Forensic Analysis of Ordered Data Structures on the Example of JPEG Files

Thomas Gloe

Institute of Systems Architecture, Technische Universität Dresden
01062 Dresden, Germany
thomas.gloe@tu-dresden.de

***Abstract*—JPEG file format standards define only a limited number of mandatory data structures and leave room for interpretation. Differences between implementations employed in digital cameras, image processing software, and software to edit metadata provide valuable clues for basic authentication of digital images. We show that there exists a realistic chance to fool state-of-the-art image file forensic methods using available software tools and introduce the analysis of ordered data structures on the example of JPEG file formats and the EXIF metadata format as countermeasure. The proposed analysis approach enables basic investigations of image authenticity and documents a much better trustworthiness of EXIF metadata than commonly accepted. Manipulations created with the renowned metadata editor ExifTool and various image processing software can be reliably detected. Analysing the sequence of elements in complex data structures is not limited to JPEG files and might be a general principle applicable to different multimedia formats.**

I. Introduction

Methods to determine the source or to detect manipulations of an image are of growing importance whenever authenticity is relevant, like in court or in the news industry. Recent attacks against Canon's and Nikon's professional DSLR camera image authentication systems demonstrate weaknesses of in-camera solutions to acquire provable authentic images[1]. The general lack of authentication systems in most digital cameras gave raise to image forensics purely relying on the available image data.

The ample body of image forensic literature focusses on statistical methods that analyse either device characteristics or image processing artefacts [1]. Basic analysis methods of the image file format and belonging metadata received less attention [2]. For example, metadata embedded in JPEG files is commonly accepted as being less reliable with very limited use in image forensic investigations [1]. Faking metadata is believed to be possible without leaving suspicious traces and, unsurprisingly, a German court decided that EXIF metadata [3] as part of JPEG camera images represents no probative fact to proof the origin and the time of acquisition of an image [4]. The possibility to create perfect imitations of deterministic computer-generated artefacts, like JPEG compression settings and metadata, is a problem relevant to computer forensics in general [5]. Considering analysed characteristics in current state-of-the-art image file forensic methods, we will discuss possibilities to create 'undetectable' forgeries using commonly available image processing tools. The observed weaknesses call for an exploration of the limits of image file forensics. Investigating JPEG and EXIF formats in detail, we identified new characteristics based on the file structure robust against manipulations using existing tools. In consequence, generating 'undetectable' forgeries is much more complicated than commonly expected and requires advanced knowledge of file formats. The proposed analysis approach gives investigators a tool for basic image authentication, which can be easily adapted to different sorts of multimedia formats.

Starting with an overview of current state-of-the-art methods to analyse deterministic computer-generated artefacts of JPEG files and EXIF metadata in Sec. II, Sec. III discusses possibilities to create convincing forgeries undetectable using present image file forensic approaches. Based on our practical test setup (Sec. IV), we investigate the structure of JPEG files and EXIF metadata and introduce new characteristics in Secs. V and VI, respectively. Section VII summarises our investigations and discusses implications for the practice.

II. State-of-the-Art Image File Forensic Analysis

The JPEG standard [6] defines both, a lossy compression scheme and a file format, and is implemented in all digital cameras to store images of natural scenes. Only a limited number of devices allows to use proprietary raw image formats as alternative (to avoid compression). Raw image files cannot be displayed directly and require further processing typically resulting in a loss of forensic characteristics of the original file format. Thus, present approaches focus on the omnipresent JPEG standard[2] and state-of-the-art image file forensic methods are based on the analysis of basic compression parameters, thumbnail images, and metadata stored in a JPEG file:

Quantisation tables control the loss of information during compression and are stored as individual part of each JPEG file. Digital cameras and image processing software (or groups

[1] http://www.elcomsoft.de/nikon.html, http://www.elcomsoft.de/canon.html

[2] A discussion on statistical analysis approaches of JPEG compression artefacts is out of the scope of this paper and a comprehensive list of references created for Ref. [7] is made available online at http://forensics.inf.tu-dresden.de/ddimgdb/publications/jpeg.

WIFS'2012, December, 2-5, 2012, Tenerife, Spain.
978-1-4673-2287-4/12/$31.00 © 2012 IEEE.

thereof) use their own customised quantisation tables, which can be analysed for basic image source identification [8], [9].

Some image processing tools leave the *thumbnail image* of JPEG files untouched preserving accurate details about the original content [10]. Thumbnail images existing in a JPEG file are typically stored itself as complete JPEGs and the implementation to generate thumbnails differs between digital cameras. Kee and Farid propose to investigate basic thumbnail parameters, like dimensions and quantisation tables, together with a statistical analysis of the thumbnail processing history to distinguish between original and altered JPEG images [11].

The state-of-the-art approach [2] to analyse *EXIF metadata* combines counts of standard and non-standard EXIF entries as well as parser errors in the EXIF metadata together with basic JPEG parameters (*image dimensions*, quantisation tables, *subsampling parameters* and *Huffman tables*) of the main image and the standard EXIF thumbnail to aggregate a set of 576 features.

Another branch of image file forensic methods is file carving, where elements of the file structure are analysed to restore deleted images on a storage medium [12], [13]. Similar to carving methods, we will investigate in detail the structure of JPEG files and EXIF metadata, and identify structural differences between groups of camera models and image processing software as new characteristics for image authentication.

III. Counter Image File Forensic Tools

Creating manipulated images falsely detected as authentic by present file forensic methods requires counter forensic capabilities to adjust image dimensions, quantisation tables, Huffman tables and subsampling parameters of the main image and the standard EXIF thumbnail. Furthermore, tools to edit EXIF metadata like time of acquisition or serial numbers are necessary. While an advanced counterfeiter might try to build his own tool to create 'undetectable' forgeries, we think a more realistic scenario would exploit available software packages. To the best of our knowledge, no single solution is available to adjust all characteristics and we use a chain of tools instead.

Counterfeiters trying to alter authentic camera images can use Gimp and its capabilities to save JPEG images with original compression settings and original EXIF metadata. While the basic file characteristics are then consistent to the original file, Gimp does minor but forensically obvious updates of EXIF information. For example, the software tag is changed to 'Gimp + version number' and screen resolution tags are set to the dpi setting of the desktop. Instead of using the EXIF metadata stored by Gimp, `Jhead`[3] can be used to reintegrate the authentic EXIF metadata of the source image.

Convincing forgeries would also update the thumbnail with the forged image content. Required compression parameters to recreate the thumbnail can be extracted from the thumbnail of the authentic image using `JPEGsnoop`[4]. Based on the extracted compression settings, `cjpeg` of the JPEG reference library libJPEG[5] can be used to save a downscaled version of the manipulated image. With the help of `ExifTool`[6] or Jhead the created 'authentic' thumbnail can be reintegrated in the EXIF metadata. Considering the mentioned state-of-the-art file forensic characteristics, the final JPEG image is valid and indistinguishable from never post-processed images.

This tool chain can be easily adopted to arbitrary images, which are either not taken by the desired source device or which were previously altered without using the original settings. Only one additional authentic image is required for reference to extract necessary image file forensic characteristics with JPEGsnoop. Similarly to the creation of valid thumbnails, a combination of cjpeg, Jhead and ExifTool allows to recreate an altered image with consistent characteristics. With the availability of authentic images as supplementary material of camera reviews or product descriptions on the Internet, obtaining reference material is easy. Also image search engines available online can help to find appropriate material, e.g., Goolge's image search or Flickr's camera finder.

Independent of image manipulations, the tool ExifTool allows to edit almost any entry of metadata. For example, a counterfeiter might try to alter the time of acquisition in order to prove his presence at a specific point in time. ExifTool adds no additional information to metadata entries and images are detected as 'authentic' using present image file forensic tools.

IV. Test Setup

In our effort to build up the 'Dresden Image Database' [14] and to make experimental results of state-of-the-art forensic techniques based on images therein available to the interested public, we searched for new characteristics to defeat weaknesses of current image file forensic methods and make counterfeiting more challenging. In this study, we used the auxiliary set of images of the JPEG scene (4,666 images) and all available images of natural scenes (16,956 images). Images of the JPEG scene were captured with one device of each available camera model, while iterating over all combinations of compression, image size and flash (enabled/disabled) settings. The analysed images are created using a set of 26 camera models with altogether 73 devices. Table I gives a short overview. We note that space limitations permit a detailed list of cameras and images and refer the interested reader to the comprehensive description of the database in Ref. [14].

File characteristics of JPEG images stored by image processing software are analysed using images of the JPEG scene acquired in JPEG and uncompressed raw mode with Nikon D200 and Ricoh GX100 (8 images). An automatic script was created to resave each authentic image with each investigated image processing software (c.f. Tab. I) and all available combinations of JPEG compression settings. More than 32,000 images were generated and analysed in this study.

[3] http://www.sentex.net/~mwandel/jhead/

[4] http://www.impulseadventure.com/photo/jpeg-snoop.html

[5] http://www.ijg.org

[6] http://www.sno.phy.queensu.ca/~phil/exiftool/

TABLE I
DIGITAL CAMERA MODELS IN THE 'DRESDEN IMAGE DATABASE' [14] AND SOFTWARE WE FOCUS ON IN THIS STUDY.

make	**models**
Agfa	DC-504, DC-733s, DC-830i, Sensor505-X, Sensor530s
Canon	Ixus 55, Ixus 70
Casio	EX-Z150
FujiFilm	FinePix J50
Kodak	M1063
Nikon	CoolPix S710, D200, D70, D70s
Olympus	μ1050SW
Panasonic	DMC-FZ50
Pentax	Optio A40, Optio W60
Praktica	DCZ5.9
Ricoh	GX100
Rollei	RCP-7325XS
Samsung	L74wide, NV15
Sony	DSC-H50, DSC-T77, DSC-W170
software	**versions**
ExifTool	8.59
Gimp	2.6.11
IrfanView	4.30
Jhead	2.87
cjpeg (libJPEG)	8c
Paint.NET	3.08
PaintShop Pro	8.10, X3
Photoshop (abbr. PS)	CS2, CS3, CS4, CS5, Elements 9 (abbr. E9)

V. SEQUENCE OF JPEG DATA STRUCTURES

The JPEG standard [6] defines a file format to store all information necessary to decompress a JPEG image. While state-of-the-art methods focus on the analysis of compression settings, we investigate the structure of the file format to identify new forensically exploitable characteristics.

The implementation of the full-featured file format *JPEG Interchange Format* (JIF) [6] is quite complex and alternatives covering only subsets of all possibilities are more common: *JPEG File Interchange Format* (JFIF) [15] and *JPEG/EXIF* [3]. Differences in the three file formats affect only some specific elements and allow a straightforward forensic analysis.

Basic structures of JPEG file formats are marker segments [6] and Tab. II indicates mandatory and optional elements in the three file formats. All parameters necessary for JPEG decompression are stored in a specific marker segment. The begin of each marker segment is indicated by an identifier—the marker id. Common abbreviations for marker ids are denoted using typewriter font and 16 bit short values starting with `0xFF` indicate a specific marker id in a JPEG file. Data encapsulated in marker segments can be either compression parameters (e.g., in `DQT`, `DHT` or `SOF`) or application specific metadata, like thumbnails or EXIF metadata (e.g., in `APP0(JFIF)` or `APP1(EXIF)`). Additionally, some markers solely consist of the marker id and indicate state transitions necessary to parse the file format. For example, `SOI` and `EOI` indicate the start and end of a JPEG file and all other JPEG markers have to be placed between these two mandatory markers. Furthermore, restart marker `RST`n might occur only within the compressed data stream directly following the `SOS` marker segment.

Based on our test data, we analysed all available images and extracted all information included in the file format with our own file parser. Most of the extracted data confirm differences between compression parameters of different sources already known in the literature. More interestingly, we found more than 500 adaptive quantisation tables in thumbnails of images acquired with Casio EX-Z150, Nikon S710, Praktica DCZ5.9, Ricoh GX100 and Samsung L74wide. This impressive number clearly puts the feasibility to compile comprehensive databases of camera- and software-specific parameters stored in JPEG files (or proprietary raw files [16]) into question.

TABLE II
COMMON JPEG FILE MARKERS. MANDATORY MARKERS FOR A SPECIFIC FILE FORMAT ARE INDICATED BY ×. NOTE, ALSO JIF REQUIRES A ENTROPY ENCODING TABLE, BUT IS NOT RESTRICTED TO `DHT`.

marker id	short value	JIF	JFIF	EXIF	description
`SOI`	`0xFFD8`	×	×	×	start of image
`APPn`	`0xFFE`n				application data
`APP0`	`0xFFE0`		×		(e.g., JFIF)
`APP1`	`0xFFE1`			×	(e.g., EXIF)
`DQT`	`0xFFDB`	×	×	×	quant. tables
`DHT`	`0xFFC4`	(×)	×	×	Huffman tables
`SOF`	`0xFFC`n	×			start of frame
`SOF`	`0xFFC0`		×	×	(baseline DCT)
`SOS`	`0xFFDA`	×	×	×	start of scan
`DRI`	`0xFFDD`				restart interval
`RST`n	`0xFFD`n				nth restart
`COM`	`0xFFFE`				comment
`EOI`	`0xFFD9`	×	×	×	end of image

Here, we will focus on the sequence and occurrence of marker segments. Tables III–IV summarise results of our analysis in main images and thumbnails created with digital cameras and image processing software, respectively. Note, `APP` marker segments can be used to store different types of additional information (e.g., metadata in EXIF, `APP1(EXIF)`, or XMP format, `APP1(XMP)`). We denote identifying character strings found at a segment's beginning (e.g., JFIF or EXIF) where available. The standard location for thumbnails is the 'image file directory 1' (IFD1) of the `APP1(EXIF)` segment (c.f. Sec. VI) and is used by all digital cameras and image processing software considered in this study[7]. In several camera images, we observed additional thumbnails in the maker note EXIF entry of `APP1(EXIF)`, in Flashpix segments `APP2(FPXR)` or as post thumbnail appended to the end of the main image after `EOI`. Among the investigated image processing software, only Photoshop stores a second thumbnail in the manufacturer-specific `APP13(PS3)` segment. Some software (IrfanView, Paint.NET and PaintShop Pro 8.10) was not able to update the original thumbnail in EXIF IFD1 and we excluded camera-specific sequences in Tab. IV.

All analysed main images acquired with digital cameras start with a sequence of `SOI`, `APP1(EXIF)` segments indicating the consistent use of the JPEG/EXIF file format. Following the EXIF standard, corresponding thumbnails are stored without any `APP` segment in their own sequence. Depending on the camera model, occurrence and sequence of segments differ in main images as well as thumbnails. Some models store additional `APP` segments in the main image and the behaviour

[7]The JFIF standard [15] defines its own location to store thumbnails in `APP0(JFIF)`, but the investigated software stored a thumbnail always in an `APP1(EXIF)` segment even when the used file format is JFIF.

TABLE III
SEQUENCE OF JPEG MARKER SEGMENTS IN THE MAIN IMAGE AND CORRESPONDING THUMBNAILS ACQUIRED WITH DIGITAL CAMERAS.

camera model / **type of thumbnail**: camera model	sequence of JPEG marker segments
EX-Z150	SOI APP1(EXIF) APP15(TEXT) DQT DHT SOF0 SOS EOI
M1063	SOI APP1(EXIF) APP2(FPXR) $n\times$ DQT SOF0 DHT DRI SOS RSTn EOI
DC-733s, DC-830i	SOI APP1(EXIF) APP5 APP6 DQT DRI DHT SOF0 SOS EOI
Sensor505-X, EX-Z150, CoolPix S710, Optio A40, Optio W60, DCZ5.9, L74wide, H50, T77, W170	SOI APP1(EXIF) DQT DHT SOF0 SOS EOI
DC-504, Sensor530s	SOI APP1(EXIF) DQT SOF0 DHT COM SOS EOI
FZ50	SOI APP1(EXIF) DQT SOF0 DHT DRI SOS EOI
DC-504, Sensor530s, Ixus 55, Ixus 70, FinePix J50, D200, D70, D70s, μ1050SW, GX100, RCP-7325XS, NV15	SOI APP1(EXIF) DQT SOF0 DHT SOS EOI
EXIF maker notes: Optio W60	SOI DHT DQT SOF0 SOS EOI
EXIF IFD1: DC-733s, DC-830i, Sensor505-X, EX-Z150, CoolPix S710, Optio A40, Optio W60, DCZ5.9, L74wide, H50, T77, W170; **EXIF maker notes**: EX-Z150, CoolPix S710, Optio A40; **post thumb**: L74wide	SOI DQT DHT SOF0 SOS EOI
EXIF IFD1: DC-504, Sensor530s	SOI DQT SOF0 DHT COM SOS EOI
EXIF IFD1: Ixus 55, Ixus 70, FinePix J50, M1063, D200, D70, D70s, μ1050SW, FZ50, GX100, RCP-7325XS, NV15; **EXIF maker notes**: D200, D70, D70s, GX100, RCP-7325XS; **APP2 (FPXR):** M1063; **post thumb**: μ1050SW, NV15	SOI DQT SOF0 DHT SOS EOI

of Casio EX-Z150 cameras depends on quality settings. Kodak cameras store an additional Flashpix thumbnail [3], which is split into several APP2(FPXR) segments. Interestingly, two Agfa models store a comment segment COM in the thumbnail, but not in the main image. All other thumbnails include only marker segments relevant to store compression parameters.

The huge variety of sequences of marker segments created by image processing software in Tab. IV reflects the large number of possible combinations of compression settings. Most image processing software employs the JFIF format including cjpeg (c.f. 'undetectable' forgeries in Sec. III). While digital cameras store JPEG using baseline DCT (SOF0), image processing software provides more sophisticated alternatives, like progressive DCT (SOF2) or lossless JPEG compression (SOF3). Often separate DQT and DHT segments are stored for each table. Photoshop adds a lot of manufacturer-specific APP segments. Few programs allow to store images in the JPEG/EXIF format and even then the observed sequences of segments are different to patterns of authentic camera images.

Summarising our observations, we identified five not strictly standardised aspects, forming new interesting artefacts for basic image authentication:

1) *not all segments are mandatory* and different combinations thereof can occur,
2) *segments can appear multiple times*, for example, quantisation tables can be either encapsulated in one single (digital camera images) or multiple separate DQT segments (often employed by image processing software),
3) *the sequence of segments* is generally not fixed, with exception of the required combination of SOI, APP0 for JFIF and SOI, APP1 for EXIF at the start of a file,
4) *JPEG thumbnails exist in different segments and employ their own complete sequence of marker segments*,
5) *arbitrary data can exist after EOI* of the main image without any influence on image processing software (e.g., we found unknown data structures looking like debug information (Rollei) and post thumbnails).

Analysing our new artefacts allows us to correctly distinguish between authentic and forged camera images (including the approach of Sec. III). None of the investigated software allows to recreate the introduced characteristics consistently and makes the creation of perfect forgeries much more difficult. Compiling a comprehensive database covering different camera models and image processing software will allow source identification at a coarse scale (groups of camera models and groups of image processing software).

VI. SEQUENCE OF EXIF DATA STRUCTURES

Besides the JPEG/EXIF file format, the EXIF standard defines the organisation of metadata including camera properties and employed image acquisition settings [3]. Current cameras implement EXIF versions 2.1 or 2.2 (the most recent version is 2.3), which differ in the number of standardised entries.

EXIF metadata is stored in the APP1(EXIF) segment and Tab. V depicts basic structures. Starting with the EXIF identifier code, a TIFF header specifying the byte order of the stored data and an offset to the 0th image file directory (0th IFD) follows. IFDs are the basic elements used to store different types of metadata. Each IFD starts with the number of entries, followed by a sequence of entries and ends with an offset to the next IFD. An additional data segment is located directly after each IFD to store values > 4 byte. The standard defines the 0th IFD, the EXIF IFD and the 1st IFD as mandatory. The 0th IFD stores basic image information and offsets to all other IFDs. The EXIF IFD contains standard metadata and, optionally, manufacturer-specific maker notes. The 1st IFD is reserved for the standard thumbnail. Additionally, a GPS IFD, an interoperability IFD or proprietary IFDs might exist.

The standard structure of an IFD entry (c.f. Fig. 1) consists of a tag (identifying the content), the employed data type, the count of the stored values, and, if all values fit into 4 bytes, the values itself. Otherwise an offset determines the location of all corresponding values in the IFD's data segment. Typically, maker notes in the EXIF IFD and manufacturer-specific IFDs use different proprietary data structures and specifications are only available by reverse engineering.

We decoded and analysed all stored standard EXIF metadata in our test set and found interesting differences between digital cameras, image processing software and metadata editors. Using the count of metadata entries similar to [2] resulted

TABLE IV

SEQUENCE OF JPEG MARKER SEGMENTS IN THE MAIN IMAGE AND ALL CORRESPONDING THUMBNAILS STORED WITH IMAGE PROCESSING SOFTWARE. IRFANVIEW, PAINT.NET AND PAINTSHOP PRO 8.10 PRESERVES THE ORIGINAL CAMERA-SPECIFIC EXIF IFD1 THUMBNAILS AFTER RESAVING AND WE EXCLUDED UNMODIFIED THUMBNAILS FOR CLARITY. IDENTIFYING CHARACTER STRINGS FOUND AT THE BEGINNING OF APP MARKER SEGMENTS ARE ABBREVIATED: PHOTOSHOP 3.0 (PS3), HTTP://NS.ADOBE.COM/XAP/1.0/ (XMP), ADOBE_CM (ACM) AND ICC_PROFILE (ICC). DUE TO SPACE LIMITATIONS, WE CUT VERY LONG SEQUENCES TO THE FIRST 13 MARKER SEGMENTS INDICATED BY DOTS (…).

software / **type of thumbnail**: software	sequence of JPEG marker segments
PS CS4	SOI APP0(JFIF) APP1(EXIF) APP13(PS3) APP1(XMP) APP14(Adobe) DQT SOF0 DRI DHT SOS RSTn EOI
PS CS4	SOI APP0(JFIF) APP1(EXIF) APP13(PS3) APP1(XMP) APP14(Adobe) DQT SOF2 DHT SOS SOS SOS SOS …
PS CS2, CS3, CS4, E9	SOI APP0(JFIF) APP1(EXIF) APP13(PS3) APP1(XMP) APP2(ICC) APP14(Adobe) DQT SOF0 DRI DHT SOS RSTn EOI
PS CS2, CS3, CS4, E9	SOI APP0(JFIF) APP1(EXIF) APP13(PS3) APP1(XMP) APP2(ICC) APP14(Adobe) DQT SOF2 DHT SOS SOS SOS …
Gimp **(with original compression settings)*, Paint.Net	SOI APP0(JFIF) APP1(EXIF) APP2(ICC) DQT DQT SOF0 DHT DHT DHT DHT SOS EOI
Gimp	SOI APP0(JFIF) APP1(EXIF) COM DQT DQT SOF0 DHT DHT DHT DHT DRI SOS EOI
Gimp	SOI APP0(JFIF) APP1(EXIF) COM DQT DQT SOF0 DHT DHT DHT DRI SOS …
Gimp	SOI APP0(JFIF) APP1(EXIF) COM DQT DQT SOF0 DHT DHT DHT DHT SOS EOI
Gimp	SOI APP0(JFIF) APP1(EXIF) COM DQT DQT SOF2 DHT DHT DRI SOS RSTn DHT …
Gimp	SOI APP0(JFIF) APP1(EXIF) COM DQT DQT SOF2 DHT DHT SOS DHT SOS DHT …
Gimp, IrfanView, Paint.Net	SOI APP0(JFIF) APP1(EXIF) DQT DQT SOF0 DHT DHT DHT DHT SOS EOI
Gimp, IrfanView	SOI APP0(JFIF) APP1(EXIF) DQT DQT SOF2 DHT DHT SOS DHT SOS DHT SOS …
IrfanView	SOI APP0(JFIF) APP1(EXIF) DQT SOF0 DHT DHT SOS EOI
IrfanView	SOI APP0(JFIF) APP1(EXIF) DQT SOF2 DHT SOS DHT SOS DHT SOS …
PS CS4	SOI APP0(JFIF) APP12(Ducky) APP1(XMP) APP14(Adobe) DQT SOF0 DHT SOS EOI
PS CS4	SOI APP0(JFIF) APP12(Ducky) APP1(XMP) APP2(ICC) APP14(Adobe) DQT SOF0 DHT SOS EOI
Gimp	SOI APP0(JFIF) COM DQT DQT SOF0 DHT DHT DHT DRI SOS EOI
Gimp	SOI APP0(JFIF) COM DQT DQT SOF0 DHT DHT DHT DRI SOS RSTn EOI
Gimp	SOI APP0(JFIF) COM DQT DQT SOF0 DHT DHT DHT SOS EOI
PaintShop Pro 8.10	SOI APP0(JFIF) COM SOF0 DQT DHT SOS EOI
PaintShop Pro 8.10, PaintShop Pro X3	SOI APP0(JFIF) COM SOF2 DQT DHT DHT DHT SOS DHT SOS DHT SOS …
cjpeg (libJPEG)*, Gimp, IrfanView / **EXIF IFD1**: Gimp	SOI APP0(JFIF) DQT DQT SOF0 DHT DHT DHT DHT SOS EOI
IrfanView	SOI APP0(JFIF) DQT DQT SOF2 DHT DHT SOS DHT SOS DHT SOS DHT …
IrfanView	SOI APP0(JFIF) DQT SOF0 DHT DHT SOS EOI
IrfanView	SOI APP0(JFIF) DQT SOF2 DHT SOS DHT SOS DHT SOS SOS …
PaintShop Pro X3	SOI APP0(JFIF) SOF0 DQT DHT SOS EOI
PaintShop Pro X3	SOI APP0(JFIF) SOF3 DHT DHT SOS EOI
PS CS5	SOI APP1(EXIF) APP13(PS3) APP1(XMP) APP2(ICC) APP14(Adobe) DQT SOF0 DRI DHT SOS RSTn EOI
PS CS5	SOI APP1(EXIF) APP13(PS3) APP1(XMP) APP2(ICC) APP14(Adobe) DQT SOF2 DHT SOS SOS SOS SOS …
PaintShop Pro 8.10, PaintShop Pro X3	SOI APP1(EXIF) SOF0 DQT DHT SOS EOI
APP13(PS3) & **EXIF IFD1**: PS CS2, CS3, CS4, E9	SOI APP0(JFIF) APP13(ACM) APP14(Adobe) DQT SOF0 DRI DHT SOS RSTn EOI
APP13(PS3) & **EXIF IFD1**: PS CS5	SOI APP13(ACM) APP14(Adobe) DQT SOF0 DRI DHT SOS RSTn EOI
EXIF IFD1: PaintShop Pro X3	SOI SOF0 DQT DHT SOS EOI

TABLE V

STANDARDISED STRUCTURE OF AN APP1(EXIF) MARKER SEGMENT. THE TABLE DEPICTS THE EXPECTED SEQUENCE OF IFDS. NOTE, GPS AND MANUFACTURER-SPECIFIC (MAN.) IFDS ARE OPTIONAL.

entry group	content
EXIF identifier	character string 'Exif'
TIFF header	byte order (little- or big-endian), number 42, offset to 0th IFD
0th IFD	*entries of 0th IFD* defining general image properties, like image dimension, and offsets to other IFDs: EXIF IFD, GPS IFD (optional), manufacturer non-standardised IFDs (optional) and 1st IFD storing the standard thumbnail
0th IFD data	storage for data of 0th IFD greater than 32 bit
EXIF IFD	*entries of EXIF IFD* including version, camera settings and manufacturer-specific maker notes
EXIF IFD data	storage for data of EXIF IFD greater than 32 bit
GPS IFD	*entries specific to GPS IFD* including GPS coordinates
GPS IFD data	storage for data of GPS IFD greater than 32 bit
man. IFD	*entries specific to manufacturer non-standardised IFDs*
man. IFD data	storage for data of man. IFD greater than 32 bit
1st IFD	*entries specific to 1st IFD* describing the thumbnail properties
1st IFD data	storage for data of 1st IFD greater than 32 bit
thumbnail	thumbnail data with its own sequence of JPEG marker segments

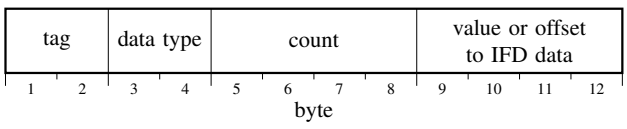

Fig. 1. Standard structure of an IFD entry.

in 13 different classes of camera models. However, we could distinguish between 23 classes (of 26 models) by considering the sequence of entries with respect to tag and data type. We observed constant sequences of entries, including tag, data type and often offsets[8], for images of each investigated camera model except DC-830i and D70. Differences in settings of sharpness (DC-830i) and white balance (D70) resulted in altogether two constant sequences for both models and complicated the creation of comprehensive reference material.

The most important observation is the change and reorganisation of EXIF data by image processing software and metadata editors. While the investigated image processing software rarely provides functionality to edit metadata, if at all, the number of preserved EXIF entries changes with the selected (compression) options. For example, Photoshop adds a software tag and removes proprietary maker notes; IrfanView always uses its preferred data type for image dimensions. Based on the sequence of EXIF entries, low false negative rates to detect post-processed images are possible and occur only when changes of image processing software have no effect on camera metadata (e.g., model GX100 uses the same data type for image dimensions as IrfanView).

[8]Constant offsets point to a location before the first non-standard thumbnail.

TABLE VI
SELECTION OF EXIF ENTRIES OF MODEL CANON IXUS 70.

tag	tag interpreted	data type	count	offset	value
entries of 0th IFD – authentic camera image					
0x010F	Make	2	6	122	Canon
0x0110	Model	2	22	128	Canon DIGITAL IXUS 70
0x0112	Orientation	3	1	-	1 (top + left-hand)
0x011A	XResolution	5	1	160	180/1
0x011B	YResolution	5	1	168	180/1
0x0128	ResolutionUnit	3	1	-	2 (inches)
0x0132	DateTime	2	20	176	2009:01:07 20:14:51
0x0213	YCbCrPositioning	3	1	-	1 (centred)
0x8769	EXIF IFD pointer	4	1	186	-
entries of 0th IFD – tag DateTime edited with exiftool (all differences are bold)					
0x010F	Make	2	6	122	Canon
0x0110	Model	2	22	128	Canon DIGITAL IXUS 70
0x0112	Orientation	3	1	-	1 (top + left-hand)
0x011A	XResolution	5	1	**150**	180/1
0x011B	YResolution	5	1	**158**	180/1
0x0128	ResolutionUnit	3	1	-	2 (inches)
0x0132	DateTime	2	20	**166**	**2012:07:01 23:59:59**
0x0213	YCbCrPositioning	3	1	-	1 (centred)
0x8769	EXIF IFD pointer	4	1	186	-

More interestingly, we are also able to detect manipulations of EXIF entries like DateTime, at least when ExifTool is used. Table VI illustrates differences between authentic and edited images. Unexpectedly, ExifTool does not only update a selected tag, it always rewrites the whole `APP1(EXIF)` segment, reorders the tags with increasing id in each IFD and removes redundancies introduced by the camera. For example, in Tab. VI offset values changed in the 0th and all other IFDs (space limitations permit a complete list). Testing the consistency of the known constant sequence of entries makes the detection of EXIF manipulations possible and much more reliable than commonly believed. Metadata editors leaving forensic exploitable artefacts can be easily identified by comparing an edited authentic image byte-wise to the original version. Yet also camera models using the same structure like a metadata editor might exist. We could detect altered versions of Sensor505-X and L74wide images only in about half of the cases in which ExifTool adds padding data to the end of the thumbnail in the 1st IFD.

Summarising our investigations, we identified the following four new artefacts for the forensic analysis of EXIF metadata:

1) the *byte order* is different between camera models (and the default setting employed by ExifTool),
2) *sequences of IFDs and corresponding entries* (including tag, data type and often also offsets) appear constant in images acquired with the same model and differ between different sources,
3) some manufacturers use *different data types* for the same entry,
4) *raw values of rational types* differ between different models, but still result in the same interpreted values (e.g., $200/10$ is equal to the rational value $20/1$).

VII. CONCLUDING REMARKS

Within this paper, we investigated the structure of image file formats on the example of JPEGs. We document differences in sequences of JPEG and EXIF data structures and introduce new characteristics for basic image file authentication. Considering the investigated software, it seems currently not possible to create unsuspicious forgeries of JPEG images without advanced programming skills. The discussed characteristics invalidate the common assumption about the unreliability of metadata (at least for ExifTool and the considered image processing software). Forensic investigators should still interpret all analysis results with caution to avoid false accusations and, we have to note, that Jhead allows to alter EXIF tag DateTime without leaving any traces. Our approach makes the creation of convincing forgeries considerably more difficult and provides a valuable method for the toolbox of forensic investigators.

Sequences of data structures might be relevant to all kinds of multimedia data formats and a first investigation of MPEG-4 videos acquired with mobile phones (Motorolla Milestone, Palm Pre) indicate that similar analysis principles are applicable to the authentication of video formats as well.

REFERENCES

[1] Husrev T. Sencar and Nasir Memon, "Overview of state-of-the-art in digital image forensics," in *Statistical Science and Interdisciplinary Research*, Indian Statistical Institute Platinum Jubilee Monograph. World Scientific Press, 2008.

[2] Eric Kee, Micah K. Johnson, and Hany Farid, "Digital image authentication from JPEG headers," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 1066 – 1075, 2011.

[3] Japan Electronics and Information Technology Industries Association, *JEITA CP-3451, Exchangeable image file format for digital still cameras: Exif Version 2.2*, April 2002.

[4] "Court decision district court of Munich, Germany, in German, Landgericht München I, 21.5.2008, Az. 21 O 10753/07," May 2008.

[5] Rainer Böhme, Felix Freiling, Thomas Gloe, and Matthias Kirchner, "Multimedia forensics is not computer forensics," in *Proceedings of IWCF 2009*, Zeno J. M. H. Geradts, Katrin Y. Franke, and Cor J. Veenman, Eds., 2009, vol. 5718, pp. 90–103.

[6] ISO/IEC IS 10918-1, ITU-T Recommendation T.81, *Information technology — Digital compression and coding of continuous-tone still images — Requirements and guidelines*, 1992.

[7] Thomas Gloe, "Demystifying histograms of multi-quantised DCT coefficients," in *2011 IEEE International Conference on Multimedia and EXPO (ICME 2011)*, 2011.

[8] Hany Farid, "Digital image ballistics from JPEG quantization: A followup study," Tech. Rep. TR2008-638, Department of Computer Science, Dartmouth College, Hanover, NH, USA, 2008.

[9] Jesse D. Kornblum, "Using JPEG quantization tables to identify imagery processed by software," *Digital Investigation*, vol. 5, pp. S21–S25, 2008.

[10] Steven J. Murdoch and Maximillian Dornseif, "Hidden data in Internet published documents, far more than you ever wanted to tell," in *21. Chaos Communication Congress 2004*, 2004.

[11] Eric Kee and Hany Farid, "Digital image authentication from thumbnails," in *Proceedings of SPIE: Media Forensics and Security II*, Nasir D. Memon, Jana Dittmann, Adnan M. Alattar, and Edward J. Delp III, Eds., 2010, vol. 7541, p. 75410E.

[12] Husrev T. Sencar and Nasir D. Memon, "Identification and recovery of JPEG files with missing fragments," *Digital Investigation*, vol. 6, pp. S88–S98, 2009.

[13] Andrew B. Lewis, "Reconstructing compressed photo and video data," Tech. Rep. UCAM-CL-TR-813, University of Cambridge, Computer Laboratory, Cambridge, United Kingdom, 2012.

[14] Thomas Gloe and Rainer Böhme, "The 'Dresden Image Database' for benchmarking digital image forensics," in *Proceedings of the 25th Symposium On Applied Computing (ACM SAC 2010)*, 2010, vol. 2, pp. 1585–1591.

[15] Eric Hamilton, *JPEG File Interchange Format Version 1.02*, C-Cube Microsystems, 1778 McCarthy Blvd., Milpitas, CA 95035, 1992.

[16] Matthias Kalms, Thomas Gloe, and Christopher Laing, "File forensics for raw camera image formats," in *Proceedings of the 6th Conference on Software, Knowledge, Information Management and Applications (SKIMA 2012). Chengdu University, China, 9-11 September, 2012*. 2012, in press, Inder Science Publisher.