To appear in Information Visualization, ***10***:2, 2011

Visual Analysis for Textual Relationships in Digital Forensics Evidence

T.J. Jankun-Kelly* David WIlson*† Andrew S. Stamps* Josh Franck‡
Jeffrey Carver§ J. Edward Swan II*

Mississippi State University and University of Alabama

ABSTRACT

We present a visual analytic framework for exploring the relationship of textual evidence for computer forensics. Based upon a task analysis study performed with practitioners, our tool addresses the inefficiency of searching for related text documents on a hard drive. Our framework searches both allocated and unallocated sectors for text and performs some pre-analysis processing; this information is then presented via a visualization that displays both the frequency of relevant terms and their location on the disk. We also present a case study that demonstrates our framework's operation, and we report on an informal evaluation conducted with forensics analysts from the Mississippi State Attorney General's Office and National Forensics Training Center.

Keywords: computer forensics, visualization, visual analytics, treemaps, tag clouds

1 INTRODUCTION

Computer forensics investigation is a growing field. Computer crime is rapidly increasing [1–3] and difficult to prosecute [4,5]. Investigators suffer from amounts of data and a plethora of media types that make automated tool development difficult, leaving the burden of analysis on the human investigator. Current practice requires hours or days to potentially find all evidence of interest as current tools present only the data on the disk with little additional structure to support the analytic process. These research challenges suggest an opportunity for visual analysis that this paper in part addresses.

Computer forensic investigation follows a straightforward workflow. Images of digital media are processed by tools such as EnCase [6], AccessData's Forensic Toolkit (FTK) [7], or the Autopsy Forensic Browser [8,9]; specialized tools that extract only text (but do no further visualization) such as bintext [10] may also be used. Next, these tools are used to manually search through areas of interest, such as suspicious directories, registry or system settings, web browser history and caches, and other locations. Finally, investigators sift through these results to find evidence, which may lead them to additional searches as above—this evidence is eventually condensed into a report of findings. This process requires significant knowledge of the ideal locations to search on the disk; however, it also requires laborious searching that could be augmented by the analysis tools.

There have been varying approaches to provide some automated pre-analysis to forensics investigators. Indices of common or suspected keywords on the disk image are most commonly generated to assist in text search. For example, in FTK, single word searches may be performed on an indexed image; conjunctive searches may be built out of multiple single-word searches. Such conjunctive searches allow an investigator to find all of the files that contain all of the words of interest, but do not provide information on where those words occur within any individual file or in relation to each other. In addition, the frequency of the terms is also important but is only provided in the single term searches. The richness of the relationships between the search terms and the data on the disk motivated the visual analysis framework we present here.

In this article, we expand the description of our textual relationship visual analysis framework presented previously [11]. By studying how forensics investigators use their tools, we have identified the aspects of textual relationship analysis that could benefit from visual analysis. Our framework allows investigators to see files in which a set of words exist, monitor the frequency of terms in those files that may point to additional evidence, and filter those terms based upon inter-word relationships or type (such as currency, email addresses, and so on). This information is presented visually in a manner that leverages an investigator's familiarity with the hierarchy while providing a rich visualization. In this expanded paper, we provide additional details on the observational study that guided the design of our framework; in addition, we provide feedback from investigators on the suitability of the tool for their work. Additionally, we update the system description and performance for its current implementation. Overall, our analytic framework aims to reduce the effort required to find textual evidence of interest.

*Department of Computer Science and Engineering, Bagley College of Engineering, Mississippi State University. Email: tjk@acm.org, {ass78,dw152}@msstate.edu, swan@acm.org

†Now at Microsoft

‡Department of Psychology, Mississippi State University. Email: jaf210@msstate.edu

§Department of Computer Science, University of Alabama. Email: carver@cs.ua.edu

2 Related Work

Current computer forensic systems are primarily text-based; our premise, supported by results elsewhere in security visualization [12], is that visualization can augment such approaches. In this work, we focus on hard disk forensics of textual data; graphical media (e.g., in pornography cases) and audio media (e.g., in piracy cases) are beyond our scope. Visualization for computer forensics is a relative new area of research with a smaller body of work than general security visualization. Most similar to our work is that of Teerlink and Erbacher [13, 14] which uses two displays of the hard drive: The first uses a matrix of squares that are shaded by the file's metadata (e.g., last access time); the second uses a treemap [15, 16] combined with color to show the metadata-enhanced file hierarchy. Our approach also uses a treemap-like display, but unlike Teerlink and Erbacher's metadata focus, our framework is designed for inter- and intra-file textual analysis. For forensic text analysis across file contents, Schwartz and Liebrock [17] use a histogram-like adaptation of Tile-Bars [18] to depict the distribution of terms in a disk image. Their approach uses terms pre-provided by the user which is less useful for dynamic term exploration; additionally, the set of files containing the terms are not depicted in the disk hierarchy nor are unallocated clusters depicted. Our analytic framework indexes both the allocated and unallocated disk space, allowing for dynamic visualization of new search terms while also depicting file contents to suggest new terms.

3 Visualization and System Design

3.1 Pre-Design Task Analysis

Before we created our visual analysis framework, we studied the use of forensics tools by investigators to guide our design. A contextual analysis [19–21] of forensics practitioners was performed: Officers were provided a laptop with forensics software and a synthetic disk image and asked to "do what they do normally" in their analysis (Figure 1, left). A think-aloud protocol was used. The officers were were asked to verbalize all their thoughts in a stream-of-consciousness manner during performance of the task; the audio, along with video and key/mouse movements, were recorded by our testing apparatus, which could later be analyzed using our testing software described by our previous work (Figure 1) [22]. In summary, the software provided a synched display of the webcam video, the screen shot of the application (with mouse events highlighted), and our log of keyboard and mouse events which we annotated later during our analysis described below. This software was developed in house for this project.

Initially, we chose a think-aloud protocol because in-process verbalization has been shown to provide task-specific information about reasoning that only occurs during the task; such information is not available in retrospection [23, 24]. In addition, such protocols are unobtrusive [23]. The analysis of verbal responses was intended to provide a detailed view of the officers' problem-solving and decision-making processes during performance of the task, using perhaps 10–15 different classifications of verbal utterances. However, due to many factors detailed elsewhere [22], we were limited to three practitioner participants. This small set of participants limited the validity of any formal between-subject analysis of the verbal data. Thus, we used the verbal, screen capture, and key/mouse data in a qualitative and informal manner to classify the officer's interactions with the forensics software.

For each utterance/user interaction, we classified the action performed into one of four general categories: Selection, Manipulation, Search, and Note taking. Selection refers to direct interactions such as selecting a file for view, navigating lists or other elements, and so on. Manipulation refers to actions that alters the view or format of viewed data, such as switching from ASCII to hex or examining metadata. Search (broken into Name, Email, or Other) refers to the use of the included keyword search system. Finally, Note (broken into Copying, Linking data, or Other) refers to the use of the included note-taking system—this category includes all stated decisions on which information to classify as evidence. A total of 493 separate actions were coded and classified from the three hour-long sessions (Figure 2).

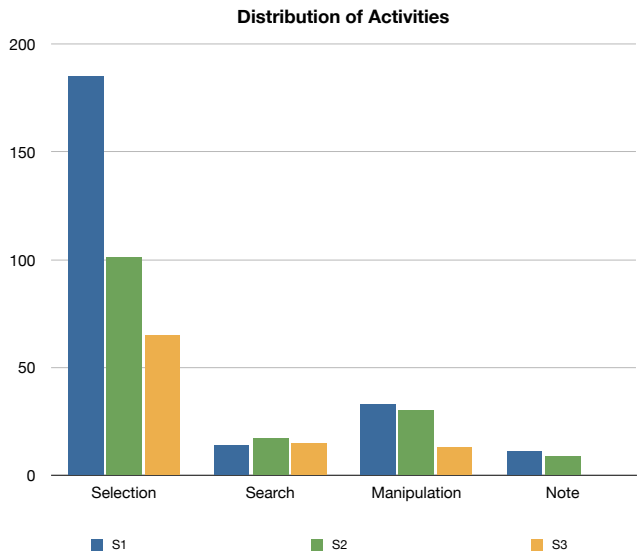

Figure 2: Results of our classification of user activity during our forensics tasks analysis of three officers. Selection and navigation of search terms dominated the activity.

Though we cannot discuss any quantitive findings, our informal analysis did discover trends to inform our design. Selection actions proved to be by far the most common action, accounting for 71% of all actions performed by participants. Delving into the recorded data, we found that files containing search terms returned in a list were generally accessed sequentially with no other systematic strategy evident. This sequential search of search results indicates that searches were not filtered sufficiently to narrow the search

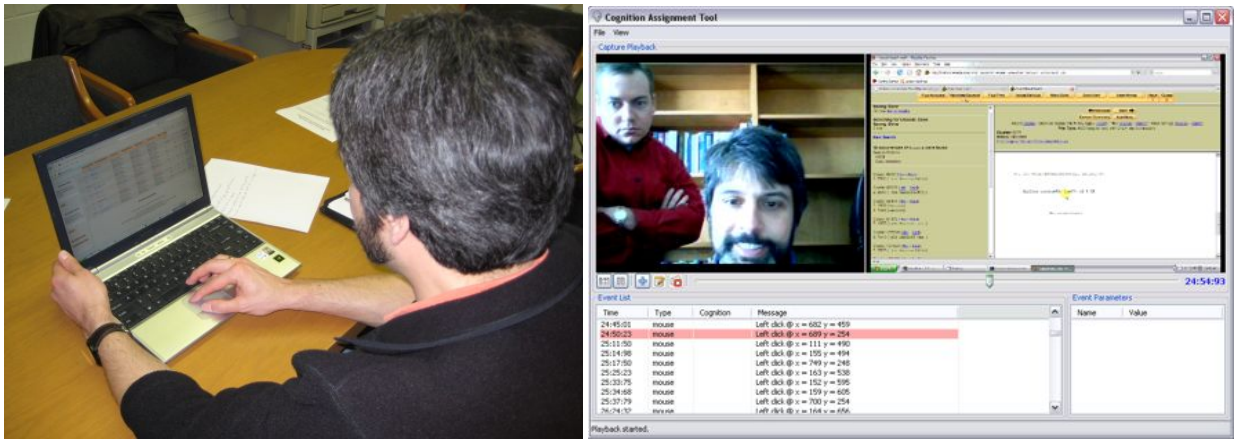

Figure 1: Experimental setup for our task analysis. *Left*: Experimental rig with keyboard, mouse, and video capture. *Right*: Analysis tool use to process the trials. Webcam input is in the upper-left, screen capture video in the upper right, mouse events on the lower left, and observer notes and analysis on the lower right.

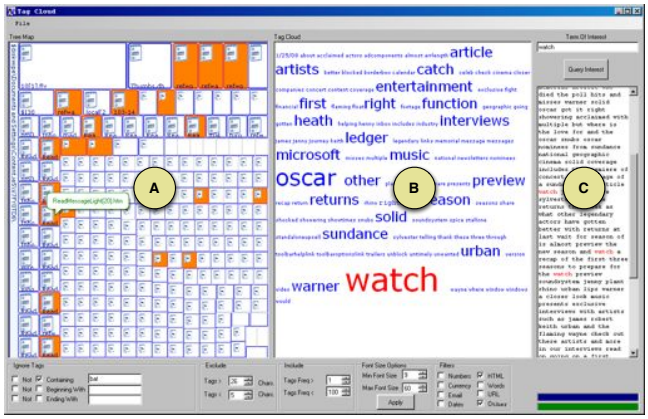

Figure 3: Our text forensics visualization system. It consists of a treemap-like depiction of the file hierarchy (a), a tag-cloud of terms in selected files (b), and the direct search interface (c). File metadata, conjunctive search, and cluster visualization are also available.

to evidence-bearing files. Analyzing the searches corroborates this finding—when text evidence was found, manipulation (interaction with the file) and search (finding additional terms) were of similar frequency with fewer selection events.

From our analysis, it was decided that the greatest impact on performance would be derived from increasing the efficiency of the search process. For example, while a name or amount may be found, finding the other names/amounts related to them took significant searching. It is also important to detail where this information was on the disk for evidence-collecting purposes and to find similar data. The design of this visualization is presented next.

3.2 Visualization

Our visualization has three main panes:

- A search-sensitive file hierarchy (Figure 3a).
- A tag cloud of terms in the selected files (Figure 3b).
- An interface to search for terms directly on the disk or within a file (Figure 3c).

In addition, contextual information about the selection, conjunctive search, and disk cluster-based visualization of search terms are also available. Each of these views is linked so that changes to one is reflected in the other. Our visualizations extend extant methods (treemaps and tag clouds) with an eye towards solving our specific forensics analysis problems; they are also tailored for our audience of forensics officers. These methods are discussed next.

Search-Sensitive Hierarchy. The hierarchy-view (Figure 3a) contextualizes the search and provides visual cues of where searches items are found and where other evidence could be located. Our depiction uses a modified squarified treemap [25]; node size is based upon the number of non-filtered words in a file; no binary media files or files without text are displayed. The treemap was modified in two ways to assist in its usability. First, we use icons to distinguish between files and directories; the icons are similar to those used by standard file browswers. This was done in order to provide a familiar starting point to forensic officers as the views are similar to those in everyday experience, other than the differing element size; in addition, the icons provide an at-a-glance difference between an end-point in the hierarchy (a file) and one with children (a directory). Secondly, the left-hand side of the treemap provides the context for upper-levels in the directory (as opposed to eliding them or displaying them surrounding the child directory). This both saves screen space (as opposed to a containing view) and has some familiarity (such as the left-to-right opening hierarchical displays used in OS X).

To foster dynamic exploration, the hierarchy is "search sensitive." The tree visually highlights the location of terms selected in the tag cloud or conjunctive search. The size of the nodes can be adjusted based upon selected tag cloud or searched terms; this highlights areas where evidence-bearing material is located on the disk. Currently, the displayed size of the files/directories is fixed even if words are filtered out by later operations; we are currently investigating dynamically resizing the nodes but have yet to find a method that

Figure 4: Tag cloud for a selected directory. Selecting terms here will highlight files containing the term.

does not potentially shuffle around the display disruptively. Selecting a file in the display also triggers a change in the tag cloud to reflect the terms in the newly selected file.

Term Tag Cloud. When a file is selected, a tag cloud depicts the terms within the file (Figures 3a, 4; see System Infrastructure for details). Word size is based on frequency; more frequent terms are given more display space. To account for perceptual weighting, we use a quadratic falloff to determine a word's point size; this correction creates a linear perceptual fall-off in area. Words selected in the view will be highlighted in the textual display of the file; in addition, words can be selected to be removed from the view. Highlighted terms (in red) match search terms from the conjunctive search view.

As text may be found in different file types, we provide additional filters to either select a subset of terms or remove terms that are unneeded. For example, in web browser caches, textual "noise" from HTML markup pollutes the view. To reduce this noise, words are identified during term extraction as HTML tags, email, currency, and so on. These term types can then be used to filter out words or show only words matching a given type. Additional filters may be added dynamically for types of words, word length, minimum or maximum word frequency, etc. The filtering is propagated to all other views.

Search Interface. The search interface is the alternate starting point for the system, allowing an investigator to directly search for a term. A list of files and directories containing the term is listed here; files are also highlighted in the treemap, and the contents of any selected file or directory are displayed in the tag cloud. For more advanced word searches, see the contextual search below. Alternatively, the text of the selected file with the primary search terms highlighted may be shown, as done in Figure 3c.

Contextual View. The contextual view displays metadata about the selected file or directory. This includes ownership, permissions, file/directory size, its creation and modification

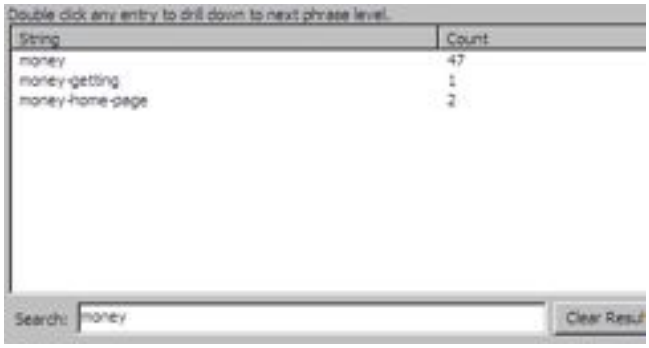

Figure 5: Our contextual search interface. As a word is entered, phrases that begin or end with that term are shown.

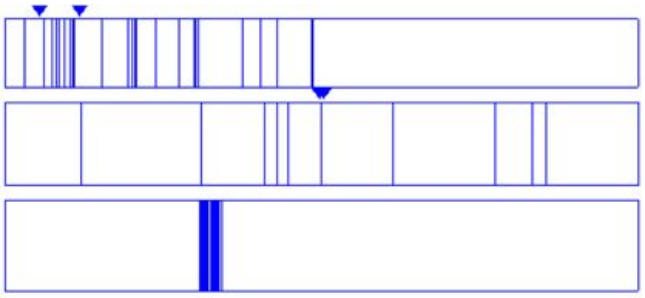

Figure 6: Cluster view of the selected terms. The top layer shows the entire disk, the middle layer a selected subregion (between the triangles), and the bottom a cluster-by-cluster depiction. Clusters with the search term are highlighted blue.

size, and other similar information. This space is also used to show the contents of a selected file as desired. When a term is selected, metadata for how often the term occurs in the selected file and over the entire disk is provided.

Contextual Search View. When searching for a term, the context in which the term occurs may be important. The contextual search view (Figure 5) provides this functionality. Given an initial term such as "money," the view populates with found two-word phrases that start or end with that term. Selecting one of these phrases will generate a list of phrases that start or end with the selected phrase and so on. Currently, only naïve stemming of the text terms is performed via a "like" match in the database; in the future, more advance stemming such as that used by Parallel Tag Clouds [26, 27] would provide more robust searches. Selecting the phrases also highlights the files that terms occur in within the various views. This phrase-based search supports finding specific mixed terms such as "investment fraud" that a pure conjunctive-based search would group with non-phrases.

Cluster View. Our final view (available as a separate window) displays where information is located on the physical disk; it is used to highlight clusters associated with the currently selected terms in the tag cloud view (Figure 6). This view primarily benefits identifying where deleted files bearing evidence are located as they will not appear in the hierarchy view. The display is essentially a horizontal rectangle

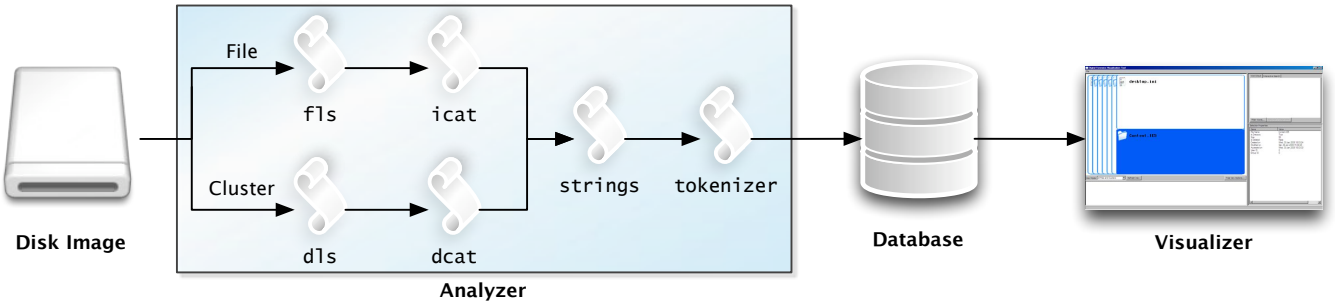

Figure 7: Workflow of our forensics visual analytics system. The Analyzer processes the text from the disk image and stores it in a database that the Visualizer depicts.

that contains the entire range of clusters at one time as individual lines (or rectangles for contiguous cluster ranges). There are two levels of zoom that can be achieved through clicking this initial single rectangle. If the user clicks (and optionally drags horizontally) over a region of the cluster view, two additional rectangular regions will appear in the same space as the original view, both containing different zoom levels. The middle zoom region shows whatever arbitrary region was selected by clicking or dragging in the topmost cluster view. Based on that selection, the third zoom pane's selection is determined, which depicts a one-to-one vertical line to pixel rendering of the center of the selected middle zoom region. The magnification window within the middle zoom region can moved around for more precise investigation in the lowest-level area.. Triangular glyphs that are part of the upper and middle views delineate where the magnification windows in the preceding zoom level are located. The location and other metadata for the selected cluster are displayed in the contextual view.

3.3 System Infrastructure

To provide interactive exploration, our visual analytic framework consists of two primary applications (the Analyzer and the Visualizer) built around different tools. Figure 7 summarizes the application workflow. The Analyzer processes disk images for string tokens, writes these to a database with metadata identifying the file or cluster corresponding to the cluster, and then the Visualizer depicts the disk image as discussed previously. The details of these two systems are detailed here.

For the Analyzer preprocessor, we make extensive use of the Sleuth Kit [28,29] to extract the file structure, unallocated sectors, and textual data. The file hierarchy and unallocated clusters are treated separately before merging their data with our tokenizer, itself written in Python. For a given disk image, we determine the file structure via the `fls` tool (which indicates all the files in a directory); walking over this structure, the contents of the file in the image are extracted via `icat` and the textual content of this stream is distilled via the UNIX `strings` utility (which extracts only the printable text). A similar process is used for unallocated clusters: Clusters are enumerated via `dls` (which walks over disk clusters), their content is extracted via `dcat`, and their text is distilled via `strings`. The strings are then processed by our tokenizer, which separates the lines into tokens, identifies each token as a word, number, US currency, URL, email address, or unreadable symbol, and then stores it in a MySQL (formerly SQLite) database. Each token is stored with a reference to the previous and next token in the file; the file or cluster the text belongs to is also stored for later reference by the visualization. The current implementation requires roughly 20 hours to index all the clusters on a 4.5 GB disk; for a subsection of interest (such a 127 MB web cache directory), it takes about 162 minutes. This is a one-time process and is comparable to current indexing tools, but we are examining means to accelerate its performance.

The Visualizer, now written in Visual Basic, .NET, and WPF/DirectX (updated from our Python/wxPython/OpenGL version reported previously) accesses the MySQL database directly. The Visualizer uses the database exclusively; it does not require access to the original disk image. Data is requested as needed. Most interactions are responsive: Querying a 125 MB image takes 13 sec for a given term across the image. Treemap rendering is responsive, taking less than 2 sec to transition between sparse and very dense (600+ nodes). Rendering the tag cloud is interactive as terms are indexed; this behavior has been optimized since our previously reported implementation [11].

4 Case Study: Email Investment Fraud

To demonstrate our visual analysis framework, we provide a small case study. Previously, we generated an investment fraud cases where a fictitious criminal, William Slick, used the email distribution services of an intermediary "abacus55" to commit fraud. We created several test email accounts on different web-mail services and simulated standard web-browsing and email behavior with the fraudulent behavior interjected. For this case study, we use a subset of the disk that focused on the 125 MB web-cache. In this case study, we present how analysts could use our tool to find the email fraud evidence.

Given the sparse details of the case (it is only known that

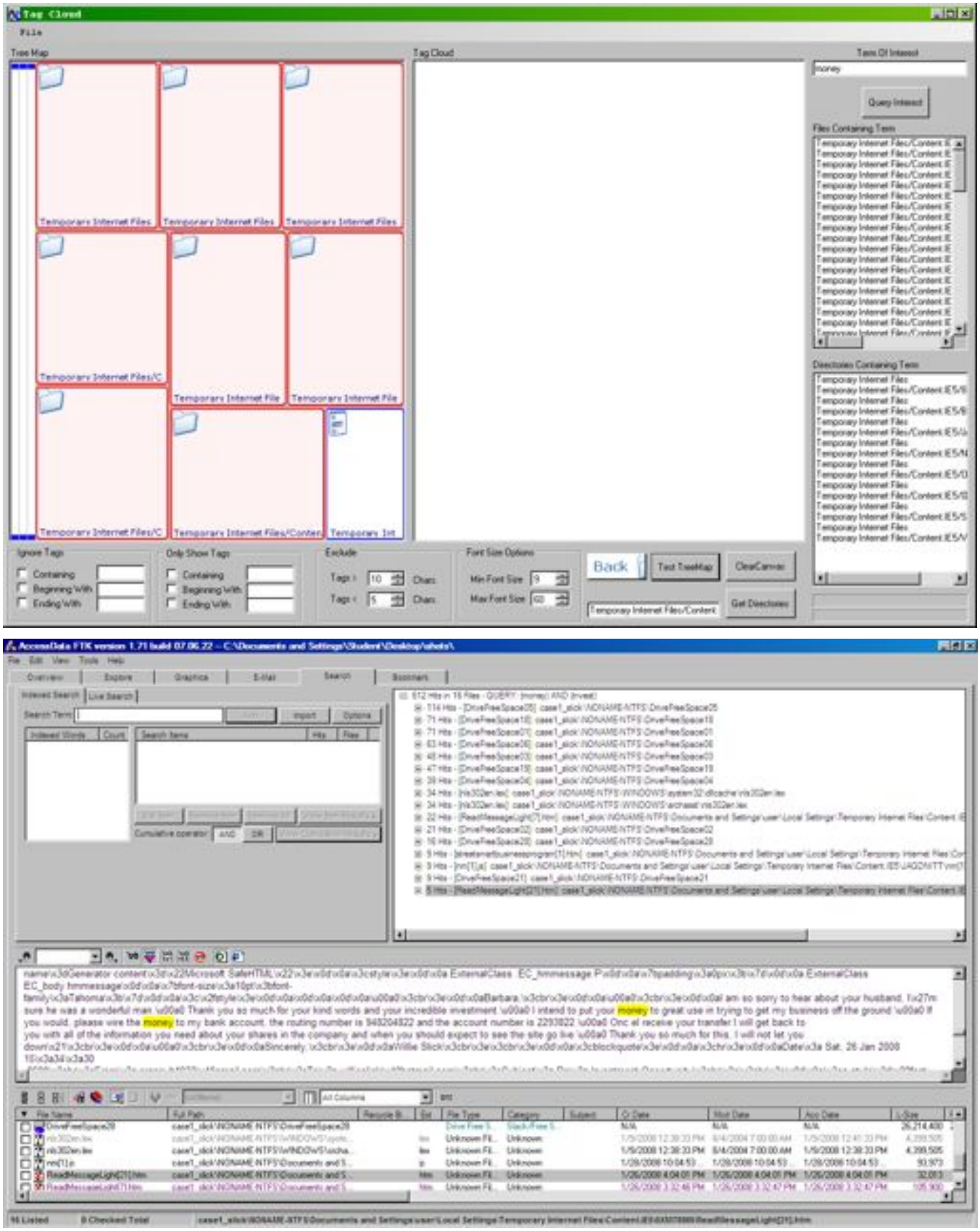

Figure 8: *Top:* Initial search for the term “money” on the disk The list on the right enumerates the occurrences of the term. *Bottom:* The same search performed with FTK. Note the FTK view does not visually display the hierarchy or show related terms (such as “money getting”).

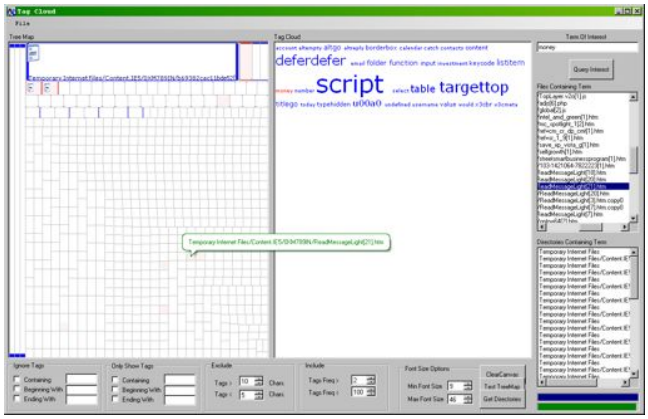

Figure 9: Filtered tag cloud for one of the selected values for the term "money."

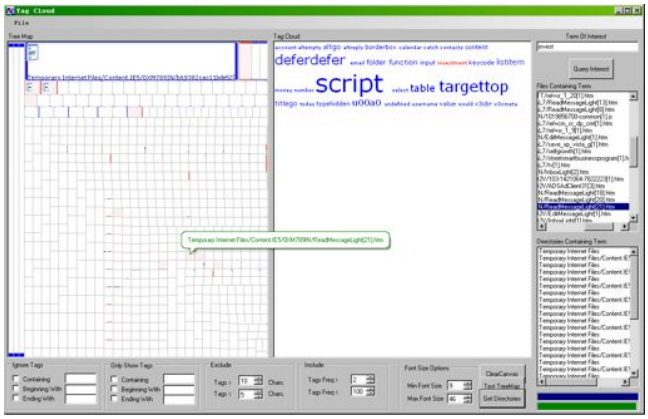

Figure 10: Filtered tag cloud for one of the selected values for the term "invest" in the same file as the occurrence of "money."

fraud occurred), a search for fraud-related terms is the first course of action. An initial search for "money" turns up hits in several files across the web cache directory (Figure 8, top). In current tools, such as FTK, a similar initial search returns just the list of files with no additional contextual relationships (though the contents of the selected file is shown, Figure 8, bottom). We can search for conjunctive terms such as "money investment" or "money getting" to narrow down his list. Choosing one of the files, we can generate a tag cloud its contents: It is mostly a mix of HTML codes for the web-mail page and text from the rest of the cached messages (Figure 4). By selecting only word terms, filtering out common HTML and web-mail tags, and looking for words with a minimum frequency, the tag cloud confirms that money was mentioned several times in the selected file (Figure 9).

Given that fraud is the suspected crime, other terms related to fraud can be searched. For example, a search for "invest" turns a few hits on "investment." Due to its small frequency, "investment" would be given little screen space in the tag cloud; like all such clouds, it suffers from the issue of hiding infrequent terms. To address this, we can filter out terms that are more frequent than a given threshold, allowing us to focus on infrequent terms. Such a search displays the "investment" hit in the same file that the "money" terms were found (Figure 10). Now that we have likely found an evidence file, we can search for specific numeric amounts (by choosing to display only currency data) or related emails. For example, if we select email addresses, the address of our suspect is clearly identified: `willieslick@hotmail.com`. If desired, we can inspect the contents of the file directly, find other email addresses, or search for additional related files on the disk.

5 Evaluation

To validate our approach, an informal evaluation by our forensics practitioners was performed. Two investigators from the Mississippi State Attorney General's Office and three forensics instructors at the National Forensics Training Center at Mississippi State provided feedback. Each analyst was provided a half-hour demonstration with additional unstructured time to interact with the tool, ask questions, and provide feedback, which was recorded by the interviewer.

Overall, feedback was highly positive. Analysts indicated that the tool would better structure their searching workflow. Whereas previously the steps would be Search→Process File→Expand Search→Iterate terms, they saw the tool as directly indicating which files are of interest and which are not via the highlighting and the treemap node size indicating the amount of matching text. They did not find the treemap disorienting, which was helped by the indicator of directory depth to the left (the vertical bars, one for each parent directory, in Figures 8a–10. The tag cloud was seen as a means to rapidly search for terms in a selected file through direct search (as opposed to searching visually or through the database).

Several suggestions for improvement were provided by our analysts and are currently being addressed. For example, our filtering based upon type of tag (email, currency, etc.) could be improved by additional important types, including common credit card patterns and other financial signifiers. We are investigating means to build custom searches into the databases such as through regular expressions, which were also requested. Additional ways to visually encode common metadata, such as access or modification times, were also desired; this display would adapt methods already used by Teerlink and Erbacher's work [13, 14] for such a purpose.

6 Conclusions and Future Work

In this work, we have detailed a visual forensics analysis tool for finding textual relationships amongst files on a disk image. The framework includes an Analyzer component that extracts word frequency, their distribution on the disk, and their type, and a Visualizer that depicts these relationships. A treemap modified with characteristics of a normal file browser and a coordinated tag cloud facilitate the exploration of the text data. To motivate these design decisions, we have included details of a pre-design contextual study with forensics officers; the feedback from practitioners we have included indicates that our design choices were effective.

The feedback from our investigator colleagues has already provided several avenues of future work. In addition, we are examining several other directions. First, our goal with this project is to empirically motivate and validate the visualization; while we have done the former, the latter validation is still needed. While we have confidence from our case studies and expert feedback that the visualization benefits forensic analysis, we do not have any quantitative confirmation. We are currently working with the Mississippi State Forensics Training Center for this task. Secondly, there are technical improvements to the Analyzer we are investigating. Primary amongst these are reducing the processing time; one idea is to use FPGAs to generate the terms while the disk is being imaged as was done for an image file search [30]. In addition, we wish to improve the processing of the text (i.e., its stemming) to better match terms of interest. Finally, there are improvements to the visual representation to consider. We have already discussed providing visual feedback of the file metadata within the treemap views. Improvements to the textual view for showing the text relations is also being considered; for example, we may be able to use approaches such as word trees [31] or phrase nets [32] to assist in visually depicting the conjunctive search relations.

ACKNOWLEDGEMENTS

We gratefully acknowledge the support of the numerous law enforcement officers that either participated in the study, worked with us on the visualization design, or answered questions from us. We also thank Dr. David Dampier, Kendall Blaylock, Jansen Cohoon, and Gary Cantrell of the National Forensics Training Center at Mississippi State for their assistance. The work is funded by a National Science Foundation CyberTrust grant #CNS-0627407.

REFERENCES

[1] Bequai, A. (2002) Syndicated crime and international terrorism. *Computers and Security*, 21(4):333–337. 1

[2] Kessler, G. and Schirling, M. (2002) Computer forensics: Cracking the books, cracking the case. *Information Security*, pp. 68–81. 1

[3] Wolfe, H. (2003) Computer forensics. *Computers and Security*, 22(1):26–28. 1

[4] Householder, A., Houle, K., and Dougherty, C. (2002) Computer attack trends challenge internet security. *IEEE Computer*, 35(4):5–7. 1

[5] Thompson, R. (1999) Chasing after 'petty' computer crime. *IEEE Potentials*, 18(1):20–22. 1

[6] EnCase. http://www.guidancesoftware.com/products/ef_index.aspx. Last checked May 2009. 1

[7] AccessData, Forensic toolkit 2.0. http://www.accessdata.com/forensictoolkit.html. Last checked May 2009. 1

[8] Carrier, B., Autopsy forensic browser. http://www.sleuthkit.org/autopsy/. Last checked May 2009. 1

[9] Carrier, B. (2004) *Know Your Enemy*, chap. Ch. 11: Computer Forensics Basics. Addison Wesley, 2nd edn. 1

[10] Foundstone, bintext. http://www.foundstone.com/us/resources/proddesc/bintext.htm Last checked October 2010. 1

[11] Jankun-Kelly, T. J., Wilson, D., Stamps, A., Franck, J., Carver, J., and Swan II, J. E. (2009) A visual analytic framework for exploring relationships in textural contents of digital forensics evidence. *Proceedings of the 6th International Workshop on Visualization for Computer Security*, pp. 39–44. 1, 5

[12] Goodall, J. R. (2009) Visualization is better! a comparative evaluation. *Proceedings of the 6th International Workshop on Visualization for Computer Security*, pp. 57–67. 2

[13] Teelink, S. and Erbacher, R. F. (2006) Improving the computer forensic analysis process through visualization. *Communications of the ACM*, 49(2):71–75. 2, 7

[14] Teelink, S. and Erbacher, R. F. (2006) Foundations for visual forensic analysis. *Proceedings of the 7th IEEE SMC Workshop on Information Assurance*, pp. 192–196. 2, 7

[15] Bederson, B. B., Shneiderman, B., and Wattenberg, M. (2002) Ordered and quantum treemaps: Making effective use of 2D space to display hierarchies. *ACM Transactions on Graphics*, 21(4):833–854. 2

[16] Shneiderman, B. (1992) Tree visualization with treemaps: a 2-d space-filling approach. *ACM Transactions on Graphics*, 11(1):92–99. 2

[17] Schwartz, M. and Liebrock, L. M. (2008) A term distribution visualization approach to digital forensic string search. Goodall, J. R., Conti, G. J., and Ma, K.-L. (eds.), *Proceedings of the Fifth International Workshop on Visualization for Computer Security*, vol. 5210, pp. 36–43, Springer. 2

[18] Hearst, M. A. (1995) Tilebars: Visualization of term distribution information in full text information access. *CHI*, pp. 59–66. 2

[19] Hackos, J. T. and Redish, J. C. (1998) *User and Task Analysis for Interface Design*. John Wiley & Sons, Inc. 2

[20] Hix, D. and Hartson, H. R. (1993) *Developing User Interfaces: Ensuring Usability through Product & Process*. John Wiley & Sons, Inc. 2

[21] Mayhew, D. (1999) *The Usability Engineering Lifecycle: a Practitioner's Handbook for User Interface Design*. Morgan Kaufmann Publishers. 2

[22] Jankun-Kelly, T. J., Franck, J., Wilson, D., Carver, J., Dampier, D., and Swan II, J. E. (2008) Show me how you see: Lessons learned from studying computer forensics experts for visualization. Goodall, J., Conti, G., and Ma, K.-L. (eds.), *Proceedings of the Fifth International Workshop on Visualization for Computer Security*, vol. 5210, pp. 80–86, Springer. 2

[23] Ericsson, K. A. and Simon, H. A. (1993) *Protocol Analysis: Verbal Reports as Data*. Bradford Books/MIT Press. 2

[24] VanSomeren, M. W., Bernard, Y. F., and Sandberg, J. A. C. (1994) *The Think Aloud Method: A Practical Guide to Modeling Cognitive Processes*. Academic Press. 2

[25] Bruls, M., Huizing, K., and van Wijk, J. J. (2000) Squarified treemaps. *Proceedings of the Joint Eurographics/IEEE TVCG Symposium on Visualization 2000*, pp. 33–42. 3

[26] Collins, C., Viéagas, F. B., and Wattenberg, M. (2009) Parallel tag clouds to explore and analyze faceted text corpora. Stasko, J. and van Wijk, J. J. (eds.), *IEEE Symposium on Visual Analytics Science and Technology 2009*, pp. 91–98. 4

[27] Porter, M. F. (1980) An algorithm for suffix stripping. *Program*, 14(3):130–137. 4

[28] Carrier, B. (2005) *File System Forensic Analysis*. Addison Wesley. 5

[29] Carrier, B., The Sleuth Kit. http://sleuthkit.org/sleuthkit/. Last checked May 2009. 5

[30] Dandass, Y. S. (2007) Hardware-assisted scanning for signature patterns in image file fragments. *40th Annual Hawaii International Conference on System Sciences*, p. 268, IEEE Computer Society. 8

[31] Wattenberg, M. and Viégas, F. B. (2008) The word tree, an interactive visual concordance. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1221–1228. 8

[32] van Ham, F., Watenberg, M., and Viéagas, F. B. (2009) Mapping text with phrase nets. *IEEE Transactions on Visualization and Computer Graphics*, 15(6):1169–1176. 8