Analysis of redundant traces for consistency

With examples from electronic messaging and graphical images

Fred Cohen
California Sciences Institute and
Fred Cohen & Associates
Livermore, CA 94550
fc at all dot net

***Abstract*—*This paper is about the detection of inconsistencies and consistencies in redundant traces to detect forgeries, demonstrate forensic soundness, and lend weight to assertions made by forensic examiners performing analysis.*

Keywords-digital forensics, trace consistency, analysis

I. Introduction, overview, and background

The requirements for the use of scientific evidence through expert opinion in the United States and throughout the world are based on principles and specific rulings that dictate, in essence, that the evidence be (1) beyond the normal knowledge of non-experts, (2) based on a scientific methodology that is testable, (3) characterized in specific terms with regard to reliability and rates of error, (4) that the tools used be properly tested and calibrated, and (5) that the scientific methodology is properly applied by the expert as demonstrated by the information provided by the expert. [1][2][3][4] This paper offers an approach to meeting these criteria for digital forensic evidence (DFE).

Digital forensic evidence comes in the form of "traces". A trace is a sequence of bits that are put forth by a party and asserted as being the result of some process undertaken on some digital system. An "event" is a statement, document, or any other item of import to the case.

Without redundancy, a trace is little more than a "bag of bits". Redundancy is inherent in human and current computer language and fundamental to the notion of syntax. Without redundancy, reliability cannot be assured, because alteration of a single bit anywhere completely changes the semantics of the bag of bits. Digital systems hardware, instruction sets, memory pointers, software, languages, and protocols all have redundancy. Computers often store traces of activities in the form of access, write, and creation dates, logs produce audit trails, files often have date and time indicators within records they store, and sequences of writes leave traces in the structures of links between allocated areas within the file system. Network use often produces traces including time stamps and URL records on intermediate and end computers, records of address lookups, flow logs, performance impact indicators, and interference patterns. [5][6][7] In short, there are many traces in computer systems and networks of activities that take place. The challenge to the DFE examiner is to exploit this redundancy to find revealing traces and confirmations of the consistency of those traces.

The science of digital forensic evidence examination is based on a set of theories that we call "information physics", a mathematical model, and a process by which the examiner makes hypotheses based on events and traces, performs experiments or applies mathematical analysis to confirm or refute the hypotheses, and reports results.[7] To do these activities, the examiner uses tools that meet legal standards. Examination is broken down into analysis, interpretation, attribution, and reconstruction. This paper focuses on analysis, a process by which purely mathematical methods are used to identify consistency and inconsistency in traces and events. It acts only on traces and events that are formulated into logical statements that can be evaluated in conjunction with the traces.

Examination normally starts with assumptions based on events, and as examination proceeds, hypotheses are confirmed or refuted based on traces. It exploits the redundancy inherent in the bag of bits to identify useful traces based on structure that permits further analysis. Consistency tends to lend weight to the accuracy of the asserted events, while inconsistency tends to refute the asserted events. This is analogous to testing digital systems, and testing research may therefore be revealing.

A model of digital forensic evidence processing has been proposed [8] that identifies a schedule S over a set of {L, R, H, E, T, C, D, P, R}. A legal statute, or law (L) that may or may not be violated based on a logic expression L:{l1, ..., ln}, R:{r1, ..., rm}, LxR→[F|T], where the truth of LxR implies that a charge of violation is warranted based on the defined legal criteria. (LxR⇒V). The hypothesized claims (H=(H1, ..., Hn}) are supported by hypothesized event claims (E: {E1, ..., Eo}]), each of which consists of a set of indicated events from the set of all events [∀e, e∈E*] within and outside of the digital system [∀Ex∈E, Ex:(ex1∈E*, ..., exp∈E*)], and that, when put together, purport to constitute a demonstration that the relevant legal requirement is met. There is the set of possible digital traces from existing evidence T:(t1, ...,tq), each consisting of sets of values of sets of bits from the overall collection of bits available as digital forensic evidence. There is an internal consistency relation C:TxT→[-1...1] between traces that identifies the extent to which different traces are entirely consistent (1), unrelated (0), or entirely inconsistent (-1) with each other, and a demonstration consistency relation D:TxE*→[-1..1], that relates T and E and which may tend to confirm or refute hypothesized event sets as being completely inconsistent with traces (-1), completely consistent with traces (1), or not revealing (0). There is a finite set of forensic procedures P:{p1, ..., pn}, ∀p∈P, p→ c⊂C, p→d⊂D, p→c⊄C, p→d⊄D available to the forensic examiner. Procedures are normally implemented using methods and tools that have properties.

Each procedure has the potential to act on any subset of T and to produce false positives (make), false negatives (miss), or correctly find the presence or absence of subsets of C and/or D. Each party has finite resources R:(T,$,C,E). Procedures consume time, money, capabilities, and expertise, and each of these elements limit the ability of the parties to fully examine the space of possibilities. A schedule sequence S:(s1, s2, ...),∀s∈S, s:(l⊂L, r⊂R, h⊂H, e⊂E, t⊂T, c⊂C, d⊂D, p⊂P, r⊂R, t, t') exists where t and t' bound the time period for each step in the schedule, and only subsets of L, R, H, E, T, C, D, P, and R are available within that time frame.

The sizes of elements of this model limit analysis. In particular, (1) L is finite and usually small; (2) R is typically simple and is almost always expressible as a boolean function with some metrics or thresholds; (3) H is defined by documents provided, and the courts prevent ongoing alteration H beyond some time within the schedule; (4) E can be very large, but in most cases it is a few hundred to a few thousand events that are asserted, including statements made by the parties in depositions, testimony, and elsewhere; (5) T, in its totality, is the size of all sets of all states of all digital automata in existence at all relevant times. The total number of traces for m bits of data is then ∑(m!n)2n for n=1 to m, so nothing approaching complete coverage can be attained for almost any legal matter. C is the size of T squared, (|T|)2 and D is the size of T times the size of the power set of E. P is the size of all possible instruction sequences executed on all subsets of T and E in the context of all possible initial memory states over a defined time. Thus C, D, and P are too large to realistically cover as well. R and S act to constrain process, and this effectively limits all aspects of efforts by the examiner to gain understandings of T, C, and D and limits the application of P. A challenge identified in [8] is to identify subsets of P that tend to reveal elements of C and D with values near the extremes of 1 and -1 so as to support or refute events in E and thus support or refute claims in H which are ultimately probative with regard to V. This model will be used to discuss the analysis process.

II. Feature and characteristic detection, extraction, and analysis

Starting with the results of [8], we know that for any real forensic examination, we will need to find p⊂P that allows us to identify revealing c⊂C and/or d⊂D. In the "bag of bits" case, we can review computational complexity of known procedures and, based on assumptions about syntax and semantics, particularize the procedures and complexity measures to specific consistency and inconsistency detection problems relevant to the matter at hand. In this context, content has characteristics, like the file and data structures associated with the operating environment, and features, like the specific content of a file and its meaning in context. A structured file, (e.g., a document) has characteristics (e.g., document type, syntax, etc.) and features (e.g., the combinations of words, spelling errors, etc.). Unstructured content, (e.g., a graphical image file) also has characteristics (e.g., pixel count) and features (e.g., areas that look like eyes, tables, or grass). Traces don't inherently have features other than total size or characteristics other than the specific bits included. But as assumptions are made based on events and analysis, additional features and characteristics are defined which may be consistent or inconsistent with the assumptions. The process of making and testing assumptions is largely the process of analysis..

A. *Trace typing*

Traces are commonly "typed" before being further analyzed. The syntax of the media typically leads to examining portions of traces as groups, such as blocks or other subsequences. This in turn leads to identification of likely types (e.g., file systems, files, embedded files, logs, messages, etc.). Typing effort is fundamental to the creation of further assumptions used for further examination of traces.

Typing of media is usually based on headers placed by FSMs to make identification and proper use easy. Headers may be inconsistent with the content or otherwise misleading. Header or other meta-data examination, file names, and similar analysis are almost all O(1). For files or embedded file systems, headers are also used. Type determination in the storage hierarchy is typically equivalent to spanning a tree. Other methods of typing include syntax analysis by (1) content examination using methods such as the "JDLR" analysis techniques from ForensiX [9], (2) statistical analysis such as information content measures [10] or more specific statistics, which are normally linear time O(n+m) for n different types and m bits of content, or (3) FSMs built to parse different syntaxes, such as multiple lexical analyzers, which is also O(m+n).

Inconsistencies within the type information is problematic in that (1) there are many possible causes, and (2) without a consistent set of types, the analysis is reduced to all possible interpretations of all possible traces. In most legal matters, type information is indicated by events (e.g., files are asserted to be from an party's Windows system). This establishes an event that can be tested as to type by analysis of traces. However, consistency of traces with the events, does not make the result unique. There could be covert content, the content might have different interpretation in a different context, and the event information may be incomplete or imprecise. Virtualization may be used or the system might have been bootstrapped from different media at different times. Thus the underlying FSMs operating are not definitive, even though consistency is maintained within the context observed.

B. *Exact copies, regular expressions, and similar analyses*

For obvious redundancy, (i.e., exact copies at defined locations), finding duplicates is an O(n) bit sequence match with C=1 or C=-1. A consistency measure more tolerant of deviations might identify 1=identical, -1=inverted, and linear interpolation, but this is problematic in that in the non-continuous digital space, a single bit can completely change the syntax and semantics of content, a single shifted byte may produce C=0 or thereabouts, as will compressed or encrypted data. Searching for a string within a larger string is also O(t+k) where t is |trace| and k is |key|.[11] This constitutes a substantial portion of the current digital forensic

analysis effort for cases involving child pornography, possession of contraband information, or similar sorts of offenses. Searching the same trace for multiple strings or sets of patterns that can be written as regular expressions can be linear time through the sequential machines described in [11], but many tools do not use these methods effectively and end up at O(n(k+t)) where n is the number of expressions being sought instead of at O(nk+t). A wide range of similar search methods that gain faster time for repetitive searches of the same traces are identified in [12] and have been substantially improved upon since then. Hashing and similar methods make searching for keys O(k) after initial O(t) hash table creation. While regular expressions are often useful for structured text, for other formats this is far less effective.

Any Backus-Naur Form (BNF) syntax [13] can be parsed by an LALR parser [12] in O(n) time. This applies to most Internet RFCs and many other language specifications in widespread use. Parsing operations may yield type C and D inconsistencies. For forensic purposes, each syntactic component at each level of recursion should be linked to the trace it reflects. This is not commonly done today. In the process of parsing, errors may occur. These errors demonstrate either a parsing procedure fault or a type C or D inconsistency. If these errors stop the processing of the trace, the examiner must find another way to continue. There may be many ways of interpreting a trace in light of such faults, and this adds to the complexity of analysis. In general, it makes parsing as complex as the possible interpretations of languages. Complexity results here do not apply to error handling in LALR parsing, and current tools do not handle such errors well for forensic purposes.

For unstructured data (e.g., pictures, sounds, representations of real-world content captured as depictions through conversions), exact matches are far more interesting. The notion of parsing a picture is very different from that of parsing structured data, but some forms of parsing are used (e.g., detect lines, identify shapes, etc.). These analysis methods are completely different than structured data methods, few of them have linear complexity with the number of bits in the image, and the notion of consistency is far more complex, going to the issue of what the image represents rather than the mere presence of bits in locations. Identical copies can be detected with methods that are linear time for a fixed set of comparisons, such as the search for known images of child pornography, graphical images like icons, and tagents placed in to digital output by printers and then scanned in using higher resolution imaging devices. [14] But these are the exceptions rather than the rule for such analysis. As a simple experiment, we repeatedly scanned the same piece of paper 9 times on the same flatbed scanner without delay and without moving the paper. Each resulting scan file varied in length and content from every other scan file. At the level of 16 byte chunks, the files differed in 99.96% of chunks. The first 256 bytes were identical headers in all of the scans, and only 153 other chunks matched across files, these matches distributed throughout the files and across different pairs of files. Even the same input device yields different outputs for the same source, so exact matching is clearly a problem for these sorts of inputs.

C. Equivalent content in different formats

To search for equivalent content across formats and inexact matches, equivalence classes are defined and traces mapped into the classes. If the classes can be parsed by an LALR parser, linear time results apply as above, but not all equivalence classes can be so specified. Even date and time stamps come in many formats, and since timing and ordering is a key issue in many legal matters, such equivalence classes are important to be able to map. Even in similar records like "Received:" message headers a standard format is not always used, time zone indicators are optional, offsets from universal coordinated time (UTC) are sometimes in incompatible formats, and ordering is affected by differentials of time settings across systems. Parsing anomalies constitute inconsistencies between the traces and events, but inconsistencies between messages within archives may also show inconsistencies with events. The complexity of detecting format differences depends on the specification of the format. In most structured data cases, time to detect pattern inconsistencies is linear in the number of patterns, since the patterns are specified in BNF or an equivalent format. But comparison of subsets of trace (e.g., different emails within a mailbox) to find commonalities is far more complex because each has to be compared to each other. This is O(pt2) where t is the number of trace subsequences examined and p is the number of partial traces per trace subset.

The most common approach to reconciling different formats for value comparison is "normalization". A common format commensurable from other formats is chosen and traces are translated to the common format. For ordered entities, like date and time stamps, where there is a strict "≤" relation, selecting a common format such as "YYYY-MM-DD-HH:mm:ss.pppp..." is particularly useful because it sorts both alphabetically and numerically to the same ordering as time. Similar methods can be applied for ordering in other cases where there is a ≤ relation. Sorting its then O(cn log(n)) where c is the complexity of the comparison method used to determine the ordering relationship. A database approach may help in some cases (e.g., the fields within a header), but more complex analytical processes will likely not gain advantages. The transformations of traces into normalized forms makes them more suitable for analysis, but without the ability to link them back to the original traces, they are problematic for forensic purposes. Also, the normalization process should track differences in original formats to help find inconsistencies between traces (e.g., events assert traces made by the same mechanism have format differences is a D inconsistency).

D. Generating characteristics and features of traces

For structured data, even if formats don't violate syntax specifications, content may vary and provide indicators of origins. (e.g., name and version number, message ID, IP address, field formats, etc.). These may be examined by searching for presence or absence, typically using a regular expression or similar descriptive method and executing a linear time detection algorithm. This generates sets of characteristics of traces that can be related. Searching for

characteristics over the same trace is the sum of the search times, and is linear in the combined search pattern sizes for LALR parses and the total size of the traces examined. Characteristics and features are often recursively sought (e.g., message files generate message separators, header areas, and bodies; header areas generate ordered headers, headers generate fields, etc.). Different characteristics and features apply to each of the different parsed items and thus different algorithms are applied to perform different sorts of consistency tests for each syntax element. Cross-item consistency testing is also feasible (e.g., separator date and time stamps vs. "Received:" header date and time stamps). Automated mechanisms usually have structured syntax, while humans tend to have less structured and more error-prone syntax. People tend to be good at differentiating obvious automated from obvious human messages, but this may be very time consuming for large collections. As a result, automation may be fruitfully applied to try to differentiate these. These are typically LALR parses with resulting linear complexity O(n) for trace length n, but they must properly handle errors to differentiate automated from human mechanisms. Reliability figures for such analysis are not presently available.

Human syntactic elements are used when people generate content, but while standard language analysis has become quite advanced in recent years, analysis of messages used in messaging systems today has grown to include specific syntactic elements used for short message service (SMS) and similar low bandwidth or hard-to-enter data mechanisms, such as cellular phones and instant messaging systems. These messages tend to have abbreviated syntactic elements. LOL in parsing them. A linguistic database and syntax structure analysis capability has yet to be demonstrated to facilitate this sort of analysis for forensic purposes.

A common thread among many of these methods is to break the content into smaller chunks, which we will consider syntax elements, or symbols in the symbol set. Matches between counts and frequencies of symbols are commonly used to detect similar messages. Symbol pairs, triples, and more generally, n-tuples may be sought to find similar phrasing. This is particularly useful for finding common sequences across messages. Some pseudo-random generation methods may be detected by looking for sequences containing one of each of sets of different collections of symbols, such as words, in sequences. In essence, all of these techniques are of complexity O(n log n) where n is the number of symbols, for any given symbol set. But the complexity goes up as the number of different symbol sets increases. Since the total number of possible symbol sets is O(m) where m is the number of bit sequences that can be chosen for symbols, and the number of bit sequences identifiable is the size of the space of traces, the complexity of the general class of all such matches is too high to be practical.

For unstructured data, such as graphical images, different sorts of characteristics are generated. [15][16][7] These unstructured data features are quantifiable in fixed time or linear time in the number of pixels in the image. Recent results from companies like Google have provided linear time parallelizable image characteristic analysis and searches for terms like "house" or "dog", these based largely on recent development in human cognition. [17] But these methods, while useful for generating initial identifications that can be examined in more depth, are not forensically viable today because the mechanisms that drive them are not characterized in terms of reliability for purpose.

For graphical images, derivative traces may be generated by analytical processes and grouped together as well. Just as we can build up syntactic entities in artificial data sets, naturally created data can be built up from lower level components to higher level syntactic entities which can be compared for consistency. For example, shadow detection has been used to determine whether images areas are consistent in terms of apparent sources of lighting. [18] Searching for tagents associated with particular printer types and particularization to specific printers with particular time stamps [14] is an example where image data is structured after low-level traces are translated into higher level syntactic elements. The Google approach to image analysis may also be used to identify features [17] and these analytical results may be compared to events such as statements about the appearance of an object to help guide the investigator in identifying type D inconsistencies.

Features that are not so easily analyzed include properties of the image used for human comprehension and features that can be mathematically characterized but not easily located by automation. For example, shadows in images may be used to show the source of lighting, and the apex of the features and their shadows can be used to determine if different light sources are involved in different parts of an image but they are hard to detect automatically; reflections from eyeballs, silver spoons, and similar highly reflective surfaces in pictures can be mapped into images of what is reflected in them and compared to each other to find composite images, but the analysis and identification of these features is quite complex and not highly automated today [18]; finding areas within images and converting them into maps of real-world objects takes more than linear time; analysis of facial features and other similar biometrics requires substantial analysis to find the features, even though mapping into a database of features is then relatively fast; and tampering detection by blur estimation has been shown successful. [19] Image authentication systems have been proposed for tracing images to sources, and detection of sources have been experimentally performed with limited success. [20] The complexity of these methods is greatly reduced when manufacturers assist in the creation of identifying transforms within their devices.

III. Consistency analysis of characteristics and features

Once characteristics and features are identified, extracted, and analyzed in preliminary ways, whether for structured or unstructured data, the analysis focuses on identifying consistencies and inconsistencies of those characteristics and features in the general sense, and in many cases, the more specific correlation of identical, similar, and related types of features and characteristics within and between content and sources. There are many different

approaches that may be used, and each has the potential to point out different consistencies and inconsistencies.

A. Ordering assumptions and detecting out of order entries

Time is a physical reality that impacts almost every case because most legal issues involve causality in one form or another. Such simple rules as "A caused B implies that A precedes B in time" are very powerful when there is a great deal of data related to times and events. Time is sometimes complicated in digital forensic analysis because the time bases that create time stamps within different systems and mechanisms may be of different formats, be from different time zones, have different clock skews from accurate times as defined by standards bodies, and so forth. In addition, time sequences within computers may be complicated by prior state, loads, external and internal states, inputs, and processing, user intervention, and alteration of traces between their origin and delivery to the examiner.

When regular expressions can be used to describe times, extracting times is O(n) in the trace length. Detecting the presence of out-of-order entries is O(nc) where c is the complexity of the comparison operation and n is the length of the trace. The number of possible original orderings is, in general, the set of all graphs with nodes corresponding to time stamps. This means that rehabilitating evidence by identifying mechanisms that cause ordering failure may be O(n!) where n is the number of times. Mechanisms like file locks, which force sequential output, may cause processes to output in a different order than the order of the time stamps they generate, which forces a looser notion of inconsistency based on ordering that allows for finite time differences and results in a POset rather than a strict ordering.

Recent work in the analysis of overlay patterns of disk writes shows that ordering of file writes can be limited by examining existing patterns of file storage areas on disk. [5] More detailed analysis of time sequencing from traces to validate digital time-stamps has also been done. [6]

B. Sourcing and travel patterns

Sourcing and travel patterns for messages based on headers have been analyzed and they require O(n2/m) time and space where n is the number hops all messages combined took and m is the number of messages [21][7] Sources, destinations, parsed subsets, or other traces may also be used in conjunction with timing information to identify such things as performance effects and other similar damage-related issues asserted to be self-identifying from the traces. [21] In this case, the generation of the relevant data is easy, but finding a process that might determine the effect requires some sort of averaging or other consolidation of traces and analysis of a relationship between the events claimed and the traces.

C. Consistency checks across related records

Consistency checks across related records has been explored [22][7] and time to compare records once associated is linear with the number of entries after sorting, or O(xn log(n)) in total where x is the time to associated records. When identical indicators are present, X is fixed time and the sort is the limiting factor. Audit trails correlation was undertaken in the 1980s [22] with results indicating that creating false but consistent audit trails from existing audit trails is quite difficult.

D. Anchor events and external correlation

Anchor events and external correlation allows traces to be tied across systems and to third parties that the examiner can testify about. [21] Time consistency between records then helps to provide internal trace consistency, geographic location, or time zone. Differentials and jitter in time analysis has also been considered. [7] The complexity of doing these sorts of machine-to-machine differentials of records across machines is O(n) where n is the number of time indicators used, assuming that the time indicators are reconcilable to a common format in O(1) time.

E. Differentials and jitter

For time-related analysis, the common format used for reconciliation of records must also be reconciled with time zone differences including time zone change times. It is prudent practice to translate times into UTC to avoid translation and comparison problems when multiple time zones are in play. Time zones are not always uniquely reconcilable, and this leads to exponential growth in the number of POsets within the delta of the time zones. The same is true for jitter or system-to-system differences, which may also change from time to time.

Differential time is useful in reconciling ordering issues. A standard initial time, such as the zero time of a common clock (Jan 1, 1401 is commonly used), is used as a zero, all times are translated into offsets from the zero date at the maximum clock granularity desired, and times are expressed in clock ticks from the zero time. Translation into a common time frame is O(1) for each time indicator. In analysis of traces, sequences of traces are associated with time offsets from prior or subsequent trace. This is then revealing with respect to sequencing and timing, and allows the calculation of jitter in O(n) time where n is the number of time stamps.

The correlation of these traces other than for strict ordering is problematic. Gathering statistics on means and deviations are O(n), but these are not particularly well suited to the types of errors that occur in digital systems, which generally fail in step functions rather than having deviations from a norm based on random stochastic processes. In cases we have seen, there are many instances of messages delayed by days purported to be delivered by the same process that, during the same time frame, delivered seemingly equivalent messages through the same paths in a matter of seconds. In one example, a message was delivered a second time after 6 months of delay between its original delivery. This is an inconsistency with "normal" behavior, but did not, in the particular instance, indicate anything nefarious. It appeared to be the result of a restoration from an old backup where residual data from the mail transfer agent triggered a resend of and old, already sent message.

IV. Building sieves and counting things

Much of the work of forensic analysis consists of the examiner building sieves to extract specific traces from larger traces and counting different things within traces.

A. Derivative traces

In many cases, the form of original evidence is incompatible with efficient analysis, but translation into a different format greatly improves efficiency. The date and time stamp example above is only one example of the general method of creating derived traces that are back-referenced to original traces, and performing analysis on derived traces.

B. Translations

Common translations include, without limit, translations into other character sets (e.g., EBCDIC to ASCII), multi-line to single-line translations for continuation lines, removal of hyphens used at the end of a line to continue a word on the next line, and combining sequences of separator characters into a single "TAB" character. All of these are O(n) in the length of the trace, but all depend on assumptions about the type and syntax of the trace.

C. Counting things

Many analytical processes in digital forensics involve counting, and there are many common counting errors.[7] Counting is generally O(1) complexity up to a maximum threshold, beyond which the mechanisms for counting fail and produce either errors or wrong results.

D. Combined mechanisms and error handling

When mechanisms are combined and each can produce errors, some of which are not automatically detectable (e.g., a search specification was imperfect), the combined results may produce undetected errors. As sieves, counts, and derived traces are combined into more and more complicated instruments, these instruments tend to become increasingly fragile. Error handling is currently of unknown complexity.

V. Summary, conclusions, ande further work

Clearly the study of the use of redundant traces for consistency is only in its infancy and the available methods for analysis and correlation of these redundant traces are already substantial. At a fundamental level, it seems clear that redundancy is key to consistency analysis and that systematic identification and analysis of redundancy may lead to a more complete theory and practice of consistency analysis.

This paper only covers the rudimentary forms of analysis in widespread use today and further work is needed to characterize other classes of consistency checking in analysis, including analysis of effects of parallelism, detection is similarity rather than more precise matches, addressing issues of mixed symbol sets and other similar environmental factors, analysis of possible consistencies and inconsistencies of missing traces and use of this to guide future events, validation requirements for the methods used.

References

[1] The U.S. Federal Rules of Evidence.

[2] Daubert v. Merrell Dow Pharmaceuticals, Inc., 509 U.S. 579, 125 L. Ed. 2d 469, 113 S. Ct. 2786.

[3] Frye v. United States, 293 F 1013 D.C. Cir, 1923

[4] [4] Reference Manual on Scientific Evidence - Second Edition - Federal Judicial Center, http://air.fjc.gov/public/fjcweb.nsf/pages/16

[5] Svein Yngvar Willassen, "Timestamp Evidence Correlation", Presentation at IFIP WG 11.9 International Conference on Digital Forensics, 2008

[6] Svein Yngvar Willassen, "Hypothesis-based investigation of digital timestamps", chapter in Advances in Digital Forensics IV, Ray and Shenoi ed., Springer, ISBN# 978-0-387-84926-3, 2008.

[7] F. Cohen, "Digital Forensic Evidence Examination", ASP Press, 2009, ISBN#1-878109-44-8.

[8] F. Cohen, "Two models of digital forensic analysis", IEEE/SADFE-2009, Fourth International IEEE Workshop on Systematic Approaches to Digital Forensic Engineering, Oakland Conference, Oakland, CA, USA, May 21, 2009

[9] F. Cohen, "ForensiX", The ForensiX Just Doesn't Look Right (JDLR) mechanism is detailed in the source distribution available in http://all.net/ForensiX/Forensix.tar

[10] S. Moody and R. Erbacher, "SADI – Statistical Analysis for Data type Identification", 3rd International Workshop on Systematic Approaches to Digital Forensic Engineering, 2008.

[11] P. Weiner, "Linear pattern matching algorithm". 14th Annual IEEE Symposium on Switching and Automata Theory: 1-11. (1973).

[12] D. Knuth, "The Art of Computer Programming, Volume 3, Searching and Sorting", 1973, Addison-Wesley.

[13] Crocker, D. and P. Overell, "Augmented BNF for Syntax Specifications: ABNF", RFC 4234, October 2005. This also references RFC 733 and 822 as source documents. Available at http://www.ietf.org/rfc/rfc4234.txt

[14] D. Schoen, "Investigating Machine Identification Code Technology in Color Laser Printers", 2005, The Electronic Frontier Foundation, available at: http://www.eff.org/wp/investigating-machine-identification-code-technology-color-laser-printers

[15] Rudolf L. van Renesse, "Optical Document Security", 3rd edition, 2005, ISBN 1-5805-258-6, Artech House, Boston, London.

[16] F. Meng, X. Kong, and X. You, "A New Feature-based Method for Source Camera Identification", IFIP WG 11.9, International Conference on Digital Forensics, 2008 appearing in "Advances in Digital Forensics IV", I. Ray and S. Shenoi, Ed.

[17] Tom Dean, "Disruptive Perspectives on Biological and Machine Vision", Keynote Address at HICSS 42, Jan 5-8, 2009.

[18] Hany Farid, "Digital Image Forensics", National Academy of Sciences, Annual Meeting Symposium, Legal/Forensic Evidence and Its Scientific Basis, April 25, 2006, p[resentation available at: http://progressive.playstream.com/nas/progressive/2006am-forensic-farid/Hany_Farid.html

[19] Dun-Yu Hsiao, Soo-Chang Pei, "Detecting Digital Tampering by Blur Estimation", Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05), 2005.

[20] I-Chuan Chang Bor-Wen Hsu and Chi Sung Laih, "A DCT Quantization-Based Image Authentication System for Digital Forensics", Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05), 2005.

[21] F. Cohen, "Issues and a case study in bulk email forensics", Fifth annual IFIP WG 11.9 International Conference on Digital Forensics, 2009/01/27, appearing in "Advances in Digital Forensics V" I. Ray and S. Shenoi, Ed., 2009

[22] F. Cohen, "A Note on Detecting Tampering with Audit Trails", IFIP-TC11, "Computers and Security", 1996