System for the Proactive, Continuous, and Efficient Collection of Digital Forensic Evidence

By

Clay Shields, Ophir Frieder and Mark Maloof

From the proceedings of

The Digital Forensic Research Conference

DFRWS 2011 USA

New Orleans, LA (Aug 1st - 3rd)

DIGITAL INVESTIGATION 8 (2011) S3–S13

available at www.sciencedirect.com

ScienceDirect

journal homepage: www.elsevier.com/locate/diin

Digital Investigation

A system for the proactive, continuous, and efficient collection of digital forensic evidence

Clay Shields, Ophir Frieder, Mark Maloof*

Department of Computer Science, Georgetown University, Washington, D.C. 20057, USA

Keywords:
Proactive forensics
Evidence collection
Deleted file logging
Search
File similarity

ABSTRACT

The historical focus of forensics research and tools on digital systems that are seized from a suspect misses the fact that in centrally controlled networks it is possible to proactively and continuously collect evidence in advance of any known need. We present a proof-of-concept for PROOFS, the first proposed continuous forensic evidence collection system that applies information retrieval techniques to file system forensics. PROOFS creates and stores *signatures* for files that are deleted, edited, or copied within such a network. The heart of each signature is one or more *fingerprints*, generated based on statistical properties of file contents, maintaining semantics while requiring as little as 1.06% of the storage space of the original file. We focus on text documents and show that PROOFS has a high precision of 0.96 and recall of 0.85 with stored fingerprint sizes of less than 375 bytes. The two contributions of this work are that we show that common environments exist where proactive collection of forensic evidence is possible and that we demonstrate an efficient and accurate mechanism for collecting evidence in those environments.

1. Introduction

Digital forensic investigations have historically been a reactive measure taken by law enforcement to confirm or refute a hypothesis about what actions a user performed in the past. Officers physically seize relevant systems and image them to ensure preservation of criminal evidence. They later manually perform an analysis to recover files or file fragments that can support or refute the investigative premise. Current operating systems do not purposefully collect and preserve information specifically to assist in an investigation. Instead, forensic tools examine recovered files and fragments that exist only as a side effect of normal system activity. The existence of such data is dependent on non-deterministic system and user activity — data that could answer a question may never even be created; if they are, they may be overwritten or otherwise lost.

As forensic examinations become more common on internal networks and the investigative environment changes, the opportunity arises for *proactive* collection of forensic data surrounding questions of violation of organizational policy, such as IP theft, misuse, or embezzlement. Focusing on internal networks changes the forensic paradigm as investigators now have access to systems prior to the time they are imaged for investigation; there are currently few tools and little research addressing this shift from a distributed manual task to a centralized, network one. An organization, be it corporate or governmental, can install software on its own systems that continuously and securely preserves forensic data against future need. This improves investigative

* *Corresponding author*.
E-mail addresses: clay@cs.georgetown.edu (C. Shields), ophir@ir.cs.georgetown.edu (O. Frieder), maloof@cs.georgetown.edu (M. Maloof).
1742-2876/$ – see front matter © 2011 Shields, Frieder & Maloof. Published by Elsevier Ltd. All rights reserved.
doi:10.1016/j.diin.2011.05.002

accuracy in two ways: firstly, it ensures that some relevant evidence is generated and preserved; and secondly, it allows storage of that data at a central point in the network, reducing examination complexity. While there is a cost to maintaining these forensic data, they can reduce overall costs by providing a readily available source of evidence to help direct the use of expensive examiner time. United States law states that employees have little to no expectation of privacy on organizational equipment, making such collection legal (US v. Ziegler, 2007).

We present a proof-of-concept for a system named Proactive Object Fingerprinting and Storage (PROOFS). PROOFS is the first system for proactively collecting evidence that is designed to answer questions about user actions; it is intended to be the Google of network investigations. PROOFS continuously creates and stores *object signatures* for digital objects, such as files or email messages, on systems in a centrally controlled network. Each object signature contains metadata about signature creation and the object as well as one or more *fingerprints* that serve to accurately identify the object while being convenient to store and search. Fingerprints are created using a novel application of information retrieval techniques. They are able to match objects across extensive edits and are searchable by keyword in the case of textual objects. In this initial proof-of-concept, we focus on textual fingerprints, but relevant techniques exist for audio, video, and executables among other file types.

There are two main contributions in this work. First, we show that environments exist where proactive collection of forensic evidence is possible. Though proactive collection may seem a natural progression, it is a major shift from the current forensic model in which systems are not available to the investigator until seized; this shift opens a new area for research and tool development. Second, we demonstrate an efficient and accurate mechanism for collecting evidence in those environments and show how it assists common investigations.

2. Overview

In PROOFS, each computer in the network is modified to create an object signature when a file is: closed after writing; deleted; or copied across file system boundaries. Additionally, entry and exit points to the network, such as an email server or proxy server, are modified to produce signatures when an object transits the domain border. The signatures are securely written to a centralized forensic database and remain even when the object used to create them is deleted. This allows investigation based on file contents even for files that are no longer anywhere on the network.

When an investigation is needed, the PROOFS forensic database becomes a valuable resource that can support a variety of investigations, and the evidence is admissible in the same way other log file entries are. This database does for network investigations what Google does for web pages. It provides a central, searchable resource that is used to guide the actions of examiners, both providing additional evidence and allowing focus on specific systems most likely to yield evidence though further manual examination.

Given even a fragment of a file or other digital object, PROOFS can identify with high accuracy any system on the network that ever held a copy of any version of that object. It can identify who created a document and when and where they did so. It can help identify which user copied or transmitted the object, when, and to where. Given a set of keywords, PROOFS can identify which systems in the network ever held documents that matched those keywords, and can rank users and systems in order of their frequency of use of those terms. These activities help direct forensic examiners as to where to look more deeply, making better use of their expensive skills.

The cost and accuracy of using PROOFS is related to the size of the signatures and how accurately they are able to match the related objects. In this first work, we focus on text documents and show that PROOFS can return 96% of relevant documents and that 85% of the documents returned are matches even when stored fingerprint sizes are fewer than 375 bytes in length. We show that PROOFS is accurate in matching documents that are edited and contain changes by using an automated document summarization tool to reduce documents in word count before creating fingerprints; accuracy remains high even when copies retain as few as 40% of their original words. We further show that source documents can be matched to fragments that have had as much as 50% of content deleted before fingerprinting. Our future work includes developing fingerprints for non-text documents, including audio, video, and executables.

3. Background

Digital forensic tools were developed in response to an organic need. Investigators in both civil and criminal cases, as well as from intelligence agencies, required tools to retrieve information from seized computers to help resolve hypotheses about the past actions and future intentions of the computers' users. Current forensics tools are very good at retrieving a wide variety of artifacts that can reflect user actions: existing files; deleted files, including temporary backup application files, print spool files, and web and mail cache files; and operating systems artifacts such as registry entries and file system metadata (Boyd and Forster, 2004; Buchholz and Spafford, 2004; Carrier, 2005).

It is important to note that files are copied very frequently in normal system operation, and that often there are multiple copies not usually visible to the user. For example, a Word document that is edited, emailed, and opened within the network will generate signatures multiple times while being edited as copies are saved in the background for crash recovery; when Word is closed the file will be fingerprinted; any temporary copies made by the sending mail client will be fingerprinted; it will be fingerprinted at the mail server; fingerprinted upon delivery to the receiving client; and fingerprinted when closed if it is opened in an editable mode for reading. New examiners are often surprised at the number of copies left of many files.

However, the existence of data that are not stored as part of the file system is subject to non-deterministic system activity that can cause it to be lost. For example, operating system

updates occur at unpredictable intervals and overwrite previously deleted files, making their data unavailable to future examination. Many procedural steps in forensic response are designed to minimize this type of loss, from ceasing system use when the need for an investigation becomes apparent to using write blockers when examining media. Because forensic tools were developed to deal with impounded systems, they can only be reactive. It would be absurd to expect that subjects of external investigations would install tools in advance to assist an investigation that might occur at some point in the future; *we expect that individuals would never do so.*

As digital forensics has evolved as a discipline, forensic examinations of computers other than those seized from a subject have become increasingly common. Organizations that administer their own computers and network perform forensic examinations on their own systems in response to legal demands, reported policy violations, or intrusion detection alerts (Luoma, 2006). In fact, insider attacks are considered the most frequent and damaging type of attack (Keeney et al., 2005). In these cases, the organization has the control of a system long before the event that initiates an investigation and can install proactive forensic collection tools to continuously collect data to support future investigations. Because it is organizational equipment, such collection is legal in the United States.

With proactive data collection loss of data can be prevented. We show that it is possible to capture and maintain a large amount of useful and relevant information very cheaply. Storing forensic data at a centralized point also speeds investigations and thus makes them cheaper. Trained digital forensic examiners are expensive to hire, with rates well over $100 per hour in the United States. The cost of collecting and storing data is very low compared to the hourly cost that would be incurred hiring examiners to recover data after the fact, particularly for investigations that span multiple computers across a network.

4. Operating environment and security analysis

A proactive approach is most effective when the user of the computer who is a potential subject does not have administrative access to the system. While this environment is alien to many academics, it is very common in organizational settings where an IT department provides and administers systems for users. To best describe our expected environment we posit a *subject model* that is analogous to an attacker model. We assume:

- The subject does not administer the system they use.
- The system has software installed the user cannot stop or modify.
- The system does not include software that can thwart data collection.
- The subject cannot install their own software.
- If encryption is needed, full disk encryption with organizational key escrow can be used. PROOFS can then still fingerprint individual files.
- Data collected is not accessible by the user, as a result of being cryptographically signed and sent to a central database. When off-network, the data is stored inaccessibly to the user until connectivity is regained.

4.1. *Security analysis*

The goal of PROOFS is not to provide complete forensic information for every possible security violation. In fact, we do not expect PROOFS logs to withstand compromise of operating system security breaches one might expect in an outside intrusion. Instead, we focus primarily on violations of organization security policies that currently are difficult to enforce through existing computer security policy. Because PROOFS is dependent on existing operating security mechanisms for collection and transmission of data, compromise of the operating system means that any data collected after privilege violation is at best suspect, and at worst purposefully misleading — as is true with other logging mechanisms. We therefore do not expect to gather full data on adversarial users, and in some cases, such as investigations regarding malicious system administrator, may not gather any information. These cases will have to be dealt with by manual examination, which is the current state of the art.

The bulk of organizational forensic investigations occur because of violations of organizational policy. This includes cases such as IP theft (Bumiller, 2010; Neuman, 2010); violation of acceptable use policies through misuse such as file sharing, harassment, or visits to sexually-oriented sites; external requests to preserve information for pending legal action; and loss of equipment such as laptops or mobile devices. These cases require only insider access, and do not require the user to be sophisticated enough to defeat system protections; in fact, the majority of the computer-using population does not have the technical expertise to do so. Attempting to avoid use of any of the dictionary terms is difficult, as they are selected from the common vernacular of the organization and span thousands of terms. Additionally, the subject cannot control what documents are delivered to their computer and indexed, so avoiding terms in downloaded documents may be impossible. Even where users do have the knowledge to attack the system, strong disincentives exist against doing so. Well-protected systems include anti-virus programs that frequently identify attack tools, and users who are caught attempting to violate security risk not only being fired, but face potential civil and criminal penalties. We therefore argue that for common investigations, PROOFS will be active and will gather useful evidence.

For sophisticated insiders and outside attackers who breach system security, data gathered before privilege escalation might still be useful once the means and time of compromise are discovered via a manual forensic examination. Files that were seen to be part of the attack and that were transferred into the network or copied from removable media proceeding the attack and indexed by PROOFS can provide clues as to how the escalation attack occurred, and can help identify other systems that contained the same file.

5. Proactive forensic data collection using PROOFS

We propose a novel, proactive forensic evidentiary collection system called *Proactive Object Fingerprinting and Storage* (PROOFS). The heart of PROOFS is the ability to create *signatures* that contain information about a digital object that was created, edited, or deleted anywhere on the internal network. PROOFS stores and uses these signatures as a new and robust source of forensic information. They contain a variety of metadata about the event that caused the signature to be created and about the object of interest. Most importantly, they contain one or more *fingerprints* of the object that is created based on its statistical properties. Fingerprints are designed to be small to store but robust enough to match objects that have some degree of difference between them. While it is possible to create fingerprints for a wide variety of file formats, such as video files, audio files, and still images as discussed in Section 10, we focus on documents that contain text. Text fingerprints are a form of semantics-preserving lossy compression that represents an inverted index into the dictionary used in their creation. They are created using information retrieval techniques (Grossman and Frieder, 2004) as follows.

Dictionary creation First, a dictionary of statistically significant terms is created using a training set of existing documents taken from the system where PROOFS is to be deployed. These statistics would be gathered from files on a shared network drive; from files from backup systems; and from text-containing files on individual systems.

Terms in the dictionary are selected on the basis of their *inverse document frequency*, or IDF, which is the log of the total number of documents divided by the number of documents containing the term. Terms with low IDFs are too common to be useful in distinguishing documents; terms with very high IDFs tend to be too specific to differentiate between a set of documents. We refer to *normalized IDFs* which have been normalized to a range of 0–1 based on the highest IDF seen in a collection.

Determination of appropriate IDF settings can be made by experimentation. The sample files are compared to locate duplicates using an accurate duplicate detection algorithm. Next, a series of dictionaries and fingerprints are created based on different IDFs and used to find duplicate files. The settings that provide the most accurate results in sufficiently small storage space can be chosen. The resulting dictionary is shared across all systems that are to be fingerprinted.

Over time the best terms to store in the dictionary might change as new topics become important and old ones fall out of use. There are two ways to deal with this. The first is that important new terms can be appended to the end of the dictionary. When comparing fingerprints of differing lengths, the older fingerprints can be padded out with zero bits, preserving backward compatibility. As an alternative, the dictionary can be periodically reconstructed with a more current document set and redistributed across the network to all fingerprint creating clients. This allows dictionaries to avoid repeated growth, but requires storing multiple dictionary versions for future use in interpreting old signatures.

Fingerprint creation Once the dictionary has been created, text fingerprints are created. Each document is tokenized to extract tokens of interest. For files that contain text but have a proprietary file format, such as Microsoft Word or PDF, a technology like Oracle's Outside In can be used to extract the text. The text is next processed to remove symbols and numbers and to remove common stop words. Text can also be stemmed to find word roots, and tokens of only a particular length kept. The type of processing done depends on the type of document being processed and what the tokens of interest are expected to be. This set of processed tokens is used to create the fingerprint. PROOFS creates a bit-vector fingerprint in which each bit represents the presence or absence of a term from the dictionary in the document. These fingerprints are of a fixed length equal in bits to the number of terms in the dictionary. The bit-vector fingerprints are very sparse and are stored after compression with gzip which reduces their size by an average of 83%. It is possible to trade storage space for faster matching of fingerprints but here we optimize for minimum storage requirements. We show that storage of compressed fingerprints for real-world data can be less than 375 bytes long.

Keyword searching Because the bit-vector fingerprint is an inverted index into the dictionary used to produce it, keyword searches are simple. An investigator can extract the terms used in the dictionary and select relevant terms for a boolean search overall fingerprints. Due to the simplicity of this operation, we do not discuss it in detail.

Fingerprint matching Fingerprints can be matched against each other. For bit-vector fingerprints we use a cosine similarity measure, which is the measure of the cosine of the angle between two vectors represented in n space. Our experiments specify a parameter that represents the maximum cosine of the angle between the two vectors that can be considered a match. In PROOFS it is possible for an investigator to control the precision and recall of their searches to a degree by varying this parameter. Increasing it results in higher precision but a lower recall. An investigator can use this fact to perform initial, highly precise searches of the PROOFS database, then expand the scope of the searches by decreasing the matching parameter. These later searches will return a wider variety of matches which are less accurate but contain more possibilities.

Signature creation and size One or more fingerprints are combined with metadata to create a signature. The metadata varies depending on what event caused the signatures creation. For a file edit or file deletion, we would expect the signature to contain the fingerprint, user identifier, file name, file path, a time stamp for the event, a machine identifier (such as a name, IP address, or MAC address), a code for the event that caused the signature, and for copies, the source path and source file name. For email, the signature could contain the fingerprint, the source and destination addresses and cc or bcc addresses, a time stamp, and pointers to any fingerprints created for attachments.

The size of signatures is important to the overall storage cost of the system. We believe that signatures are best stored in a centralized relational database for easy searching. Our estimates of the signature size reflect database storage rather than a naive approach where all text is stored. For the largest

fingerprints we used, which were 375 bytes, the total signature size easily can be 1024 bytes. Many of the fields in the signature would be used so commonly that separate database tables would be used to store them. The signature would therefore consist of the fingerprint, a fingerprint creation time stamp, and a series of fields that were keys indexing other tables. Assuming that each entry in the database is 512 bytes and that we are using 64-bit integers for identifiers and time stamps, the signature field would contain the 8-byte ID for the field in the table, the 375-byte fingerprint, an 8-byte time stamp, and up to 15 other keys to other tables. The amount of other data written into other tables is dependent on the overlap in information between signatures.

System overhead We have showed elsewhere that the storage and CPU overhead for PROOFS are low (Shields et al., 2010). Traces for a campus email server with 8700 active users show that signature storage would require 202 GB per year, or 1 cent per user. Other systems require similarly low storage. CPU overhead, as simulated using file system traces, is also low.

5.1. *Required modifications*

The first step in collecting forensic data is to instrument local systems to create fingerprints and signatures, described above, in response to system activity. We believe that signatures should be collected when an object is modified, deleted, or copied. Modifying the operating system to perform these actions is not a major undertaking, as it involves only patching the close system call for files that were opened for writing and the unlink system call for files being deleted. Hooks into the kernel like these already exist in operating systems that index changed files on a file system, and it is within the capability of an organization that provides and administers its own systems to implement.

We do not expect that these simple modifications will result in capture of every potential piece of evidence, particularly since we are not attempting to capture information from memory, and because we only index files when closed correctly. A user could potentially thwart collection by keeping information only in memory, or by closing files by power cycling the computer. We believe that this would make the computer less usable, and would not be of significant help in hiding evidence of the organizational policy violations discussed in Section 4.1.

Systems beyond individual desktop computers and file servers require other modifications. Email servers would create signatures for incoming and outgoing email messages. Any attachments to emails would fingerprinted separately and linked to the parent email. A proxy server would host a web cache and create signatures for objects accessed from within the network.

A central database server is used to store the signatures that are created by the other network components. It is possible to estimate the required storage space by examining the workloads of local systems, and we expect that 10 GB per desktop system per year and 1 TB per email server per year would be a reasonable initial size as shown in Shields et al. (2010). In cases in which a system cannot reach the database, such as an off-network laptop, the signatures can be written to a file system partition the user cannot access, or can be written using non-tamperable logging (Schneier and Kelsey, 1999). In either case, the signatures should be uploaded when the connection becomes available.

5.2. *Improved forensic investigative activities*

The signatures we propose have four main axes of investigation: time, users, systems, and object data. Time readings come from the time stamps made during signature creation; user information from the user ids or email addresses in the metadata; system information from the machine identifiers; and information about objects from the signatures and includes both metadata and saved content. Cross-referencing these different axes provides a variety of ways to examine activity. Investigations based on user activity over time are very common and tools have been developed that do this; where necessary, information recovered from multiple systems is correlated to determine distributed user activity (Buchholz and Falk, 2005; Olsson and Boldt, 2009). What has not been possible in the past, and what PROOFS allows, is investigating activity based on file content, even when the files of interest no longer exist on the system. This supports common existing forensic steps and introduces new forensic possibilities. We describe how PROOFS can be used here.

Insider IP theft An employee leaves a company; shortly thereafter, they start at a competing company (Neuman, 2010). The examiner is asked if they took proprietary documents with them to their new position. Using PROOFS to find files owned by that user and the actions that occurred, the examiner is able to show that a set of files were copied to a USB drive. Analysis of the signatures shows the keywords in the fingerprints were related to an internal project of interest to the competitor.

Misuse A company employee is receiving threatening emails on a work account. The context of the mail indicates it is another employee, but the emails are sent through a webmail account that does not include a source IP. The examiner can use the text of the emails to find all computers on the network that ever contained the text of the message in web cache. This provides a starting point for a deeper investigation as to who is sending the messages.

Intrusion response A system is found to have been compromised by an external attacker. A forensic examination is performed, and a fragment of a README file that was part of the rootkit and was fingerprinted before the rootkit stopped PROOFS is recovered. A signature of the fragment is created and is used to quickly identify all other compromised systems based on signatures in the database.

Lost equipment An examiner is informed that a laptop has gone missing. Using PROOFS she retrieves a list of files that were on the laptop. She then uses PROOFS to identify copies of the files on other computers in the network. For those files with no retained copies, profiles of the contents are extracted by fingerprint keyword. The resulting information is used to determine the potential damage caused by the lost equipment.

Examination support While conducting a forensic examination, a fragment of a file is found that contains compelling evidence toward answering the investigative hypothesis.

However, the fragment is in unallocated space and the metadata about which user created it and when is lost. The examiner is able to create a signature from the fragment and consult the PROOFS database to find what files on the system matched, when they were deleted, and who the owner of the files were. This narrows the investigative effort.

6. Fingerprint performance

Experimental settings The ability to create fingerprints that can be accurately matched across changes to an object lies at the heart of PROOFS. To prove the concept, we implemented PROOFS fingerprinting in Java to experiment with creation of dictionaries and measure their matching accuracy. For each experimental trial, we created a new dictionary using 5% of the files in the data set. Files were chosen randomly without replacement. The dictionary created was then trimmed to remove tokens outside a specified IDF range. We recorded the bit-vector fingerprint length as it is equal in bits to the number of dictionary terms.

We then selected 2000 files that were not used for the dictionary and created bit-vector fingerprints of each. In some cases, as described below, we then manipulated the files to simulate edits or deletions in the text, then re-fingerprinted them. Once created, we compared the fingerprints of the unmodified source files to all other fingerprints using cosine similarity matching with a parameter (normalized from 0 to 100) representing the minimum score needed to be considered a match. For each experiment, we computed the average precision and recall over 20 trials. Where error bars are plotted they represent the 95% confidence interval.

Precision, recall, and F-measure Accuracy was measured by computing the *precision* and *recall* of each trial; as these terms are more commonly used in information retrieval than the forensics community they are explained here. The precision is the fraction of relevant documents matched divided by all documents retrieved; this is also the number of true positives over the sum of true positives and false positives. Recall is the fraction of relevant documents matched out of all relevant documents; or true positives divided by the sum of the true positives and false negatives. They are standard measures of accuracy in information retrieval systems, and higher scores are better. As a point of comparison, Google has been measured to have a precision of 0.29 and recall of 0.20 for domain-specific searches (Shafi and Rather, 2005); our system shows precision and recall of over 0.96 and 0.85, respectively. The *F-measure* is the weighted harmonic mean of precision and recall and is used to show overall performance in a more compact form due to limited space.

6.1. Data sets

We present results from experimental runs using the Enron email data set (Klimt and Yang, 2004), which is a collection of forensically retrieved emails released following the investigation into the collapse of the company. This data set is the most representative publicly available forensic data set. Results from other data sets are available in our technical report (Shields et al., 2010).

Among other operations, PROOFS is designed to find matching documents, so we needed to ensure that the data sets contained matches to find. The Enron data set contains many natural matches, as it contains copies of many emails as seen by the sender and multiple recipients. To determine which pairs of files constituted matches we used I-Match (Chowdhury et al., 2002) to identify duplicate documents. Automated duplicate finding was required because there were too many documents to compare manually. These results were later used as a point of comparison to determine whether the matches found under PROOFS were correct.

The necessary use of I-Match as a baseline to determine duplicates means that our results contain a potential source of error. In our best case, PROOFS will only be seen as doing as well as I-Match. In cases where we do better and find appropriate duplicate documents that I-Match did not, it will show as reduced accuracy for PROOFS.

6.2. Matching original documents

Our initial experimental runs were done to determine the trade-offs between dictionary size and accuracy. We use this to determine the optimal IDF settings for creation of compact but accurate dictionaries. We ran a series of experiments where we created dictionaries given either a 1, 5 or 10% sample of the data set at a variety of normalized IDF ranges. The selected files were tokenized by replacing punctuation with white space, removing known stop words, using Porter stemming (Porter, 1997), limiting the length to between 6 and 20 characters, then removing any purely numeric tokens. The minimum normalized IDF started at 0.1 and increased in increments of 0.1 up to 0.6. For each minimum IDF we tested a range of maximum IDFs from 0.2 incrementing by 0.1 up to 0.9. Ranges where the maximum IDF was less than or equal to the minimum IDF were ignored. We used the results of these experiments to choose a set of parameters for use in our other experiments, all based on a 5% initial sample. These are shown in Table 1 and described below.

The resulting points show the F-measure versus dictionary size overall IDF ranges. We note that one would normally expect to see higher accuracy as the sample size increases. This is not universally true in our plots as each plotted point represents a selected IDF range. In cases where the range is small and terms in that range not distinctive, larger sample sizes under perform smaller sample sizes with a wider IDF range. This is the cause of apparent outliers in the plot.

The Enron data set can be accurately fingerprinted with dictionaries of less than 10,000 entries. It is clear, as seen in

Table 1 – Data set statistics and parameters selected.

Enron data set			
#Files	517,412	#Terms	5548
Avg. size	2777	Avg. precision	0.85
Std dev.	23,798	Avg. recall	0.96
Min IDF	0.1	Max IDF	0.7

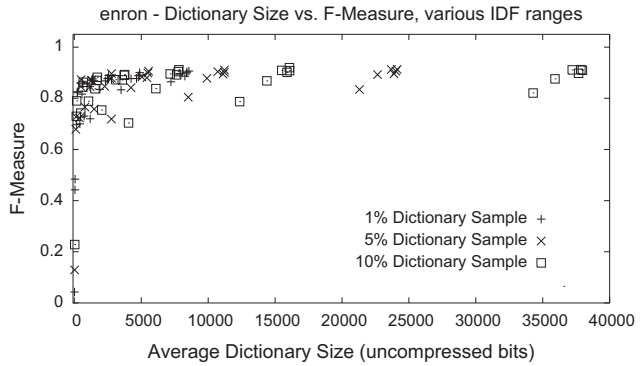

Fig. 1 – Enron F-measure vs. dictionary size, various IDF ranges.

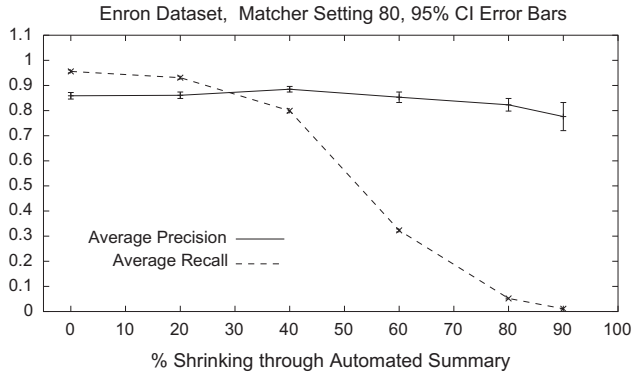

Fig. 2 – Enron automatic summarization, matcher 80.

Fig. 1, that there is a larger trade-off involved with this most diverse data set. While fingerprints can be made very accurate the fingerprint size grows significantly, not leveling off until it reaches in excess of 10,000 bits. We chose to reduce the accuracy slightly in favor of smaller fingerprints, and chose a minimum IDF of 0.1 and a maximum IDF of 0.7. This yielded an average precision of 0.8500 $\pm$ 0.0170, an average recall of 0.9555 $\pm$ 0.0074, and an average fingerprint size of 5548 bits $\pm$ 29 bits before compression.

To put these results in context, given a fingerprint PROOFS is able to return at least 96% of relevant documents. Of the documents returned, at least 85% will be relevant to the source document. Again, as a point of reference we mention that the measured precision and recall of Google is 0.29 and 0.20 (Shafi and Rather, 2005). PROOFS will perform more accurately as it can select terms that are specific to the organization running it, and it does not have to index the world.

6.3. *Matching edited documents*

Our initial results showed that we attain high accuracy in matching unedited but related documents within a large collection. In many forensic situations, however, we wish to follow source documents across edits or other changes. We now demonstrate that fingerprints are resilient to edits while maintaining high accuracy.

Automatically summarized documents Our first experiment with edited documents is designed to show that PROOFS can accurately match fingerprints between a source document and one that has been shortened by editing. Given the large number of documents included in our data sets, it was infeasible to have them all edited by humans; therefore, we used automatic document summarization software to simulate human edits that shorten the source file. We conducted additional experiments, omitted for space, that show this is a reasonable approximation of human activity (Shields et al., 2010). There are a variety of tools that do this, including those in the Apple Summarization Service and in Microsoft Word. For ease of use in our experiments, we used the classifier4j tool (Lothian, 2005). In small-scale testing it performed similarly to the other tools mentioned but was compatible with our Java-based experimental framework. We created edits of source files at specified reductions of 20, 40, 60, 80, and 90% in length and matched fingerprints of those files to the source files.

The effect of varying the matcher parameter is clearly visible. With the higher matcher parameter of Fig. 2, the precision and recall are above 80% with the file shortened up to 40%; beyond that, recall falls off dramatically. Using the lower matcher parameter of Fig. 3, the precision starts just above 60%, but recall is above 95% and remains high with the document shortened up to 60%. This variability is a desirable feature. PROOFS can create and store signatures without regard to what matcher parameter will be used. When an investigation becomes necessary, the investigator can start with a high matching parameter for better precision, and reduce it to see a wider variety of less-precise results as desired.

Overall, PROOFS supports high accuracy in matching documents edited for size to the original source document. In all the data sets it was possible to achieve a precision and recall above 80% for documents that have been shortened by 40%; for some data sets, precision and recall were above 90% even when documents were shortened by 60%.

6.4. *Matching file fragments*

A common investigative task for forensic examiners is to search for keywords in seized media, which often finds

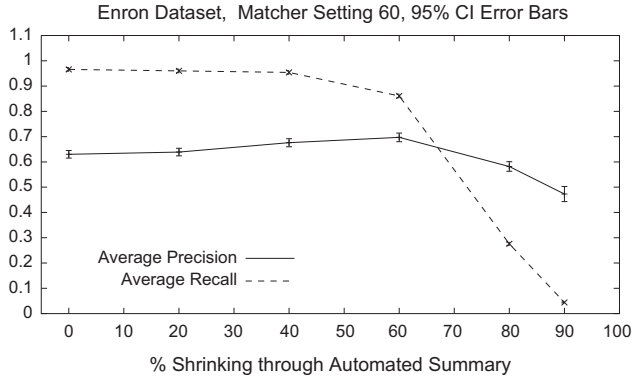

Fig. 3 – Enron automatic summarization, matcher 60.

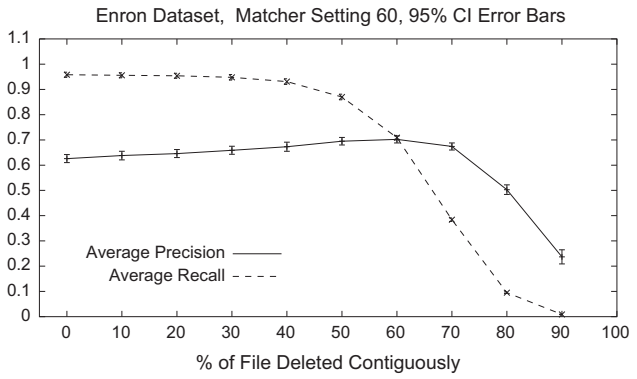

Fig. 4 – Accuracy with deleted sections, matcher 60.

fragments of files in swap space or in unallocated disk space. While the contents of fragments can often be pivotal in resolving a case, they lack any context about their source or creation. PROOFS greatly improves the ability of investigators to make use of file fragments by allowing retrieval of this context, including the likely user and time of creation.

We demonstrate this by an experiment in which we removed a contiguous section of text from the source file. The size of the deletion was specified as a percentage of the source file and was taken from the file at a random position. We also conducted experiments taking sections from the start or end of the file, but have omitted the results due to space constraints; the results are very similar.

The Enron data set results again demonstrate the effectiveness of changing the matching parameter. With the lower parameter, shown in Fig. 4, recall remains above 85% for deletion sizes up to 50%, but precision is below 70%, meaning an investigator would be faced with a wider number of possible matches to winnow through. With the higher matcher parameter, seen in Fig. 5, the precision is well over 80% with 60% of the file deleted, but recall drops below 70% after only 30% of the file is removed.

These results show that PROOFS is highly effective at matching recovered fragments of text documents to their source, adding significant context to a common investigative activity.

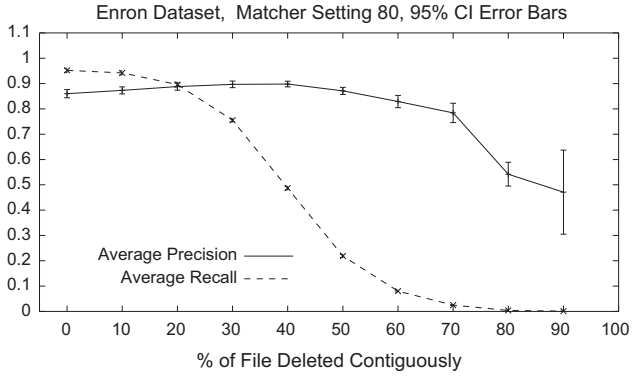

Fig. 5 – Accuracy with deleted sections, matcher 80.

7. Comparison to existing methods

Our proactive approach to collecting and storing forensic information is a novel application of information retrieval techniques. In this section, we demonstrate that PROOFS fingerprints are superior to a naive approach. Space prohibits inclusion of results using I-Match signatures (Chowdhury et al., 2002), but we have shown that I-Match is significantly less resilient to matching altered files (Shields et al., 2010).

Naive approach A natural approach to preserving forensic data would be to compress everything, store it, and index it for later searching. We argue here that doing so would require significantly more storage than PROOFS. In small-scale testing on a set of 300 Word documents, 500 PDF files, and 800 text files, each type of file compressed at different rates. Using gzip, Word documents were compressed by an average of 75%; text documents by 65%, and PDF files by only 16%. In a naive approach that gzipped all files, the *best* approach might save 75% of normal space required of all documents. Indexes across these files would require additional space, reducing the savings.

In PROOFS, we perform three steps that reduce the amount of storage required for each document. First, we extract text from documents containing formatting. For PDF and Word files, this reduced the file size by approximately 75%, though plain text documents do not benefit from this step. Second, we extract fingerprints from the tokens in the text. In our experiments, the average uncompressed fingerprint size was 25% of the average file size, for a relative reduction of 75%. Finally, the fingerprints, being a sparse bit vector, are also compressed. The average compression of the fingerprints using gzip was 83%.

For formatted documents, PROOFS reduces the storage required to an average of 1.06% of the source, for an effective reduction of 98.94% from the original. For plain text, the storage required is 4.25% of the original, for a 95.75% reduction in required storage over the base document. No exiting compression mechanism can perform this well. Additionally, PROOFS signatures do not require any additional indexing.

Overall, this means that we might expect a naive approach to require at least 5 times the amount of storage for text files; for formatted documents, we would expect a 23 times increase, *excluding* indexes. Even with large disks, this is a prohibitive cost.

8. Related work

PROOFS is novel because it is the first collection system to preserve evidence about the contents of files. Operating over file contents, rather than binary data, provides a better view of what a user said, knew, or did. We have discussed approaches to proactive evidence collection previously (Shields, 2010).

Other systems have proposed or implemented continuous collection of system events, and a variety of audit and logging systems are used to collect and store information about running systems (Bishop, 1996; Picciotto, 1987; Wee, 1995). Log collection and analysis is an important part of existing investigations but log systems are not designed specifically to support forensic examinations (Herrerias and Gomez, 2007; Sandler et al., 2008; Takahashi and Xiao, 2008).

Past work has recognized that hashing a file does not allow matching of closely related documents. An approach called *fuzzy hashing* (Kornblum, 2006) or *similarity matching* (Roussev et al., 2007) has been developed to help address this. These approaches can identify files with similar binary segments but will not identify files with common text in different formats, such between PDF and Word format. PROOF operates on content and will identify similar files even if the format has changed.

Network forensics systems have been proposed that collect data from the network. Some collect information about the contents of packets to allow attribution of payloads (Ponec et al., 2007), but such contents are not tied to a particular system or user. Other systems maintain information about files sent to a central anti-virus scanner (Oberheide et al., 2008), but do not describe what information is maintained there nor if it is practical to store full copies of all files. Additionally, records are kept only of files that have not been previously opened, so copying and renaming known files will not create entries. These are actions that are associated with theft of intellectual property, a common forensic investigation.

There has been past discussion of proactive forensics, though no common agreement on the meaning of the term has been reached. In some cases, it has meant data collection to detect an insider threat based on user behavior (Bradford et al., 2004; Bradford and Hu, 2005; Paintsil, 2007). Others use it in the sense of continuous collection (Aggarwal et al., 2005; Neuman, 2010) but do not present a complete proactive solution.

Duplicate document detection has been an area of past work within the information retrieval community (Chowdhury et al., 2002; Cooper et al., 2002; Heintze, 1996; Manber, 1994). The forensics solution space differs from past work due to the need for a small, storable fingerprint that is more robust to matching across edits.

9. Limitations and future work

We believe that PROOFS can provide a new, deep, and valuable source of forensic information for a centrally controlled network, but that it will *complement* rather than replace existing forensics tools. This is because PROOFS is probabilistic. Examiners will still need to perform examinations to confirm the PROOFS data. In addition, there is a wide variety of evidence that is commonly used but which is not present in PROOFS, such as file access times, so other tools will be required to recover all relevant evidence.

Our text fingerprints currently focus on storing information about statistically important terms. We will investigate construction of fingerprints that provides additional information about text files. One approach is to extract all words from the document and use the list as input to a Bloom filter. Storing the resulting hash will allow investigators to probabilistically probe the hash for the existence of words or terms that are not in the dictionary but which might otherwise be important to an investigation. A different approach is to create a fingerprint that is a count of the number of times a particular pattern is matched in the file. This can be used to determine if a deleted file contained things like social security numbers, and if so how many.

Other future work will focus on creating small, robust signatures for other types of digital objects, including executable, source code, images, audio, and video. In many cases, there is work indexing and searching for these types of files (Burges et al., 2005; Hoad and Zobel, 2006; Srinivasan and Sawant, 2008), which we can adapt to the forensic environment. We will also examine improving the accuracy of text searches by creating and combining multiple fingerprint types in a single signature.

10. Conclusion

We have presented a system named Proactive Object Fingerprinting and Storage (PROOFS) that continuously and efficiently creates fingerprints based on the contents of files. Fingerprints for text files are small to store; we have shown that for the Enron data set, a collection of over half a million documents, fingerprints can be as small as 375 bytes but retain high accuracy. They are also able to match with fingerprints created from related input files with high accuracy.

Our results show that we can match fingerprints of unedited source files with a recall of greater than 95% and a precision of greater than 85%. Fingerprints are also robust against editing, and we have shown that for documents shortened by automated summarization, fingerprints can still match accurately when the file is only 60% of its original length in the Enron data set. Fingerprints can also be matched when significant portions of the source file have been deleted; we have shown that for the Enron data set, at least 20% of the file can be removed and still be matched.

When combined with metadata such as file names, time stamps, and user information into a signature, fingerprints allow central storage of information useful to forensic examinations on a centrally controlled network. Investigators can track system activity based on user, system, time, and novelly, by the contents of objects either present on or deleted from the systems on the network.

Acknowledgments

We would like to thank Wade Tandy and Chris Wacek for early development of the proof-of-concept code.

REFERENCES

US v. Ziegler, http://caselaw.findlaw.com/us-9th-circuit/1102459.html; January 2007. United States Court of Appeals, Ninth Circuit. No. 05–30177 (1-30-07).

Aggarwal S, Henry P, Kermes L, Mulholland J. Evidence handling in proactive cyberstalking investigations: the PAPA approach. In: Systematic approaches to digital forensic engineering. 2005. p. 165–176.

Bishop M. A standard audit trail format. In: National Information Systems Security'95 (18th) proceedings: making security real. 1996. p. 136.

Boyd C, Forster P. Time and date issues in forensic computing: a case Study. Digital Investigation February 2004;1:18–23.

Bradford PG, Hu N. A layered approach to insider threat detection and proactive forensics. In: Annual Computer Security Applications Conference (ACSAC). Tuscon, AZ: December 2005.
Bradford P, Brow M, Perdue J, Self B. Towards proactive computer-system forensics. In: International conference on information technology: coding and computing. 2004. p. 648–652.
Buchholz F. Falk C. Design and Implementation of zeitline: a forensic timeline editor. In: Digital Forensics Research Workshop. 2005. p. 1–7.
Buchholz F, Spafford E. On the role of file system metadata in digital forensics. Digital Investigation 2004;1:298–309.
Bumiller E. In: Ex-Hacker, editor. Army leak suspect is turned in. New York Times, http://www.nytimes.com/2010/06/08/world/08leaks.html; June 2010.
Burges C, Plastina D, Platt J, Renshaw E, Malvar H. Using audio fingerprinting for duplicate detection and thumbnail generation. In: Proc. acoustics, speech, and signal processing. 2005. p. 1–4.
Carrier B. File system forensic analysis. Addison-Wesley Professional; 2005.
Chowdhury A, Frieder O, Grossman D, McCabe MC. Collection statistics for fast duplicate document detection. ACM Transactions on Information Systems 2002;20:171–91.
Cooper JW, Coden AR, Brown EW. Detecting similar documents using salient terms. In: Proceedings of the eleventh international conference on information and knowledge management. 2002. p. 245.
Grossman D, Frieder O. Information retrieval. 2nd ed. Springer; 2004.
Heintze N. Scalable document fingerprinting. In: USENIX workshop on electronic commerce. 1996.
Herrerias J, Gomez R. A log correlation model to support the evidence search process in a forensic investigation. In: second international workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'07). April 2007. p. 31–32.
Hoad T, Zobel J. Detection of video sequences using compact signatures. ACM Transactions on Information Systems January 2006;24:1–50.
Keeney M, Kowalski E, Cappelli D, Moore A, Shimeall T, Rodgers S. Insider threat study: computer system sabotage in critical infrastructure sectors. Tech. rep., U.S. Secret Service and SEI at CMU. May 2005.
Klimt B, Yang Y. Introducing the Enron Corpus. In: First conference on email and anti-spam (CEAS). 2004.
Kornblum J. Identifying almost identical files using context triggered piecewise hashing. Digital Investigation 2006;3:91–7.
Lothian N. Classifier4j, http://classifier4j.sourceforge.net; February 2005.
Luoma V. Computer forensics and electronic discovery: the new management challenge. Computers & Security March 2006;25:91–6.
Manber U. Finding similar files in a large file system. In: Proceedings of the USENIX Winter 1994 technical conference. Berkeley, CA: 1994.
Neuman W. A man with muffin secrets, but no job with them. New York Times, http://www.nytimes.com/2010/08/07/business/07muffin.html; August 2010.
Oberheide J, Cooke E, Jahanian F. CloudAV: N-version antivirus in the network cloud. In: Proceedings of the 17th USENIX security symposium. San Jose, CA: July 2008.
Olsson J, Boldt M. Computer forensic timeline visualization tool. Digital Investigation September 2009;6:S78–87.
Oracle, Oracle outside in technology. http://www.oracle.com/technology/products/content-management/oit/oit_all.html.
Paintsil AB. Insider threat detection: a proactive forensic approach. Master's thesis, Stockholm University/The Royal Institute of Technology, Stockholm. Sweden. May 2007.
Picciotto J. The design of an effective auditing subsystem. In: Proceedings of the 1987 symposium on security and privacy. 1987. p. 13–22.
Ponec M, Giura P, Brönnimann H, Wein J. Highly efficient techniques for network forensics. In: Proceedings of the 14th ACM conference on Computer and Communications Security. 2007. p. 150–160.
Porter MF. An algorithm for suffix stripping. Readings in Information Retrieval; 1997:313–6.
Roussev V, III, Richard G, Marziale L. Multi-resolution similarity hashing. In: Digital Forensics Research Conference (DFRWS). 2007. p. 105–113.
Sandler D, Derr K, Crosby S, Wallach DS. Finding the evidence in tamper-evident logs. In: 2008 Third international workshop on systematic approaches to digital forensic engineering. May 2008. p. 69–75.
Schneier B, Kelsey J. Secure audit logs to support computer forensics. ACM Transactions on Information and System Security (TISSEC) 1999;2:159–76.
Shafi S, Rather R. Precision and recall of five search engines for retrieval of scholarly information in the field of Biotechnology. Webology August 2005;2(2).
Shields C, Frieder O, Maloof M. A novel system for the proactive, continuous, and efficient collection of digital forensic evidence. Tech. Rep. CSTR-20100415-1, Georgetown University. 2010.
Shields C. Towards proactive forensic evidentiary collection. In: Hawaii International Conference on System Sciences (HICSS). January 2010.
Srinivasan SH, Sawant N. Finding near-duplicate images on the web using fingerprints. In: Proceeding of the 16th ACM international conference on Multimedia. 2008. p. 881.
Takahashi D, Xiao Y. Complexity analysis of retrieving knowledge from auditing log files for computer and network forensics and accountability. In: 2008 IEEE International Conference on Communications. May 2008. p. 1474–1478.
Wee C. LAFS: a logging and auditing file system. In: Annual computer security applications conference. 1995. p. 1–10.

Clay Shields is an associate professor in the Computer Science Department at Georgetown University, and is Director of the Georgetown Institute for Information Assurance. Prior to arriving at Georgetown, he was an assistant professor at Purdue University, where he held his first academic position after earning his Ph.D from the University of California Santa Cruz. Before graduate school, Clay was an infantry officer with the 101st Airborne Division, and earned his undergraduate degree in electrical engineering from the University of Virginia. His research efforts focus on computer forensics and network security.

Ophir Frieder holds Robert L. McDevitt, K.S.G., K.C.H.S. and Catherine H. McDevitt L.C.H.S. Chair in Computer Science and Information Processing and is Chair of the Department of Computer Science at Georgetown University. His research interests focus on scalable information retrieval systems spanning search and retrieval and communications issues. He frequently consults for industry and government and for key intellectual property litigation; his systems are deployed in commercial and governmental production environments worldwide. In 2007, Springer Science and Business Media designated his co-authored book entitled "Information Retrieval: Algorithms and Heuristics" with the "Top Selling Title" award. He is the recipient of the 2007 ASIS&T Research in Information Science Award and a recipient of the 2008 IEEE Technical Achievement Award. He is a Fellow of the AAAS, ACM, and IEEE.

Mark Maloof is an associate professor in the Department of Computer Science at Georgetown University. His research interests include machine learning, data mining, on-line learning algorithms, concept drift, and applications of machine learning and data mining to computer security. He led the effort that

established Georgetown's first graduate programs in computer science, and is the director of the department's Master's and Ph.D programs. In 2004, he shared with Zico Kolter the award for the best application paper at KDD for their work on detecting malicious executables. In 2007, he shared with Greg Stephens a Program Innovation Award from the MITRE Corporation for their work on detecting insider threats. Mark has served as a consultant to industry, government, and nonprofit organizations.