Chapter 6

A NEW APPROACH FOR CREATING FORENSIC HASHSETS

Marcelo Ruback, Bruno Hoelz and Celia Ralha

Abstract The large amounts of data that have to be processed and analyzed by forensic investigators is a growing challenge. Using hashsets of known files to identify and filter irrelevant files in forensic investigations is not as effective as it could be, especially in non-English speaking countries. This paper describes the application of data mining techniques to identify irrelevant files from a sample of computers from a country or geographical region. The hashsets corresponding to these files are augmented with an optimized subset of effective hash values chosen from a conventional hash database. Experiments using real evidence demonstrate that the resulting augmented hashset yields 30.69% better filtering results than a conventional hashset although it has approximately half as many (51.83%) hash values.

Keywords: Forensic hashsets, data mining, decision trees

1. Introduction

The amount of data stored on modern computer systems is increasing rapidly. According to Hinshaw [5], with the advent of the Internet, the amount of stored data is doubling every nine months, which is half the time specified by Moore's Law. Digital forensic investigations are directly affected by the massive increase in stored data. A major challenge for law enforcement agents is completing digital forensic analyses in a timely manner for prosecution [1].

A common method to reduce the amount of data to be analyzed in digital forensic investigations is to remove files that are clearly irrelevant, especially those related to general purpose applications, samples, templates and documentation. Another method is to extract files that have potentially incriminating content such as child pornography images,

G. Peterson and S. Shenoi (Eds.): Advances in Digital Forensics VIII, IFIP AICT 383, pp. 83–97, 2012.

malware and steganography applications [2]. However, to use these filtering methods, it is necessary to have a database of known files. In particular, the hash values [13] of known irrelevant files and known incriminating files are computed and stored in a "known files database" (KFDB). These hash values or "hashsets" are used to filter large forensic images for irrelevant and incriminating files, thereby reducing the scope of a forensic investigation.

The current approach is to update a known files database with hashsets for every new software release or update [7, 8]. However, it is very difficult to accomplish this task in a satisfactory manner given the great diversity of software. In addition, law enforcement and government entities in each country or geographical region must make strong efforts to monitor software specific to their languages and markets, and generate the required hashsets.

Storage media encountered in criminal investigations contain a good sampling of the software used in a particular country or geographical region as well as other files (e.g., child pornography images and anti-forensic applications) that are frequently used by criminals. As such, these storage media are excellent candidates for harvesting hashsets to be used in a known files database.

This paper describes the application of data mining (DM) techniques to identify irrelevant files from a sample of computers from a country or geographical region. The hashsets corresponding to these files are augmented with an optimized subset of effective hash values chosen from a conventional hash database. Augmenting the database with only effective hash values results in a leaner database with reduced computational overhead, but good filtering performance. Experiments using real evidence demonstrate that the resulting augmented hashset yields 30.69% better filtering results than a conventional hashset although it has approximately half as many (51.83%) hash values.

2. Forensic Hashsets

Cryptographic hash functions are widely used in digital forensics to uniquely identify files. This enables files to be filtered using databases of known file hashes [2, 8]. The principal digital forensic tools, FTK, EnCase and Sleuth Kit, group files into two categories based on file hashes. The first category comprises the "known" or "ignorable" files that are part of common applications and are of no specific interest to a forensic investigation. The second comprises "alert" or "notable" files that may be of interest to an investigation because of their content or because they are associated with applications that are commonly used to hide information or hinder forensic investigations (e.g., software encryption, steganography and file wiping software).

The most prominent known files database is the National Software Reference Library (NSRL) [9]. The NSRL contains file hashes from various sources and is available in the form of a reference data set (RDS) that is published quarterly by the National Institute of Standards and Technology (NIST). The July 2011 release of NSRL RDS (version 2.33) contains 20,129,213 hashes of known files.

Other well-known databases are HashKeeper (now defunct), which was maintained by the U.S. National Drug Intelligence Center; and the Known File Filter (KFF) library provided by AccessData.

Most forensic tools allow the addition of new entries to a known files database as well as modifications to the hashset classification (ignorable/notable). However, the manipulation of hash database data is often difficult and expensive.

Hashsets can also be obtained from other sources, including vendors, law enforcement agencies and government entities. In larger organizations, there is a need to constantly update hashsets to obtain uniform results and maintain quality control. To accomplish this task effectively, a central database that serves as reference for the entire organization could be created.

However, as Roussev, *et al.* [12] note, the computational cost of searching for known file hash values can be prohibitive:

> *"In digital forensics, success has come from using databases of hash values of known system and application files, such as those maintained by NIST. But it is debatable if this approach will work when the databases contain billions of hash values – would it be necessary to compute clusters just to perform hash searches?"*

For this reason, implementing one all-encompassing database to maintain hash values is not feasible. Methods must be implemented to extract only the hashsets that are effective at filtering known files in forensic investigations.

3. Related Work

The absence of suitable known files databases for use in non-English speaking countries is a major problem. Kim, *et al.* [7] have presented an approach for building a reference data set for use in South Korea. Their approach involves two phases: (i) study the effectiveness and consistency of data from the NSRL; and (ii) create a consolidated Korean RDS from NSRL data complemented with metadata and file hashes associated with Korean software. However, a deficiency with the Korean RDS and the NSRL is that they are based on metadata and hash values of known files that are extracted directly from software installation packages. Our approach attempts to address this deficiency by using data mining to improve the quality of a hash database without having to rely on software packages.

Other related work by Hoelz, *et al.* [6] uses distributed agents to reduce the scope of a KFF search. However, our approach, which uses data mining, represents an improvement over the multi-agent system approach.

4. Data Mining and Digital Forensics

Beebe and Clark [1] have emphasized the need to use data mining techniques to deal with the large volumes of data encountered in digital forensic investigations:

> *"Terabyte-sized data sets are already challenging analysts and investigators. Therefore, an active stream of research extending data mining research to digital forensics and digital investigations is desperately needed."*

The term "data mining" is used by many researchers as a synonym for "knowledge discovery in databases," primary because data mining is a key step in knowledge discovery from databases [11]. According to Fayyad, *et al.* [3], knowledge discovery is the process of identifying new, non-trivial, valid, understandable and potentially useful patterns in data. The process of knowledge discovery, which is interactive and iterative, involves several steps: (i) selection; (ii) preprocessing; (iii) transformation; (iv) data mining; and (v) interpretation.

The primary goals of data mining applications are prediction and description [3]. Prediction is focused on finding patterns in data in order to build a model that can predict future values of variables. Description involves finding patterns that can describe the mined data in a human-understandable format.

Classification is a prediction method in data mining that can be formalized as the task of obtaining a target function f that maps each attribute set x to a pre-defined class label y [14]. A classification method typically uses a learning algorithm to identify the model that best describes the relationship between the set of attributes and the resulting class label. The input data used in the learning process is usually divided into two parts. One is the "training set," which is used as input to the learning algorithm to construct the classification model. The other is the "test set," which is used to evaluate the accuracy of the results and to validate the model generated using the training set.

Decision trees are a powerful set of algorithms for learning and classification [4]. These algorithms can handle high dimensionality data and facilitate the exploratory discovery of knowledge. Representing knowledge in the form of a tree is intuitively appealing. Moreover, the learning and classification phases are simple and fast.

5. Proposed Approach

Our proposed approach attempts to improve the effectiveness of a known files database by implementing three methods based on file relevance. The first is the application of data mining techniques to identify new ignorable files in a sample of computers from a particular country or geographical region. The second is the selection of hashsets from conventional hash databases by considering the files that effectively represent the software used in the given country or geographical region. The third is the use of the most recent hashsets from conventional hash databases based on their statistical significance.

Computers encountered in criminal investigations are good candidates for the sample set. Since these computers are drawn from diverse locations and segments of society, they should offer a good representation of commonly used software. Of course, it is important to ensure that the files in the sample computers are not infected by malware. The files collected from the sample set can be used to identify new irrelevant files and to help separate files associated with obsolete and rarely used software from conventional hashsets.

A data mining process is used to find patterns in the data and to create a model that allows the classification of the sample files according to their relevance. The hash values of files that are identified as irrelevant may be added to the database as ignorable files.

Knowledge supplied by digital forensic experts is used to identify the relevant attributes for the classification model. Also, expert knowledge is used to classify sample files for the training set and test set required by the decision tree (DT) algorithm. The result is a leaner but more effective known files database that increases the number of ignorable files that are filtered while reducing the computational overhead.

The following sections describe the approach in detail. In particular, the sections discuss the preliminary tests performed to assess the feasibility of identifying new extraneous files, the solution architecture, and the decision support process that takes into account file relevance.

5.1 Preliminary Study

If identical versions of a file are found on multiple unrelated computers, then there is a high probability that the file is a common one that does not add useful information to an investigation. If, the file metadata

Table 1. Preliminary results.

Description of Files	**Total**
1. Total unique files	871,590
1.1 Filtered as "ignorable"	79,473
2. Total unique files present in more than one piece of evidence	58,541
2.1 Filtered as "ignorable"	18,692
2.2 Filtered as "alert"	43
2.3 "Unknown"	39,806
2.3.1 Temporary Internet files	**4,078**
2.3.2 Other files	**35,728**

corroborates the hypothesis that the file is irrelevant, then the file is a strong candidate for inclusion in the database as an ignorable file.

To evaluate this assumption, a preliminary study was conducted to check the number of files encountered in forensic investigations that could be deemed as ignorable. In this study, hard drives from nineteen criminal cases investigated by the Brazilian Federal Police were examined. The NSRL RDS 2.31 and KFF databases were used to classify the files in the hard drives as "ignorable," "alert" or "unknown."

Table 1 presents the results of the study. Note that a total of 58,541 unique files were present in multiple pieces of evidence, but only 18,735 of these files were identified as ignorable or alert by the NSRL and KFF databases. In other words, nearly 68% (39,806 of 58,541) of the files present in multiple evidence sources were not filtered by NSRL or KFF. Clearly, a data mining approach could be used to improve the classification of ignorable (and alert) files, which would enhance the effectiveness of file filtering.

5.2 System Architecture

Our approach utilizes three hash databases of ignorable files: (i) a database containing ignorable files and ignorable temporary Internet files identified by our data mining process; (ii) a database containing effective hashsets from conventional known files databases; and (iii) a database containing the most recent hashsets from conventional known files databases.

Figure 1 provides an overview of the data mining process used to create the first database. Files from a sample set of computers are analyzed and correlated with evidence and case information. This information is input to the data mining process along with the files reviewed and classified by digital forensic experts as "alert" files, hashsets from exter-

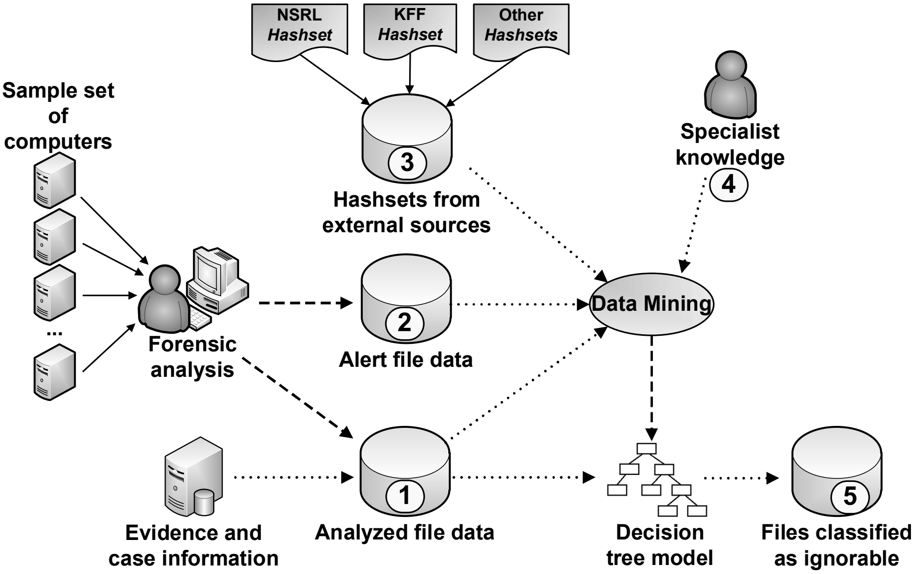


Figure 1. Identifying ignorable files from a sample set of computers.

nal sources and specialist knowledge. The resulting decision tree model identifies the files that should be marked as "ignorable."

The second database, which comprises effective hashsets, is derived from hashsets present in conventional known files databases. Only the hashsets that are referenced by files from the sample set of computers are included in the database.

The third database contains hashsets that were recently added to conventional known files databases. These hashsets typically correspond to brand new software releases. Specialist knowledge is required to assess "recency" and determine the specific hashsets that should be included in the database. Note that this database complements the database of effective hashsets because it includes files that are new, but not yet statistically significant in the evidence sample.

The final known files database used to filter files consists of the three databases listed above. We refer to this composite database as the "proposed hashset database" (PHDB).

5.3 Knowledge Discovery Process

Figure 2 illustrates the knowledge discovery process used to generate the decision tree model that ranks the relevance of files in the sample set. The process comprises the six knowledge discovery steps described by Fayyad, *et al.* [3]. Steps 1 and 2 correspond to selection and trans-

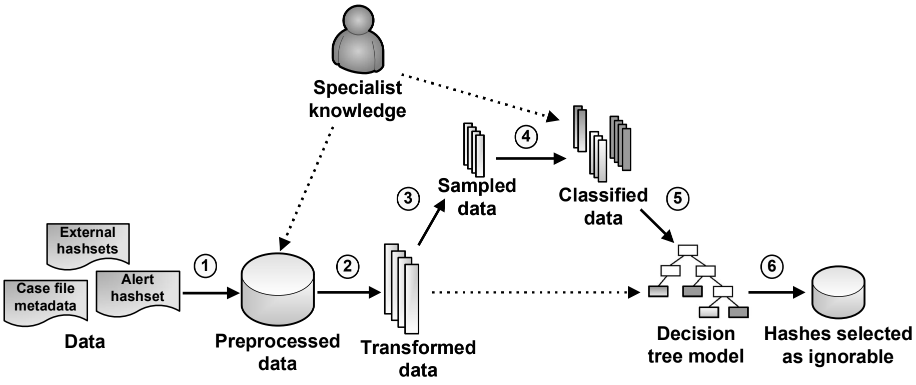


Figure 2. Knowledge discovery process.

formation. Steps 3 through 6 are related to the data mining process itself, and cover sampling, classification, induction and the application of the decision tree model. Note that expert knowledge is of paramount importance in forensic investigations. This knowledge is used to model and create the database. Also, knowledge specific to operating systems is incorporated to enhance classification.

- **Step 1. Selection and Preprocessing:** The first step is to collect, extract, transform and load all the data into a previously modeled database. The files present in the sample set are collected with the help of forensic tools that, in addition to extracting the metadata from the file system, include relevant information about file contents, including (i) its type based on signature analysis; (ii) correspondence between the type signature and type extension; and (iii) properties such as compression and encryption. All this information collected in the "evidence files sample database" (EFSDB).

- **Step 2. Transformation:** The second step is to transform and consolidate the preprocessed data. The data mining process is used to rank the relevance of each hash value that is related to one or more occurrences of a file in the sample. Therefore, data should be consolidated and grouped for each hash value and all the attributes should be expressed as a percentage of the total number of files that have the same hash value in the database.

- **Step 3. Sampling:** The EFSDB can contain hundreds of thousands, even millions of distinct hash values. These values should be classified by an expert in order to be used by the decision tree algorithm. A subset of the complete database was used to simplify the experiments and reduce the computational overhead.

- **Step 4. Sampled Data Classification:** The files in the sample are manually classified by forensic experts into three categories: relevant (R), ignorable (I), and ignorable temporary Internet cache (T).

- **Step 5. Decision Tree Model Induction:** The files are divided into the training and test sets used to derive and test the model created by the decision tree algorithm.

- **Step 6. Decision Model Application:** The decision tree model is used to classify files (identified by their hash values) and identify ignorable files in the EFSDB.

The data mining process can be adjusted to classify the relevance of files based on a greater or lesser degree of confidence. For example, a minimum percentage of repetitions of a file in different media could be established for the file to be considered ignorable. Actions that represent positive adjustments to the classifications made by the decision tree model are: (i) increasing the size of the sample; (ii) enhancing the validity of the classification results by drawing on multiple experts; and (iii) performing detailed data preparation.

6. Experimental Results

Experiments were conducted to validate the data mining approach for improving hashsets. The experimental sample comprised 100 computers from actual cases. A total of 4,045,808 files constituting the EFSDB were extracted using FTK. Several open source tools were used for knowledge discovery. They included Pentaho Data Integration for Step 1 in the knowledge discovery process, PostgreSQL for the database systems, RapidMiner for data mining, and NSRL RDS 2.32 for identifying known files in the sample and as a PHDB benchmark.

After the data in the EFSDB was consolidated, a sample of 12,528 file entries was chosen from the 242,557 unique hash values that appeared in at least two distinct samples. The files in the sample that were not identified by the NSRL hashset were manually classified by a forensic expert. The manually classified files were used as input to the C4.5 algorithm [10] that created the decision tree model.

Figure 3 presents a portion of the decision tree model obtained after executing the C4.5 algorithm. In particular, it shows the top branch of the decision tree for the files whose *percent_internetcachefolders* values

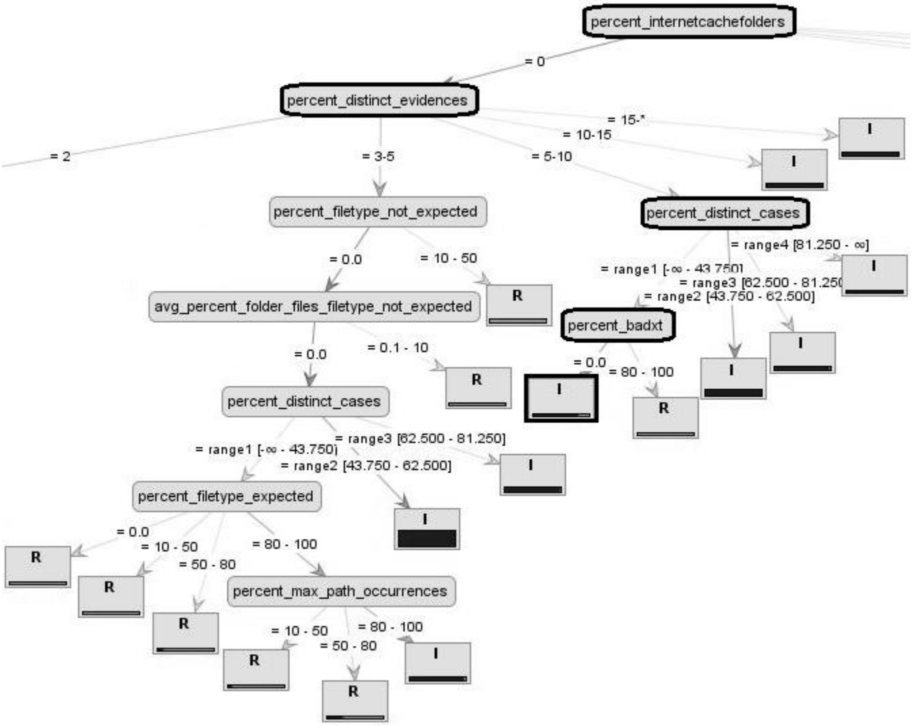

percent_internetcachefolders
= 0
percent_distinct_evidences
= 15-*
= 10-15
I
= 2
= 3-5
= 5-10
I
percent_filetype_not_expected
percent_distinct_cases
= range4 [81.250 - ∞]
= 0.0
= 10 - 50
= range1 [-∞ - 43.750]
= range3 [62.500 - 81.250
I
= range2 [43.750 - 62.500]
avg_percent_folder_files_filetype_not_expected
R
percent_badxt
I
= 0.0
= 0.1 - 10
= 0.0
= 80 - 100
I
R
I
percent_distinct_cases
R
= range3 [62.500 - 81.250]
= range1 [-∞ - 43.750]
= range2 [43.750 - 62.500]
I
percent_filetype_expected
I
= 0.0
= 10 - 50
= 80 - 100
R
= 50 - 80
R
percent_max_path_occurrences
R
= 10 - 50
= 80 - 100
= 50 - 80
R
I
R

Figure 3. Sample decision tree.

are zero, not including those files whose *percent_distinct_evidences* values are equal to two. The rules used in the decision tree model were examined by experts and deemed to be appropriate.

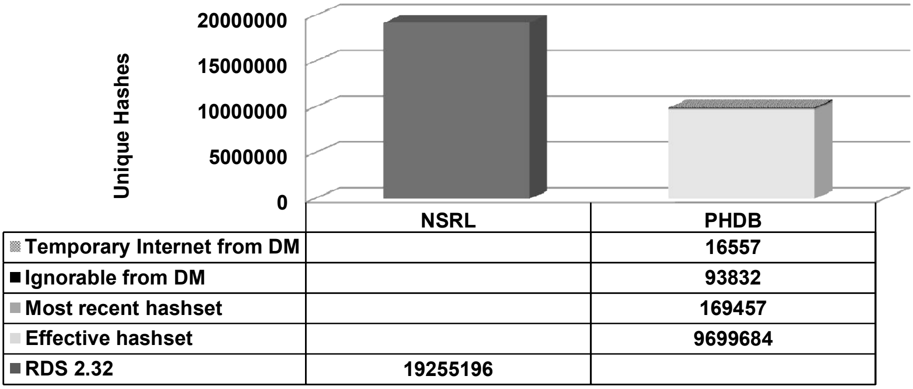
The parameters used to generate the decision tree model were chosen empirically after several tests. The parameters and their values are: minimal size of a node to allow a split = 12; minimal leaf size = 6; maximal tree depth = 10; and number of pre-pruning alternatives = 6. The training and test sets were selected using a bootstrapping method with five validations and a sample ratio of one. According to Tan, *et al.* [14], at each validation, the bootstrapping method chooses the training set items with replacement; the remaining items are incorporated in the test set.

The decision tree classification process starts at the root node, traverses the tree in a top-down manner and the following branch is chosen based on test conditions for each internal node. To clarify the process, consider the highlighted leaf node in Figure 3. This node classifies as ignorable (I) all the files whose *percent_internetcachefolders* values equal zero, *percent_distinct_evidences* values are between 5% and 10%, *per-*

Table 2. Decision tree attributes.

Attribute	**Meaning**
percent_internetcachefolders	Percentage of files with the given hash value found in Internet cache folders
percent_distinct_evidences	Percentage of total distinct pieces of evidence where files with the given hash value were found
percent_distinct_cases	Percentage of total distinct investigation cases where files with the given hash value were found
percent_badxt	Percentage of files with the given hash value whose name extension do not match the file signature

cent_distinct_cases values are between 0% and 43.75%, and *percent_badxt* values equal zero. Table 2 lists the meanings of these decision tree attributes.



	NSRL	PHDB
▨ Temporary Internet from DM		16557
■ Ignorable from DM		93832
■ Most recent hashset		169457
□ Effective hashset		9699684
■ RDS 2.32	19255196	

Figure 4. Total unique hash entries in the NSRL RDS 2.32 and PHDB hashsets.

Figure 4 gives details about the proposed hashset database (PHDB), which was created by applying the decision tree model to the evidence files sample database (EFSDB) to identify ignorable hashes, and then adding the effective and recent hashsets from the NSRL RDS 2.32 database. The final PHDB contains 9,979,530 unique hashes, only 51.83% of the number in the NSRL RDS 2.32 database.

Files and metadata extracted from nine computers were used to compare the identification results obtained using the PHDB and NSRL RDS 2.32 hashsets. The nine computers were also involved in actual investigations, but were not part of the initial sample set.

Table 3 presents the results of the comparison. The *HD* column lists the hard drives from the nine computers used in the comparison. Column *A* lists the number of unique files in the nine drives. Column *B* lists the

Table 3. Comparison of results using the PHDB and NSRL RDS 2.32 hashsets.

HD	**Files** **(A)**	**NSRL** **(B)**	**C1**	**C2**	**C3**	**C4**	**PHDB** **(C)**	**% Gain** **(C vs. B)**
A	54,770	12,794	12,578	5	309	1,206	14,093	10.15
B	71,570	19,394	19,195	28	4,163	1,210	24,568	26.68
C	197,874	22,445	22,443	29	2,072	1,889	26,404	17.64
D	95,118	31,814	31,740	1,665	13,090	1,074	45,904	44.29
E	189,867	81,780	77,018	1,666	20,142	507	97,667	19.43
F	80,394	11,938	11,933	10	4,369	307	16,609	39.13
G	144,219	12,747	12,702	23	7,064	1,071	20,837	63.47
H	82,709	20,693	20,690	17	5,762	783	27,235	31.61
I	45,898	1,018	1,018	282	5,687	464	7,169	604.22
	962,419	214,623	209,317	3,725	62,658	8,511	280,486	30.69

number of ignorable files based on the NSRL RDS 2.32 hashset. Column C lists the number of ignorable files based on the PHDB hashset. Note that $C = C1 + C2 + C3 + C4$ where $C1$ denotes the effective hashsets, $C2$ the recent hashsets (all files identified by $C2$ were also identified by $C1$), $C3$ the ignorable files based on data mining, and $C4$ the ignorable temporary Internet files based on data mining (Figure 4).

According to Table 3, the overall performance using the PHDB hashset is 30.69% better than the performance obtained with the NSRL RDS 2.32 hashset, when measured in terms of percent gain ($(C-B)/B \times 100$). This improvement is seen despite the fact that the PHDB hashset has just 51.53% of the number of hashes as the NSRL RDS 2.32 hashset (9,699,684 vs. 19,255,196) as shown in Figure 4.

Figure 5 also shows the results comparing the performance obtained using the PHDB and NSRL RDS 2.32 hashsets for files recovered from the nine test computers.

The PHDB entries were also checked by experts as part of the experiment. The experts verified that all 62,658 files identified as ignorable in the PHDB hashset were in fact ignorable. As an example, Table 4 presents some attributes of files extracted from Computer E whose SHA-1 hash values start with `0x000` and were identified as ignorable using the PHDB hashset, but not ignorable using the NSRL RDS 2.32 hashset.

7. Conclusions

Current methods for creating and maintaining known files databases, especially in non-English speaking countries, are not very effective. A promising solution is to harvest hashsets by using data mining techniques

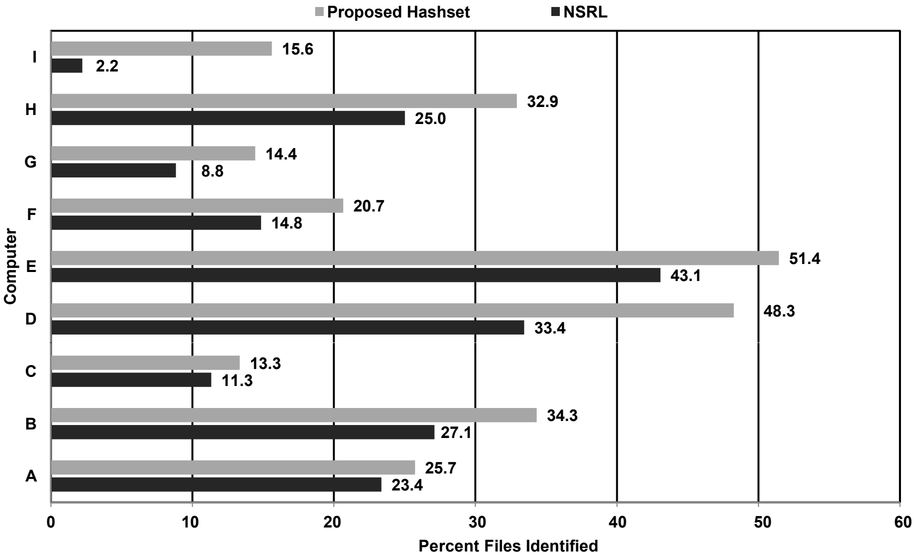


Figure 5. Total unique hash entries in the PHDB and NSRL hashsets.

on storage media encountered in criminal investigations in a particular country or geographical region. These storage media contain a good sampling of software as well as other notable files that are frequently used by criminals.

Experiments demonstrate that excellent filtering results are obtained using a composite known files database comprising hashsets harvested by data mining, and effective and recent hash values from a conventional hash database. In particular, the composite hashset yields 30.69% better filtering results than a conventional hashset although it has approximately half as many (51.83%) hash values.

Our future work will refine the data mining approach and incorporate additional expert knowledge to enhance the decision tree model, with the ultimate goal of creating leaner, but highly effective, known files databases.

References

[1] N. Beebe and J. Clark, Dealing with terabyte data sets in digital investigations, in *Advances in Digital Forensics*, M. Pollitt and S. Shenoi (Eds.), Springer, Boston, Massachusetts, pp. 3–16, 2005.

[2] S. Bunting, *EnCase Computer Forensics – The Official EnCE: EnCase Certified Examiner Study Guide*, Sybex, Hoboken, New Jersey, 2007.

Table 4. Sample files identified as ignorable using the PHDB hashset.

Name	Path	Type
x86_netfx-shfusion_dll_b03f5f7f11d50a3a_6.0.6002.18005_none_5ab3d7c5cc906c6a.manifest	\Windows\SoftwareDistribution\Download\829570003409c6fd8183f615cab83b0b\	XML
mstscax.dll	\Windows\winsxs\x86_microsoft-windows-t..s-clientactivexcore_31bf3856ad364e35_6.0.60 00.21061_none_2e516291e1cf33e3\	Executable
drvmain.sdb	\Windows\winsxs\x86_microsoft-windows-a..ence-mitigations-c5_31bf3856ad364e35_6.0.6001.18165_none_0be a5d21f274f4a9\	Unknown
WsmCl.dll	\Windows\System32\	Executable
NlsLexicons0414.dll	\Windows\winsxs\x86_microsoft-windows-naturallanguage6_31bf3856ad364e35_6.0.6000.16710_none_9be9c78e2d9d5d54\	Executable
MMSUIPlugin_001.ico	\Program Files\TIM Web Banda Larga \plugins\MMSUIPlugin\	Unknown
SMSUIPlugin_001.ico	\TIM Web Banda Larga\plugins \SMSUIPlugin\	Unknown

[3] U. Fayyad, G. Piatetsky-Shapiro and P. Smyth, From data mining to knowledge discovery: An overview, in *Advances in Knowledge Discovery and Data Mining*, U. Fayyad, G. Piatetsky-Shapiro, P. Smyth and R. Uthurusamy (Eds.), AAAI Press, Menlo Park, California, pp. 1–34, 1996.

[4] J. Han and M. Kamber, *Data Mining: Concepts and Techniques*, Morgan Kaufmann, San Francisco, California, 2006.

[5] F. Hinshaw, Data warehouse appliances: Driving the business intelligence revolution, *Information Management*, vol. 14(9), p. 30, 2004.

[6] B. Hoelz, C. Ralha and R. Geeverghese, Artificial intelligence applied to computer forensics, *Proceedings of the ACM Symposium on Applied Computing*, pp. 883–888, 2009.

[7] K. Kim, S. Park, T. Chang, C. Lee and S. Baek, Lessons learned from the construction of a Korean software reference data set for digital forensics, *Digital Investigation*, vol. 6(S), pp. S108–S113, 2009.

[8] S. Mead, Unique file identification in the National Software Reference Library, *Digital Investigation*, vol. 3(3), pp. 138–150, 2006.

[9] National Institute of Standards and Technology, National Software Reference Library, Gaithersburg, Maryland (`www.nsrl.nist.gov`).

[10] J. Quinlan, *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, California, 1993.

[11] L. Rokach and O. Maimon, *Data Mining with Decision Trees: Theory and Applications*, World Scientific, Singapore, 2008.

[12] V. Roussev, G. Richard and L. Marziale, Class-aware similarity hashing for data classification, in *Advances in Digital Forensics IV*, I. Ray and S. Shenoi (Eds.), Springer, Boston, Massachusetts, pp. 101–113, 2008.

[13] B. Schneier, *Applied Cryptography*, John Wiley, New York, 1995.

[14] P. Tan, M. Steinbach and V. Kumar, *Introduction to Data Mining*, Addison-Wesley, Boston, Massachusetts, 2005.