Android Digital Forensics Tool

1. Introduction

With the widespread use of smartphones, Android devices have become a primary source of digital evidence in criminal investigations, corporate incident response, and civil disputes. These devices store large volumes of sensitive user data such as messages, call records, application usage, browsing history, and multimedia files. Extracting this data in a forensically sound manner is increasingly challenging due to modern Android security mechanisms, encryption, and permission restrictions.

This project presents an **Android Digital Forensics Tool**, an open-source, Python-based acquisition and analysis suite designed to extract, parse, and visualize forensic evidence from Android devices running Android 5.0 up to Android 14 and later. The tool is optimized for both Windows and Linux environments and supports both rooted and non-rooted devices.

2. Problem Statement

Traditional Android forensic acquisition tools often suffer from one or more of the following limitations:

- Heavy reliance on root access, which is not always available or legally permissible
- Limited support for modern Android versions
- Fragmented outputs requiring external tools such as spreadsheets or third-party viewers
- Lack of automation and poor case organization
- Minimal verification of forensic integrity

There is a need for a unified forensic solution that:

- Works on both rooted and non-rooted devices

- Uses multiple extraction techniques to maximize evidence recovery
- Presents results in a readable and investigator-friendly format
- Maintains forensic soundness through hashing and structured case management

3. Project Objectives

The main objectives of this project are:

1. To design a multi-method Android forensic acquisition tool capable of extracting user, application, and system data.
2. To support both rooted and non-rooted Android devices without requiring invasive modifications.
3. To parse raw forensic artifacts into meaningful, readable evidence.
4. To provide an integrated report viewer and timeline analysis interface.
5. To ensure forensic integrity through hashing and organized evidence storage.

4. System Overview

The Android Digital Forensics Tool is built using Python and leverages Android Debug Bridge (ADB) for device communication. It combines multiple data extraction strategies and presents the results through a graphical user interface (GUI).

Supported Platforms

- Windows
- Linux

Supported Android Versions

- Android 5.0 to Android 14+

5. Extraction Methodology

To maximize evidence recovery, the tool uses a **multi-vector extraction strategy**. If one method fails due to system restrictions, the tool automatically falls back to alternative techniques.

5.1 Extraction Methods

1. Backup-Based Extraction (Non-Root)

- Uses standard Android backup mechanisms
- Requires user confirmation on the device
- Suitable for most modern devices
- Safely retrieves data without elevated privileges

2. Root-Based Extraction

- Direct access to protected system directories
- Faster and deeper data recovery
- Allows access to databases inside `/data/`
- Recommended only for legally authorized rooted devices

3. API-Based Content Queries

- Uses Android content providers
- Extracts SMS, call logs, contacts, and usage statistics
- Works when backups are blocked on newer Android versions

4. Shared Storage Acquisition

- Automatically retrieves media and documents from public directories such as:
  - `/sdcard/DCIM`
  - `/sdcard/Documents`
  - `/sdcard/Downloads`

6. Extracted Evidence Categories

6.1 Communication Artifacts

- SMS and MMS messages with timestamps and sender information
- Call logs including incoming, outgoing, and missed calls
- Contacts and phone numbers
- User dictionary entries, which may reveal frequently typed words or names
- Calendar events with scheduling information

6.2 Application and System Intelligence

- Installed third-party applications with installation dates
- Application usage statistics to establish activity timelines
- Permission analysis highlighting apps with sensitive privileges
- Chrome browser history including URLs and visit times

6.3 File System Evidence

- Photos, videos, and documents from shared storage
- User-generated files commonly relevant in investigations

7. Analysis and Reporting Features

A major strength of this tool is its **built-in analysis and reporting interface**, eliminating the need for external analysis software.

7.1 Built-in Report Viewer

- Centralized dashboard for all extracted evidence
- No reliance on CSV or spreadsheet tools

7.2 Timeline Analysis

- Unified chronological view of:

- Messages
- Calls
- App usage
- System events

7.3 Filtering and Search

- Filter evidence by category (SMS, Calls, Calendar, App Usage)
- Full-text search for keywords, phone numbers, or names

7.4 Detailed Inspection

- Ability to view complete message bodies
- Preserves special characters and emojis
- Supports detailed evidence review

8. Smart Extraction Engine

The tool includes automation features designed to support professional forensic workflows:

- Automatic detection of rooted vs non-rooted devices
- Intelligent method switching when extraction fails
- Structured case management using time-stamped folders
- MD5 hash generation for all extracted files to preserve integrity

9. Installation and Usage

9.1 Prerequisites

- Python 3.8 or higher
- Android Debug Bridge (ADB) installed and added to system PATH

9.2 Installation Steps

1. Clone the repository:

```
git clone https://github.com/Start-5/AndroidForensicsTool.git
```

2. Install required dependencies:

```
pip install -r requirements.txt
```

9.3 Running the Tool

```
python gui_windows.py
```

9.4 Evidence Extraction Workflow

1. Connect Android device via USB
2. Verify connection using the built-in check
3. Select extraction method (Backup or Root)
4. Parse and generate report
5. Analyze evidence using the integrated viewer

10. Limitations

Despite its strengths, the tool has some limitations:

- Some Android versions restrict full backups for certain applications
- Root-based extraction depends on device configuration
- Encrypted app data may not always be accessible on non-rooted devices
- Requires user interaction during backup confirmation

11. Legal and Ethical Considerations

This tool is intended strictly for:

- Educational use
- Research
- Authorized forensic investigations

Users must ensure they have proper legal authorization such as device ownership, explicit consent, or a valid warrant before extracting data. Unauthorized use may violate privacy laws and regulations.

12. Conclusion

The Android Digital Forensics Tool provides a practical, flexible, and modern solution for Android forensic acquisition and analysis. By combining multiple extraction techniques with a user-friendly reporting interface, it addresses many limitations of traditional Android forensic tools.

Its support for both rooted and non-rooted devices, structured case management, and built-in analysis capabilities make it a valuable asset for students, researchers, and forensic practitioners.