A Visual Analytic Framework for Exploring Relationships in Textual Contents of Digital Forensics Evidence

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

Mississippi State University and University of Alabama

ABSTRACT

We describe the development of a set of tools for analyzing the textual contents of digital forensic evidence for the purpose of enhancing an investigator's ability to discover information quickly and efficiently. By examining the textual contents of files and unallocated space, relationships between sets of files and clusters can be formed based on the information that they contain. Using the information gathered from the evidence through the analysis tool, the visualization tool can be used to search through the evidence in an organized and efficient manner. The visualization depicts both the frequency of relevant terms and their location on disk. We also discuss a task analysis with forensics officers to motivate the design.

Index Terms: I.3.3 [Computer Graphics]: Picture/Image Generation; I.3.8 [Computer Graphics]: Applications—Visualization; K.6.m [Management of Computing and Information Systems]: Miscellaneous—Security

1 INTRODUCTION

Computer forensic investigation is still a relatively young field, and in its youth lies a lack of sophisticated evidence analysis methods. The difficulty in analyzing forensic evidence is that it is hard to know what kind of information to expect on digital media, so coming up with a trusted model of automated analysis is theoretically complex. As a result, most forensics tools just present information about files, the filesystem in which the files are contained, and other aspects of the digital media so that the investigator has to do all of the analysis him or herself. Since there could be a very large amount of information stored on a digital storage device, it could take hours or days to find all relevant information that could be used to support a computer crime case. The rapid increase of computer-based crime [4, 16, 23] combined with the difficulty to prosecute such crimes [14, 22] presents several research challenges.

The most common process forensic investigators use to locate information is to load up images of digital media into a program such as EnCase [1], AccessData's Forensic Toolkit (FTK) [2], or the Autopsy Forensic Browser [6, 8] and then use the tools in them to search for useful information, either by manually looking through the directory hierarchy, registry, web browser history and cache, and other common locations where evidence could be found. It is important to note that copies of the original hard drive (disk images) are used exclusively in this analysis to preserve the integrity of the original evidence. Anything of interest that officers find searching the image is added to their virtual evidence set which will then be condensed into a report of their findings. Using this process requires an investigator to be very familiar with the areas of digital media in which useful information might be stored, and also that they meticulously investigate all of these areas to ensure that nothing has been left out.

While this process has been used successfully for forensic investigators all over the world, a certain level of automated pre-analysis would likely be a welcome addition to the investigator's toolkit. Currently the most common form of automated pre-analysis is for a keyword index to be generated from all of the textual contents of the digital media so that it can be searched using a relatively naive method. In FTK, only individual words can be searched for at any given time, but conjunctive searches can be built out of multiple single-word searches. This allows the investigator to locate files containing all of the terms together, but the terms may or may not reside contiguously in the file. This makes it more difficult to find contextually-bound search terms such as 'investment fraud' opposed to just 'fraud.'

The intention of this project is to expand upon this simple method by allowing an investigator to see all of the files and clusters in which a set of words all exist, and also to allow them to easily find information that they may not have been looking for by showing them the most prevalent words found in the evidence. By visually depicting the importance of words and their relation to other information in the evidence, the investigator will be able to find new potentially useful information with much less effort expended on their part.

2 RELATED WORK

Commercial and open source tools currently provide a text-based platform for forensics officers. It is our premise that visualization will augment these tools to improve forensic analysis. Our focus is in the realm of hard disk forensics. There a few initial efforts in this area. Teerlink and Erbacher's work [20, 21] present hard disk data two ways. The first uses a matrix of squares, one for each file, each luminance-encoded based upon a chosen metadata attribute (e.g., last modified date or size); their second display uses a treemap [3, 19] to show the same information within its hierarchical context (not all files are shown in this case, and the color scheme is altered). While this method does assist in locating files with unusual metadata characteristics, the issue we are investigating requires inspection of the file's contents. For looking at file contents, Schwartz and Liebrock [18] provide a histogram-like version of Tile-Bars [12] for finding the distribution of search strings across a disk image. Their visualization is based upon *a priori* strings (i.e., ones provided by the user), and is thus less beneficial when searching for evidence beyond that initial set. In addition, the locations of the files are only given as strings. The visualization proposed here provides both the context of the searched strings on the disk while facilitating discovery of terms of interest.

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

6th International Workshop on Visualization for Cyber Security 2009
11 October, Atlantic City, New Jersey, USA
978-1-4244-5415-0/09/$25.00 ©2009 IEEE

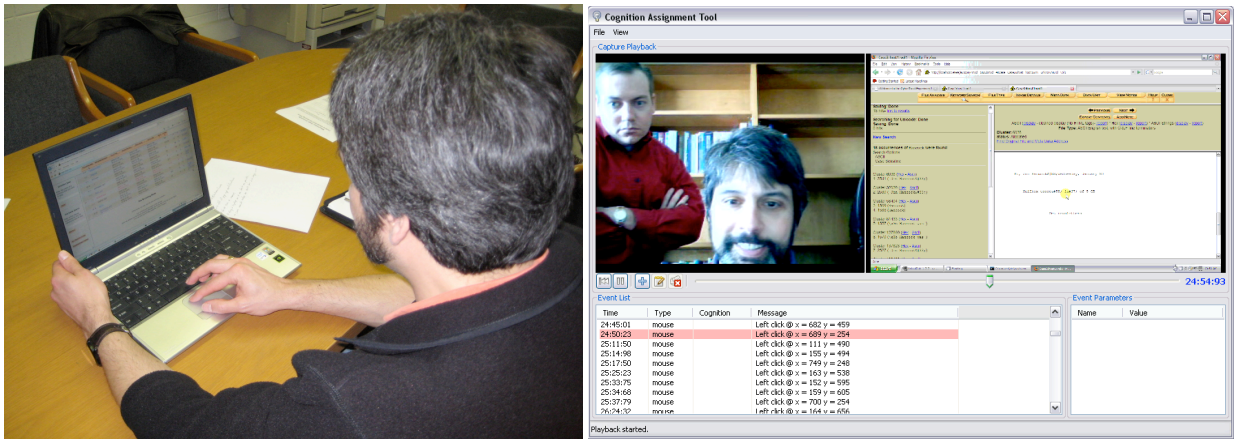

Figure 1: Experimental setup for our task analysis. *Left*: Experimental rig with keyboard, mouse, and video capture. *Right*: Analysis tool we developed to process expert trials. Webcam input is in the upper-left, screen capture video in the upper right, mouse events on the lower left, and observer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

3 Visualization and System Design

3.1 Pre-Design Task Analysis

To provide a basis for the design of our textual-relationship visual analytics tool, we first performed a contextual analysis [11,13,17] with three forensics detectives to understand how they utilize tools to find textual data; we reported the design of this study previously [15]. Officers were given a laptop with the Autopsy forensics software and our two email investment fraud test cases and asked to "do what they do normally" in their analysis (Figure 1 left). Interactions with the system were recorded and they were allowed to take notes and encouraged to verbalize their process for the observer. We then used custom software we developed to analyze the video, screen-capture, and input logs in a central interface to perform our analysis (Figure 1 right).

Although three subjects does not produce enough power to do traditional hypothesis testing, we saw very definite trends to guide us in our design. For coding the user's interactions, we categorized user actions as either Selection (e.g., menu selection, scrolling, and navigation), Manipulation (e.g., changing the type of view (hex vs. ASCII) and examining metadata), Search (e.g., textual search using the interface), or Note (e.g., analog or digital note taking); Search and Note were further broken down by what was being searched for (Name, Email, or Other for Search; Copying, Linking data, or Other for Note). In all three cases, Selection predominated with two to four times as many events; this is especially true for sessions where little evidence bearing data was found. When evidence was found, searching and manipulation had similar frequency. Searching was spread amongst name and email address searches when evidence was found.

Based upon our observations, we decided to focus on visualizations that facilitate drawing the user to the effective searches. 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 depicts three major pieces of information:

- A search-sensitive file hierarchy (Figure 2a).
- A tag cloud of terms in the selected files (Figure 2b).

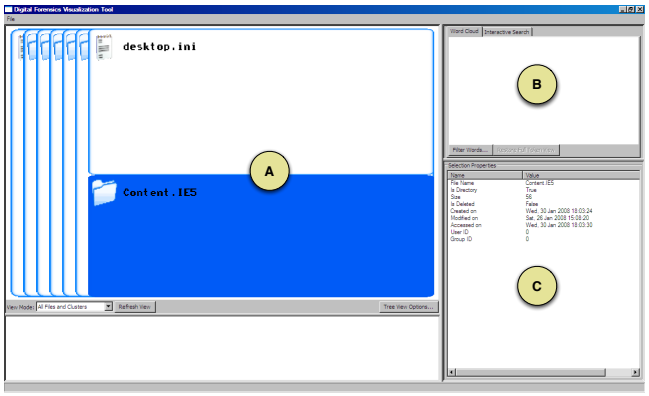

Figure 2: Our textual contents forensics visualization system. It consists of a treemap-like depiction of the hierarchy (a), a tag-cloud of terms in selected files (b, currently empty), and metadata about the selection. The contextual search and cluster views are not shown.

- Contextual information about the search (Figure 2c).

In additional, 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 provides the context for a search. It uses a modified squarified treemap [5]; node size is based upon the number of word occurrences in a file by pass the user's initial filters; thus, no image or primarily-binary files are included in the view. We modified the treemap two ways. First, we distinguish between file and directory using icons similar to those found on major operating systems. 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 hierarchy (a file) and one with children (a directory). Secondly, the left-side 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

Figure 3: Tag cloud for a selected webmail cache file. Selecting terms here will highlight files containing the term.

(as opposed to the containing view) and has some familiarity (such as the left-to-right opening hierarchical displays used in OS X).

The hierarchy view is "search-sensitive" since the location of terms selected in the tag cloud or conjunctive search are indicated by highlights in the tree-view. The user can choose to update the node sizes to reflect selected tag cloud terms or searched terms; this facilitates finding 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 methods which 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 is generated based upon the parsed terms (Figure 3, see System Infrastructure for term generation details). The size of a word in the display is based upon its frequency; larger terms occur more often in the file. We use a quadratic falloff to determine word size; the area of the word decreases linearly with smaller frequency. 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.

Since text on a disk originates from different file types, we have additional filters that a user can apply. For example, the example in Figure 3 is from a browser cache of webmail browsing. Such files contain significant textual "noise" such as HTML tags or custom markup for the site. Thus, we provide filters that can be enabled for common data: HTML, email, etc. In addition, users can add additional filters for specific words manually or by selecting a term in the tag cloud. The user can also specify the maximum numbers of terms to show, specify a minimum frequency of occurrence between a word is shown, and specify a minimum and maximum word length for filtering. The filtering is propagated to all other views.

Contextual View The contextual view displays metadata about the selected file or directory. This includes ownership, permissions, file/directory size, its creation and modification 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 Sharing the same space as the tag cloud view, the contextual search view is used to search for a specific set of terms (Figure 4). Starting with a given term, such as "money", the list view populates with phrases that start or end with that term. Selecting one of these terms will populate the list with the next set of phrases starting/ending with the same two words and so on. The user can then bring up a list of which files that term is found in and highlight them within the hierarchy view. This phrase-based search supports finding specific mixed terms such as "investment fraud" that pure conjunctive-based search would group with non-phrases.

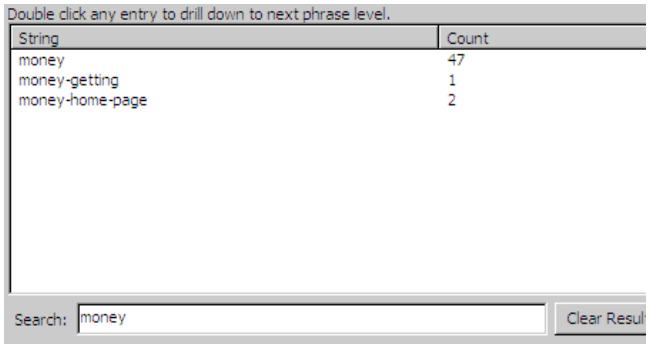

Figure 4: Our contextual search interface. As a word is entered, phrases that begin or end with that term are shown.

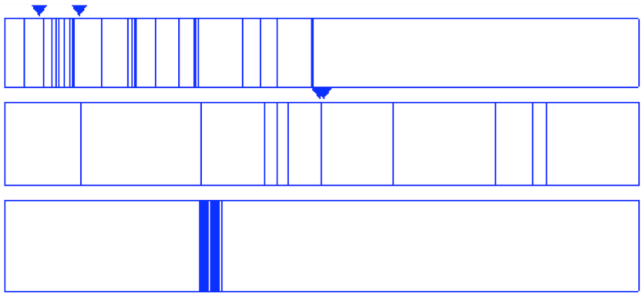

Figure 5: Cluster view of the selected terms. The top layer shows the entire disk, the middle layer a selected subregion (between the triangles), and the bottom a cluster-by-cluster depiction. Clusters with the search term are highlighted blue.

Cluster View Our final view 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 5). 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 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 panes selection is determined, which depicts a one-to-one vertical line to pixel rendering of the center of the selected middle zoom region. The middle zoom region can also be clicked to move around the lowest-level zoom view for more precise investigation. Triangular glyphs above both of the upper-most views delineate where the zoom regions in the following zoom level are coming from. The location and other metadata for the selected cluster is displayed in the contextual view.

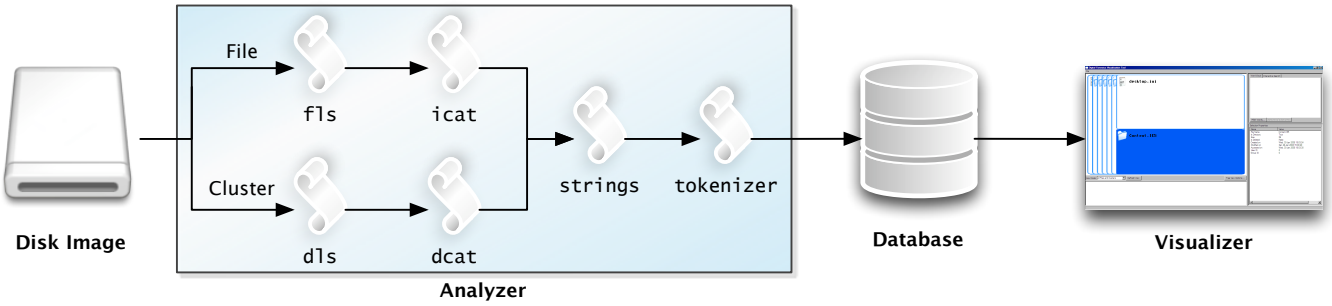

Figure 6: Workflow of our forensics visual analytics system. The Analyzer processes the text from the disk image and stores it in a database that the Visualizer depicts.

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 6 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 [7, 9] 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; walking over this structure, the contents of the file on the image is extracted via `icat` and the textual content of this stream is distilled via the UNIX `strings` utility. A similar process is used for unallocated clusters: Clusters are enumerated via `dls`, their content extracted via `dcat`, and their text distilled via `strings`. The strings are then processed by our tokenizer, which separates the lines into tokens, identifies the tokens as a word, number, US currency, URL, email addresses, or unreadable symbols and stores these in our SQLite database. For each token, the next and previous token is also referenced for contextual text search, and the corresponding file and cluster is stored with metadata for said file and cluster also recorded for the visualization. The current implementation requires roughly 20 hours to index all the clusters on a 4.5GB disk; for a subsection of interest (such a 4.5MB web cache directories), it takes about 20 minutes. This is a one time process, but we are examining means to accelerate its performance.

The Visualizer, written in Python utilizing the wxPython cross-platform interface library and OpenGL for hardware accelerated rendering, extracts information from the database using the Elixir ORM. 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 with no noticeable lag in hierarchy navigation or contextual searches. Extracting and rendering the tag cloud is the most costly operation, requiring roughly 5 seconds for displaying 1000 items; this lag only occurs the first time the cloud is calculated. Filtering reduces this time, and it is unlikely that an analyst will need to see or be able to make sense of that many tokens in a single tag cloud.

4 Case Study: Email Investment Fraud

To demonstrate our visual analysis framework, we provide a small case study. For our task analysis, we generated an investment fraud cases where a fictitious criminal William Slick utilized the email distribution services of an intermediary "abacus55" to commit the 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. The 4.5GB Analyzer-processed disk image of this information was then provided for analysis. All the participants knew of the case was that fraud of some case was suspected and that it was committed via the internet; the names and specifics were not detailed. 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, a search for fraud related terms is the first course of action. An initial search for "money" turns up hits in several files, most of them in the web cache directory (Figure 7). The contextual search also shows two right-continuation phrases, "money-getting" (one hit) and "money-home-page" (two hits). "Money-getting" seems promising; selecting it takes us to the file containing it. The tag cloud for the file contains a mix of HTML codes for the web-mail page and text from the rest of the cached messages (Figure 3). 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 13 times in the selected file (Figure 8).

Given that fraud is the suspected crime, other terms related to fraud can be searched. For example, a search for "investment" turns up one hit. Due to its small number, it 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 that 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 9). We note (from the metadata view on the lower right) that this is the only occurrence of "investment" on the disk. 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 suspected is clearly identified: `willieslick@hotmail.com` (Figure 10). 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 Conclusions and Future Work

We have described a visual analytics tool for finding relationships amongst text terms on a disk image. The Analyzer component finds term occurrence, frequency, and contextual relationships both in files and unallocated clusters while the Visualizer depicts the relationships. We use a modified treemap display for the hierarchy and provide several interactions with a tag cloud display for drilling down to terms of interest; these combine with other functionality to find related terms. We have demonstrated the efficacy of our contributed framework via a case study and motivated its design via our task analysis with forensic practitioners.

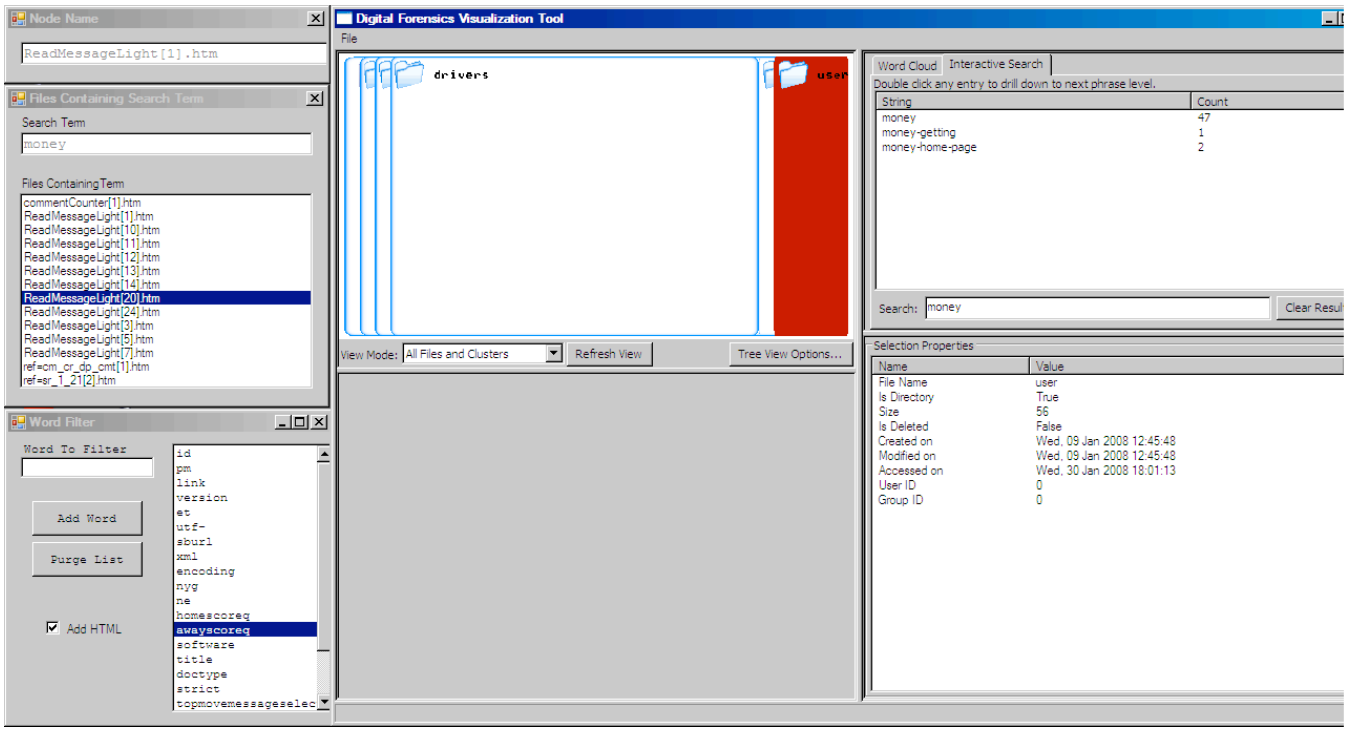

Figure 7: Initial search for the term “money” on the disk The list on the left enumerates the occurrences of the term.

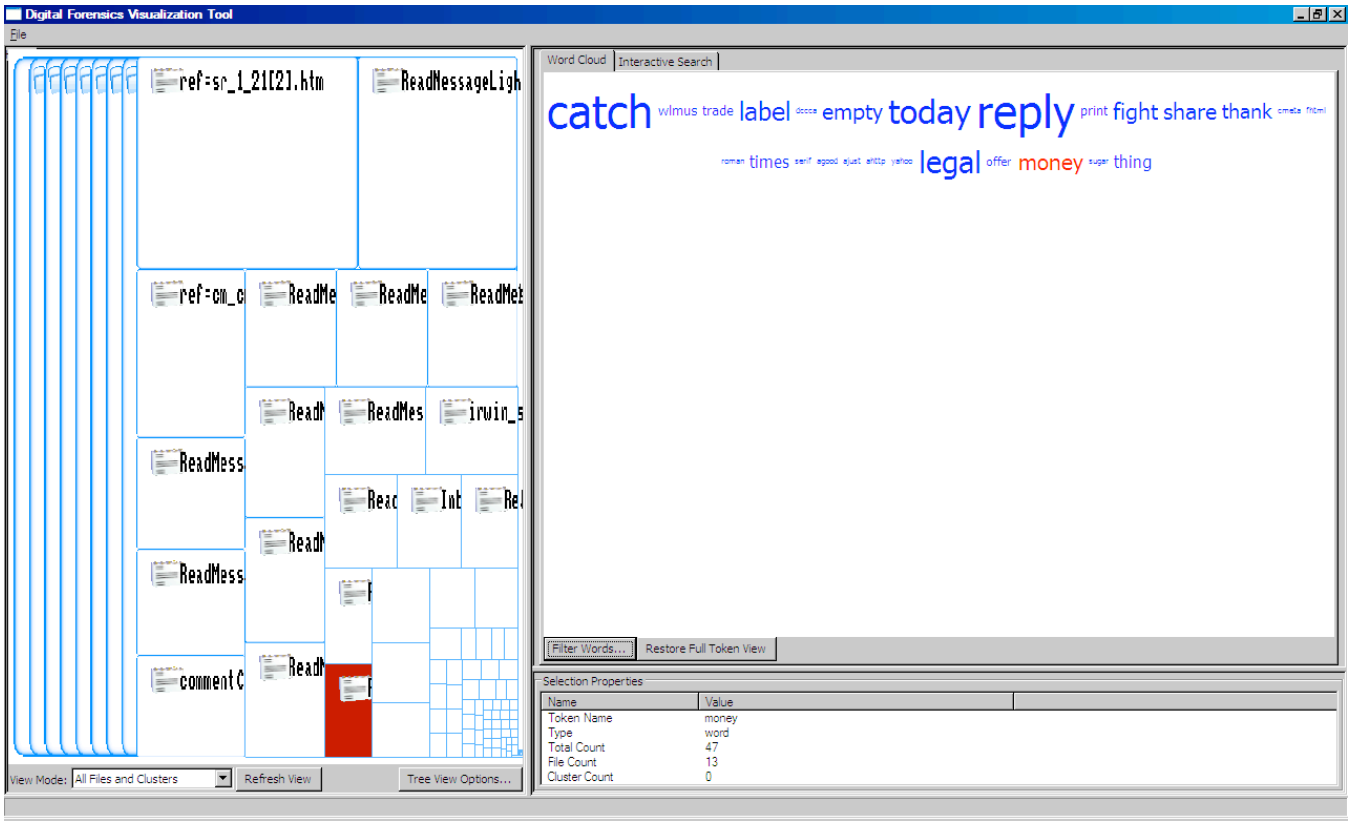

Figure 8: Filtered tag cloud for one of the selected values for the term “money.”

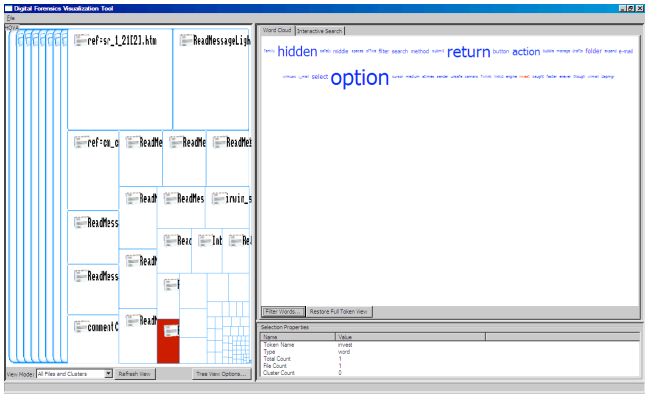

Figure 9: Filtered tag cloud for one of the selected values for the term "invest" in the same file as the occurrence of "money."

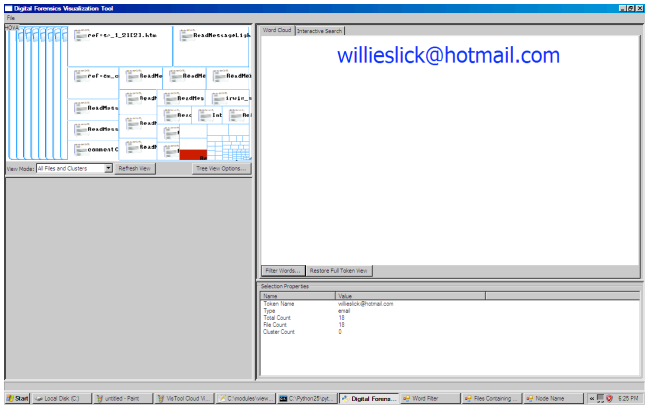

Figure 10: Filtered tag cloud for selecting only email addresses; the address matches our suspect.

For the future, there are three primary avenues of research. First, as this project is part of a larger Analyze-Visualization-Validate research framework; we are pursing validation of the effectiveness of the visualization. While we have confidence from our case studies and domain analysis that the visualization benefits forensic analysis, we do not have any quantitative evaluation of this belief. We are currently working with the Mississippi State Forensics Training Center for this task. Second, in terms of the technical details, we are investigating ways to speed up the caching and display of search terms. While our current solution works, we believe there is still more we can do to eek out performance. We also have plans to speed up the initial Analyzer phase; one idea is to use FPGAs to generate the terms while the disk is being imaged as was done for image file search [10]. Finally, there are elements of the visual display we wish to improve. The unallocated cluster visualization needs more refinement; the scale of the entire disk requires something more than our three-stage depiction. In addition, it is worth pursuing how additional metadata, such as those shown by Teerlink and Erbacher's work [20, 21], can be integrated meaningfully into the display.

Acknowledgements

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

References

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

[2] AccessData. Forensic toolkit 2.0. `http://www.accessdata.com/forensictoolkit.html`. Last checked May 2009.

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

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

[5] M. Bruls, K. Huizing, and J. J. van Wijk. Squarified treemaps. In *Proceedings of the Joint Eurographics/IEEE TVCG Symposium on Visualization 2000*, pages 33–42, 2000.

[6] B. Carrier. Autopsy forensic browser. `http://www.sleuthkit.org/autopsy/`. Last checked May 2009.

[7] B. Carrier. The Sleuth Kit. `http://sleuthkit.org/sleuthkit/`. Last checked May 2009.

[8] B. Carrier. *Know Your Enemy*, chapter Ch. 11: Computer Forensics Basics. Addison Wesley, 2nd edition, 2004.

[9] B. Carrier. *File System Forensic Analysis*. Addison Wesley, 2005.

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

[11] J. T. Hackos and J. C. Redish. *User and Task Analysis for Interface Design*. John Wiley & Sons, Inc., New York, 1998.

[12] M. A. Hearst. Tilebars: Visualization of term distribution information in full text information access. In *CHI*, pages 59–66, 1995.

[13] D. Hix and H. R. Hartson. *Developing User Interfaces: Ensuring Usability through Product & Process*. John Wiley & Sons, Inc., New York, 1993.

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

[15] T. J. Jankun-Kelly, J. Franck, D. Wilson, J. Carver, D. Dampier, and J. E. Swan II. Show me how you see: Lessons learned from studying computer forensics experts for visualization. In J. Goodall, G. Conti, and K.-L. Ma, editors, *Proceedings of the Fifth International Workshop on Visualization for Computer Security*, volume 5210 of *Lecture Notes in Computer Science*, pages 80–86. Springer, September 2008.

[16] G. Kessler and M. Schirling. Computer forensics: Cracking the books, cracking the case. *Information Security*, pages 68–81, 2002.

[17] D. Mayhew. *The Usability Engineering Lifecycle: a Practitioner's Handbook for User Interface Design*. Morgan Kaufmann Publishers, San Francisco, 1999.

[18] M. Schwartz and L. M. Liebrock. A term distribution visualization approach to digital forensic string search. In J. R. Goodall, G. J. Conti, and K.-L. Ma, editors, *Proceedings of the Fifth International Workshop on Visualization for Computer Security*, volume 5210 of *Lecture Notes in Computer Science*, pages 36–43. Springer, 2008.

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

[20] S. Teelink and R. F. Erbacher. Foundations for visual forensic analysis. In *Processings of the 7th IEEE Workshop on Information Assurance*, pages 192–196, 2006.

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

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

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