RAMS

Reliability, Availability,
Maintainability, and Safety

This is the Title of my Thesis

Your Name

December 2012

PROJECT THESIS

Department of Production and Quality Engineering

Norwegian University of Science and Technology

Supervisor 1: Professor Ask Burlefot

Supervisor 2: Professor Fingal Olsson

Contents

Chapter 1

Introduction

File recovery from digital data storage devices has been a hot topic among the Digital Forensics field. Traditional data storage devices make use of a file system, in order to manage contained data, their available space and to maintain location of files. When the storage device and their file system are intact, it is quite simple to recover data from them. This is mainly because file systems make use of meta-data in order to track information for their files. Meta-data can contain information such as creation date, data struc- ture (e.g directory or regular file), file type, file owner, size, last modified date etc. In a real life forensic case it is highly unlikely that file meta-data will be present, or they might be corrupted or deleted. It became clear for the digital forensic community that an alternative, more realistic approach must be used.

1.1 File Carving

File carving is a forensics technique that recovers files based on their content, without relying on their meta-data. File carving process involves two steps. File validation and file reconstruction.[1]. During the recovery procedure, we must first validate the type of the file and then apply the appropriate reconstruction technique. In this thesis, only the validation techniques are of our interest. By examining the byte-content and/or the structure of a file [22], file validation techniques are used to classify its type. Several file types contain common structures like headers, footers (named Magic Number Matching) [7][3], fields that specify file attributes like color or size etc.(Data Dependency Resolving [3]), that can be used to identify the type of the file.

Additionally, another approach is to apply statistical analysis techniques and algorithms, which use the complete byte set of a file, creating a fileprint for every file type. Some examples are the n-Gram Analysis [9], the Byte fre- quency analysis (BFA) algorithm and the Byte frequency cross-correlation (BFC)[4]. The aforementioned techniques have some profound weaknesses. The Magic Number Matching and the Data Dependency Resolving approaches make general type classification infeasible. This is due to the fact that not every file-type contain such structures. Furthermore, n-Gram Analysis and both BFA and BFC were designed to be applied in a complete file or a pre-defined part of it, which retains all of its content. Hence, they depend on files internal structure and characteristics.

1.2 Problem Formulation

So why this is a problem? The answer lies in file systems behaviour and file fragmentation. When we delete a file from a media storage, the data are not actually removed. The sectors in which the file was stored still contain the same data, although the file system marks them as unallocated [2]. Which means that the next time a new file is created, the file system is free to use these sectors, which are marked as unallocated, to store the new file. But if the new file is bigger than the old one, and the file system tries to store it starting from the same sector entry as the deleted one, it wont have enough space to store it. So the file system will allocate all the sectors of the previous deleted file, while the remaining data which do not fit, will be stored to other unallocated sectors. This results to file fragmentation. In a forensic file recovery case, it is more probable that the files that must be recovered are fragmented. Validation techniques which use the complete file content are high unlikely to provide aid to forensic examiners. Hence, an alternative approach to file type classification must be taken. File fragment classification is a technique that uses only a small fragment of a file, in order to determine its type. Ergo, file fragmentation is not a problem any more as this approach is independent from files overall structure. Although in theory file fragment classification looks like an ideal approach, in practice current solutions that use this approach could not yield good results[6][22]. One reason that file fragment classification is difficult, is due to the complex container files. Complex container files like TAR, ZIP, RAR, PDF etc. contain other primitive file types, making general fragment

classification difficult. Moreover, a fragment might contain more data which are strongly related to the files content than the files structure.

1.3 Objectives

The main objectives in this project are:

1. Test the hypothesis that by analysing only the printable ASCII characters plus the tab, newline and carriage return character, accuracy of classification algorithms can be enhanced for document-type fragments.

2. Create a more accurate algorithm for identifying document-type fragments than the available ones. In particular text, xls, doc and pdf files are of our interest and we try to improve their classification accuracy.

1.4 Algorithms Requirements

The design requirements for our classifying algorithm are as follows:

1. Speed - Relatively fast compared to current techniques

2. Accuracy - Algorithm should be as accurate as possible by minimizing false positives in classification of file fragments

3. Operate in 512 bytes, same as the sector size of a hard drive

1.5 Approach

Most of the current techniques on fragment classification use the full byte content of a file for both the training and classification procedures. Since we intend to create an algorithm which would be able to yield better accuracy results for fragments that originate from a document file type, we want to test the hypothesis that by using only the printable ASCII characters (32 >= b <= 126) plus the tab, newline and carriage return of a fragment we could achieve better results

regarding text fragment classification. The aforementioned special characters are a behavioural trait of a document so we expect that their occurrence in conjunction with all the other printable characters will be more frequent in a document file.

In order to test our hypothesis and our results we have to use one of the current classification algorithms. Additionally, since our main goal is to design a classification algorithm which will satisfy the previously mentioned requirements, we should carefully choose a currently available algorithm with the potential of modifying and creating a custom more effective version of it.

Our algorithm of choice is the Byte Frequency Analysis algorithm. More about the reasons of this choice can be found in the next chapter.

It has been observed that BFA, although extremely inaccurate, classifies a big amount of fragments that belong to a document file as text. We will make use of the BFA which will take under account only our special subset of ASCII characters among with 3 more variations of it and try to enhance its accuracy on classifying document-file fragments as text. We used 4 different variations of the BFA mainly for 2 reason. The first one is to compare the results with the results of the BFA that Shahi used for file fragment classification and find out if our hypothesis is correct. The second one is to choose the variation of it that yields the best results regarding text fragment classification. Literally this is going to be the first step of our algorithm so accuracy of the BFA will affect the accuracy of our final algorithm.

Next we will isolate all fragments classified as text and analyse them in order to find patterns which will eventually result in special metrics that could help us to design our algorithm.

Chapter 2

Related Work

2.1 Related Work

Here I pu th related work

Chapter 3

Experimental Setup

3.1 Data Set

The data set we used for our training and testing is derived from Garfinkels[] coprus, Wikipedia and Academic Earth[] and is the exact same coprus that Shahi[] used for his testing set.The set is comprised of 10 different file types with a size of about 1GB each. We split this corpus in a 9-1 ratio for the training and testing set respectively. Furthermore, we divided both the testing and training set files in 512-byte blocks, which we refer to them as fragments. We used the training set to train our fingerprints and apply statistical analysis in order to discover useful patterns and the testing set to test all variations of our BFA algorithm. More detailed information about our data set can be found in Table X.

3.2 Byte Frequency Analysis(BFA) Algorithm

BFA [McDaniels] is a statistical learning algorithm that was initially developed to analyse and classify whole files. It was not meant to be used in file fragments. By counting the number of instances of each byte in a file of a certain type, BFA uses this frequencies to create a representative average value for each byte instance among with their respective correlation strength. This results in a fingerprint for this particular file-type. Then during the classification procedure, the input file is compared with every fingerprint and an accuracy level is created for each file type. BFA classifies the file to the file type that corresponds to the highest accuracy level. Shahi

trained and tested BFA with file fragments of 512-byte size and his results show that although the algorithm is pretty bad for broad fragment classification, it is quite good in identifying fragments that belong to document-type files as text. We use a BFA which will train our fingerprints with the bytes that corresponds to the printable ASCII characters plus the tab, line break and carriage return instead of the complete byte-set of the fragments. This BFA will be only the first step of our algorithm and we intend to use additional metrics after this point. Taking under account the speed requirement,BFA seems as a very good candidate since it is quite a lightweight technique compared to heavier machine learning algorithms. Moreover, as we already stated, BFA seems to classify a big ammount of document-type fragments as text. Shahi tested several classification algorithms in the same corpus. BFA was also tested among with Byte Frequency Correlation algorithm, n-Gram Analysis and Conti et al. algorithm. The results show that BFA has the highest accuracy at classifying document-type fragments as text but the most important attribute of it is that it also has fewer false positives.

Chapter 4

BFA Variations

4.1 Variation 1 - Training with our special ASCII subset

In this variation we created 10 fingerprints which were trained using the training set, one for each file type. We used only the printable ASCII characters (32 b 126) among with the tab(9), new line (10) and the carriage return(13) characters. The results can be found in table 4.1. This variation of BFA classifies 589,758 fragments as text which corresponds to the 30.4% of the initial corpus. 501,012 of them are fragments that come from pdf, xls, doc and text files and 88,746 fragments originate from the other 6 file types. This means that in the set that is classified as text we have an 85% true positive of identifying document-type fragments as text with 15% false positives. This 85% of true positives corresponds to the 66.7% from the total pdf, xls, doc and text files of our corpus.

4.2 Variation 2 - 4-Ratio categories of our special ASCII subset

During our research we thought that it would be interesting to analyse the distribution of bytes that belong to our special ASCII subset in the fragments of our training set. Depending on the percentage of our special ASCII subset in a fragment, the fragment was assigned to one of 4 ratio categories. 0-25%, 25-50%, 50-75% and 75-100%. The results of this analysis can be found in table 4.2. As it seems fragments from certain file types are more likely to belong to certain ratio categories. For example almost all text fragments(99.95%) contain more than 75% of our

Table 4.1: BFA Results - Fingerprints with printable ASCII characters

	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
num.of fragments	189,732	204,795	190,055	177,887	204,728	193,352	195,608	195,608	195,656	195,653
pdf	27.9	52.3	0.0	20.3	48.1	0.2	35.3	40.7	46.5	44.1
zip	20.2	26.6	0.0	13.3	28.0	0.1	24.9	29.2	24.7	28.2
text	21.3	4.9	98.0	50.4	4.4	95.5	14.1	6.0	7.1	7.2
doc	14.4	4.2	0.5	7.1	5.2	0.2	9.7	7.9	8.7	5.8
mp4	1.7	0.6	0.0	0.2	0.8	0.0	0.4	0.5	0.4	0.5
xls	1.2	0.0	1.4	0.8	0.1	3.9	1.0	0.2	0.0	0.1
ppt	3.2	2.2	0.0	1.8	2.7	0.0	3.3	3.3	2.7	2.9
jpg	0.5	0.1	0.0	0.1	0.0	0.0	0.1	0.1	0.0	0.1
ogg	2.8	2.2	0.0	1.4	3.0	0.0	2.8	3.0	2.7	2.7
png	6.8	6.9	0.0	4.6	7.7	0.0	8.3	9.1	7.2	8.5
Unclassified	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Total = 34% Perc = 33										

Table 4.2: Training Set Analysis

ratio	pdf	zip	text	doc	mp4	xls	ppt	jpg	ogg	png
0 - 25%	9,327	347	235	528,661	3,130	1,054,503	114,968	7,842	785	11,875
25 - 50%	1,332,849	1,680,052	436	768,686	1,585,760	576,755	1,547,585	1,685,320	1,684,877	1,674,301
50 - 75%	86,583	370	181	8,834	18	31,595	10,106	1,305	287	1,787
75 - 100%	265,275	2	1,621,682	161,133	0	21,521	10,785	4,410	5	4,850
Total:	1,694,034	1,680,771	1,622,534	1,467,314	1,588,908	1,684,374	1,683,444	1,698,877	1,685,954	1,692,813
0 - 25%	0.55	0.02	0.01	36.03	0.20	62.61	6.83	0.46	0.05	0.70
25 - 50%	78.68	99.96	0.03	52.39	99.80	34.24	91.93	99.20	99.94	98.91
50 - 75%	5.11	0.02	0.01	0.60	0	1.88	0.60	0.08	0.02	0.11
75 - 100%	15.66	0	99.95	10.98	0	1.28	0.64	0.26	0	0.29

special ASCII subset and almost all xls fragments less than 50%. Undoubtedly this is completely reasonable. Text files are mostly comprised of plain text and Excel sheets, with their cell-like structure, contain less printable characters. And this analogy is more obvious in a 512-byte fragment. That finding can be used as a metric to improve current classification techniques and we are going to elaborate more on this later in this document. Based on the analysis results we thought that would be interesting to divide the fragments of our training set in 4 such categories. Then for each category and for each file type we created their respective fingerprints. So we ended up with 40 fingerprints, 4 for every file type. The algorithm checks first the ratio of our special ASCII subset of the input fragment and according to its value it compares the fragment with the fingerprints of the respective category.

Chapter 5

Summary and Recommendations for Further Work

In this final chapter you should sum up what you have done and which results you have got. You should also discuss your findings, and give recommendations for further work.

5.1 Summary and Conclusions

Here, you present a brief summary of your work and list the main results you have got. You should give comments to each of the objectives in Chapter 1 and state whether or not you have met the objective. If you have not met the objective, you should explain why (e.g., data not available, too difficult).

This section is similar to the Summary and Conclusions in the beginning of your report, but more detailed—referring to the the various sections in the report.

5.2 Discussion

Here, you may discuss your findings, their strengths and limitations.

5.3 Recommendations for Further Work

You should give recommendations to possible extensions to your work. The recommendations should be as specific as possible, preferably with an objective and an indication of a possible approach.

The recommendations may be classified as:

- Short-term
- Medium-term
- Long-term

Appendix A

Acronyms

FTA Fault tree analysis

MTTF Mean time to failure

RAMS Reliability, availability, maintainability, and safety

Appendix B

Additional Information

This is an example of an Appendix. You can write an Appendix in the same way as a chapter, with sections, subsections, and so on.

B.1 Introduction

B.1.1 More Details

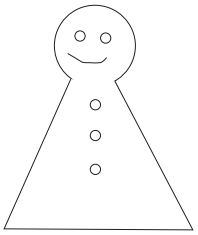
Bibliography

Lundteigen, M. A. and Rausand, M. (2008). Spurious activation of safety instrumented systems in the oil and gas industry: Basic concepts and formulas. *Reliability Engineering and System Safety*, 93:1208–1217.

Rausand, M. and Høyland, A. (2004). *System Reliability Theory: Models, Statistical Methods, and Applications*. Wiley, Hoboken, NJ, 2nd edition.

Curriculum Vitae

Name:	**Your Name**
Gender:	Female
Date of birth:	1. January 1995
Address:	Nordre gate 1, N–7005 Trondheim
Home address:	King's road 1, 4590 Vladivostok, Senegal
Nationality:	English
Email (1):	your.name@stud.ntnu.no
Email (2):	yourname@gmail.com
Telephone:	+47 12345678



Your picture

Language Skills

Describe which languages you speak and/or write. Specify your skills in each language.

Education

- School 1
- School 2
- School 3

Computer Skills

- Program 1

- Program 2
- Program 3

Experience

- Job 1
- Job 2
- Job 3

Hobbies and Other Activities