Detecting Threatening Insiders with Lightweight Media Forensics

Simson L. Garfinkel
Dept. of Computer Science
Naval Postgraduate School
Arlington, VA 22203
Email: slgarfin@nps.edu

Nicole Beebe, Lishu Liu, and Michele Maasberg
Department of Information Systems and Cyber Security
University of Texas San Antonio
San Antonio, TX 78249
Email: First.Last@utsa.edu for all authors

***Abstract*—This research uses machine learning and outlier analysis to detect potentially hostile insiders through the automated analysis of stored data on cell phones, laptops, and desktop computers belonging to members of an organization. Whereas other systems look for specific signatures associated with hostile insider activity, our system is based on the creation of a "storage profile" for each user and then an automated analysis of all the storage profiles in the organization, with the purpose of finding storage outliers. Our hypothesis is that malicious insiders will have specific data and concentrations of data that differ from their colleagues and coworkers. By exploiting these differences, we can identify potentially hostile insiders.**

Our system is based on a combination of existing open source computer forensic tools and datamining algorithms. We modify these tools to perform a "lightweight" analysis based on statistical sampling over time. In this, our approach is both efficient and privacy sensitive. As a result, we can detect not just individuals that differ from their co-workers, but also insiders that differ from their historic norms. Accordingly, we should be able to detect insiders that have been "turned" by events or outside organizations. We should also be able to detect insider accounts that have been taken over by outsiders.

Our project, now in its first year, is a three-year project funded by the Department of Homeland Security, Science and Technology Directorate, Cyber Security Division. In this paper we describe the underlying approach and demonstrate how the storage profile is created and collected using specially modified open source tools. We also present the results of running these tools on a 500GB corpus of simulated insider threat data created by the Naval Postgraduate School in 2008 under grant from the National Science Foundation.

I. Introduction

Hostile insiders represent a serious and ongoing threat for organizations worldwide. Trusted insiders such as Aldrich Ames and Robert Hanssen have done much damage to the US because of their decision to share classified information to which they had access with enemies of the United States. More recently, Bradley E. Manning has stood trial for collecting large amounts of classified information on his SIPRNet computer system, copying it to removable media, and then providing that data to the WikiLeaks organization. In all of these cases, the lack of sufficient internal monitoring and controls failed to stop the internal collection and external dissemination (often referred to as *exfiltration*) of highly sensitive documents.

Insiders engaged in illegal activities represent another kind of hostile insiders that organizations need to address. In July 2010, for example, the Inspector General of the Department of Defense released a 94 page report detailing how a group of more than 50 federal employees and contractors with high-level security clearances were involved in the collection and distribution of child pornography—some of it on government owned computer systems. [1]. Involvement with any kind of pornography on government systems makes the offenders subject to the threat of blackmail, which endangers national security. Other federal agencies have had problems with insiders collecting and then exfiltrating social security numbers and even credit card numbers.

We believe that there is a single distinguishing characteristic in many of these cases: as a preparation to exfiltrating data, insiders will collect it on a single system, typically a workstation under their control [2]. As such, the storage profile of this workstation changes in significant ways. First, the workstation suddenly has a collection of documents that makes it significantly different from the other workstations in the organization—it becomes an outlier. Second, the workstation changes from its historical norms.

Prior attempts at insider detection have relied on measuring information access patterns or the detection of signatures based on historical incidents. These systems have been expensive to develop and maintain, and have been subject to numerous false positives and false negatives.

Based on our experience in media forensics, we are researching a new approach for detecting hostile insiders. This approach uses *lightweight media forensics* to create a *storage profile* for each computer in an enterprise. These profiles are reported to a central management console on a regular but unpredictable basis. The management console then performs routine lightweight datamining and outlier detection—that is, it identifies *which* workstations are different from the norm, as well as the *reason* that they are different.

II. Architecture

Our novel architecture is based on a light weight media forensics agent that covertly collects and securely reports an inventory of the contents of each mass storage device (e.g., hard drive or SSD) in the enterprise, and a datamining solution that performs unsupervised clustering and outlier analysis. Outliers are reported at a management console to the security administrators, who can then investigate further.

A. Light Weight Media Forensics Agent

Media Forensics is a term of art used in the law enforcement and intelligence communities to describe digital forensics applied to computer media such as hard drives, portable storage devices and optical storage. As practiced today, media forensics is a slow, labor-intensive process. Analyzing a single 200GB hard drive might take a seasoned analyst 10-20 hours.

We are using other approaches to media forensics to develop a system that can largely function without human involvement. Our system combines three elements: bulk data analysis, random sampling, and automated file system metadata extraction.

1) **Bulk Data Analysis.** This approach ignores the structure of files on the hard drive and instead analyzes the bulk data on the drive from the first sector to the last. We have developed a program called *bulk_extractor* [3] that can scan a hard drive for email addresses, credit card numbers, social security numbers, keywords, and other kinds of sensitive information. The program can also perform *fragment type identification* using specialized algorithms [4] and by recognizing known content through identification of block hashes [5]. The results are tabulated into a histogram (Figure 1 shows an example of the email histogram).
2) **Random Sampling.** This approach also ignores files, but instead of examining each sector on the hard drive, it randomly samples between 1% and 2% of the drive. Using this approach we can rapidly calculate the drive's *forensic inventory*—for example, the amount of the drive dedicated to storing PDFs, Microsoft Word files, still images, videos, and encrypted content.
3) **Automated File System Metadata Extraction.** This approach walks the file system of a hard drive and extracts metadata for each file into a single XML block using Digital Forensics XML [6]. The metadata includes file names, file modification times, and document properties such as file creator, file print time, and other information.

Using these approaches, we are building a *local surveillance agent* that will covertly monitor the forensic contents of each enterprise workstation on a regular-but-unpredictable basis (every 1-3 days, for example), which the user will not be able to detect or disable. Built upon Google's Rapid Response (GRR) Framework [7], the agent will create a management report for each workstation consisting of:

- An estimation of the total # of email addresses on the hard drive, and the most popular 100.
- An estimation of the total # of credit card numbers on the hard drive.
- An estimation of the total # of social security numbers on the hard drive.
- An estimation of the percentage of the drive devoted to JPEGs, video, and encrypted data.
- The number of allocated files on the drive, their file extensions, and their type as determined by file extension, internal magic numbers (using [8]), and the file type determined using our support vector machine-based classifier.
- Document metadata information.

This management report will be securely sent to the management console using either an existing management framework or a lightweight transport system based on web services.

```
# Filename: pat-2009-12-02.E01
# Feature-Recorder: email
# Histogram-File-Version: 1.1
n=1775  pat@m57.biz     (utf16=45)
n=818   terry@m57.biz   (utf16=51)
n=531   charlie@m57.biz (utf16=59)
n=412   premium-server@thawte.com (utf16=9)
n=291   mozrepl@hyperstruct.net   (utf16=95)
n=198   inet@microsoft.com
n=197   cps-requests@verisign.com (utf16=1)
n=174   certificate@trustcenter.de
n=152   info@valicert.com
n=142   mfc@uk.ibm.com
n=131   hewitt@netscape.com
n=130   feste@feste.org
n=102   t93940@gmail.com (utf16=12)
n=97    ips@mail.ips.es
n=90    pat@www.ms        (utf16=90)
n=87    server-certs@thawte.com
n=76    mal@lemburg.com
n=74    pat@hit.ge        (utf16=74)
n=70    silver-certs@saunalahti.fi
n=70    someone@example.com      (utf16=70)
n=70    someone@microsoft.com    (utf16=37)
n=68    alex@nitroba.com         (utf16=8)
```

Fig. 1. A portion of the email address histogram found on the drive pat-2009-12-02, part of the NPS M57-Patents corpus. The email addresses in the m57.biz and nitroba.com domains are fictional; the others are part of software distributions and are already in the public domain. The notation (utf16=NN) means that NN of the email addresses found on the media were in UTF-16; the remainder were in UTF-8 or ASCII.

B. Histogram Processing

In previous work PI Garfinkel has shown that frequency distribution histograms are of significant use in forensic investigations [9]. For example, a frequency histogram of email addresses found on a hard drive readily identifies the drive's primary user and that person's primary contacts.

Histogram generation is integrated with the feature recording system so that histograms can be created for any feature. Our design further allows histograms to be generated from substrings extracted from features using regular expressions. For example, *bulk_extractor* creates a histogram of search terms provided to Google, Yahoo, and other popular search engines. Histograms of search terms are particularly useful when conducting an investigation, as they reveal the intent of the computer's user.[1]

Histogram analysis is a powerful tool that supports our goal in using *bulk_extractor* as a tool for detecting hostile insiders. Histogram analysis gives both our outlier analysis tool and the human analyst contextual information that can be used in rating information that is discovered through lightweight media forensics.

[1]For example, at the 2008 murder trial of Neil Entwistle, prosecutors introduced evidence that Entwistle had performed Internet searches for murder techniques just three days before his wife and child were found murdered [10].

C. Outlier Analysis

Although every hard drive is different, our previous experiences have shown that hard drives with unusual statistical properties invariably warrant further analysis. Figure 2 shows data from a previous study regarding the inadvertent release of confidential information through the sale of used computer equipment that had not been properly sanitized. To conduct this study approximately 250 used hard drives were purchased on the secondary market (e.g., used computer stores, computer swap meets and pawn shops) and then scanned for credit card numbers. The graph shows the number of total CCNs on each drive (blue) and the number that were distinct (red). As can be seen, while most hard drives in the collection contained no credit card numbers or just a few, 4 drives contained hundreds and 3 drives contained thousands. By using outlier analysis it was possible to rapidly find those drives that needed to be manually investigated. Details can be found in [11].

We hypothesize that a similar approach can be used to find some kinds of hostile insiders—specifically insiders that are collecting information on specific computers either for personal use that is inconsistent with organizational norms (as is the case with pornography), or collecting information with the intent of later exfiltration (as is the case with those stealing sensitive information).

We are exploring the use of cluster analysis to identify statistical outliers. Being a statistical outlier in this context indicates the outlier media (e.g., a specific employee's workstation) has a storage profile unlike others in the organization, or unlike itself from an historical perspective. Cluster analysis is a widely accepted technique that is commonly and successfully used in fraud detection and network intrusion detection. It presents three important benefits:

1) **Discovery, not signature-based.** Clustering algorithms are unsupervised. That is, they do not require a priori knowledge of known signatures, which is extremely important in the *insider threat* detection context. This is because we know from analysis of past cases that there are significant differences across classes of insiders, as well as variation at the individual level within classes [12] [2].
2) **Scalability.** While we are proposing a lightweight solution in terms of extractor and transport agents, data for analysis could be significant in large, busy organizations. Highly scalable clustering algorithms have been developed that scale linearly $O(n)$, or even logarithmically $O(\log(n))$ in some cases, and that handle noisy data sets well.
3) **Meaningful results.** Some machine learning techniques, while effective (for prediction, for example) are difficult to interpret and explain. It is often important to understand and be able to explain why the algorithms partitioned data in a certain way, or predicted a certain outcome. This is critically important when launching further inquiries as a result of anomaly detection findings.

Our outlier analysis and identification system should create organizational and sub-organizational, level storage profiles based on actual use. We expect that the system will accommodate real-world changes in organizational baselines over time. The system should also be able to detect statistical outliers pertaining to one workstation relative to the rest of the workstations in an organization.

The system should be capable of simultaneously considering individual, sub-organization, role-based, and organizational levels of analysis. For example, perhaps one employee's storage profile is not a statistical outlier considering the organizational dimension alone, but is a statistical outlier when considering the job function and organization level profile simultaneously. In short, several analytical variations will be permissible to facilitate more granular outlier detection.

Last, we plan to expand the system so that it can detect statistically anomalous changes in individual storage profiles over time. So, perhaps the *insider's* storage activity is inconspicuous enough to thwart statistical detection at the organizational level. The individual, time-based detection mechanisms can be more sensitive since the variance is lower. Hence, more subtle changes in storage activity can be detected via lightweight media forensics considering changes to the workstation alone over time.

All of this should be possible because outliers statistically deviate from other observations. Observations that are not outliers conform to an underlying statistical distribution. Several challenges exist in designing a threat detection system via cluster-based outlier detection, including algorithm detection and feature selection.

D. Algorithm Selection

Proper algorithm selection and adaptation is critical to successful outlier detection. Clustering algorithms partition observations, but the manner in which the 'space' is partitioned varies between algorithms. This impacts cluster quality and algorithm efficiency. Data clustering methods, independent of whether the data being clustered is textual (i.e. unstructured) or structured, can be grouped into five categories: partitioning, hierarchical, density-based, grid-based, and model-based [13]. Each approach has advantages and disadvantages.

a) Partitioning methods: for example, k-means and k-medoids clustering, group data into mutually exclusive clusters that minimize the intra-cluster variation, while maximizing inter-cluster variation. Partitional clustering is an iterative process, re-computing cluster and item similarity measures and reassigning items until convergence occurs (ideally, or until a maximum number of iterations is reached). Partitioning methods are a commonly used clustering method, due to their speed, simplicity, and scalability [13]–[15]. This is due to low computational expense (time complexity, running time) and space (i.e. memory) requirements.

The disadvantages of partitioning methods, however, include:

- tendency toward spherical clusters, thus impacting cluster quality (i.e. validity and meaningfulness) [16], [17];
- tendency toward balanced clusters, which can also impact cluster quality [16];
- difficulty dealing with noisy data and outliers [13];
- dependence on initial cluster selection, which reduces cluster stability and reliability [13], [17];
- its overall tendency to result in local minimums instead of global minimums (i.e. inter-cluster variation is minimized

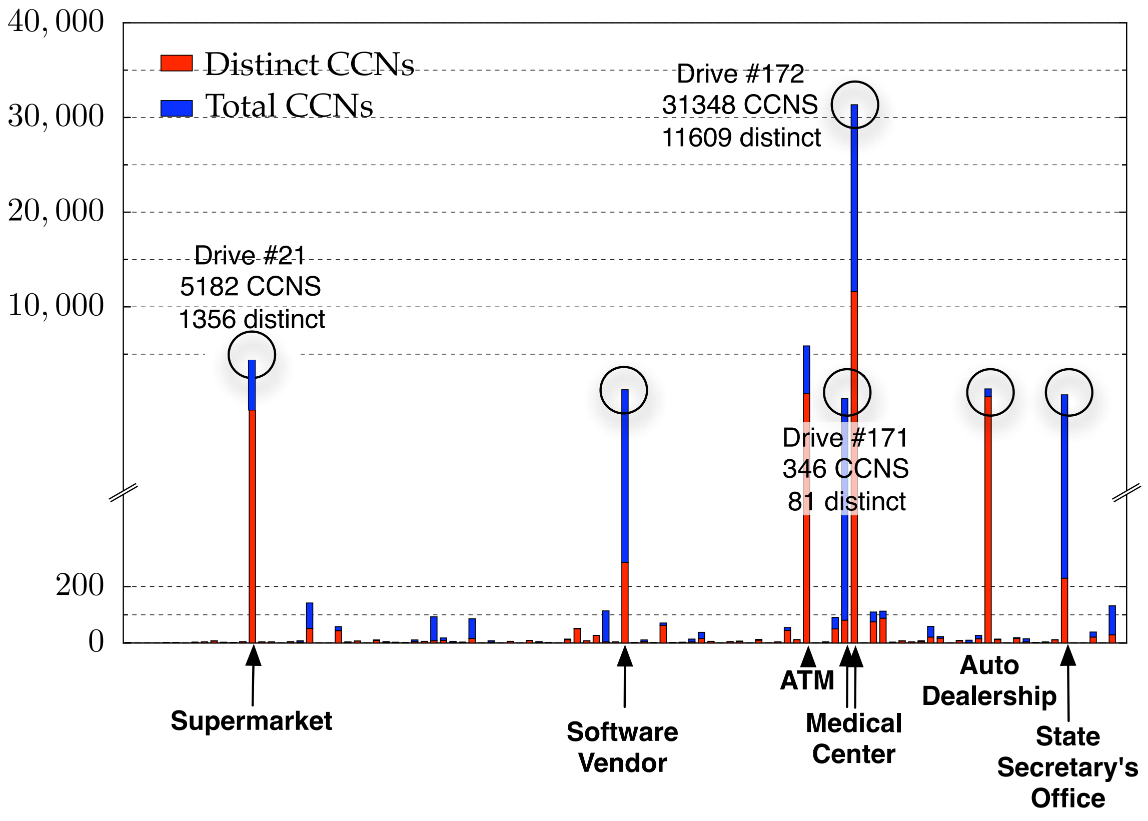

Fig. 2. This graph shows the number of unique credit card numbers and the total number of credit card numbers found on 250 different hard drives acquired on the secondary market. As can be seen most hard drives have relatively small number of credit card numbers. Some drives, however, have large numbers of credit card numbers. For the research project that produced this graph, outliers were examined to understand the process failures within each organization that resulted in the data being inadvertantly released. This same approach could be used to identify outlier drives inside an enterprise network that might be indicative of a hostile insider.

locally, but overall cluster variation is not minimized/optimized) [17].

b) Hierarchical clustering: is often preferred over partitional clustering, due to its ability to achieve improved cluster quality [17]. Hierarchical clustering uses a dendogram-based approach and either forms clusters (i.e. bottom-up, agglomerative approach) or divides clusters (i.e. top-down, divisive approach) by calculating pair wise item similarity. The computational expense and space requirements associated with such calculation, however, greatly limit its scalability. The computational expense of discriminative methods (those utilizing pair wise similarity measurements) is a quadratic function of the number of inputs being clustered: $O(n^2)$, where O is the data set containing items $\{o1, o2, o3 \ldots o(n)\}$ [13], [14]. In contrast, the computational expense of partitional approaches is a linear function of the number of inputs: $O(n)$ [13], [14].

c) Density-based clustering approaches: use density functions as local cluster criterion. They are advantageous over partitional clustering, in the sense that they handle noise better, can discover arbitrarily shaped clusters, and produce higher quality clusters [18]. Their primary disadvantage is computational expense and space requirements. Like hierarchical clustering, their computational cost is a quadratic function of the number of inputs, limiting their scalability to large data sets [13], [18].

d) Grid-based clustering: approaches quantize the data space into a finite number of grids, which subsequently form clusters. Such approaches are advantageous because they are insensitive to data input order and are computationally efficient, with run time scaling linearly with data set size, i.e., $O(n)$ [13]. The primary disadvantage of grid-based clustering approaches, however, is cluster quality/accuracy. Cluster quality is hindered by: (1) the simplicity of clustering computations; (2) the necessarily straight cluster boundaries that do not usually coincide with the inherent structure of the data; (3) relatively arbitrary assignment of data to clusters at or near grid vertices; and (4) difficulty in selecting the optimum grid resolution.

e) Model-based clustering: is the fifth category of clustering approaches. Model-based clustering is generative in nature. That is to say, the approaches assume parametric and probabilistic patterns exist in the data and clusters are modeled accordingly. Model-based clustering utilizes machine learning (e.g. neural networks) and/or statistical approaches (e.g. conceptual clustering and Bayesian clustering) [13]. Because of the probabilistic, parametric assumptions underlying the clustering process and mathematics, model-based approaches are advantageous for the following reasons:

- they are usually insensitive to input order;

- they are mathematically capable of detecting and handling outliers and generally noisy data;
- they can model non-spherical and/or unbalanced clusters, and
- they are designed to achieve global optimization (i.e. they minimize intra-cluster variance and maximize inter-cluster variance from a global perspective).

The computational expense of model-based clustering approaches varies between approaches, but is often higher order [19]. A notable exception to this is Kohonen's Self-Organizing Map (SOM) approach-an unsupervised neural network approach [20]. Implementations of SOM for clustering scale linearly $O(n)$ with data set size and sometimes even scale logarithmically $O(\log(n))$ [21]. They also handle noisy data well [20].

We are now experimenting with the a self-organizing neural network algorithm to partition the observation space and detect statistical storage anomalies, because the data will undoubtably be very noisy, and the inherent data structure and clusters will most likely be non-spherical, unbalanced, and highly skewed. A self-organized neural network algorithm, such as a Kohonen SOM or adaptation thereof, have the advantage of producing results that can be readily interpreted. SOMs use vector quantization and non-linear mapping to transform mutli-dimensional input into two-dimensional, navigatable output, that is easy to understand by the user. The user is able to understand why observations clustered together in low-dimensional space, whereas traditional neural networks act as black boxes, wherein the underlying structure of the cluster remains unclear. Since this application will alert analysts to possible threats via statistical anomaly detection, it is imperative that the analysts can interpret clusters of interest.

The threat detection engine (outlier analysis system) that we are building should dynamically *map* the organization's *normal* storage profiles. The map will be dynamic in the sense that *normal* will change over time as the organization changes, missions change, and as a result of daily changes in low-level operations. The clustering algorithm facilitates a dynamic mapping since it *learns* with each new input and each new mapping iteration. It will not *learn* from anomalies, as they are statistically identified as outliers due to their vector distance (also known as quantization error) from its best matching neuron (map node). The system will have the capability of re-introducing the observation and adapting the map, should the observation be deemed in fact normal. An important aspect of the research will be adapting and tunding traditional SOM algorithms to best fit the proposed context and empirically determining the most appropriate anomaly detection threshold for the analyst.

E. Feature Selection

Another critical design consideration in cluster-based outlier detection is feature selection. Features are the characteristics of an observation. Improper feature selection significantly compromises the effectiveness of the clustering algorithm. Selecting non-essential features reduces the signal to noise ratio and diminishes the ability of the clustering algorithm to detect anomalies. Omitting essential features leaves important classifiers out of the equation. Effective multi-dimensional clustering for outlier detection necessitates the right features be selected.

The research should empirically validate hypothesized features itemized previously in section 2A. Our hypothesized features fall into two categories: (1) media forensics features, and (2) user features. Media forensics features include those previously discussed, including total numbers of email addresses, credit card numbers, etc., allocated file type information and distribution, drive allocation percentage, document metadata, etc. User features may include organizational unit, job function, security clearance, access permissions, and other aspects specific to an organization.

We hypothesized the media forensics feature based on known facts from documented insider threat cases. For example, email remains one of the top 3 methods for exfiltrating data [22] - used in 30% of documented insider cases reviewed by the CERT Insider Threat Center [23]. Thus, we will look for email-based anomalies concerning addresses found on the hard drive. File type information (e.g. file extension) has proven relevant in insider cases, whether indicative of the type of data stored, or indicative of attempts to hide types of data stored [24]. File type distribution can also provide indication of inappropriate collections of graphic images and video files, which may be an exploitation risk, or an indication of covert channeling via steganography [24]. Abnormally large amounts of encrypted data, or unaccounted for spikes in the amount of encrypted data stored can also be an indicator of unauthorized data collection and exfiltration [24]. Past insider threat cases in the financial sector have demonstrated a common pattern of collecting personally identifiable information (PII) (e.g. social security numbers and credit card number), for the purpose of transforming the data into formats preferred by consumers of the data [25]. Last, past research has shown the value of user features, such as user role, in detecting breaches by insiders [26].

III. Status of Project

At the present time we are just at the beginning of this project. Work to date has consisted largely of making modifications to the *bulk_extractor* to allow its use in a distributed manner, and improvements to the file fragment type identification work so that we can more reliably characterize information sampled from the storage media.

As part of adopting *bulk_extractor* to work with the GRR Framework we have substantially cleaned the code, largely eliminating the use of global variables and improving the isolation between various program functions. The result of this effort will be splitting of the previously monolithic *bulk_extractor* program into several functional modules: one module samples the disk, one module runs the scanners, and finally a separate "plug-in" module for each scanner. Although we will continue to be able to run *bulk_extractor* as a monolithic program, we will also be able to use it as a library that can be called from the Python programming language.

As part of our effort to develop reliable technology for file fragment type identification we have attempted to replicate the work of Gopal *et al.* [28], Zhulyn *et al.* [27], and Beebe *et al.* [29]. The code Gopal *et al.* developed is algorithmically equivalent to using bigrams as features and Liblinear [30] "S

	S2			S3			c
Features	Train.Time	Pred.Time	accuracy	Train.Time	Pred.Time	accuracy	
unigrams	19m 9.518s	4.208s	55.99%	29m 46.439s	4.162s	48.20%	256
bigrams	5h 22m 21.391s	31.286s	68.12%	4h 34m 39.545s	32.649s	68.26%	1024
trigrams	174h 46m 5.795s	7m 47.676s	62.76%	211h 8m 47.311s	7m 23.068s	70.19%	1024
uni+bi	7h 39m 46.240s	36.019s	68.68%	3h 54m 36.043s	37.834s	67.06%	256
FS5	7h 51m 35.550s	35.111s	69.83%	7h 27m 34.618s	36.697s	68.92%	256

TABLE I. EFFICIENCY AND EFFECTIVENESS OF DIFFERENT FEATURE SETS FOR FILE FRAGMENT TYPE IDENTIFICATION WITH LIBLINEAR S2 AND S3. FS5 IS THE FEATURE SET OF CONCATENATED UNIGRAMS+BIGRAMS+ ENTROPY, KOLMOGROV COMPLEXITY, MEAN BYTE VALUE, HAMMING WEIGHT, AVERAGE CONTIGUITY BETWEEN BYTES, AND LONGEST BYTE STREAK [27]

3" (L2 regularized L1 loss function, dual solver). Across 52 file and data types, the result using liblinear that approximates the Gopal approach is 68.26% accuracy in our experimentation (see Table 1). We found significantly improved accuracy using trigrams instead of bigrams, but the training time was orders of magnitude longer. We are also developing an improved approach for training.

Overall, the results that we have seen to date are rather similar across loss functions/solvers, as well as feature sets. Nonetheless, we are currently concluding that the "FS5" feature set (unigram+bigram+ entropy+Kolmogrov complexity+mean byte value+Hamming weight+average contiguity between bytes+longest byte streak) and the L2 regularized L2 loss function, primal solver is the optimal approach. We believe that we can further improve our accuracy through the use of summarized pattern features and by eliminating untrainable data.

Summarized pattern features were introduced by Collins [31]. The basic idea is to reduce the diversity of particular features by mapping individual characters to character classes. For example, all lower case letters could be replaced with their upper case equivalent, all digits could be replaced with the digit "5", and so on. Although summarized patterns were originally developed for named entity recognition [32], Mayer applied them to file fragment identification with great success [33]. Mayer also developed an approach for detecting and training on a small vocabulary of relatively long n-grams in a corpus of exemplars. For example, although the space of 9-grams is too sparse to effectively train a SVM, adding the specific 9-gram "endstream" to the mix will be highly useful when training a SVM to detect PDF files.

We believe that training can be further improved by removing from the training set so-called untrainable data such as the zlib-compressed regions of PDF and Microsoft Open Office XML files, as well as the Huffman-encoded regions of JPEG files. We are currently developing a tool that will act as a filter and automatically remove such data from our training set.

IV. Acknowledgments and Disclaimer

The views expressed in this document are those of the authors and do not reflect the official policy or position of the Department of Defense, the Department of Homeland Security, or the U.S. Government. This research was funded in part under DHS S&T Proposal BAA 11-02-TTA 04-0108-I.

V. References

[1] Inspector General, Department of Defense, Defense Criminal Investigative Service, "Report of investigation," Jul. 25 2007. [Online]. Available: http://www.dodig.mil/fo/Foia/PDFs/OperationFlickerReportsJuly2010pdf.pdf

[2] A. P. Moore, D. Cappelli, T. C. Caron, E. D. Shaw, D. Spooner, and R. F. Trzeciak, "A preliminary model of insider theft of intellectual property," 2011.

[3] S. Garfinkel, "Digital media triage with bulk data analysis and bulk_extractor," *Computers & Security*, vol. 32, pp. 57–72, Feb. 2013.

[4] S. Garfinkel, A. Nelson, D. White, and V. Roussev, "Using purpose-built functions and block hashes to enable small block and sub-file forensics," in *Proc. of the Tenth Annual DFRWS Conference*. Portland, OR: Elsevier, 2010, pp. S13–S23. [Online]. Available: http://simson.net/clips/academic/2010.DFRWS.SmallBlockForensics.pdf

[5] J. Young, K. Foster, S. Garfinkel, and K. Fairbanks, "Distinct sector hashing for target detection," *IEEE Computer*, pp. 28–35, Dec. 2012. [Online]. Available: http://simson.net/clips/academic/2012.IEEE.SectorHashing.pdf

[6] S. Garfinkel, "Digital Forensics XML," *Digital Investigation*, vol. 8, pp. 161–174, Feb. 2012.

[7] M. Cohen, D. Bilby, and G. Caronni, "Distributed forensics and incident response in the enterprise," in *Proceedings of the 2011 DFRWS Conference*, 2011. [Online]. Available: http://dfrws.org/2011/proceedings/16-348.pdf

[8] I. F. Darwin, "Libmagic," Aug. 2008. [Online]. Available: ftp://ftp.astron.com/pub/file/

[9] S. L. Garfinkel, "Forensic feature extraction and cross-drive analysis," in *Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS)*. Lafayette, Indiana: Elsevier, Aug. 2006. [Online]. Available: http://www.dfrws.org/2006/proceedings/10-Garfinkel.pdf

[10] Associated Press, "Testimony expected on Neil Entwistle's computer activity the day of the murders of his wife, baby," Jun. 19 2008, last accessed Dec. 3, 2011. [Online]. Available: http://www.foxnews.com/story/0,2933,368604,00.html

[11] S. L. Garfinkel, "Design principles and patterns for computer systems that are simultaneously secure and usable," Ph.D. dissertation, MIT, Cambridge, MA, Apr. 26 2005.

[12] D. Cappelli, A. Moore, R. Trzeciak, and T. J. Shimeall, "Common sense guide to prevention and detection of insider threats 3rd edition–version 3.1," *Published by CERT, Software Engineering Institute, Carnegie Mellon University, http://www. cert. org*, 2009.

[13] H. Jiawei and M. Kamber, "Data mining: concepts and techniques," *San Francisco, CA, itd: Morgan Kaufmann*, vol. 5, 2001.

[14] S. Zhong and J. Ghosh, "Generative model-based document clustering: a comparative study," *Knowledge and Information Systems*, vol. 8, no. 3, pp. 374–384, 2005.

[15] Y. Zhao and G. Karypis, "Topic-driven clustering for document datasets," in *SIAM International Conference on Data Mining*, 2005, pp. 358–369.

[16] S. Guha, R. Rastogi, and K. Shim, "Rock: A robust clustering algorithm for categorical attributes* 1," *Information Systems*, vol. 25, no. 5, pp. 345–366, 2000.

[17] C. Lin and M. Chen, "Combining partitional and hierarchical algorithms for robust and efficient data clustering with cohesion self-merging," *IEEE Transactions on Knowledge and Data Engineering*, pp. 145–159, 2005.

[18] A. Hinneburg and D. Keim, “A general approach to clustering in large databases with noise,” *Knowledge and Information Systems*, vol. 5, no. 4, pp. 387–415, 2003.

[19] A. Roy, S. Govil, and R. Miranda, “A neural-network learning theory and a polynomial time rbf algorithm,” *Neural Networks, IEEE Transactions on*, vol. 8, no. 6, pp. 1301–1313, 1997.

[20] T. Kohonen, “The self-organizing map,” *Proceedings of the IEEE*, vol. 78, no. 9, pp. 1464–1480, 1990.

[21] P. Koikkalainen and E. Oja, “Self-organizing hierarchical feature maps,” in *Neural Networks, 1990., 1990 IJCNN International Joint Conference on*. IEEE, 1990, pp. 279–284.

[22] J. S. Okolica, G. L. Peterson, and R. F. Mills, “Using author topic to detect insider threats from email traffic,” *Digital Investigation*, vol. 4, no. 3, pp. 158–164, 2007.

[23] M. Hanley and J. Montelibano, “Insider threat control: Using centralized logging to detect data exfiltration near insider termination,” 2011.

[24] K. Roy Sarkar, “Assessing insider threats to information security using technical, behavioural and organisational measures,” *information security technical report*, vol. 15, no. 3, pp. 112–133, 2010.

[25] A. Cummings, T. Lewellen, D. McIntire, A. P. Moore, and R. F. Trzeciak, “Insider threat study: Illicit cyber activity involving fraud in the us financial services sector,” 2012.

[26] M. Maybury, P. Chase, B. Cheikes, D. Brackney, S. Matzner, T. Hetherington, B. Wood, C. Sibley, J. Marin, and T. Longstaff, “Analysis and detection of malicious insiders,” DTIC Document, Tech. Rep., 2005.

[27] O. Zhulyn, S. Fitzgerald, G. Mathews, and C. Morris, “Using nlp techniques for file fragment classification,” in *Proceedings of the 2012 DFRWS Conference*, 2012, pp. S44–S49. [Online]. Available: http://www.dfrws.org/2012/proceedings/DFRWS2012-5.pdf

[28] S. Gopal, Y. Yang, K. Salomatin, and J. Carbonell, “Statistical learning for file-type identification,” *Machine Learning and Applications, Fourth International Conference on*, vol. 1, pp. 68–73, 2011.

[29] N. Beebe, “Sceadanv1.0 systematic classification engine and data analysis,” Jun. 30 2012. [Online]. Available: http://www.dfrws.org/2012/proceedings/DFRWS2012-challenge-winner.pdf

[30] R.-E. Fan, K.-W. Chang, C.-J. Hsieh, X.-R. Wang, and C.-J. Lin, “LIBLINEAR: A Library for Large Linear Classification,” *Journal of Machine Learning Research*, vol. 9, pp. 1871–1874, 2008.

[31] M. Collins, “Ranking algorithms for named-entity extraction: Boosting and the voted perception,” *Proc. Association for Computational Linguistics*, 2002.

[32] D. Nadeau and S. Sekine, “A survey of named entity recognition and classification,” pp. 3–26, 2007, last accessed Dec. 3, 2011. [Online]. Available: nlp.cs.nyu.edu/sekine/papers/li07.pdf

[33] R. C. Mayer, “Filetype identification using long, summarized n-grams,” Master’s thesis, Naval Postgraduate School, Monterey, CA, Mar. 2011.