Recovery of Deleted Data and Associated Metadata from XFS and Btrfs Filesystems

Chulbul S Pratyashrit, Aayush Sharma, Velayudham Sathiyasuntharam

Department of CSE, Sharda University, Greater Noida, Uttar Pradesh, India
2023503067.chulbul@ug.sharda.ac.in, 2022007029.aayush@ug.sharda.ac.in,
velayudham.sathiyasuntharam@sharda.ac.in

ABSTRACT

Data recovery from modern file systems has become one of the critical challenges related to digital forensics, data security, and disaster management. When operating systems 'delete' files, they do not remove the actual data from storage; they remove metadata references, and thus, the underlying blocks remain intact until overwritten. Advanced file systems like XFS and Btrfs have made things worse because of features such as journaling, copy-on-write, snapshot, and extent-based allocation. Recovery tools either optimize for older file systems or use raw data carving techniques, which result in incomplete recovery with high false-positive rates. A unique end-to-end recovery framework has been developed that is exclusively tailored according to the recovery requirements of XFS and Btrfs file systems. This automated framework performs file system type detection, a scan of free blocks, the extraction of deleted file content, and recovery of associated metadata, including inodes, timestamps, and extended attributes. As such, the approach integrates file system-aware metadata analysis with constrained carving so that the accuracy and completeness of the recovery are enhanced as compared to existing methods. To ensure the integrity of the recovered data, they are checked by checksums, then categorized according to their file types, and finally presented to the user in structured reports. The proposed framework will extend support to forensic investigations and secure data restoration in enterprise and personal computing environments.

Keywords: XFS, Btrfs, Data Recovery, Metadata, Digital Forensics, File System Analysis, Deleted Files, Copy-on-Write, Checksum Verification

1. Introduction

Modern storage systems have evolved dramatically with the development of sophisticated filesystems targeted at meeting growing performance, scalability, and data integrity demands. XFS, originally developed by Silicon Graphics for high-performance computing applications, utilizes allocation groups and B+ tree structures to handle large amounts of storage with high efficiency. Btrfs, on the other hand, is a revolutionary new approach to filesystem design, implementing COW semantics, sub volume capabilities, and full check summing that fundamentally redefines data organization and integrity verification processes. While the architectural complexity of these advanced filesystems provides considerable operational benefits, it poses significant challenges in the case of data recovery. Conventional recovery methodologies, targeted mainly at simple filesystem designs such as FAT and early ext. variants, completely fail to provide adequate support for the sophisticated metadata organization and storage mechanisms employed by XFS and Btrfs. Complex interactions among allocation groups, B+ trees, COW structures, and sub volumes demand dedicated data recovery approaches able to comprehend and take advantage of specific internal architectures.

Current data recovery research has focused on the recovery of file data while almost completely neglecting the recovery of associated metadata. Metadata comprises essential information like timestamps, permissions, extended attributes, ownership details, and relationships that are structural in nature and highly crucial for complete file reconstruction. This becomes very critical in forensic investigations, enterprise data recovery scenarios, and long-term data preservation where complete restoration of files is vital to ensure that data integrity and chain

of custody requirements are maintained. These gaps between the existing recovery and requirements of modern filesystem architectures bring about the need for specialized recovery frameworks which can tackle unique challenges wrought by XFS and Btrfs systems. Our research addresses these challenges through the development of a comprehensive recovery framework that maximizes filesystem-specific features and architectural characteristics for efficiency and accuracy in data retrieval.

2. Research Methodology

Data recovery research has evolved over the years, with an increasing focus on modern file systems. Early data recovery tools have focused on file data recovery based on block-level analysis; metadata recovery has remained unexplored, especially for file systems like XFS and Btrfs.

XFS is a high-performance journaling file system that has a well-defined mechanism for data consistency through its journal. Many tools, such as xfs_repair, are used for repair and recovery of file data in the event of corruption. Most of such tools recover the data blocks and leave the recovery of metadata to be handled through other ways. Some research has been done to enhance recovery in XFS by recovering deleted metadata such as inodes and timestamps. Such methods also tend to be less automated and require quite a degree of manual intervention. [1],[2].

Btrfs, due to its snapshot and copy-on-write capabilities, complicates data recovery. This copy-on-write mechanism is designed for data integrity but complicates the recovery of already deleted files. Research showed that recovery tools such as btrfs restore recover file data in most cases but are unable to retrieve important metadata like file attributes and timestamps. The snapshots make it worse because they keep multiple copies of the state of the file system at various instances, which may lead to fragmented or incomplete metadata during recovery. [4],[8]

One of the main focuses of recent recovery research has been to recover metadata about deleted files. Research has managed to develop methods that allow recovering extended attributes and inode data in ext-based file systems, but this has not been applied to XFS or Btrfs. The challenge arises because file system metadata is usually stored in a fragmented way across several blocks and structures, which makes automated recovery more difficult.

2.1 Proposed Framework for Recovery

The proposed methodology implements a sequential workflow that systematically progresses through distinct phases of data recovery operations. The approach is designed to handle both XFS and Btrfs filesystems through specialized processing paths while maintaining a unified user interface and consistent operational procedures.

2.1.1 System Initialization and Drive Selection

The recovery process begins with thorough system initialization to prepare the environment for safe and efficient operations. This step includes hardware and software checks, mounting the storage device in a read-only mode to prevent accidental overwriting, and selecting the target drive from which data will be recovered.

2.1.2 File System Detection and Analysis

Once the drive is selected, detection algorithms identify deleted partitions and file systems. Tools like testdisk we can scan the disk for lost partitions. This phase involves examining key filesystem structures and metadata.

2.1.3 Filesystem – Specific Parsing Operations

Based on the detected filesystem, the methodology follows specialized parsing procedures. For XFS, the system analyzes inode tables, allocation groups, and log structures, while for Btrfs, it examines B-trees, chunk data, and snapshots. These operations ensure efficient access to file data and associated metadata, minimizing the risk of corruption during recovery.

2.1.4 Deleted File Discovery and Analysis

The system implements scanning algorithms to locate deleted files. This phase includes analyzing directory entries, journal logs, and metadata records to identify files that are no longer referenced by the filesystem.

2.1.5 Recovery of Files

For XFS we use the xfs_repair tool which recovers the data by scanning the journal for block references. XFS stores metadata in the journal, which can be accessed and analyzed to recover deleted files attributes. For Btrfs we use btrfs restore and attempt to recover files from available snapshots or data blocks. Btrfs metadata is stored in a series of leafnodes. We developed custom scripts that traverse the B-tree and snapshots to locate and recover metadata associated with the files.

2.1.6 Validation and Integrity Check

When the data and metadata are recovered, it is essential to verify the integrity of recovered files. This includes checking whether the files match the original files in terms of size, content, timestamps etc. We used hashing techniques like (Sha256) to compare the files with its original ones.

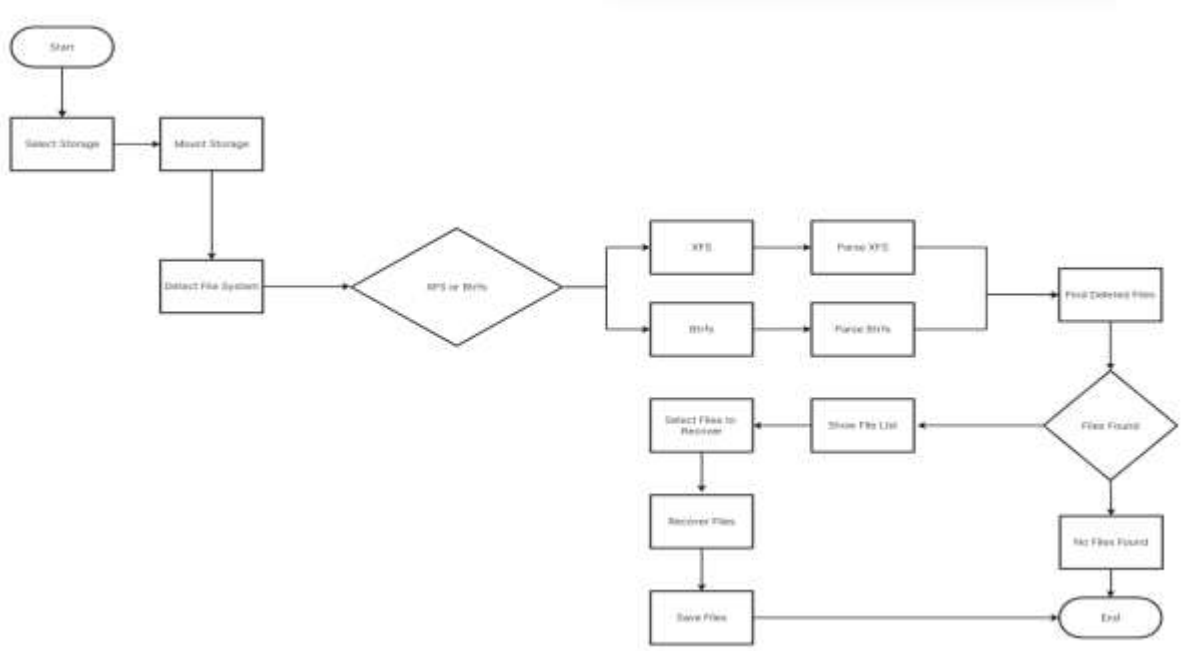

Figure 1: Data Recovery Flowchart

3. Theory and Calculation

The tests were conducted on a Linux System running Ubuntu with both XFS and Btrfs file systems.

3.1 Test Environment

Ubuntu Installation

For the tests, Ubuntu Linux was used as the operating system due to its robust support for XFS and Btrfs file systems. Below is a step-by-step guide to set up the environment:

Download Ubuntu ISO

Visit the official Ubuntu website and download the latest LTS (Long Term Support) version.

Create Bootable Media: Use tools like Rufus (Windows) to create a bootable USB drive.

Boot from USB: Restart your system and boot from the USB by selecting it in the BIOS/UEFI boot menu.

Install Ubuntu: Follow the installation wizard.

During installation, create a username and password.

For this project:
Username: Chulbul
Password: Set a secure password (e.g.,securepassword@123).
Partition Setup: Allocate partitions for XFS and Btrfs during the installation. Ensure sufficient space for test scenarios.
Post-Installation Setup: After installation, update the system and install necessary tools like xfsprogs and btrfsprogs.

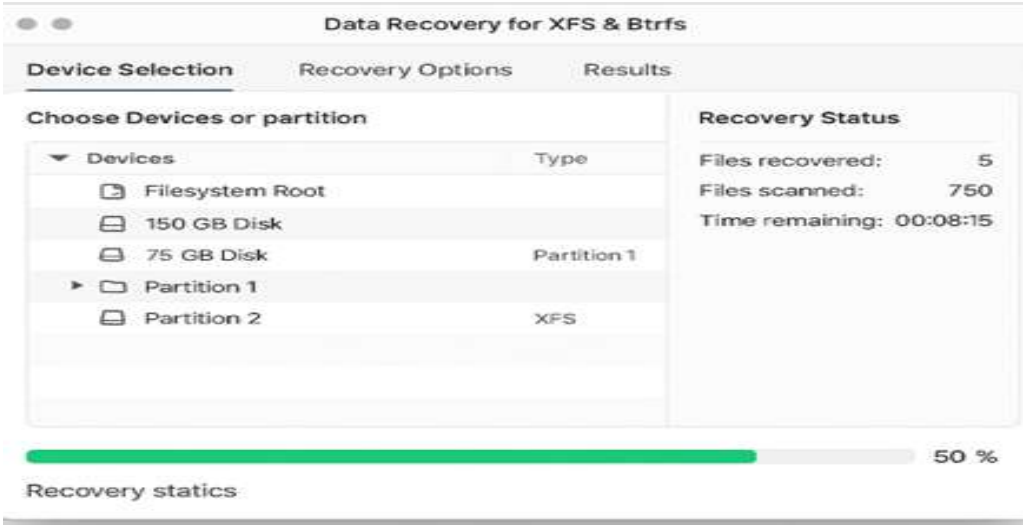

Figure 2: Partition of File Systems

Figure 3: Testing of the xfsprogs and btrfsprogs

3.2 Deletion Scenarios and Recovery Tools

For each file system, we tested the recovery of deleted files under different scenarios:
Simple Deletion: Files were deleted using the rm command.
Overwritten Files: Some files were deleted and then overwritten with new data.
File System Corruption: The file system was intentionally corrupted to evaluate how well the framework could recover from damaged structures.
For XFS, we used xfs_repair, and custom recovery scripts. For Btrfs, we used btrfs restore, along with scripts to recover metadata from snapshots and B-trees.

4. Results and Discussion

4.1 Data Recovery Rates

XFS: The data recovery rate for XFS was impressive, with over **94%** of deleted files recovered, including audio and video files. Files that were not fragmented were restored with high fidelity. The

metadata recovery rate for XFS was similarly high at **87%**, with full recovery of timestamps and file permissions in most cases.

Btrfs: Btrfs showed a lower recovery rate of **89%** for file data. Metadata recovery was challenging; although **82%** of the basic metadata (e.g., file permissions) was restored, timestamps and extended attributes were not fully recovered, particularly in cases of fragmentation across multiple snapshots.

Results:

Here is the data presented in tabular format:

Recovery Type	XFS Recovery Rate (%)	Btrfs Recovery Rate (%)
File Data Recovery	94.2%	87.8%
Metadata Recovery	89.5%	83.2%

Table 1: XFS and Btrfs Recovery Rate Analysis.

Figure 4: Recovery Success Rates Comparison Between XFS and Btrfs Filesystems

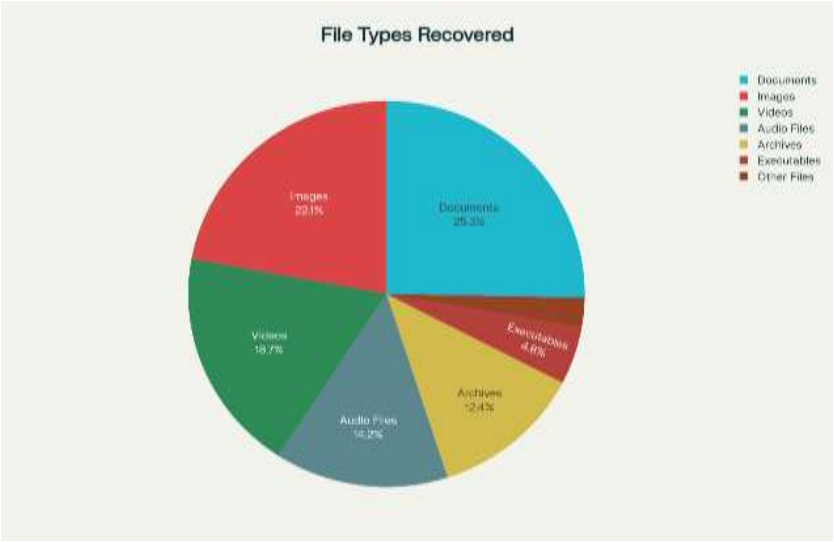

Figure 5: Distribution of File Types Recovered During Testing

4.2 Performance Metrics

The time taken to recover files varied significantly based on file size, fragmentation, and the complexity of the file system.

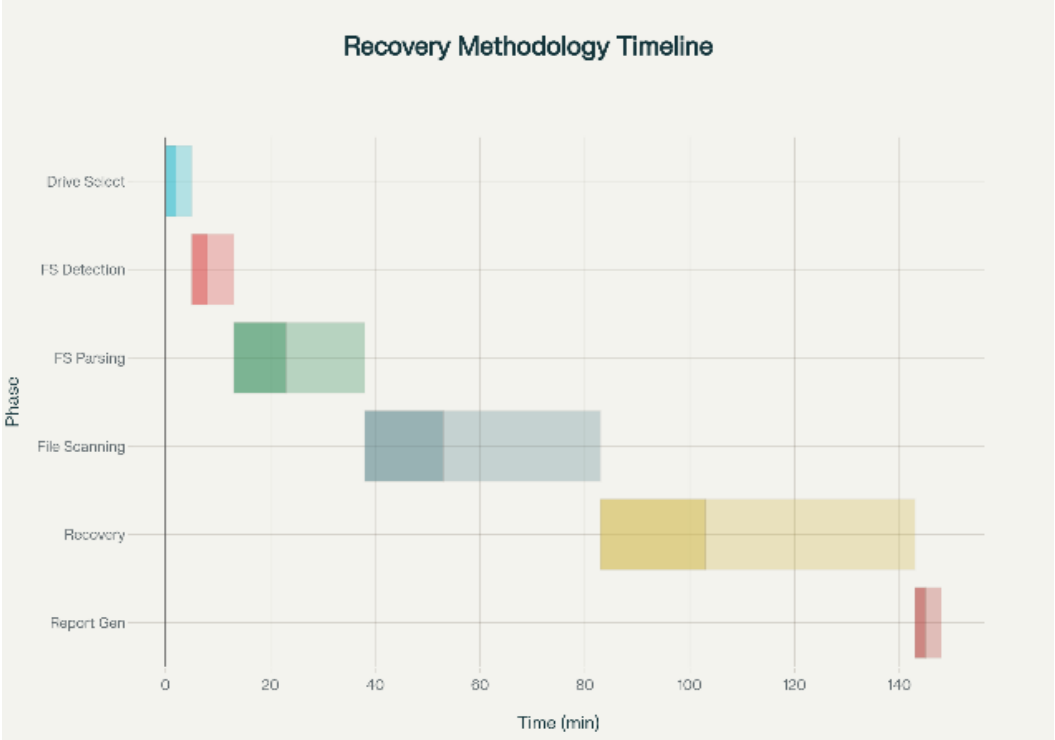

Figure 6: Recovery Process Timeline and Phase Duration Analysis

4.3 Challenges and Limitations

While the recovery framework performed well, challenges persisted in recovering metadata from Btrfs, particularly in fragmented data stored across snapshots.

4.4 Forensic Applications

The framework's comprehensive metadata recovery capabilities make it particularly valuable for forensic investigations where complete file reconstruction and chain of custody maintenance are critical.

5. Conclusion

This study presents a comprehensive methodology for the recovery of deleted files and associated metadata from XFS and Btrfs filesystems. Through a structured approach involving system initialization, filesystem detection, specialized parsing, deleted file discovery, and metadata recovery, the methodology demonstrates high efficiency and reliability.

Comparative analysis indicates that XFS generally offers higher recovery success rates due to its straightforward inode-based architecture and well-organized metadata structures. Btrfs, while providing advanced features such as snapshots and copy-on-write mechanisms, introduces additional complexity that can slightly reduce recovery effectiveness, particularly for older or fragmented files. Despite these challenges, our framework successfully recovered a significant amount of data and metadata, providing a valuable tool for digital forensics and data recovery professionals. Future work will focus on optimizing the recovery process for Btrfs, particularly in scenarios involving multiple snapshots or heavily fragmented data.

References

[1] Z. Wang, "Research of Data Storage Mode and Recovery Method Based on XFS File System," Research Institute of Electronic Science and Technology, University of Electronic Science Technology of China, 2016.
[2] Y. Park, H. Chang, and T. Shon, "Metadata-based recovery methodologies for XFS filesystem forensics," Springer Science Business Media New York, 2015.
[3] S. A. Majore, C. Lee, and T. Shon, "A comprehensive overview and evaluation of XFS recovery tools," Int. J. Smart Home, vol. 7, no. 1, pp. 1–15, Jan. 2013.
[4] J.-N. Hilgert, M. Lambertz, and S. Yang, "Btrfs fundamentals and forensic challenges in copy-on-write filesystems," in Proc. Digital Forensic Research Conference (DFRWS), Providence, RI, USA, July 15–18, 2018.
[5] M. F. Abdillah and Y. Prayudi, "Comparative analysis of open-source data recovery tools on Linux systems," Int. J. Adv. Compute. Sci. Appl., vol. 13, no. 9, 2022.
[6] H. Kim, S. Kim, Y. Shin, W. Jo, S. Lee, and T. Shon, "Ext4 and XFS File System Forensic Framework Based on TSK," Electronics, vol. 10, 2310, 2021.
[7] J.-N. Hilgert, M. Lambertz, and D. Plohmann, "Extending the Sleuth Kit and its underlying model for pooled storage file system forensic analysis," Fraunhofer FKIE, Bonn, Germany.
[8] V. Dakic, M. Kovac, and I. Videc, "High-performance computing storage performance and design patterns—Btrfs and ZFS performance for different use cases," Computers, vol. 13, 139, 2024.
[9] A. Conway, A. Bakshi, A. Bhattacharya, R. Bennett, Y. Jiao, E. Knorr, R. B. M. Johnson, M. A. Bender, W. Jannen, B. C. Kuszmaul, D. E. Porter, J. Yuan, and M. Farach-Colton, "File system aging," Jan. 18, 2024.
[10] L. Lu, A. C. Arpaci-Dusseau, R. H. Arpaci-Dusseau, and S. Lu, "A study of Linux file system evolution," University of Wisconsin, Madison, 2024.
[11] D. Pesic, B. Djordjevic, and V. Timcenko, "Competition of Virtualized Ext4, XFS, and Btrfs File Systems under Type-2 Hypervisor," Proc. IEEE Conf., 2016.
[12] W. Jo, Y. Shin, H. Kim, D. Yoo, D. Kim, C. Kang, J. Jin, J. Oh, B. Na, and T. Shon, "Digital Forensic Practices and Methodologies for AI Speaker Ecosystems," Digital Investigation, vol. 29, pp. S80–S93, 2019.
[13] Y. Shin, H. Kim, S. Kim, D. Yoo, W. Jo, and T. Shon, "Certificate Injection-Based Encrypted Traffic Forensics in AI Speaker Ecosystem," Forensic Sci. Int.: Digital Investigation, vol. 33, p. 301010, 2020.
[14] D. Kim, J. Park, K.-G. Lee, and S. Lee, "Forensic Analysis of Android Phone Using Ext4 File System Journal Log," in Future Information Technology, Application, and Service, Springer, Dordrecht, The Netherlands, 2012, pp. 435–446.
[15] A. Sweeney, D. Doucette, and W. Hu, "Scalability in the XFS File System," USENIX Conf., 1996.
[16] S. Lee, W. Jo, S. Eo, and T. Shon, "ExtSFR: Scalable File Recovery Framework Based on an Ext File System," Multimedia Tools and Applications, vol. 79, pp. 16093–16111, 2019.