MetaFor: Metadata Signatures for Automated Remote File Identification in Forensic Investigations

M.P. Roberts and J. Haggerty

School of Computing, Science & Engineering, University of Salford, Greater Manchester, M5 4WT
matt.concordia@gmail.com; J.Haggerty@salford.ac.uk

Abstract

The increased use of the Internet to store data ensures that it provides a valuable resource for a forensics examiner during an investigation. Of particular interest is evidence related to the dissemination of indecent images of children that are spread via social networking sites and Web fora. This paper posits a novel approach, MetaFor, which using a Web crawler searches for metadata signatures for automated identification of files residing on remote Web servers. In this way, it may identify potential repositories of illegal images or sources of evidence related to traditional crimes, such as utilising geo-location metadata to identify digital pictures taken during a crime in progress. This approach differs from other forensic signature schemes in that it utilises JPEG header metadata rather than image or file data as the basis of a signature. In this way, MetaFor can be extended to search for unknown files that may be relevant to an investigation. In order to demonstrate the applicability of the approach, this paper applies the approach to a case study of two Web servers and presents the results.

Keywords

Digital forensics, signature analysis, image files

1. Introduction

Today's reliance on technology has brought many economic and cultural benefits, but it also harbours many technical and social challenges. One major benefit of this wide scale adoption of technology is the speed and volume of data and information that may be shared amongst hosts. However, this has given rise to concerns over paedophile activity and the spread of illegal digital pictures, in particular indecent images of children, via social networking sites and Web fora (see for example, BBC, 2012).

Owing to the number and prolific nature of the media files themselves, forensic examiners have a hard, and at times disturbing, duty to find and identify indecent images of children disseminated online. Limited time and budgets make thorough manual searching unrealistic or at best, a time-consuming and costly process due to the amount of data that must be searched. Moreover, most image identification techniques involve computationally expensive algorithms in order to assess image data resident in a file in an attempt to evade anti-forensics techniques employed by a suspect. This paper presents a novel application, MetaFor, by which a forensics examiner may run an automated Web crawler search of Web servers for known suspicious or illegal images by utilising signatures formed from JPEG metadata. In this way, it may automatically identify potential repositories of illegal images or sources of evidence related to traditional crimes, such as utilising geo-location metadata to identify digital pictures taken during a crime in progress. As will be discussed in section 3, this approach has the added functionality of extending the search to unknown images residing online.

This paper is organised as follows. Section 2 discusses related work. Section 3 presents an overview of the system and describes the signature scheme. Section 4 presents the results of applying the approach to a case study. Finally, we make our conclusions in section 5 and discuss further work.

2. Related work

Commonly used computer forensic tools, such as Forensic Toolkit (FTK) (Access Data, 2013) and EnCase (Guidance Software, 2013) are used for storage media analysis of a variety of files and data types in fully integrated environments. For example, FTK can perform tasks such as file extraction, make a forensic image of data on storage media, recover deleted files, determine data types and text extraction. EnCase is widely used within law enforcement and like FTK provides a powerful interface to the hard drive or data source under inspection, for example, by providing a file manager that shows extant and deleted files. Whilst these applications provide a robust forensic analysis, they are not designed to perform automated retrieval and analysis of potential evidence residing on remote Web servers.

Due to the volume of potential evidence that may require analysis, there is a requirement for automated approaches for file identification. The tools above enable searches of file hashes. However, due to the vulnerability of this approach to anti-forensics techniques, other signature schemes have been proposed. For example, FORSIGS (Haggerty & Taylor, 2007) searches for sixteen random bytes located in a single memory location which forms the file signature. This has been extended to online searches in FORWEB (Haggerty et al, 2008) which employs a Web spider to crawl through a web page to collect links to image files. These images are then downloaded and assessed using the FORSIGS algorithm. Alternatively, Mohamad & Deris (2009) use a single-byte marker and a twenty-point reference for signature detection. These signature approaches can also be extended to search within slack space on the storage media (Holleboom & Garcia, 2011).

Content-based image retrieval (CBIR) has been the subject of research for some time. CBIR is concerned with identifying an image file based on locating objects that are held within the image. Approaches, such as Sportiello & Zanero (2011), use a support vector machine (SVM) which is trained by human-directed input to recognise objects by grouping them based on attributes within the image file. These algorithms have had some success when applied to "stock" photos that are clean and usually uncluttered. Other schemes focus on identifying individual sources of digital images, such as cameras. For example, Chang-Tsun Li and Li (2012) propose a couple-decoupled photo response non-uniformity approach to improve the accuracy of device identification and for image content integrity verification. Kang et al (2012) propose a sensor pattern noise (SPN) approach for camera identification.

All of the above techniques analyse the image data or the whole of the file for their analysis. Alternative approaches propose the use of header information and in particular the EXIF data to identify files. For example, Kee et al (2011) posit that it would be possible to tell if the data held within the image header had been modified based on the fingerprint left by the make and model of the camera. This research created a 284-value signature taken from the EXIF data, thumbnail image, quantization values, and Huffman codes. Alternatively, Fan et al (2011) propose a scheme based on statistical analysis to detect image manipulation. However, these approaches have in common that they are not designed for forensic investigations, and in particular, the identification of images online.

3. MetaFor

This section provides an overview of the MetaFor approach. First, it describes the MetaFor architecture. In particular, it identifies the main features of the application. Second, it posits the signature scheme utilising metadata resident in JPEG image headers. Moreover, it describes how the scheme can be extended to search for unknown image files that may reside online that could provide further evidence to the forensics examiner. In this way, it may be used to automatically search for unknown online repositories of illegal images or digital pictures to support ongoing investigations.

A. System overview

The MetaFor architecture can be split into three main sections. First is the user interface to be used by the forensic examiner. Second is the signature management code that extracts, stores and manages the signatures to match against. Last is the search component itself. This part is responsible for searching a Web server for images and extracting any EXIF data for comparison with the signatures held locally. Figure 1 provides an overview of the system architecture.

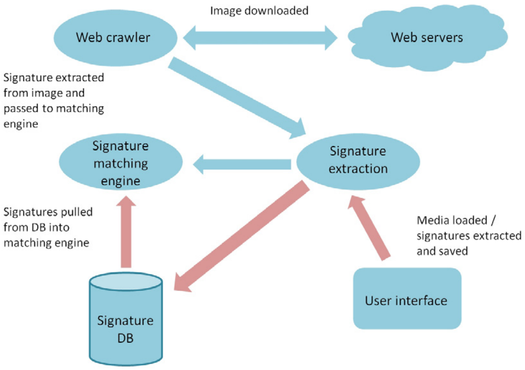


Image downloaded
Web crawler
Web servers
Signature extracted from image and passed to matching engine
Signature matching engine
Signature extraction
Signatures pulled from DB into matching engine
Media loaded / signatures extracted and saved
Signature DB
User interface

Figure 1: High level view of the system architecture

The user interface design is centred on a main page from which the configured search is initiated. Forms are used to add signatures, manage signature groups, configure the starting URLs and tune the application. The sub forms are all accessible via a menu system in the main window. The main window shows four progress bars each with two corresponding counters. These are present so that any component that is a bottleneck or that is used to a greater extent than others could be tuned by allocating more threads of this type. That is to say, if the signature extraction and comparison component queue is collecting work quicker than it can process then the ration of EXIF comparer threads can be increased to allocate more time to this process.

To enable an application to match a file against a signature, a permanent storage method needs to be established. In addition, the number of images of an illegal nature is always on the rise so there needs to be a mechanism to add more signatures and manage them appropriately. All of the permanent storage functionality is carried out “offline” in the sense that it can be separated, in temporal terms, from the main crawling and searching functionality. The database design is implemented in MySQL. The structure mirrors the abstraction of a signature being made up from many signature segments, where a segment is representative of an EXIF tag and its value. Signatures belong to a parent Signature Group which allows grouping of signatures for reporting or administrative purposes.

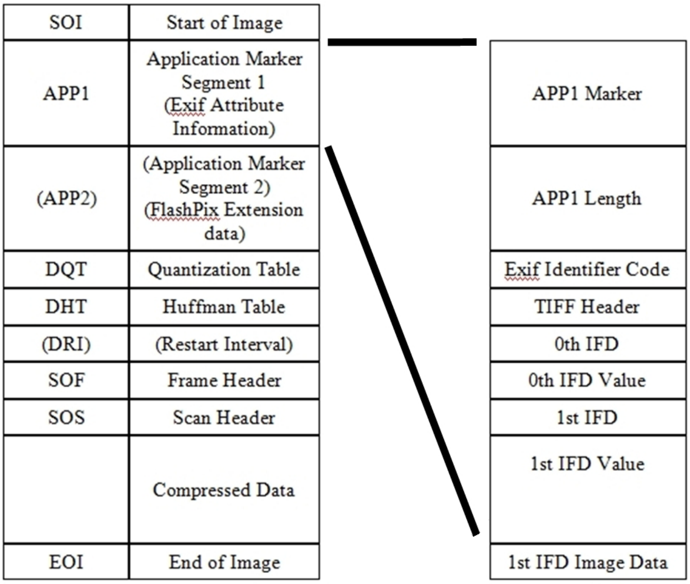
The search functionality is separated into four main components. The HTML Downloader component is responsible for sending an HTTP request to the Web server and receiving the response. The HTML Parser searches through the HTML and pulls out paths to images and links to other Web pages if the depth is set accordingly. The Image Downloader downloads part or the entire image found by the HTML Parser. Finally, the Signature Processor is the main work horse and is responsible for extracting the EXIF data from the downloaded image, converting it to a native format and then iterating through the signatures to see if any match.

B. Metadata signature scheme

Although JPEG is the commonly accepted term for the image format itself, the ITU-T Recommendation T.81 standard only covers the definition of the codec. This defines how to compress a stream of bytes (which in this case is data that represents an image) and how to decompress them back again, in order to view the image. There are four modes that the standard defines, namely: sequential Discrete Cosine Transformation (DCT) based, progressive DCT based, lossless, and hierarchical (ITU, 1992). The predominant method used over the Internet is the sequential DCT based technique.

The file format that is used to structure the JPEG compressed data is commonly the JPEG File Interchange Format (JFIF). This format defines markers within the file to designate specific file sections and their lengths. The whole image is started by a start of image (SOI) marker designated by the hexadecimal value “FFD8” and ended by end of image (EOI) “FFD9”. Other sections that are relevant are the compression data tables are held in sections DHT which holds one or more Huffman Tables. This is a process whereby commonly occurring byte patterns are replaced by a shorter “key”. These key/value pairs are stored in the header in a Huffman Table for use when decoding. The DQT section which hold the quantization tables. Most pertinent are the application specific sections (APP*n*) designated by the hexadecimal marker "FFE*n*" where *n* is the number of the group. For example the EXIF application specific group is always defined as APP1 (FFE1 in hexadecimal). This signifies the start of the metadata which is used as the basis of a signature in our scheme.

The Exchangeable Image File Format (EXIF) was included in the Japan Electronic Industry Development Association Standard for Camera File System (DCF) (JEIDA, 1998). This standard describes how metadata is stored within an image or audio file header which can contain compressed or uncompressed data. The EXIF data is contained within the Application Marker Segment (APP1) section of the JPEG file. Following the APP1 marker and length the section is defined as EXIF data with the EXIF identifier code. Following this, the EXIF data is defined in a Tagged Image File Format (TIFF) structure. This structure is made up of multiple Image File Directories that contain specific data pertaining to a subject. For instance, IFD0 is the first Image File Directory (IFD) that holds the interoperability data. This is the data that describes (amongst others) the image data structure, the image data characteristics, picture-taking conditions and details such as the date and time it was taken. IFD1 contains data relevant to the thumbnail version of the image, including the image data itself. A further IFD will go on to describe Global Positioning Satellite (GPS) data that depicts where the picture was taken if the writing device is able to capture such data. Figure 1 depicts the structure as per the 1998 JEIDA standard.



SOI	Start of Image
APP1	Application Marker Segment 1 (Exif Attribute Information)
(APP2)	(Application Marker Segment 2) (FlashPix Extension data)
DQT	Quantization Table
DHT	Huffman Table
(DRI)	(Restart Interval)
SOF	Frame Header
SOS	Scan Header
	Compressed Data
EOI	End of Image

APP1 Marker
APP1 Length
Exif Identifier Code
TIFF Header
0th IFD
0th IFD Value
1st IFD
1st IFD Value
1st IFD Image Data

Figure 1: JEIDA EXIF data structure

Data inside each IFD are depicted by structures called tags. Tags translate to an EXIF field, ultimately describing a field/value pair. These tags are identified by a unique 2-byte number to identify the field. The tag structure is made up from this identifier, a 2-byte type field that describes what data type the value pertaining to this field is, a 4-byte count that depicts how many values are included for this tag and a 4-byte offset value from the start of the TIFF header that points to the actual value for this tag. As an example the tag values describing the manufacturer of a camera held in IFD0 could look as follows and could form a metadata signature:

Bytes 0-1 (ID): 0x010f
Bytes 2-3 (Type): ASCII
Bytes 4-7 (Count): 1
Bytes 8-11 (Value Offset): 1
Byte 12-16 (Value): Canon

A metadata signature is added to a group in the Add Signature form in MetaFor in two ways. An image (or a directory full of images) can be selected and the signature(s) can be extracted from them and displayed on the form and saved to the database. Alternatively a signature can be compiled manually using logical expressions to allow a custom signature to describe a geographical area or a temporal range. For example, the following segments could be defined:

GPSLatitude : Less Than 51.861793
GPSLatitude: Greater Than 51.861731
GPSLongitude: Less Than -2.243199
GPSLongitude: More Than -2.243349
DateTimeOriginal: More Than 01 May 1973
DateTimeOriginal: Less Than 01 August 1979

This would return any pictures taken at the British serial killer Fred West's house during the height of his crimes (had digital photography been invented). This flexible scheme has the advantage over file signature approaches whereby it will identify other images outside the known images evidence base, i.e. unknown images. Therefore, it may be used to search for images that may record a crime that are available online but not known to the forensics examiner.

An issue that arises is the trade off between performance and RAM utilisation. That is to say, comparing a file with signatures that are preloaded into RAM will inevitably be faster than retrieving each signature from a database as it is needed. However, there will come a point for a given system when the number of signature objects loaded into RAM will become so big that it will be necessary to allocate some to virtual memory at the expense of performance. It is noted that reducing memory capacity is an important consideration for the development of commercially viable forensic tools (Collange et al, 2009). Thus, signatures are between 100 to 200 bytes and holding 100,000 signatures would require approximately 20 megabytes of spare RAM.

4. Case study and results

To demonstrate the applicability of the proposed approach, this section presents a case study of using MetaFor and discusses the results.

To assess the applicability of the approach, tests are carried out on two separate Web servers and utilise two groups of sixteen signatures. Firstly, the tests are applied to a 3.06 Ghz Intel Ei System development machine running an Apache Web Server. Running the tests locally eradicates any latency or network errors that may impose themselves on the results to give a more reliable assessment of the software itself with relatively constant response times. The second set of tests is carried out with the same images, but on a remote Website. These results are more realistic in terms of timings and network errors but response times could vary depending on network and Website traffic.

The Websites have the following features:

- A front page from which the search is initiated.
- Three picture galleries containing between six and ten identical images and text.
- Images that contain different file types (.png, .gif and .jpeg).
- Images that contain EXIF data and images that do not (including thumbnails).
- Two images that are identical.
- Top level menu with links to the three picture galleries.
- Secondary menu with the same links providing double circular links to all galleries.
- A broken link to an image that does not exist.
- A broken link to a web page that does not exist.
- Images that ranged from 30 kilobytes to nearly six megabytes.

The initial starting point of the search is then configured to point to each front page in turn and the crawl is initiated. The application is written with extra functionality that can be removed after testing. This functionality passes a job tracking object between each of the queues and timestamps the object as it finishes its work. Having a start and end time with each unit rather than just recording the end times prevents any confusion where threads are asleep waiting for another thread to relinquish control. This can report large processing times for a component that may have done its work very quickly. These objects are held in a chronological list and can be displayed via a reporting form. Furthermore, by taking the start time of the first object and the signature processing time of the last, we can deduce the overall time of the whole search.

The tests are run with a depth of two, meaning it will scour the front page and the pages that link from it, i.e. it will reach the gallery pages but will not follow links from thereon in. Table 1 presents the results of the signature matching against known signatures on both Web servers.

File Name	Signature present	Signature Matched
logo.png	N	-
feed.png	N	-
046.JPG	Y	10
2012-03-17%2013.48.55.jpg	Y	8
medium_2012-02-14%2008.28.08.jpg	N	-
2012-03-16%2014.12.53_0.jpg	Y	8
medium_2012-03-16%2014.12.53.jpg	N	-
IMG_0125.jpg	Y	6
medium_8.jpg	N	-
medium_2012-03-16%2011.10.03.jpg	Y	1
medium_135.JPG	N	-
IMG_0950.JPG	Y	23
medium_IMG_0953	Y	1
medium_IMG_0951	N	-
medium_IMG_0955	Y	1
edium_OtherSide.JPG	Y	1
whirlpool.jpg	N	-
medium_131.JPG	Y	1
medium_115.JPG	N	-
medium_094.JPG	N	-
medium_058.JPG	N	-
medium_046.JPG	N	-
medium 002.JPG	Y	1

Table 1: Results of signature matching

The results in table 1 suggest that images are matched against known signatures when they are present. Data to be matched to signatures is present in 11 of the files tested and these are detected by the scheme. In addition, no false positives were observed. It should be noted that this is a small data set. Future work will comprise much larger data sets of files and signatures. This is acceptable for a proof of concept, however, scaling the system for realistic use would mean thousands (if not hundreds of thousands) of signatures for comparison.

An advantage of the MetaFor approach is that the data required for the signature is located at the start of the file so the whole file does not have to be downloaded. Therefore, an image can be found, data downloaded and checked against a signature in an average of 149 milliseconds, or around seven per second on the local Web server. There is a slight increase for the remote Web server with an average of 261 milliseconds (less than four per second). Owing to the fact that the images may not be cached at the time of execution, these timings should probably be regarded as a best case scenario, and not typical.

There are a few interesting timings to note. First, the image processing time seems particularly low on the local Web server. Given that the size of the piece of the image file is 65000 bytes and the HTML/XML pages downloaded were between

1407 bytes and 8192 bytes it was expected that the processing time would be proportional. However the image downloads average around 10 milliseconds and the HTML around 300 milliseconds. Initially it was presumed that a program library was downloading asynchronously and that would explain the timings. However, further investigation into the code revealed that the start and end times must be being set before the download start and after it finishes. This behaviour is not seen on the remote server results and it is assumed that this is due to caching on the local Web server.

Second, the signature timings are unrepresentative of realistic execution times. When an image is found that has EXIF data the time reported for extracting the data and checking against the signatures is around 20 milliseconds. This test case only checks the extracted data against two groups of 16 signatures. A realistic number of signatures would be significantly higher, potentially in the order of several hundreds of thousands and this number could conceivably increase daily.

5. Conclusions and further work

Our continued reliance on the Internet as an information repository brings many benefits. However, it also gives rise to many concerns, such as the dissemination of illegal material, and in particular, indecent images of children, through social network sites and Web fora. This paper presents a novel approach, MetaFor, by which forensics examiners may run automated searches of remote Web servers for known illegal images by signatures formed from JPEG metadata. In this way, they are able to automatically detect repositories of illegal images or further sources of evidence. This approach differs from other signature schemes in that it uses file header metadata rather than image data to create the signature. In this way, MetaFor can be extended to search for unknown files. These files may support non-digital investigations, for example, searching for geographical and temporal signatures formed from the file metadata may identify digital pictures related to a traditional crime. Initial results presented in this paper, albeit on a small data set, demonstrate the applicability of the approach.

Future work aims to extend the tests to many more signatures to evaluate system performance. In this way, the potential issues of false positives and signature processing can be fully evaluated. Furthermore, as indicated in the results, performance issues in terms of Web server processing, in particular the discrepancy between image downloads and HTML processing, may also be assessed.

6. References

Access Data (2013). http://www.accessdata.com. (Accessed 10 January 2013).

BBC (2012), http://www.bbc.co.uk/news/uk-18181848. (Accessed 10 January, 2013).

Chang-Tsun Li & Yue Li (2012), “Color-Decoupled Photo Response Non-Uniformity for Digital Image Forensics”, *IEEE Transactions on Circuits and Systems for Video Technology*, Volume 22 Number 2, pp. 260-271.

Collange, S., Dandass, Y. S., Daumas, M. & Defour, D. (2009), "Using graphics processors for parallelizing hash-based data carving", *Proceedings of the 42nd Hawaii International Conference on System Sciences (HICSS'09),* Hawaii, USA, 2009, pp. 1-10.

Fan, J., Kot, A. C., Cao, H. & Sattar, F. (2011), "Modeling the EXIF-Image correlation for image manipulation detection", *Proceedings of the 18th IEEE International Conference on Image Processing (ICIP*), Brussels, Belgium, 2011*,* pp. 1945-1948.

Guidance Software (2013). http://www.guidancesoftware.com. (Accessed 10 January 2013).

Haggerty, J. & Taylor, M. (2007). "FORSIGS: forensic signature analysis of the hard drive for multimedia file fingerprints", in IFIP International Federation for Information Processing, Volume 232, *New Approaches for Security, Privacy and Trust in Complex Environments*, Venter, H., Eloff, M., Labuschagne, L., Eloff, J. & von Solms, R. (eds.), (Boston, Springer), pp. 1-12.

Haggerty, J., Llewellyn-Jones, D. & Taylor, M. (2008), "FORWEB: File Fingerprinting for Automated Network Forensics Investigations", *Proceedings of e-Forensics 2008,* Adelaide, Australia, 2008.

Holleboom, T. & Garcia, J. (2010), "Fragment retention characteristics in slack space - Analysis and measurements", *Proceedings of the 2nd International Workshop on Security and Communication Networks (IWSCN),* Karlstad, Sweden, 2010, pp. 1-6.

International Telecommunication Union (ITU) (1992), "Information Technology - Digital Compression and Coding of Continuous-Tone Still Images - Requirements and Guidelines. http://www.w3.org/Graphics/JPEG/itu-t81.pdf. (Accessed 10 January, 2013).

JEIDA (1998), "Design Rule for Camera File System, version 1.0, JEIDA-49-2-1998", http://www.exif.org/dcf.PDF. (Accessed 10 January 2013).

Kang, X., Li, Y., Qu, Z. & Huang, J. (2012), "Enhancing Source Camera Identification Performance With a Camera Reference Phase Sensor Pattern Noise", *IEEE Transactions on Information Forensics and Security*, Volume 7 Number 2, pp. 393 - 402.

Kee, E., Johnson, M. K. & Farid, H. (2011), "Digital image authentication from JPEG headers", *IEEE Transactions on Information Forensics and Security,* Volume 6 Number 3, pp. 1066-1075.

Mohamad, K. M. & Deris, M. M. (2009), "Single-byte-marker for detecting JPEG JFIF header using FORIMAGE-JPEG", Proceedings of the *Fifth International Joint Conference on INC, IMS and IDC (NCM'09),* Seoul, South Korea, 2009, pp. 1693-1698.

Sportiello, L. & Zanero, S. (2011), "File Block Classification by Support Vector Machine", *Proceedings of the Sixth International Conference on* In *Availability, Reliability and Security (ARES),* Vienna, Austria, 2011, pp. 307-312.