

Android Mobile Forensics Parser

A Bash-Based Automated Tool for Offline Android Forensic Analysis

Abstract

The widespread adoption of Android smartphones has made them a critical source of digital evidence in cybercrime investigations. Extracting and analyzing data from Android devices, however, remains a complex and time-consuming task due to the diversity of file systems, databases, and application structures. This paper presents **Android Mobile Forensics Parser**, a Bash-based forensic analysis tool designed to automate the extraction, verification, and correlation of key forensic artifacts from a mounted Android image.

The tool focuses on offline forensic analysis, ensuring evidence integrity through cryptographic hashing and structured data extraction from SQLite databases. It generates investigator-ready outputs, including CSV files, textual reports, statistical summaries, and a unified activity timeline. The proposed solution aims to support investigators, researchers, and cybersecurity students by providing a lightweight, transparent, and reproducible forensic workflow.

Keywords

Android Forensics, Digital Forensics, Mobile Device Analysis, Bash Scripting, Evidence Integrity, SQLite Analysis

1. Introduction

Mobile devices have become central to modern communication, data storage, and online activity. As a result, Android smartphones frequently serve as primary evidence sources in criminal investigations, incident response, and corporate security cases. Android forensics involves the identification, acquisition, preservation, analysis, and presentation of digital evidence extracted from Android devices.

Despite the availability of advanced commercial forensic tools, many investigators and students face challenges such as limited accessibility, high costs, and lack of transparency in proprietary

solutions. Therefore, there is a growing need for open, script-based forensic tools that allow full visibility into the analysis process.

This paper introduces **Android Mobile Forensics Parser**, a Bash-based tool that automates forensic artifact extraction from a mounted Android image while preserving evidence integrity and maintaining forensic soundness.

2. Problem Statement

Manual analysis of Android forensic images is prone to several issues:

- Time-consuming artifact extraction
- Human error during manual database parsing
- Difficulty correlating multiple data sources
- Limited understanding of automated forensic workflows among students

Additionally, many existing tools obscure internal processes, reducing transparency and educational value. This work addresses these challenges by providing a clear, script-driven forensic analysis framework.

3. Objectives

The main objectives of this project are:

1. To automate the extraction of key Android forensic artifacts
 2. To ensure evidence integrity using cryptographic hashing
 3. To generate structured outputs suitable for forensic analysis
 4. To correlate multiple artifacts into a unified forensic timeline
 5. To provide a lightweight and educational forensic solution
-

4. System Architecture and Design

4.1 Design Overview

The Android Mobile Forensics Parser operates on a **mounted Android image** with read-only access. The system is divided into the following logical components:

- User Interaction Module

- Integrity Verification Module
- Artifact Extraction Module
- Analysis and Correlation Module
- Reporting Module

The tool is implemented entirely using Bash scripting and standard Linux command-line utilities.

4.2 Evidence Integrity Verification

To preserve forensic soundness, the tool computes **SHA-256 hashes** for all files within the mounted Android image. This ensures:

- Verification of evidence authenticity
- Detection of any post-acquisition modifications
- Support for chain-of-custody documentation

The resulting hashes are stored in `evidence_hashes.txt`.

5. Forensic Artifact Extraction

The tool extracts artifacts directly from SQLite databases and system configuration files found within the Android `/data` partition.

5.1 Call Logs

- Source: `calllog.db`
- Extracted Fields:
 - Timestamp
 - Phone number
 - Call duration
 - Call type

Call timestamps are converted from Unix epoch format into human-readable date and time.

5.2 SMS Messages

- Source: `mmssms.db`
- Extracted Fields:

- Timestamp
- Sender/receiver address
- Message body

SMS records are exported into CSV format for further analysis.

5.3 Contacts

- Source: `contacts2.db`
- Extracted Fields:
 - Display name

This artifact supports identity correlation with call and SMS records.

5.4 Browser History

- Source: Chrome `History` database
- Extracted Fields:
 - Last visit timestamp
 - Visited URL

Chrome timestamps are converted from WebKit time to Unix epoch format.

5.5 Installed Applications

- Source: `packages.xml`
- Extracted Data:
 - Application package names

This artifact provides insight into user behavior and installed services.

5.6 WhatsApp Messages (Optional)

- Source: `msgstore.db`
- Extracted Fields:
 - Message timestamp
 - Message content

Additionally, URLs shared through WhatsApp are identified and extracted for further investigation.

6. Data Analysis and Correlation

6.1 Statistical Analysis

The tool performs basic statistical processing to identify:

- Most frequently called phone numbers
- Most frequent SMS senders

These statistics assist investigators in identifying key contacts.

6.2 Unified Timeline Construction

One of the core contributions of this tool is the **unified timeline**, which correlates:

- Call logs
- SMS messages
- Browser history

All events are normalized to a common timestamp format and merged into a chronological timeline, enabling investigators to reconstruct user activity patterns.

7. Output and Reporting

The tool generates both machine-readable and human-readable outputs:

- CSV files for forensic tools and spreadsheets
- Plain-text reports for quick review
- Analytical summaries and timelines

All outputs are stored in a case-specific, timestamped directory to ensure traceability.

8. Error Handling and Limitations

8.1 Error Handling

If an artifact database is missing:

- The tool skips the extraction
 - A warning message is displayed
 - Execution continues without termination
-

8.2 Limitations

- Encrypted databases are not supported
 - WhatsApp extraction requires an unencrypted database
 - Chrome history extraction depends on user profile availability
 - The tool requires a pre-mounted Android image
-

9. Forensic and Legal Considerations

The tool is designed strictly for **offline forensic analysis** and does not alter evidence. Investigators must ensure:

- Legal authorization for evidence access
 - Proper documentation of acquisition methods
 - Compliance with applicable laws and regulations
-

10. Conclusion

This paper presented **Android Mobile Forensics Parser**, a Bash-based tool for automated Android forensic analysis. The tool demonstrates that effective forensic analysis can be achieved using open-source utilities while maintaining transparency and forensic soundness.

By automating artifact extraction, integrity verification, and timeline correlation, the proposed solution provides a practical and educational platform for digital forensics investigations. Future enhancements may include support for encrypted databases, additional applications, and visualization capabilities.

```
osboxes@osboxes: ~/android_forensics_1_2025-12-22_114913
File Actions Edit View Help
analysis csv evidence_hashes.txt
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$ cd ..
[osboxes@osboxes] ~
$ cd android_forensics_1_2025-12-22_114913
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$ ls
analysis browser_history.txt call_logs.txt contacts.txt csv evidence_hashes.txt installed_apps.txt sms_messages.txt
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$ cat browser_history.txt
"2019-11-06 05:12:00" http://maps.google.com/search/warehouse-location
"2019-11-06 05:20:05" http://duckduckgo.com/?q=secure-messaging-apps
"2019-11-06 05:20:10" http://example.com/temp-email
"2019-11-06 05:20:15" http://howtodeletehistory.com
"2019-11-06 05:20:20" http://maps.google.com/search/meeting+point
"2019-11-06 05:20:25" http://encryptedchat.com/login
"2019-11-06 05:20:30" http://torproject.org
"2019-11-06 05:20:35" http://safecloudstorage.com
"2019-11-06 05:20:40" http://example.com/suspicious-doc
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$ cat contacts.txt
"Omar Hassan"
"Unknown Contact"
Delivery
"Alice Cooper"
"Bob Marley"
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$ cat sms_messages.txt
"2023-11-14 22:11:20" +2012222222222 "Are we still on for tonight?"
"2023-11-14 22:18:20" +2012222222222 "Yes. Same place. Same time."
"2023-11-14 22:27:10" +2012222222222 "Make sure no one follows you."
"2023-11-14 22:28:20" +2012222222222 "I checked, we're clear."
"2023-11-14 22:33:20" +2012222222222 "Delete this after reading."
"2023-11-14 22:36:20" +2012222222222 "Already done."
"2023-11-14 22:43:20" +2012222222222 "Bring the documents."
"2023-11-14 22:48:20" +2012222222222 "I have them."
"2023-11-14 22:53:20" +2012222222222 "This stays between us."
"2023-11-14 22:58:20" +2012222222222 "We must meet ASAP."
"2023-11-14 23:03:20" +2012222222222 "Confirm location?"
"2023-11-14 23:08:20" +2012222222222 "Safe route?"
"2023-11-14 23:13:20" +2012222222222 "Check the package."
"2023-11-14 23:18:20" +2012222222222 "Is it secure?"
"2023-11-14 23:23:20" +2012222222222 "Do not tell anyone."
"2023-11-14 23:28:20" +2012222222222 "Everything is ready."
[osboxes@osboxes] ~/android_forensics_1_2025-12-22_114913
$
```