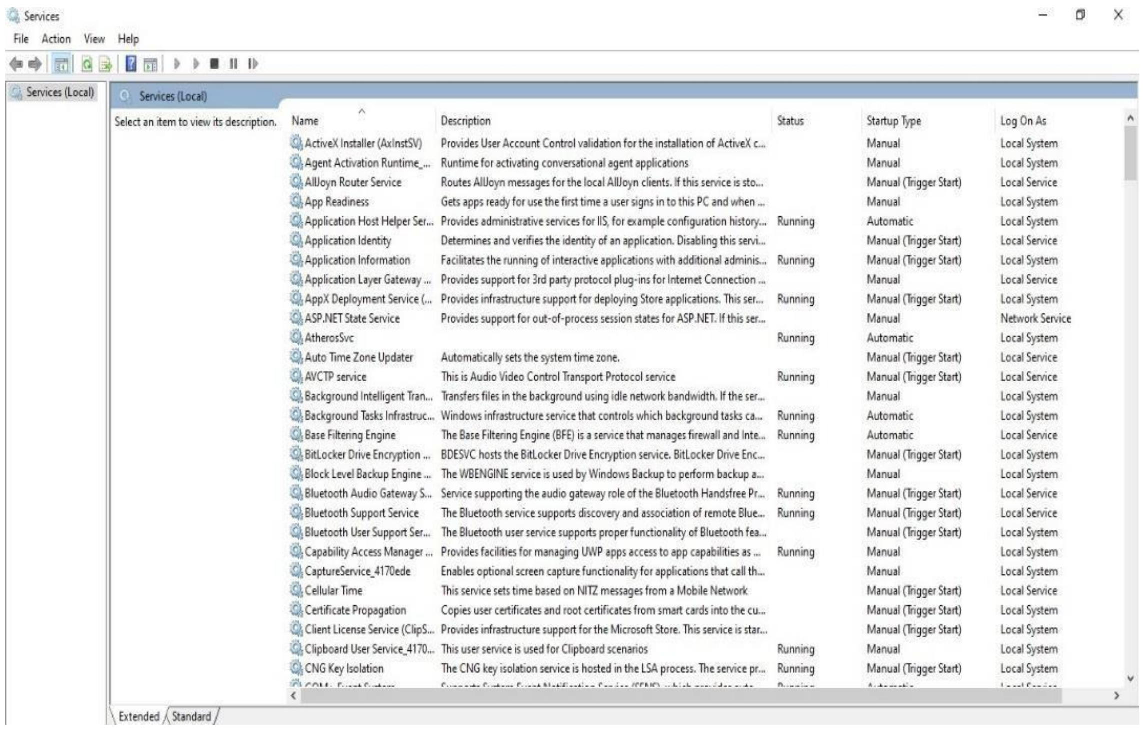
In Windows 10, to manage and review all background services, you can utilize the’services.msc’ command. This is easily accessed by pressing the Windows key along with ’R’ to open the Run window and then entering the command. Within the services interface, you can sort the services by their running status, helping you to identify which are active quickly. The names of these services are generally descriptive, providing a good initial idea of their purpose, and you can also access detailed descriptions for a clearer understanding of each service’s function. To make more informed decisions about which services may be necessary for your operations and which might be safely disabled to improve system performance and experiment outcomes. In the Windows operating system, you can manage all system services using a specific interface, commonly referred to by its file name services.msc. This management console allows you to view. all services and provides options such as start, stop, and pause each service. To stop a service, you can right-click on the desired service and select the "Stop" option from the context menu. This action halts the service until it is either manually restarted or configured to start automatically on system boot.

Similarly, in Ubuntu and other Linux distributions using the systemd system and service man- ager, services can be managed via the command line. To stop a service in Ubuntu, you can use the command systemctl stop <service name> . This command immediately stops the running service, which can be restarted or enabled to start automatically upon system boot using corresponding ‘systemctl‘ commands. This flexibility in service management across different operating systems makes it convenient for users and administrators to control and troubleshoot services as needed.



```
UNIT FILE                                 STATE           VENDOR PRESET
accounts-daemon.service                   enabled         enabled
acpid.service                             disabled        enabled
alsa-restore.service                      static          -
alsa-state.service                        static          -
alsa-utils.service                        masked          enabled
anacron.service                           enabled         enabled
apparmor.service                          enabled         enabled
apport-autoreport.service                 static          -
apport-forward@.service                   static          -
apport.service                            generated       -
apt-daily-upgrade.service                 static          -
apt-daily.service                         static          -
apt-news.service                          static          -
autovt@.service                           alias           -
avahi-daemon.service                      enabled         enabled
bluetooth.service                         enabled         enabled
bolt.service                              static          -
brltty-udev.service                       static          -
brltty.service                            disabled        enabled
colord.service                            static          -
configure-printer@.service                static          -
console-getty.service                     disabled        disabled
console-setup.service                     enabled         enabled
container-getty@.service                  static          -
cron.service                              enabled         enabled
cryptdisks-early.service                  masked          enabled
cryptdisks.service                        masked          enabled
cups-browsed.service                      enabled         enabled
cups.service                              enabled         enabled
dbus-fi.w1.wpa_supplicant1.service        alias           -
dbus-org.bluez.service                    alias           -
dbus-org.freedesktop.Avahi.service        alias           -
dbus-org.freedesktop.hostname1.service    alias           -
lines 1-34
```

Figure 2: Linux Running Service



Services
File Action View Help
Services (Local)
Services (Local)
Select an item to view its description.

Name	Description	Status	Startup Type	Log On As
ActiveX Installer (AxInstSV)	Provides User Account Control validation for the installation of ActiveX c...		Manual	Local System
Agent Activation Runtime_...	Runtime for activating conversational agent applications		Manual	Local System
AllJoyn Router Service	Routes AllJoyn messages for the local AllJoyn clients. If this service is sto...		Manual (Trigger Start)	Local Service
App Readiness	Gets apps ready for use the first time a user signs in to this PC and when ...		Manual	Local System
Application Host Helper Ser...	Provides administrative services for IIS, for example configuration history...	Running	Automatic	Local System
Application Identity	Determines and verifies the identity of an application. Disabling this servi...		Manual (Trigger Start)	Local Service
Application Information	Facilitates the running of interactive applications with additional adminis...	Running	Manual (Trigger Start)	Local System
Application Layer Gateway ...	Provides support for 3rd party protocol plug-ins for Internet Connection ...		Manual	Local Service
AppX Deployment Service (...	Provides infrastructure support for deploying Store applications. This ser...	Running	Manual (Trigger Start)	Local System
ASP.NET State Service	Provides support for out-of-process session states for ASP.NET. If this ser...		Manual	Network Service
AtherosSvc		Running	Automatic	Local System
Auto Time Zone Updater	Automatically sets the system time zone.		Manual (Trigger Start)	Local Service
AVCTP service	This is Audio Video Control Transport Protocol service	Running	Manual (Trigger Start)	Local Service
Background Intelligent Tran...	Transfers files in the background using idle network bandwidth. If the ser...		Manual	Local System
Background Tasks Infrastruc...	Windows infrastructure service that controls which background tasks ca...	Running	Automatic	Local System
Base Filtering Engine	The Base Filtering Engine (BFE) is a service that manages firewall and Inte...	Running	Automatic	Local Service
BitLocker Drive Encryption ...	BDESVC hosts the BitLocker Drive Encryption service. BitLocker Drive Enc...		Manual (Trigger Start)	Local System
Block Level Backup Engine ...	The WBENGINE service is used by Windows Backup to perform backup a...		Manual	Local System
Bluetooth Audio Gateway S...	Service supporting the audio gateway role of the Bluetooth Handsfree Pr...	Running	Manual (Trigger Start)	Local Service
Bluetooth Support Service	The Bluetooth service supports discovery and association of remote Blue...	Running	Manual (Trigger Start)	Local Service
Bluetooth User Support Ser...	The Bluetooth user service supports proper functionality of Bluetooth fea...		Manual (Trigger Start)	Local System
Capability Access Manager ...	Provides facilities for managing UWP apps access to app capabilities as ...	Running	Manual	Local System
CaptureService_4170ede	Enables optional screen capture functionality for applications that call th...		Manual	Local System
Cellular Time	This service sets time based on NITZ messages from a Mobile Network		Manual (Trigger Start)	Local Service
Certificate Propagation	Copies user certificates and root certificates from smart cards into the cu...		Manual (Trigger Start)	Local System
Client License Service (ClipS...	Provides infrastructure support for the Microsoft Store. This service is star...		Manual (Trigger Start)	Local System
Clipboard User Service_4170...	This user service is used for Clipboard scenarios	Running	Manual	Local System
CNG Key Isolation	The CNG key isolation service is hosted in the LSA process. The service pr...	Running	Manual (Trigger Start)	Local System

Extended Standard

Figure 3: Windows Running Services

4.4 Research Design

This study utilizes two Toshiba disks. One disk is part of a dual-booted Toshiba laptop running the latest Ubuntu release (24.04 LTS at the time of the study). Experiments require internet access later to observe how browsing affects deleted files.

A hurdle arose during the initial setup: Ubuntu on the Toshiba lacked the necessary wireless drivers for a environment. Fortunately, these drivers were easy to install with the appropriate knowledge [36]. Once Wi-Fi was functional, the machines were prepared for experimentation.

Throughout the experiments, raw disk images were captured at various stages for later analysis. To achieve this, snapshots were taken at crucial points. Snapshots essentially preserve a specific state of a virtual machine. This includes the entire VM configuration, encompassing hardware settings. Restoring a snapshot reverts not only the VM settings but also all changes made to the machine's disks, down to the individual file and bit level. When a snapshot is created, VirtualBox generates differencing images that solely store the modifications since the snapshot.

In this research, snapshots were chosen as the primary method for capturing the states of virtual machines due to their efficiency and space-saving properties. While cloning the entire virtual machine was another viable option, which involves creating a complete duplicate of the VM for testing various scenarios, snapshots provided several key advantages for our specific needs.

Snapshots work by recording only the changes made since the last snapshot, capturing these differences at both the file and bit level. This method is not only

more space-efficient but also saves considerable time compared to cloning, which requires duplicating the entire VM. Therefore, snapshots were strategically taken at critical points during the experiment to efficiently document changes without excessive use of storage space.

For the storage of these snapshots, we utilized the Virtual Disk Image (.vdi) format. This format is particularly useful as it allows for the seamless pausing of the VM, during which all related files are copied and preserved. These .vdi files can then be converted into a raw disk image format, making them suitable for use with various analysis tools. On Linux systems, this conversion can easily be achieved using the 'qemu-img' utility with the following command:

qemu−img convert <input vdi file > −O raw <output file name>

This conversion is essential for further forensic analysis and ensures compatibility with our processing tools, facilitating detailed examination of the virtual machine's state at each snapshot.

In this research, we utilized Oracle VirtualBox's own utility, VBoxManage, for converting .vdi files to raw image format, finding it to be more efficient than the commonly used 'qemu-img'. To verify the effectiveness and accuracy of both tools, we converted two .vdi images to raw format using each utility and then compared their file hashes. The results confirmed that both tools produced identical outputs, ensuring the reliability of our data conversion process.

Oracle VirtualBox comes with VBoxManage installed by default on Ubuntu systems, making it readily available for use right after the VirtualBox installation. To begin the conversion process, it is essential to first identify the UUID of the snapshot you wish to convert. Each snapshot in VirtualBox is uniquely identified by a Universally Unique Identifier (UUID). You can re- trieve the UUID of a snapshot by executing the following command in the terminal, replacing

'<your_virtual_machine_name>' with the name of your virtual machine: VBoxManage showvminfo <your_virtual_machine_name> −− details

This command will display detailed information about the virtual machine, including the UUIDs of all snapshots. Once the UUID of the desired snapshot is obtained, the conversion to a raw disk image can be performed with another VBoxManage command:

VBoxManage clonehd <uuid> name . img −−format RAW

Here, '<uuid>' should be replaced with the snapshot's UUID, and 'name.img' should be replaced with the filename you wish to use for the output raw image. This streamlined approach not only ensures accuracy but also enhances the efficiency of the data preparation process for subsequent forensic analysis.

To ensure minimal disruption during the experiments, a shared folder was set up between the host Ubuntu system and the two virtual machines—one running Ubuntu and the other Windows [43]. This setup facilitated easy and consistent transfer of datasets to the virtual machines by allowing files to be accessed directly from within each VM.

The experiment commenced with the transfer of a dataset named "dataset-new" into the shared folder, from where it was copied to the virtual machines. Initially, a baseline disk image of each VM was captured to record the state before any modifications. Subsequent steps involved the deletion of specific files from the VMs to simulate real-world data loss scenarios. These files varied in type and size, creating a diverse dataset crucial for comprehensive analysis.

For deleting files, we considered multiple methods applicable to both operating systems:

1 Graphical User Interface (GUI): This method involves using the file explorer to select and delete files, either by moving them to the trash and emptying them or by using Shift + Delete for permanent removal.

2 Command Line: Given its uniformity and ability to reduce variability in the deletion process, the command line was selected as the primary method for file deletion across both virtual machines

To maintain consistency in file selection across different systems, all names of the files and folders slated for deletion were cataloged in a text document titled "delete_files.txt". This document listed the relative paths of the targeted items within the dataset, providing a structured guide for the deletion process.

File deletion was executed differently on Linux and Windows to accommodate each operating system's file-handling specifics:

- Linux: On Linux systems, a Bash script was utilized where a loop iterated over each line in "delete_files.txt" using the ‘cat‘ command. Each line's content, representing a file or folder path, was stored in the ‘$i‘ variable. The files or folders were then deleted recursively with the command ‘rm -rf $i‘, where ‘-r‘ denotes recursive deletion and ‘-f‘ forces deletion without prompting for confirmation.
- Windows: For Windows, the file paths in "delete_files.txt" needed adjustment from Linux-style forward slashes (/) to Windows-style backslashes . This was accomplished before initiating a loop similar to that used in Linux, but adapted for Windows command line syntax. The loop read each line using the ‘for‘ command, which processed tokens line by line. File deletion was managed with ‘del /s /q "%i"‘, where ‘/s‘ ensures that the deletion includes all subdirectories, and ‘/q‘ suppresses confirmation prompts, facilitating a smoother and faster deletion process. These deletion methods ensured that the experiment could be carried out efficiently and uniformly across both operating systems, providing a reliable foundation for subsequent data recovery and analysis.

In this study, after files were deleted, researchers captured a disk image of the operating system to use as a baseline for their experiments. This baseline image, containing the state of the system post-deletion, allows for consistent conditions across all tests. Researchers then conducted various actions on the system, such as installing software or accessing files. Importantly, after each action, they restored the virtual machines to the original baseline state using the disk image. This method of reverting to the baseline after each test ensures that each action is evaluated under identical initial conditions, thus isolating the impact of each specific action on the recovery of deleted data. By meticulously controlling the experiment environment in this manner, the study aims to precisely analyze how different activities influence the likelihood of deleted data persisting in the system.

To study the persistence of deleted files under various conditions, researchers initially created a base disk image of the system immediately after file deletions. Following this, they conducted five distinct actions on separate virtual machines, reverting each one to the base image after every action. This meticulous approach yielded a total of seven disk images: the initial base image, one capturing the state right after deletions, and five additional images each representing the system post one of the experimental actions.

For the analysis of these images, the researchers employed a differential analysis technique. Utilizing a Python script named ‘idifference2.py‘, they analyzed differences in file allocations across all the images. The script first compared the base image to the one with the deleted files, noting any discrepancies and recording the findings in a Digital Forensics XML (DFXML) file. This file detailed changes by cataloging new, modified, and deleted files. For the purpose of their analysis, researchers focused on a selected set of data attributes including the filename, file size, the specific image containing the file, the file's offset value in the image, andthe file's MD5 hash for uniqueness. Moreover, the script was tailored to differentiate types of deleted items; using the ‘<name_type>‘ tag, it marked regular files with "r" and directories with "d". This nuanced approach allowed for precise identification and classification of deleted data, facilitating a detailed and structured analysis of how different actions affected file persistence on the system.

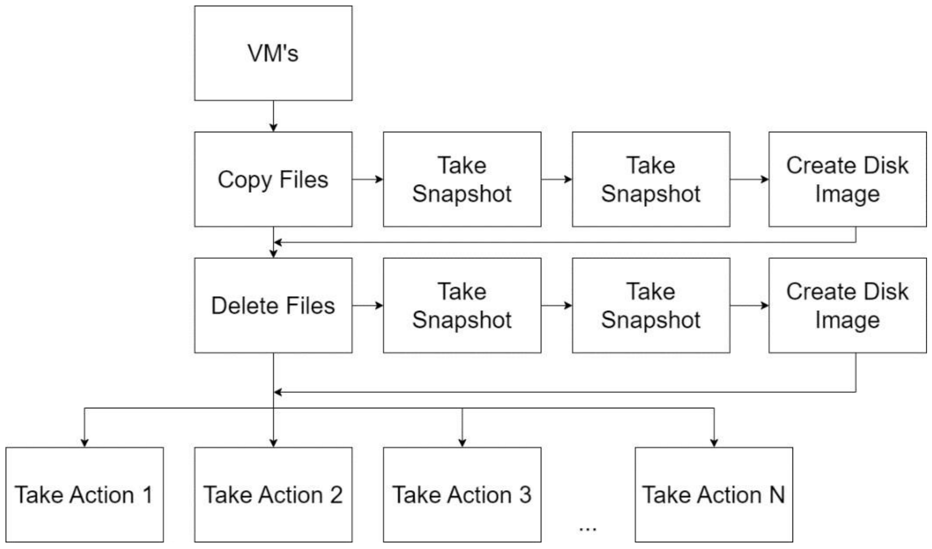


Figure 4: Working Process of Actions

4.5 Actions Performed on VM's

To explore how various activities influence the recoverability of deleted files, researchers initiated their study by generating a disk image directly after deleting selected files—this served as the base image. They then conducted a range of activities on distinct virtual machines, each designed to simulate different user behaviors or system changes. After each activity, they captured a new disk image of the system prior to reverting back to the state captured in the base image. This resulted in a collection of disk images: the original system state, the state with the deleted files, and additional images for each activity conducted.

This structured approach allowed the researchers to meticulously analyze the impact of each activity in a controlled environment. By reverting back to the base image after documenting each activity, they ensured that the starting conditions were consistent across all tests, thereby isolating the effects of individual activities on the recoverability of deleted files. This method provided a clear and reliable way to assess which activities posed the greatest risk to the permanence of file deletion.

4.6 Restarting the VM's

The researchers also investigated how the process of rebooting the system affects the recoverability of deleted files. They began by creating a base disk image following the deletion of certain files. Subsequently, they initiated a restart of the virtual machine, a procedure that encompasses shutting down applications, reloading drivers, and executing system checks. After the system had fully rebooted and a minute had passed post-login allowing the system to stabilize they captured another disk image. This additional waiting period ensured that the system was in a steady state, minimizing variables for more accurate analysis. This approach enabled them to meticulously examine the im- pact of the comprehensive reboot process on the persistence and potential recoverability of deleted files.

4.6.1 Adding new Files.

In addition to system actions, the researchers explored how the introduction of new data impacts the recoverability of deleted files. They inserted a new dataset into the virtual machines' shared folder. The objective was to determine whether the new data would occupy the same disk sectors as the previously deleted files, thereby potentially overwriting them and affecting their recovery. The specific types and quantities of files introduced during this experiment. This aspect of the study provides crucial insights into the interactions between new data writes and existing deleted file spaces on a disk, enhancing the overall understanding of data persistence and recoverability in digital storage environments.

4.6.2 Browsing on Chrome/Chromium

The researchers extended their examination to include the impact of web browsing on the recoverability of deleted files. They set up a simulated browsing session where a user engaged with popular websites such as Reddit, YouTube, and Twitch. To mimic typical user behavior more accurately, they also played random YouTube videos in the background during the session. This setup aimed to replicate the kind of disk activity that occurs during regular web browsing, focusing on the volume of data being processed and stored temporarily or permanently on the system.

To further assess the effects of substantial data downloads during web browsing, the researchers also downloaded a large dataset. Altogether, this activity resulted in about 8 GB of new data being written to the disk. This significant addition of data provides a real-world context to study how downloading and browsing can influence the disk space where deleted files once resided, offering insights into the challenges of recovering deleted data in environments with heavy internet usage.

4.6.3 Turning Off the Vm's

To thoroughly understand the changes that occur within a system during its shutdown process, administrators can employ a targeted approach. This involves shutting down the system and subsequently capturing a snapshot of the associated virtual machine (VM). This snapshot captures not only all current data and configurations of the VM but also includes a disk image that mirrors the exact state

of the virtual hard drive at the time of shutdown. By analyzing these captured files, administrators can delve deep into the systemic transformations that take place during shutdown, providing valuable insights into system behavior and potential areas for optimization or troubleshooting.

5 FINDINGS AND EVALUATION

In a comprehensive study exploring data persistence across different operating systems and disk types, research conducted experiments using both HDDs and SSDs on virtual machines. They performed identical actions on each setup and collected extensive data for analysis. A key component of their methodology involved calculating the space occupied by deleted files—measured in 512-byte sectors—and comparing the md5 hash values of these sectors against those in the original base image files.

To delve deeper into the data, the researchers selected a random sample of 100 deleted files, covering a diverse range of sizes and types, for closer examination. They also adapted their original tracing script, trace_file.py, to handle multiple files simultaneously, enhancing the efficiency of their data processing.

The study's findings revealed a significant insight: the likelihood of successfully recovering deleted files was more closely tied to the operating system used rather than the type of disk. Specifically, the recovery rates for files on Windows and Ubuntu were relatively consistent, regardless of whether the underlying hardware was an HDD or an SSD. This suggests that the OS's handling of deleted data plays a more pivotal role in data persistence than the storage medium itself.

The researchers documented their findings in a detailed manner, particularly focusing on the recovery rates after various actions were performed on the deleted

files. The initial findings indicate a substantial rate of recovery right after file deletion, with subsequent actions impacting the recovery likelihood to varying degrees. These results not only shed light on the robustness of data deletion processes across different platforms but also guide future developments in data security and recovery technologies.

Table 3: File Recovery

Action	Windows on SSD	Windows on HDD	Ubuntu on SSD	Ubuntu on HDD
Delete Files	93.22%	91.12%	94.17%	93.16%
Restart VM's	85.02%	85.12%	83.22%	83.05%
Turning Off VM's	89.50%	88.23%	88.70%	88.41%
Browsing Web	81.02%	82.02%	79.21%	79.31%
Adding Files	76.02%	78.05%	75.20%	75.21%

51 Analysis of SSD - Windows 10 OS

The research included a detailed examination of 100 files selected at random from the pool of deleted files. Analysis showed a remarkably high persistence rate of 89% for sectors between the deleted files and their original counterparts in the base image. This indicates that a substantial portion of the data remained recoverable even after files had been deleted. Additional insights into how other actions affected data persistence, though specifics are not included here. Furthermore, illustrated the recovery rates and the number of sectors successfully recovered following various actions. These findings underscore the resilience of data on digital storage mediums and highlight the challenges in ensuring complete data deletion. Such results are crucial for developing more effective data management and security protocols, particularly in environments where data sensitivity is a concern.

Table 4: Data Retrievability in Windows 10 SSD

Activity	Data Retrievability
After Turning off Vm’s	89.49
Rebooting VM’s	85.12
Browsing Chrome/Chromium	79.02
Adding Files to VM’s	72.69

5.2 Analysis of HDD - Windows 10 OS

The study analyzed 100 files selected at random, uncovering a high persistence rate of 88.22% between the sectors of deleted files and their original versions in the base image. This high rate of persistence indicates that most of the data from the deleted files remains recoverable, highlighting the difficulty in completely erasing data from digital storage media. Additional details regarding the persistence of data following other actions are documented. Moreover, its depict the percentage of files successfully recovered and the number of sectors retrieved after each action, respectively. These findings are critical, as they provide valuable insights into the effectiveness of different data deletion techniques and underscore the need for robust data security measures to protect sensitive information.

Table 5: Data Retrievability in Windows 10 HDD

Activity	Data Retrievability
After Turning off Vm’s	88.22
Rebooting VM’s	85.12
Browsing Chrome/Chromium	82.14
Adding Files to VM’s	76.08

5.3 Analysis of SSD - Ubuntu 24.04 LTS OS

The analysis of 100 randomly selected files revealed a remarkably high persistence rate of 88.75% between the deleted files and their original versions in the base image, demonstrating that a significant majority of the data from the deleted files remained recoverable. This highlights the challenges in achieving complete data erasure on digital storage mediums.To aid in visual understanding, the percentages of file recovery and the number of sectors recovered after each respective action. These insights are crucial for enhancing data security protocols and understanding the efficacy of deletion methods across different platforms and storage technologies.

Table 6: Data Retrievability Ubuntu 24.04 LTS OS SSD

Activity	Data Retrievability
After Turning off Vm’s	88.75
Rebooting VM’s	83.12
Browsing Chrome/Chromium	76.12
Adding Files to VM’s	73.12

5.4 Analysis of HDD - Ubuntu 24.04 LTS OS

The study involved an in-depth analysis of 100 randomly selected files, uncovering a significant data persistence rate of 88.45% between the deleted and the original base images. This high rate of persistence clearly demonstrates that a large portion of the data from the deleted files remains recoverable, emphasizing the challenges associated with ensuring complete data deletion. For a more comprehensive understanding of how data persistence varies with different actions.

Table 7: Data Retrievability Ubuntu 24.04 LTS OS HDD

Activity	Data Retrievability
After Turning off Vm’s	88.45
Rebooting VM’s	83.40
Browsing Chrome/Chromium	79.12
Adding Files to VM’s	75.62

6 CONCLUSION

This research examines the impact of various file systems, storage media, and user actions on the recoverability of files after they are permanently deleted. Specifically, the study utilized Ubuntu 22.04 to evaluate the Ext4 file system, and Windows 10 to assess NTFS, each running on virtual machines equipped with either an SSD or an HDD. The user actions simulated in this experiment include common activities such as shutting down the machine, rebooting, downloading files, copying files, and using disk cleaning utilities—all typical behaviors that might be expected from an average user.

To measure the persistence of deleted files, snapshots of the system were taken after each user action, from which disk images were created. The experiment involved checking 512-byte sectors within these images to determine whether the contents of previously deleted files remained detectable. Sectors retaining the same md5sum values as before deletion were considered preserved, while others were classified as non-recoverable. The focus of this analysis was on a randomly selected sample of 100 files, which varied in both file extensions and sizes.

Upon completion of these experiments, the collected data was thoroughly analyzed to draw several conclusions about data persistence under different conditions. This study provides significant insights into how data deletion is influenced by the type of file system and storage media used, as well as by the actions performed by users. These findings are essential for understanding the effectiveness of current data security measures and for developing more robust methods to ensure the complete removal of sensitive data. After analyzing the results, several conclusions that could be drawn are as follows:

- The findings of the study revealed a notable pattern: most deleted files remained recoverable even after employing system cleaning tools such as CCleaner or BleachBit. This was consistent across all file formats and both types of disks, SSD and HDD. The reason for this persistence is largely due to how these cleaning applications operate. Rather than overwriting previously deleted data, they primarily focus on removing redundant system files, leaving the deleted data
- largely untouched. Conversely, the likelihood of recovering deleted data diminished following actions that inherently involve writing new data to the disk. Such actions include shutting down or rebooting the system, engaging in web browsing, or copying new files to the disk, which are more likely to overwrite the sectors containing remnants of deleted files. This reduction in data recovery underscores the importance of implementing comprehensive data deletion methods to ensure enhanced security and privacy in sensitive data environments.
- The analysis highlighted noticeable differences in data persistence patterns between the Ubuntu and Windows operating systems across different storage media. In Ubuntu, data persistence was relatively stable, showing little variation between SSDs and HDDs across most user actions. This suggests consistent management of deleted data by the Ubuntu system, regardless of the disk type. In contrast, Windows demonstrated varied persistence levels depending on the disk type used. When users engaged in activities that involved transferring new files to the virtual machine, such as web browsing and copying files, data persistence was generally higher on HDDs than on SSDs. This could indicate that on HDDs, new data may be less likely to overwrite the remnants of deleted files, possibly due to the sequential nature of HDD storage. On the other hand, for actions not directly involving data transfer, persistence was slightly higher on SSDs. This variation may reflect the different techniques each operating system employs to manage disk writes, with Windows possibly optimizing write operations differently on SSDs compared to HDDs. These findings suggest that the choice of operating system and disk type can significantly influence the security and effectiveness of data deletion practices.
- The research underscores that the type of file system, rather than the nature of the underlying disk (SSD vs HDD), plays a more pivotal role in influencing data persistence. This conclusion is drawn from the observation that persistence levels were remarkably consistent within the same file system—Ext4 for Ubuntu and NTFS for Windows—regardless of whether an SSD or HDD was used. For example, data persistence in Ubuntu was similar across both SSD and HDD environments, and the same consistency was seen with Windows on both types of
- disks. These findings suggest that the architectural decisions and functionalities inherent to each file system are key factors in determining how effectively deleted data can be recovered, overshadowing the impact of the physical attributes of the storage media.
- The study revealed that a substantial amount of data remained recoverable even after being permanently deleted. This implies that about 8% of the sectors had been overwritten by new data, leading to some files being partially or fully overwritten. Such overwriting resulted in hash mismatches that prevented the complete recovery of all sectors. This finding highlights the challenges in achieving thorough data erasure on digital storage devices.

7 FUTURE ROADMAP

This research delved into how deleted files persist across different operating systems (Windows 10 and Ubuntu) and storage devices (SSDs and HDDs), examining various combinations to assess their impact on data recoverability. The findings from this study illuminate the intricate dynamics of file persistence, presenting a foundation for further investigative work. The section below proposes several avenues for future research that could expand upon the insights obtained, exploring deeper into the nuances of data recoverability and the factors that influence it. This could include testing additional operating systems, incorporating emerging storage technologies, or experimenting with more varied user behaviors to provide a broader understanding of data deletion efficacy and security.

- During the experiment, an unexpected pattern was observed: downloading and extracting of web data. This finding is counterintuitive, as larger volumes of data typically increase the likelihood of overwriting existing, previously deleted data. This suggests that further investigation is needed to understand the underlying mechanisms at play. It would be particularly insightful to examine how web browsers handle data downloads and extractions in contrast to the processes involved in conventional file copying. Unpacking these differences could help explain why downloading and extracting substantial amounts of data might preserve more recoverable traces of deleted files than expected.
- Broadening the scope of this research to incorporate macOS, which uses the proprietary Apple File System (APFS), offers a valuable opportunity to deepen our understanding of how different file systems influence data recoverability. By conducting a comparative analysis involving Windows, Ubuntu (ext4), and macOS (APFS), we can explore a wider array of file system philosophies and their effects on data persistence. This expanded study would enable us to assess the behavior of ext4 across both Linux and macOS environments, potentially uncovering how different operating systems interact with the same file system to affect data remanence. Including APFS in this analysis would also shed light on the distinct data management strategies that Apple employs, providing a clearer picture of how proprietary systems handle data deletion. By examining these diverse platforms, this research could uncover nuanced insights into how the interplay between operating system and file system design impacts the recover- ability of deleted data, offering valuable guidelines for both users and developers concerned with data security.

8 REFERENCES

[1] B. Carrier, File System Forensic Analysis, 1st. Place of publication not identified: Addison Wesley, 2005.

[2] B. Carrier, File system forensic analysis. Boston, Mass.: Addison- Wesley, 2005, P236– 237.

[3] S. Piper, M. Davis, G. Manes, and S. Shenoi, "Detecting hidden data in ext2/ext3 filesystems," in Advances in Digital Forensics, ser. The International Federation for Information Processing, vol. 194, 2005, pp. 245– 256.

[4] S. Lee and T. Shon, "Improved deleted file recovery technique for ext2/3 filesystem," The Journal of Supercomputing, vol. 70, no. 1, pp. 20–30, 2014.

[5] J. Sammons, The Basics of Digital Forensics the Primer for Getting Started in Digital Forensics, 1st. Waltham, Mass: Syngress, 2012.

[6] T. M. Khan, "Identifying factors affecting deleted file persistence through empirical study and analysis," Ph.D. dissertation, ProQuest Dissertations Publishing, 2017.

[7] J. H. Jones and T. M. Khan, "A method and implementation for the empirical study of deleted file persistence in digital devices and media," in 2017 IEEE 7th Annual Computing and Communication Workshop and Conference (CCWC), IEEE, 2017, pp. 1–7.

[8] A. Salam, "Internet of things for sustainable community development: Wireless com- munications, sensing, and systems," in Springer Cham, 2020, ch. 10. DOI: 'https : //doi.org/10.1007/978-3-030-35291-2'.

[9] WebbyLab, "Firmware analysis for iot devices Practical Guide," WebbyLab Blog, Mar. 2023.

[10] M. S. Barik, G. Gupta, S. Sinha, A. Mishra, and C. Mazumdar, "An efficient technique for enhancing forensic capabilities of ext2 filesystem," Digital Investigation, vol. 4, S55–S61, 2007.

[11] S. Garfinkel, P. Farrell, V. Roussev, and G. Dinolt, "Bringing science to digital foren- sics with standardized forensic corpora," Digital Investigation, vol. 6, 2009.

[12] A. Bahjat and J. Jones, "File allocation chronology and its impact on digital foren- sics," in 2023 IEEE 13th Annual Computing and Communication Workshop and Con- ference (CCWC), IEEE, Jan. 2023.

[13] O. C. Agada, I. Iganibo, J. Jones, and K. Fairbanks, "A digital body farm for collecting deleted file decay data," in Advances in Digital Forensics XVIII, vol. 653, Springer International Publishing, 2022, pp. 3–18.

[14] K. D. Fairbanks, “A technique for measuring data persistence using the ext4 file system journal,” in 2015 IEEE 39th Annual Computer Software and Applications Conference, 2015. DOI: 10.1109/compsac.2015.164.

[15] D. Farmer and W. Venema, “The persistence of deleted file information,” in Forensic discovery, Upper Saddle River, NJ: Addison- Wesley, 2007, pp. 145–148.

[16] dfxml-working-group/dfxmlpython, Dfxmlpython, n.d. [Online]. Available: https:// github.com/dfxml-working-group/dfxml_python.

[17] S. Garfinkel, A. Nelson, and J. Young, “A general strategy for differential forensic analysis,” Digital Investigation, 2012.

[18] S. S. Garfinkel, Digital forensics xml project and library, Retrieved from https:// github.com/simsong/dfxml, 2017.

[19] Oracle Corporation, Virtualbox for linux downloads, 2023. [Online]. Available: https: //www.virtualbox.org/wiki/Linux_Downloads.

[20] Oracle, Oracle vm virtualbox user manual, Release 6.0, 2019. [Online]. Available: https: //docs.oracle.com/en/virtualization/virtualbox/6.0/user/snapshots.html.

[21] Canonical Ltd., Download ubuntu desktop, 2023. [Online]. Available: https://ubuntu. com/download/desktop.

[22] Microsoft, Download windows 10 disc image (iso file), 2021. [Online]. Available: https:

[23] //www.microsoft.com/en-us/software-download/windows10ISO.

[24] Oracle, Virtualbox dynamic vs fixed size allocation, 2011. [Online]. Available: https: //forums.virtualbox.org/viewtopic.php?f=11&t=43344.

[25] Microsoft, Windows 10 requirements, 2021. [Online]. Available: https://www.micros oft.com/en-us/windows/windows-10-specifications.

[26] Microsoft, Master file table. [Online]. Available: https : / / docs . microsoft . com / en - us/windows/win32/fileio/master-file-table.

[27] Unisys Corporation, Clearpath mcp 18.0 reference manual,2017. [Online]. Availa https://public.suppor unisys

[28] .com/aseries/docs/clearpath -mcp- 18 .0/ 86000387 - 512/section-000023628.html.

[29] NIST (National Institute of Standards and Technology), Computer forensic reference data sets (cfreds). [Online]. Available: https://cfreds.nist.gov/all.

[30] University of New Haven, Cyber forensics research and education group (unhcfreg) datasets. [Online]. Available: https://www.newhaven.edu/academics/centers-institu tes/cyber-forensics- research-education-group/datasets.php.

[31] F. Breitinger, Digital forensic datasets. [Online]. Available: https://datasets.fbreitin ger.de/datasets/.

[32] F. Breitinger, Other digital forensic dataset repositories. [Online]. Available: https: //datasets.fbreitinger.de/other-repositories/.

[33] L. Zeltser, “Free malware sample sources for researchers,” 2012. [Online]. Available: https://zeltser.com/malware-sample-sources/.

[34] Textfiles.com, The textfiles.com archives.[Online]. Available: http://textfiles.com/.

[35] United States Patent and Trademark Office, Uspto bulk data storage system - patents. [Online]. Available: https://bulkdata.uspto.gov/.

[36] Tpn/pdfs: Technically-oriented pdf collection (papers, specs, decks, manuals, etc). [On- line]. Available: https://github.com/tpn/pdfs.

[37] V. G. Smith, Systemd view status of a service on linux, https://www.cyberciti.biz/ faq/systemd-systemctl-view-status-of-a-service- on-linux/, 2021.

[38] Get wifi to work after installing ubuntu or lubuntu on macbook, Available at: https: //www.amirootyet.com/post/how-to-get-wifi-to-work-after.

[39] Oracle VM VirtualBox, [title of documentation], 2020. [Online]. Available: https:// www.virtualbox.org/wiki/Documentation.

[40] Oracle Corporation, Creating, restoring, and deleting snapshots of virtual machines, Chapter in Oracle VM VirtualBox User Manual, 2020. [Online]. Available: https:// docs.oracle.com/en/virtualization/virtualbox/6.0/user/snapshots.html#snapsh ots- contents.

[41] Oracle VM VirtualBox, Cloning virtual machines, 2020.

[42] Oracle Corporation, Virtual disk image (vdi) details, Available at: https://docs.oracle. com/en/virtualization/virtualbox/6.0/user/vdidetails.html, 2020.

[43] A. Adegbodu, How to convert virtual machine image formats, DigitalOcean. [Online]. Available: https://www. digitalocean

[44] .com/community /tutorials /how- to -convert - virtual-machine-image- formats, 2019.

[45] Qemu-img man page, Die.net. Retrieved from https://linux.die.net/man/1/qemu- img.

[46] Oracle Corporation, Oracleő vm virtualbox user manual, Version 6.0, 2018. [Online]. Available: https://docs.oracle.com/en/virtualization/virtualbox/6.0/user/sharedfol ders.html.

[47] Indeed, How to delete a file using command prompt, Available at: https://in.indeed. com/career-advice/career-development/how-to-delete-a- file-using-cmd.

[48] The Linux Documentation Project, Rm(1) - linux manual page, Available at: https: //man7.org/linux/man-pages/man1/rm.1.html, 2021.

[49] DFXML Python library, Dfxml python library, GitHub. Retrieved from https://githu b.com/dfxml-working- group/dfxml_python/blob/main/dfxml/bin/idifference2.py, n.d.

[50] J. Jones, Adiff.py, Retrieved from https://github.com/jjonesu/DeletedFilePersistenc e/blob/master/python/adiff.py, 2018.

[51] Unisys, Clearpath mcp 18.0: Virtual i/o server (vios) for clearpath mcp, release 18.0, Unisys Support, 2022. [Online]. Available: https://public.support.unisys.com/aseries/ docs/ClearPath-MCP- 18.0/86000387-512/section-000023628.html.

[52] J. Jones, Trace_file.py [computer software], https://github.com/jjonesu/DeletedFile Persistence/blob/master/python/trace_file.py, 2021.

[53] Turn on or off fast startup in windows 10, Retrieved from https://www.tenforums. com/tutorials/4189-turn-off-fast-startup-windows- 10-a.html.

[54] J. Crawford, Cat dataset, Kaggle, 2022. [Online]. Available: https://www.kaggle. com/datasets/crawford/cat-dataset.