International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 3 Issue 4, April - 2014

Image Authentication using JPEG Headers

Parthiban.R
Information Security and Computer Forensics
SRM University
Chennai, India

***Abstract* - Evidence, in general is that which provides direct proof of the truth of an assertion. An obstacle in the digital evidence investigation is the investigator's ability to determine whether the pictures in question have been altered and finding the true source of the image. It is evident that smart phones are becoming the main camera of choice by many people and JPEG is the most commonly used image format by mobile phones owing to high compression value and less memory consumption. A JPEG [10] image that is produced by a camera does not just contain a picture, it also has metadata associated in its header. Metadata provides information about the image. EXIF [3] is a standard which is commonly used in mobile devices to write metadata into images. In this paper, we propose a solution to identify the device, which uses the image metadata. The proposed method has two phases; the first phase will embed the hash value of the device's IMEI into the JPEG metadata. Each device will be associated with unique IMEI [7] number hence by storing the hash of the IMEI to the image metadata we can uniquely identify the device from which the image was taken. Second phase is analysis of the metadata, during the analysis the device's IMEI will be checked against the image metadata where the device's IMEI is stored.**

*Keywords: **Metadata, JPEG, IMEI, EXIF interface***

I.INTRODUCTION

When digital images are submitted as evidence to court, as per court of law we are under obligation to prove two main things. The first one is to prove that the device seized was used to take digital image (i.e. Source of the image). Second is to prove that the image was not altered using any software. As of now, we have the ability of identifying only the model of the device but not the particular device, hence raised a situation for a unique identifier to prove the source of the image.

The latest technological fab in the recent days is the mobile cameras which are widely used by people for taking everyday pictures due to its simplicity and also since it's a cheaper alternative. These cameras have the tendency to be exploited by unethical means and during such times there arose a problem of identifying the source of the device. By default all android phones will embed only the maker details, timestamp and resolution of the image to the EXIF metadata, the maker details will have only the model number of the device. These details alone cannot be utilized to identify the device from which the image was taken.

When professional cameras are considered, the manufacturers embed the serial number of the device into the image metadata so as to identify a particular device. This can be used to match the device serial number with the serial number stored in the metadata of the image.

Past research on identifying the source of the image were based on mathematical calculation methods. Some methods explored different processing stages of the digital camera, such as sensor imperfection (PRNU, photo response non uniformity) and Error level analysis [3]. These methods will process the image and the values are compared against the known values of a particular camera. These methods though help identifying the devices are not said to be very accurate. Hence in this paper we provide a simple and efficient way to identify the source of the image using the image metadata.

II. RELATED WORK

2.1 Image Life Cycle

A digital image is a reproduction of an object formed by a lens. The light from a real scene is framed by the digital camera by focusing the lenses on the camera sensor (a CCD or a CMOS). This generates digital image signal. The light is usually filtered by the CFA (Color Filter Array) before reaching the sensor. Color filter array is a thin film on the sensor that selectively permits a certain component of light to pass through it to the sensor. Each pixel gathers only one particular main color which are Red, Green or Blue. The output from the sensor is interpolated to obtain all the three main colors for each pixel, through demosaicing process. This obtains the digital color image. The obtained signal further undergoes in-camera processing that includes white balancing, color processing, image sharpening, contrast enhancement, and gamma correction.

The processed signal is then stored in the camera memory. Most cameras save the image in compressed format for efficient storing. JPEG format is usually preferred owing to its high compression with least loss in detail.

The generated image is finally post processed, to enhance or to modify its content. Image editing can be done to an image for enhancement; the most used ones are geometric transformation (rotation, scaling, etc.), blurring, sharpening, contrast adjustment, image splicing (the composition of an image using parts of one or more parts of images), and cloning (or copy-move, the replication of a portion of the same image). After editing, the resultant image is very often stored in JPEG [10] format, so that a recompression will occur.

2.2 Metadata:

Each image has its own metadata [1]. Metadata's provide information about a picture's lineage such as color space information, type of camera used, and application notes. There are different picture formats in existence and each format has its own type of metadata. Some formats are PBM, BMP and PPM. These above mentioned types contain very little information beyond the image dimensions and color space.

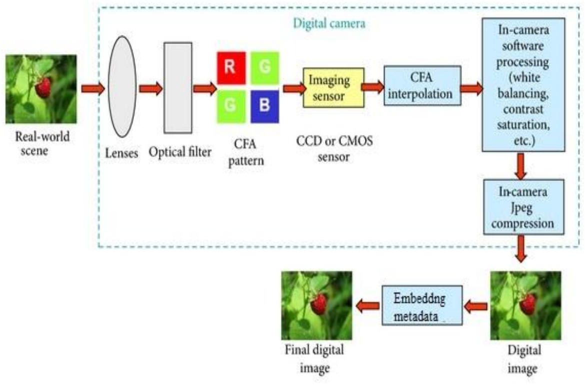

Fig. 1. Life cycle of an Image

In Fig. 1 the life cycle of an image is explained, which shows each stage of an image from the time it was captured. Comparatively, a JPEG format usually contains a wide variety of information, such as the camera's make and model, focal and aperture information, and also the timestamps.

2.3 Classes of Metadata

There are Different types known as classes of text information about a digital image, which serves a specific purpose. Some classes of metadata can be directly embedded while others have restriction. Some data formats, identify their elements by these classes, although this may not be readily apparent. Each of the following three classes of metadata become part of the image file when embedded in JPEG, TIFF, PSD, Raw or several other popular formats.

Technical Metadata

These metadata contain the technical characteristics of both the camera and the image such as shutter speed, size, color profile, ISO speed and other camera settings.

Descriptive Metadata

The descriptive metadata field contains manually added entries in the metadata. These fields contain values such as identification, location etc.

Administrative Metadata

Administrative metadata is used for the management of administrative details such as licensing, rights usage terms, specific restrictions on using an image, model releases, provenance information, such as the identity of the creator, and contact information for the rights holder or licensor.

Metadata Types

There are many different types of metadata. The most common types of metadata that we likely encounter are include File, EXIF, Maker Notes, IPTC, ICC Profiles, XMP, PrintIM.

2.4 EXIF:

The primary feature of EXIF is its ability to record camera information in an image file at the point of capture. The data fields of EXIF includes the camera's make, model, its serial number, the date and time of image capture, the shutter speed, the aperture, the lens used and the ISO speed setting. They also include other technical details, such as white balance, distance of the subject.

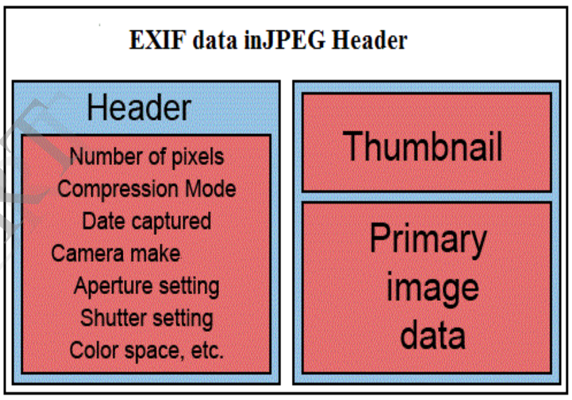

Fig. 2. JPEG Header

In Fig. 2 the JPEG header is shown with the fields Header, Thumbnail and Image data. The metadata, found in the JPEG header [9], stores multiple information's about the camera and image. According to the EXIF standard, there are five main image file directories (IFDs) into which the metadata is organized: Primary, EXIF, Interoperability, Thumbnail, and GPS.

.

III. METHODOLOGY

The proposed methodology is simple, easy to use and analyze. It is completely based on the metadata which is stored in the images, while the image was taken.
The hardware and software requirements for forensic investigation are as following.

Hardware:

- Desktop PC- forensic workstation
- 80GB SATA Hard Drive

- Android Device (Onetime use)

Software:
- Microsoft Windows XP SP3
- Android SDK

The proposed method has two phases,

1. Embedding hash of the IMEI
2. Analysis of metadata

3.1 Extracting IMEI and Hashing

Every device will have a unique ID to particularly identify them. There are many kinds of unique ID's such as MAC, Android ID, IMEI, etc. IMEI (International Mobile Equipment Identity) is a 15- or 17-digit code that uniquely identifies mobile phone. In our proposed model we are using IMEI to identify the source as spoofing it can be cumbersome. The IMEI in general is stored in the android TelephonyManager module of the android Operating System. Below is the code for android telephony manager module,

```
TelephonyManager telephonyManager =
(TelephonyManager)
getSystemService(Context.TELEPHONY_SE
RVICE);
        String          imei         =
String.valueOf(telephonyManager.getDe
viceId());

        try {
            imei = SHA1(imei); //
hashing imei
        }                       catch
(NoSuchAlgorithmException e1) {
  e1.printStackTrace();
        }                       catch
(UnsupportedEncodingException e1) {
    e1.printStackTrace();}
```

After the extraction of IMEI from the device, hashing is done. We are using SHA algorithm to hash the IMEI. Hash is done in real time, because hash of the IMEI is computed every time when the user takes a new picture. This hashing preserves the privacy of the users so as to prevent the misuse of IMEI. Hashing is the transformation of a string of characters into a usually shorter fixed-length value or key that represents the original string. Hashing is used to index and retrieve items in a database because it is faster to find the item using the shorter hashed key than to find it using the original value. It is also used in many encryption algorithms. The hashing algorithm is called hash function.

The hash function is used to index the original value or key and then used later each time the data associated with the value or key is to be retrieved. Thus, hashing is always a one-way operation. There's no need to "reverse engineer" the hash function by analyzing the hashed values. In fact, the ideal hash function can't be derived by such analysis. A good hash function also should not produce the same hash value from two different inputs. If it does, this is known as a collision. A hash function that offers an extremely low risk of collision may be considered acceptable.

3.2 Embedding IMEI into Metadata

Once the hash of the IMEI is computed, we have to embed it into the metadata. The metadata type used in android operating system is EXIF. To embed the hash to the metadata, we will have to use the exifinterface() function in the android API. The exifinterface [4] is responsible for writing metadata information into images.

```
ExifInterface exif;
try {

                  exif       =       new
ExifInterface(mediaStorageDir.getPath
() + File.separator
                          + "IMG_" +
temp + ".jpg");
    exif.setAttribute(ExifInterface.
TAG_MODEL, imei);

    exif.saveAttributes();
        } catch (IOException e) {

    exif.saveAttributes();
e.printStackTrace();
        }}
```

3.3 Analysis phase

3.3.1 Recovering HASH from Image

In the analysis phase, we are giving the evidence image as input to our analyzer. This analyzer will recover the metadata from the image; the recovered metadata from the image will have the hash of the IMEI of the device.

3.3.2 Obtaining IMEI from the Device

The confiscated device is then connected to the workstation which is used for analysis. There is no need for any specialized software module that needs to be installed in the device to get IMEI from the confiscated device. All that is necessary is an android SDK to be installed in the analysis workstation. The IMEI of the device will be obtained from the device by using android debugging bridge (ADB).

This ADB [5] is a command line tool, which allows us to communicate with the connected android device, when an ADB client is started, the client first checks whether there is an ADB server process already running. If there isn't, it starts the server process. When the server starts, it binds to local TCP port 5037 and listens for commands sent from ADB clients, all ADB clients use port 5037 to communicate with the ADB server. The server then sets up connections to all running emulator/device instances. It locates emulator/device instances by scanning odd-numbered ports in the range 5555 to 5585, the range used by emulators/devices. When the server finds an ADB daemon, it sets up a connection to that port. Each emulator/device instance acquires a pair of sequential ports an even-numbered port for console connections and an odd-numbered port for ADB connections. Hence ADB will be able to obtain the hardware details and the logs from the android device. By using the command iphonesubinfo we will be able to get the IMEI of the device via the command line.

After getting the IMEI from the device, we will have to hash it using SHA to match it against the hash recovered from the image metadata. In Fig. 3. The process of embedding the IMEI hash to the image is explained. SHA-1 is one of several cryptographic hash functions, most often used to verify that a file has been unaltered. SHA [8] is short for Secure Hash Algorithm. Here we are hashing the IMEI, in order to prevent the privacy of the user. By having the IMEI publicly available it is a risk to the privacy of the users.

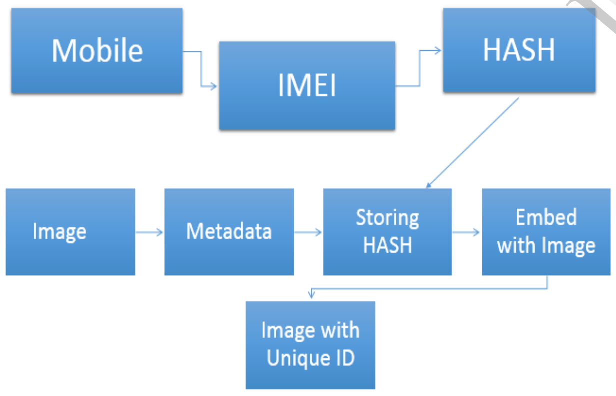

Fig.3. Embedding IMEI to Image

3.3.3 Comparing both hashes

After getting the hash from both the mobile and from the image, we will be comparing it against each other. If both hashes match, we will be able to prove that the device confiscated was used to capture the image. Thus proving that the source device from which the image was taken was indeed the confiscated device. Fig. 4 shows the various processes carried out in the Analysis Phase.

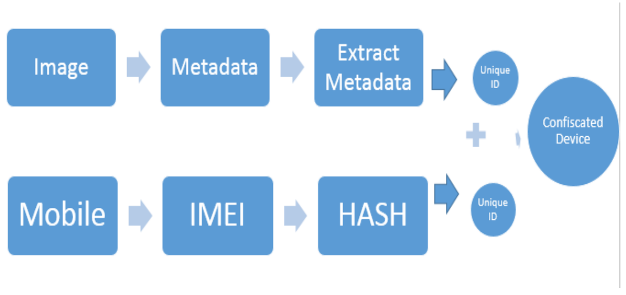

Fig. 4. Analysis Phase

IV. FUTURE WORKS

In this paper we have introduced this method as a separate application for android device. Instead, if this proposed methodology is implemented as a default application in the core camera, all the images will by default have a unique ID making it easy to prove the source of the device from which the image was taken.

V. CONCLUSION

When a digital image is submitted as an evidence to the court, the image is verified against source of the image and the originality of the image. This originality of the image can be proved by mathematical methods such as Error level Analysis and calculating Quantization tables etc. the values will be compared against the known values of the cameras. If there is no variation in the value, we can prove that this image is not altered. To identify the source of the image this paper provides a simple method. The proposed method uses the image metadata which is stored in the image. This method takes hash of the IMEI and storing it into the image data while capturing the image. Hence each image will have a unique identifier to identify the device from which the image was taken. Thus allowing us to prove the source of the evidence image.

VI. REFERENCE

[1]. http://www.photometadata.org/meta-101
[2]. http://fotoforensics.com/tutorial-ela.php
[3]. http://en.wikipedia.org/wiki/Exchangeable_image_file_format.
[4]. http://developer.android.com/reference/android/media/ExifInterface.html.
[5]. http://developer.android.com/tools/help/adb.html.
[6]. http://msdn.microsoft.com/en- us/library /xdd t0dz7%28v=vs.110%29.aspx
[7]. http://en.wikipedia.org/wiki/International_Mobile_Station_Equipment_Identity
[8]. http://en.wikipedia.org/wiki/Secure_Hash_Algorithm.
[9]. http://www.fastgraph.com/help/jpeg_header_format.html.
[10]. http://www.fileformat.info/format/jpeg/egff.html.