

**CY2002**

**Digital Forensics**

**Project**

**ANDROID ROOT DETECTOR**

**Submitted by:** MANAHIL CHOUDHRY, ASAD KHURSHID, EHTISHAM Ul-HASSAN

**Roll number:** I22-1728, I221585 ,I22-1777

**Date:** 17 Nov, 2024

**Table of Contents**

[Introduction 2](#_Toc182774293)

[Details and Steps 2](#_Toc182774294)

[Prerequisites 2](#_Toc182774295)

[Enable USB Debugging on the Android Device 2](#_Toc182774296)

[Connect the Android Device to the PC 2](#_Toc182774297)

[Running the Script 2](#_Toc182774298)

[Check Root Status 2](#_Toc182774299)

[Fetch Partition Table 3](#_Toc182774300)

[Fetch Names of Installed Apps 4](#_Toc182774301)

[Summary 6](#_Toc182774302)

# Introduction

The Android Device Analysis Tool is a Python-based script designed to assist in the analysis of Android devices through ADB (Android Debug Bridge). This tool provides essential functionalities for digital forensics and device analysis, including root detection, partition information retrieval, and installed application listing. The script is intended for use by digital forensics professionals, security researchers, and Android developers who need to gather critical information from Android devices quickly and efficiently.

# Details and Steps

This guide provides a step-by-step process to run the root detection script.

## **Prerequisites**

Before starting, ensure the following are in place:

1. Python 3.x
2. ADB (Android Debug Bridge) installed and accessible from the command line
3. An Android device with USB debugging enabled

## **Enable USB Debugging on the Android Device**

1. Open the **Settings** menu on the Android device.
2. Go to **About Phone** and tap **Build Number** multiple times (usually 7) to enable Developer Mode.
3. Return to **Settings** and open **Developer Options**.
4. Locate **USB Debugging** and enable it.
5. Confirm the warning message by selecting **OK**.

## **Connect the Android Device to the PC**

1. Use a USB cable to connect the Android device to the PC.
2. Ensure the device prompts you to allow USB debugging access. Select **OK** to allow the connection.

## **Running the Script**

1. Open the terminal or command prompt on your PC.
2. Navigate to the directory where the Python script is located.
3. Execute the script using Python (refer to the script’s user instructions for exact commands).
4. Run the script using Python: `python main.py`
5. Navigate through the menu options to perform desired operations.

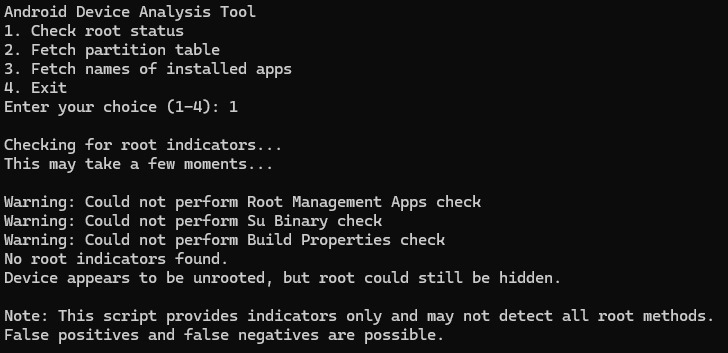
The script performs several checks and operations, including ADB connection verification, root indicator checks, partition table retrieval, and installed app listing.

# Check Root Status

This feature examines the device for indicators of root access. It performs the following checks:

* Searches for root management apps (e.g., SuperSU, Magisk)
* Checks for the presence of the `su` binary
* Examines build properties for test keys
* Verifies if the system partition is mounted as read-write

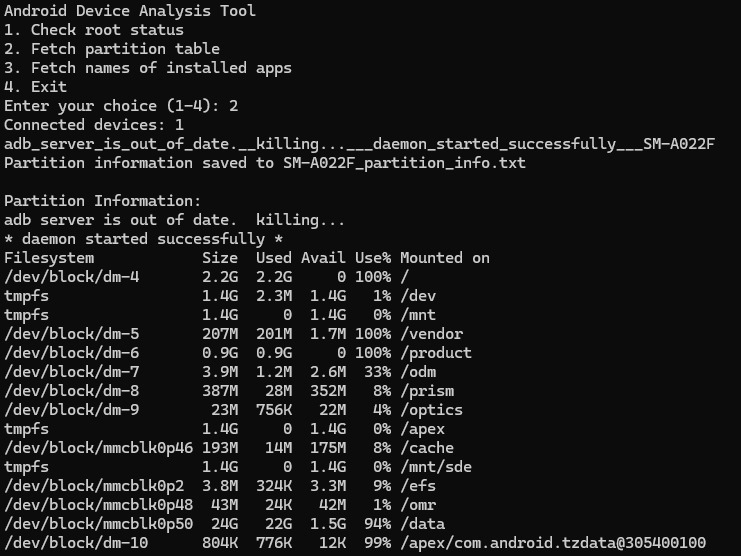
The function provides a list of detected root indicators, if any, along with a warning about potential false positives or negatives.



# Fetch Partition Table

This feature retrieves and displays the partition information of the connected device. It performs the following steps:

* Fetches the device model name
* Executes the `df -h` command to get partition information
* Saves the partition information to a text file named after the device model
* Displays the partition information on the console

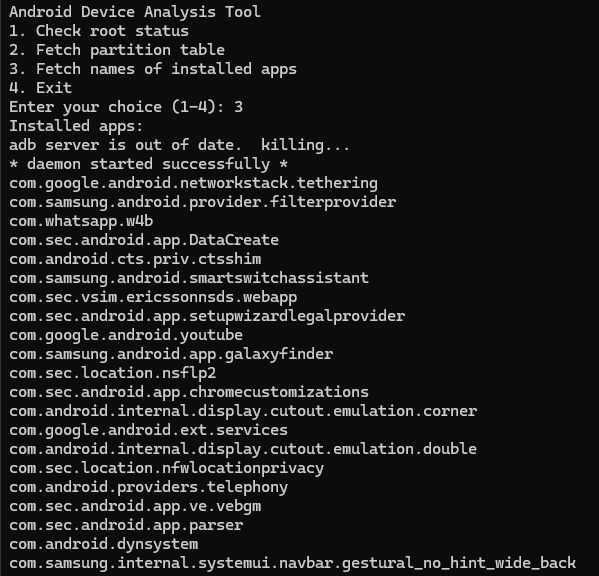
This feature is useful for understanding the storage structure of the device and identifying potential areas for further investigation.  


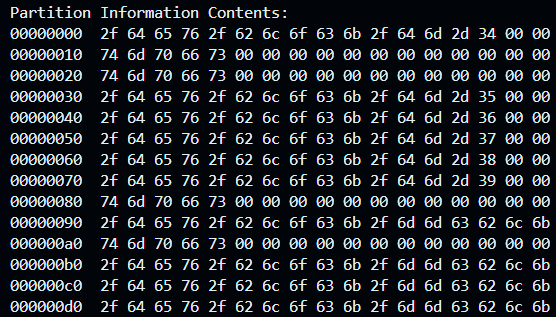
# Fetch Names of Installed Apps

This feature lists all applications installed on the device. It:

* Executes the `pm list packages` command via ADB
* Processes the output to extract package names
* Displays the list of installed applications on the console

This feature can help identify potentially malicious or unauthorized applications on the device.





# Summary

The Android Device Analysis Tool provides a streamlined approach to gathering crucial information from Android devices. Its key strengths include:

* Easy-to-use command-line interface
* Multiple analysis features in a single tool
* Root detection capabilities
* Partition information retrieval
* Installed application listing

While the tool is powerful, it's important to note its limitations:

* Reliance on ADB, which requires USB debugging to be enabled
* Potential for false positives/negatives in root detection
* Limited to information accessible without root privileges

Overall, this tool serves as a valuable asset for initial device analysis in digital forensics investigations, security audits, and Android development processes. It provides a solid foundation for more in-depth analysis and can be extended with additional features as needed.