Fragment Retention Characteristics in Slack Space - Analysis and Measurements

Thijs Holleboom, Johan Garcia

Karlstad University, Karlstad, Sweden

***Abstract*—When files are deleted, their information is not removed from the storage media. This is a well known fact, and there exist numerous undelete utilities to recover newly deleted files. When deleted files have been partly overwritten, the data from the part of the file that remains in unallocated space can be readily extracted by file carving. Such carving is often performed in forensic investigations. Furthermore, as a consequence of file system implementation specifics, there additionally exist small remains of the previous files in the space at the end of new files. In this paper we focus on these small remains of previous files, or micro-fragments, that exist even after all the space allocated to the previous file has been reallocated to new files. We derive expressions for modeling the number of micro-fragments that can be expected to be found, and perform experiments to evaluate the analytical model. The obtained results indicate good correspondence between the analytical predictions and the measured results.**

I. Introduction

When a file is deleted, most common operating systems do not actually erase the file but only change some file system meta-data to indicate that the file is no longer accessible to the user. The information in the file is still retained in the storage space previously allocated to the file, although that area is now marked as unallocated. When new files are stored on the storage device, parts of unallocated space are overwritten with the new information. Even when parts of deleted files have been overwritten, different file carving tools such as Scalpel [1] can be used to find and extract information from deleted files. These carvers examine all unallocated space and can recover partial sections of deleted files residing in space that have not yet been allocated to new files. Once all the unallocated space that belonged to the deleted file has been reallocated to new files, there will be no part of the original file left in unallocated space, and such carving no longer produces any information. Nevertheless, in many instances there are still small remnants of the files previously stored on the storage media. As the size of most files written to a storage device is not an even multiple of the allocation unit, i.e. cluster, there will be some slack space at the end of most files. As a consequence of file system implementation behavior, the last sectors of the last cluster of newly written files are in the typical case left unmodified, thus retaining information from the previously stored file. While the amount of information left in these *micro-fragments* is fairly small, it can nevertheless provide important clues. If the objective is to show the potential previous existence of a set of files on a storage medium, matching of these micro-fragments towards a set of pre-known files would be a valuable tool. The set of pre-known files may for example be a law-enforcement collection of illicit files such as child sexual abuse material. In other cases the information contained in the micro-fragments may in it self be valuable. For the case of detecting previous existence, and subsequent deletion, of some set of pre-known files a way of calculating the expected number of remains of such files for a particular parameter set would clearly be useful. This paper extends previous work in this direction by presenting extended analytical expressions for predicting the number of detectable micro-fragments for a given parameter set. Initial experiments are presented to complement the analytical evaluation, with the analytical predictions showing good correspondence to the measured results.

II. Background and related work

After all the storage space originally allocated to a deleted file has been reallocated to new files and overwritten, the only remains of the deleted file are the small micro-fragments that may exist in the file slack (also referred to as cluster slack) of new files. The micro-fragments present in slack space should not be confused with the fragmentation that frequently occur on storage medias when there is insufficient contiguous space to hold a particular file, and the file is instead fragmented across multiple areas according to the allocation strategy of the file-system.

The micro-fragments are instead caused by the fact that most file system employ allocation units, clusters, that are larger than the smallest writable part of the storage medium, the sector. The cluster size may vary, but a typical value is 4KiByte which is the default for NTFS [2]. Sectors are normally 512 bytes, although recent standards allow the phase in of 4 KiByte sectors on large hard drives. Since the amount of space allocated to a file is always a multiple of the cluster size, the last cluster of the file may not be completely full, and the last sectors in that clusters will thus not be overwritten, but retain information from the files that were previously stored at that position. Furthermore, some file systems can store small files directly in file system meta-data structures. One such case is NTFS, where files small enough are stored directly in the master file table (MFT). The exact size limit for storage in the MFT depends on the number and types of attributes attached to a file, but a value that is used as a limit is 700 bytes [3, pg 328]. While forensic analysis of MFT content is possible, it is not considered in this work.

This work concerns estimating the number of fragments that may be available for matching against a set of pre-known files. Previous work looking into partial matching of files include ssdeep [4], Forsigs [5] and MRS hashing [6]. While these approaches allow for matching between parts of files, neither has the required capabilities of efficiently matching against the small file fragments that are considered here. Another set of work is concerned with the ability to infer the file type from a fragment of a file to allow for further automated or manual analysis. Work in this direction include the OSCAR approach [7] which uses statistical measures, as do later approaches [8], [9]. Recent work in [10] propose more specialized approaches for particular file types. Work on reassembly of large fragments has shown considerable success for recovering images from non-overwritten clusters [11].

III. Analytical modeling

The number of fragments that is retained in the slack space is dependent on internal details of the file system. In [12] a terminology was introduced which also will be used in the analysis below. We start with defining the necessary terms, largely following [12].

A. Terminology and notation

First we define some terms related to these factors. There exists a set of *known files*, against which micro-fragments can be tested for matches. The set of *pre-stored files* is the the subset of files which has actually been stored on the examined device, and from which fragments can be found if the file in question belongs to the set of known files. After the pre-stored files have been deleted, the clusters allocated to them are overwritten with *new files*. These new files come from a *file source*. The file source corresponds to the process that generates the new files, which could for example be the installation of programs, copying of files or installation of an operating system. The only relevant aspect of the file source in the current context is the *file size distribution* the files from the file source has. To create our model we use the following factors:

C The cluster size. On average almost half of the last cluster belonging to a file is slack since it does not contain data belonging to the file in question.

S_{MFT} The size limit for direct storage of files in a meta-data structure such as the MFT of NTFS. Files stored in the MFT do not occupy clusters.

B The minimum detectable micro-fragment size. This will be dependent on the file system and matching technology used, and will in many cases equate to the basic sector size of 512 bytes.

$H_{1..N}$ The file sizes of the set of the N pre-stored files that are larger than S_{MFT}. The set $1..N$ is the sub-set of the known files that have actually been stored in clusters on the storage medium, thus making it possible to later detect micro-fragments from them.

D Detection area, which is the total media size occupied by the pre-stored files. This is calculated as $D = \sum_{n=1}^{N} \left(\left\lceil \frac{H_n}{C} \right\rceil C \right)$, so it takes into account the actual amount of allocated space, not just the summation of the individual file sizes.

I Number of micro-fragment-holding clusters generated by the pre-stored files. This value is calculated as $I = \sum_{n=1}^{N} \left\lfloor \frac{H_n}{C} \right\rfloor$. This formulation is a slight simplification since it may be possible to detect fragments also of the last cluster of a pre-known file. However, to simplify and to make a conservative estimate this is initially not considered.

$\bar{S}$ Average file size of new files. New files are files that are stored on the storage medium, overwriting the clusters where the pre-stored files were located.

N_C The number of clusters allocated to a file. For a file of size S this number is equal to $N_C(S) = \left\lceil \frac{S}{C} \right\rceil$.

W_R Number of end-file clusters. Each file that is not stored in MFT-like storage will have one end-cluster, which may have cluster slack.

P The probability that a new file will leave enough cluster slack space to detect a micro-fragment. If the cluster slack space for a file is less than B, the minimum detectable fragment size, no micro-fragment can be detected in the cluster slack of that file.

W_C Number of end-file clusters with micro-fragment detection potential. This value is based on W_R, but takes P into account to arrive at the number of usable cluster slack "windows" into the pre-stored files that exist in the detection area.

In a previous analysis expressions for estimating the amount of detectable micro-fragments based upon average file sizes were derived for three different size distributions of the new files. We here present an improved formulation of the expressions that takes into account that storage space is allocated in whole clusters. An analysis based upon the average number of allocated clusters is presented. This procedure improves the accuracy of the model, especially for small file sizes. In the following sections expressions for the same three file size distributions considered in [12] are given.

B. Constant file size

This is the simplest file size distribution where the file source generates files that have a constant file size of S_F. If $S_F \leq S_{MFT}$ there are no end-file clusters generated in the file system. If $S_F > S_{MFT}$, considering the discreteness of cluster allocation, the number of end-file clusters is expressed as

$$W_R^{(c)} = \frac{D}{\left\lceil \frac{S_F}{C} \right\rceil C} \tag{1}$$

which is the detection area divided by number of clusters allocated for the file size S_F. The factor $P^{(c)}$, can be directly related to S_F since $P^{(c)} = 1$ if $\lceil \frac{S_F}{C} \rceil C - S_F \geq B$ and $P^{(c)} = 0$ otherwise. The expression for $W_C^{(c)}$ thus becomes

$$W_C^{(c)} = \begin{cases} \frac{D}{\lceil \frac{S_F}{C} \rceil C} & , S_F \leq \lceil \frac{S_F}{C} \rceil C - B \wedge S_F > S_{MFT} \\ 0 & , \text{otherwise} \end{cases} \tag{2}$$

C. Uniformly distributed file sizes

This case models a file source that generates files which are uniformly distributed in an interval in the range $L_1 \cdots L_2$. Consider for example MP3 file sizes, which in many cases can be modeled by a uniform distribution between say 3 and 6 MiByte. If the file sizes are in the range $L_1 \cdots L_2$ the probability of occurrence for a file of size n is $\frac{1}{L_2 - L_1 + 1}$, and the average file size becomes

$$\bar{S}^{(u)} = \sum_{n=L_1}^{L_2} n \frac{1}{L_2 - L_1 + 1} = \frac{L_1 + L_2}{2} \tag{3}$$

The number of end-file clusters produced by filling an area of size D with files the sizes of which are uniformly distributed can roughly be estimated by dividing D by $\bar{S}^{(u)}$, which is the approach followed in [12]. There are two sources of inaccuracies here that we will address. The first one is that the amount of disc space occupied by a file always is an integral multiple of the cluster size, resulting in an effective size larger than the actual size S of the file. The second is that the number of files N needed to fill the area D itself is a probabilistic variable the mean $\bar{N}$ of which in principle has to be calculated probabilistically. We will consider these two issues in turn.

The amount of disc-space occupied by a file of size S is equal to the least number of bytes, equal to or larger than S, that is an integral multiple of the cluster size. If the number of clusters occupied by a file of size S is denoted $N_C(S)$ then the average number of clusters occupied by uniformly distributed files is given by

$$\bar{N}_C^{(u)} = \sum_{n=L_1}^{L_2} N_C(n) \times p_n = \frac{1}{L_2 - L_1 + 1} \sum_{n=L_1}^{L_2} N_C(n) \tag{4}$$

where $N_C(S) = \lceil \frac{S}{C} \rceil C$. In order to evaluate equation 4 we first introduce the symbols

$$L^{(+)C} = \left\lceil \frac{L}{C} \right\rceil C \tag{5}$$

and

$$L^{(-)C} = \left\lfloor \frac{L-1}{C} \right\rfloor C. \tag{6}$$

In other words, $L^{(+)C}$is the least number larger than or equal to L that is an integral multiple of the cluster size C, and $L^{(-)C}$is the largest number less than L that is an integral multiple of C. Now partition the range $L_1 \cdots L_2$ into three subranges: $L_1 \cdots L_1^{(+)C}$, $L_1^{(+)C} \cdots L_2^{(-)C}$, and $L_2^{(-)C} \cdots L_2$. Files with sizes in the first range will produce $L_1^{(+)C}/C$ clusters, and files with sizes in the range $L_2^{(-)C} \cdots L_2$will produce $L_2^{(+)C}/C$ clusters. File sizes in the range $L_1^{(+)C} \cdots L_2^{(-)C}$ contribute with full clusters and hence the sum over sizes in equation 4 can be replaced by a sum over clusters, multiplied by the cluster size. The result is

$$\begin{aligned} \bar{N}_C^{(u)} = \frac{1}{C} \frac{1}{L_2 - L_1 + 1} \Big(& L_1^{(+)C}(L_1^{(+)C} - L_1 + 1) \\ & + \frac{1}{2}(L_2^{(-)C} - L_1^{(+)C})(L2^{(-)C} + L_1^{(+)C} + C) + \\ & L_2^{(+)C}(L_2 - L_2^{(-)C}\Big) \end{aligned} \tag{7}$$

which is the mean of the allocated number of clusters. The formula also applies if L_1 and L_2 are in the same cluster or in consecutive clusters.

The second issue is how to estimate the number of end-clusters M produced when filling a deleted area D with files the sizes of which are uniformly distributed. The number of endclusters is equal to the number of files needed to fill the area D. Considering that a file of size S occupies $N_C(S)$ clusters we arrive at the condition that the sum of the number of clusters generated by M files is larger than D, i.e.,

$$(N_C(S_1) + N_C(S_2) + \cdots + N_C(S_M))\, C \geq D \tag{8}$$

whereas the corresponding sum for $M - 1$ files is less then D. The mean of the probabilistic variable M is approximately given by

$$\bar{M} = \frac{D}{C \times \bar{N}_C^{(u)}} \tag{9}$$

This approach is an approximation because by the same reasoning the number of picks from the uniform $(0, 1)$ distribution needed to equal or exceed the value 1 would be $\frac{1}{0.5} = 2$, because the mean of this distribution is 0.5 In reality this number of picks is, surprisingly, $e = 2.71$ [13]. However, the number of picks needed to equal or exceed the value 2 is $e^2 - e$ which deviates less from the approximate value $\frac{2}{0.5} = 4$. For larger values of the sum the error decreases even more.

In order to assess how well equation 9 describes the expected number of endclusters we modeled the process of allocating files with uniformly distributes file sizes and summing the corresponding number of clusters obtained by Monte-Carlo simulation. The results are summarized in Figure 1. The upper bound of the uniform distribution varies along the horizontal axis, the lower bound is set to 1. It is seen that the number of clusters predicted by formula 9 (full line) exactly matches the results from the computer experiment (points). Hence, the procedure of calculating the mean of the expected number of clusters by the division procedure in formula 9 is very well justified and we can set $W_R = \bar{M}$.

The small horizontal part of the graph in Figure 1 for $L \leq C$ is a consequence of the fact that there always is at least one cluster allocated for files not stored in the MFT. It can also be noted that for the case where the upper limit L_2 is less than or equal to S_{MFT} all files are stored without cluster allocation (i.e in the MFT), and thus there will be no detectable

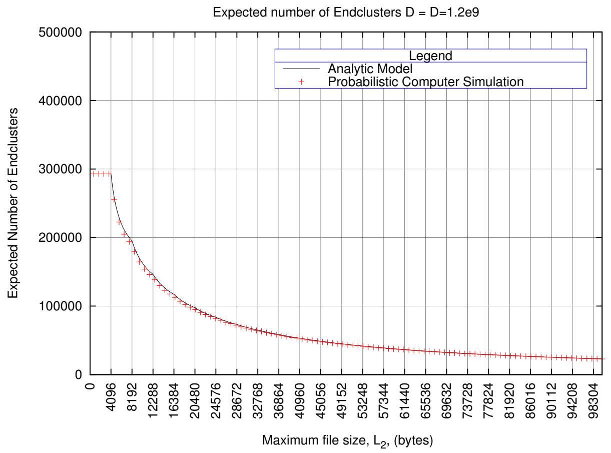

Figure 1. Probabilistic model of allocating files

micro fragments. With regards to the probability $P^{(u)}$ that the amount of cluster slack is too small to be detectable, for uniformly distributed file sizes with sufficiently large intervals, $P^{(u)}$ can be approximated by the relation between minimum detectable size and cluster size, i.e. $P^{(u)} = 1 - B/C$. The expression for $W_C^{(u)}$ thus becomes

$$W_C^{(u)} = W_R^{(u)}\, P^{(u)} = \frac{D}{\bar{N}_C^{(u)}\, C}\,(1 - \frac{B}{C}) \tag{10}$$

D. Exponentially decreasing probability distribution

It is common for file sizes to not follow a uniform distribution. With the possible exception of extremely small files of only a few bytes, which possibly are relatively rare, large files are often less common than small files. We now sketch one approach for this case, where the probability function p_n is an exponentially decreasing function of the file size n,

$$p_n^{(e)} = a\, e^{-b\, n} \tag{11}$$

The constant b is a characteristic of the distribution whereas a is a normalization constant such that

$$\sum_{n=L_1}^{L_2} p_n^{(e)} = 1 \tag{12}$$

implying that

$$a = \frac{1 - e^{-b}}{e^{-bL_1} - e^{-b(L_2+1)}} \tag{13}$$

The mean of the file size for this distribution, which strictly speaking is a geometric distribution because it is a discrete distribution, is given by

$$S^{(e)} = \sum_{n=L_1}^{L2} n\, p_n = \frac{1}{1 - e^{-b}} \times \frac{\left(L_1 - (L_1 - 1)e^{-b}\right) e^{-bL_1} - \left(L_2 + 1 - L_2 e^{-b}\right) e^{-b(L_2+1)}}{e^{-bL_1} - e^{-b(L_2+1)}} \tag{14}$$

which was derived in [12] and here given in a somewhat different, more compact form. As in the case of uniformly distributed file sizes this expression is not adequate to be used for calculating the expected number of end-file clusters because the number of files depends on the number of allocated clusters per file. In order to arrive at an expression for the mean of the number of allocated clusters we proceed as in the case of uniformly distributed file sizes and partition the range of file sizes into three subranges, exactly as done above. The contribution from subrange $L_1 \cdots L_1^{(+)C}$ becomes

$$\sum_{n=L_1}^{L_1^{(+)C}} N_C^{(e)}(n) p_n = \frac{L_1^{(+)C}}{C} \sum_{n=L_1}^{L1^{(+)C}} p_n \tag{15}$$

$$= \frac{L_1^{(+)C}}{C} \frac{e^{-bL_1} - e^{-b(L_1^{(+)C}+1)}}{e^{-bL_1} - e^{-b(L_2+1)}} \tag{16}$$

where just all probabilities could be summed because the number of clusters could be taken in front of the summation. Similarly the contribution from subrange $L_2^{(-)C} \cdots L_2$ can be evaluated, producing

$$\frac{L_2^{(+)C}}{C} \frac{e^{-b(L_2^{(+)C}+1)} - e^{-b(L_2+1)}}{e^{-bL_1} - e^{-b(L_2+1)}} \tag{17}$$

The contribution from the middle subrange $L_1^{(+)C} \cdots L_2^{(-)C}$ is somewhat more involved but a closed form can be obtained because the sizes in this range run over full clusters. The sum over sizes can be replaced by a sum over clusters followed by a sum over sizes inside each cluster where these latter sizes S all produce the same value for $N_C(S)$. Summing up, the result for the mean of the number of allocated clusters for the exponential distribution becomes

$$\begin{aligned}\bar{N}_C^{(e)} = {} & \frac{L_1^{(+)C}}{C} \frac{e^{-bL_1} - e^{-b(L_1^{(+)C}+1)}}{e^{-bL_1} - e^{-b(L_2+1)}} + \\ & \frac{e^{-b}}{e^{-bL_1} - e^{-b(L_2+1)}} \times \frac{1}{1 - e^{-bC}} \times \\ & \left(\left[\frac{L_1^{(+)C}}{C}(1 - e^{-bC}) + 1 \right] e^{-bL_1^{(+)C}} - \right. \\ & \left. \left[\frac{L_2^{(-)C}}{C}(1 - e^{-bC}) + 1 \right] e^{-bL_2^{(-)C}} \right) + \\ & \frac{L_2^{(+)C}}{C} \frac{e^{-b(L_2^{(+)C}+1)} - e^{-b(L_2+1)}}{e^{-bL_1} - e^{-b(L_2+1)}} \end{aligned} \tag{18}$$

As in the case of the uniform distribution we approximate the number of endclusters, $W_R^{(e)}$, by the expression

$$W_R^{(e)} = \frac{D}{C\, \bar{N}_C^{(e)}} \tag{19}$$

The generated number of endclusters as well as the number of generated files from the uniform and exponential distributions are plotted in Figure 2. The graphs in the figure correspond to expressions 3, 7, 14, and 18 by applying expression 9 for the expected number of endclusters, i.e., by just dividing D by the mean of the file size. The following features can be observed from Figure 2. If one assumes that no files are stored

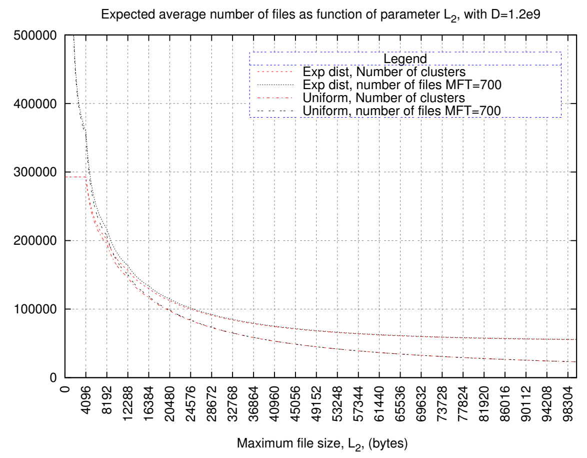

Figure 2. Expected number of endclusters

in the MFT then the number of files is equal to the number of endclusters, this is equal to setting $S_{MFT} = 0$. For small values of L_2 the number of files generated differs from the number of endclusters if the effect of storing small files in the MFT is taken into account, as expected. For small values of L_2 also the difference in number of generated endclusters between the uniform and exponential distributions becomes very small. Apparently the effect of the clusters is to even out the difference between these two distributions.

In order to calculate the probability $P^{(e)}$ that a file will leave enough space in its last cluster in order to be able to detect remains of old files all probabilities p_n such that $n - \lfloor \frac{n}{C} \rfloor C \leq C - B$ have to be summed and divided by the sum of all probabilities in this last cluster. In the case of the uniform distribution this produces the simple fraction $1 - \frac{B}{C}$ above. For the exponential distribution one obtains a somewhat more complex expression but the result is still independent of the size of the file and only depends on the global parameters of the distribution. This is a specific property of the geometrical distribution, see also [12]. The result is

$$P^{(e)} = \frac{1 - e^{-b\,(C-B)}}{1 - e^{-b\,C}} \tag{20}$$

The number of end-file clusters where cluster slack can be detected now becomes

$$W_C^{(e)} = W_R^{(e)}\,P^{(e)} \tag{21}$$

IV. EVALUATION

In the previous section different values for the expected number of end-file clusters, W_C, were derived for different distributions of the file sizes of the new files that overwrite the pre-stored files. These end-file clusters in a sense create a "window" into the underlying previous content of the storage media, with the possible exception of the case where the underlying cluster is the last cluster of the pre-stored file. The probability of hitting an end-cluster of the pre-stored file is proportional to the fraction between end-clusters and non end-clusters for the pre-stored files and hence the fraction of detectable clusters of pre-stored files can be expressed as $\frac{I\,C}{D}$.

Parameter	Value
Pre-stored file sizes	100 files, each 250KiByte
Cluster size	4 KiByte
File system	NTFS
New file sizes	10,20,40,80,800 KiByte
Replications	5

Table I
EXPERIMENTAL PARAMETERS

An expression can now be formulated for X, the expected number of detectable micro-fragments, as a combination of the fraction of detectable pre-stored clusters, the fraction of the detection area that contains end-clusters and the number of clusters in the detection area:

$$X = \frac{I\,C}{D}\,\frac{W_C\,C}{D}\,\frac{D}{C} = \frac{I\,W_C\,C}{D} \tag{22}$$

By performing experiments, the analytically derived value for X can now be compared to measured results.

A. Experimental setup

To collect data on the amount of data that is retained on the storage media, empirical measurements were performed. For all experiments, a Windows XP SP3 machine was used. Specially developed programs were used to generate pre-stored files with detectable patterns. These files were then removed, and overwritten by new files with randomized content and controlled size distribution. The parameters used for the experiments are shown in Table I.

B. Experimental results

In order to test the ability to detect remains of pre-stored files in end-clusters of new files an experiment was performed where the new files are generated with sizes that are uniformly or exponentially distributed, respectively. In Figure 3 the results of the experiment are plotted and compared with the analytical model. On the vertical axis the number of detected clusters, in the case of the experimental results, or expected clusters, from the analytical model, is given.

In these initial experiments five runs with identical parameters were made, and the corresponding five results are marked in the graph. In each run all the sizes of the created files were drawn randomly according to the parameters, causing some small variation that can be seen in the graph. Along the horizontal axis the mean of the sizes used in the actual distributions is given, which was accomplished in the following way. In the case of the uniform distribution the upper and lower boundaries are derived from the mean M by $L_1 = 0.8 \times M$, and $L_2 = 1.2 \times M$. The parameters used in the experiment were chosen accordingly.

A continuous exponential distribution was used in the computer experiment, which has the form $f(x) = 1/\lambda e^{-\lambda x}$, with a mean of $1/\lambda$. By choosing the parameter b in expression 11 equal to $1/M$ and the boundaries equal to $L_1 = 1$, and $L_2 = \infty$ we obtained equivalent values for the geometrical distribution as used in the analytical model. It was found

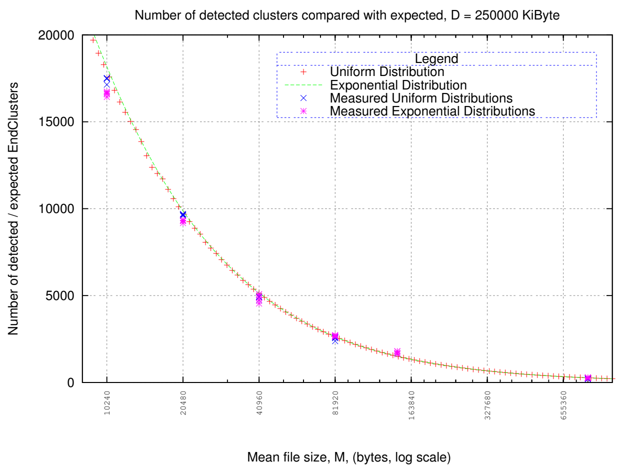

Figure 3. Results of computer experiment compared to analytical model

that the distributions are much more sensitive to the value of the mean than to the individual values of the boundaries. Also the difference between the uniform and the exponential distributions, represented in this way, became negligible, as can be seen from the figure. Note, however, that the figure not presents probability density functions because the mean is a variable here, whereas it is a constant for a given distribution.

The results from the experiment are in good agreement with the model over the whole range of values used in the experiment, as shown by the figure. Also, the experiment shows almost negligible difference between the number of detected endclusters as obtained in the two cases studied here: overwritten with uniformly distributed file sizes and overwritten with exponentially distributed file sizes.

C. *Extending with evaluation of end-cluster effects*

Up till now, the model has used the assumption that end clusters of pre-stored files cannot be detected. This is a simplifying assumption that errs on the side of caution with regards to the number of detectable micro-fragments. While the effects of this simplification is negligible if the pre-stored files are large, for cases where the pre-stored files are smaller it might result in an overly cautious estimation of the number of detectable micro-fragments. The number of pre-stored files is given by N, and each pre-stored file has one end-cluster which previously was considered to be not useful for fragment matching. However, on closer examination it can be concluded that there is a probability R that the last cluster can be used for fragment matching. Assuming that the file end location is uniformly located within end-clusters, this probability can be written as

$$R = \frac{\left(\frac{C}{B} - 1\right)\left(\frac{C}{B} - 2\right)}{2\left(\frac{C}{B}\right)^2} \tag{23}$$

Given that the fraction of the detection area that contains clusters that are no end-clusters is $\frac{IC}{D}$, the fraction that contains end-clusters is $1 - \frac{IC}{D}$. Since a fraction R of the end clusters can be used for fragment matching, the expression for expected number of fragments now becomes

$$X = \left(\frac{IC}{D} + R\left(1 - \frac{IC}{D}\right)\right)\frac{W_C C}{D}\frac{D}{C} = \left(\frac{IC}{D} + R - \frac{RIC}{D}\right)W_C \tag{24}$$

V. Conclusions

File system characteristics such as cluster size, as well as file characteristics such as size distribution, determine the expected number of micro-fragments from previously stored files that can be detected. In this paper we improve and elaborate on the first analytical model for estimation of micro-fragment retention presented in [12]. Improved expressions were presented to increase the model accuracy for small files, and to address the effect of detectable end-clusters in the pre-stored files. A set of initial experiments were performed to evaluate the analytical model against measured results. The analytical and experimental results show good agreement over the range of values used in the experiment. Although not discussed in detail, the analytical model can be further improved as pathological cases can still be constructed.

Potential extensions to this work include further experiments using a wider range of cluster sizes and file systems such as FAT32 in addition to NTFS.

Acknowledgments

The authors wish to thank Zachary Blacher for his assistance with collecting empirical data.

References

[1] G. G. Richard III and V. Roussev, "Scalpel: A frugal, high performance file carver," in *Proceedings of the 2005 Digital Forensics Research Workshop (DFRWS 2005)*, 2005.

[2] Microsoft, "Default cluster size for ntfs, fat, and exfat,," Microsoft knowledgebase: http://support.microsoft.com/kb/140365, Checked 2010-03-05.

[3] B. Carrier, *File system forensics*. Addison-Wesley, 2005.

[4] J. Kornblum, "Identifying almost identical files using context triggered piecewise hashing," *Digital Investigation*, vol. 3, no. S1, pp. 91–97, 2006.

[5] J. Haggerty and M. Taylor, "Forsigs: Forensic signature analysis of the hard drive for multimedia file fingerprints," in *Proceedings of International Information Security Conference (IFIP/SEC)*.

[6] V. Roussev, G. G. Richard III, and L. Marziale, "Multi-resolution similarity hashing," *Digital Investigation*, vol. 4, no. S1, pp. 105–113, 2007.

[7] M. Karresand and N. Shahmehri, "File type identification of data fragments by their binary structure," in *IEEE Information Assurance Workshop*, 2006.

[8] W. C. Calhoun and D. Coles, "Predicting the type of file fragments," *Digital Investigation*, vol. 5, no. S1, pp. S14–S20, 2008.

[9] C. J. Veenman, "Statistical disk cluster classfication for file carving," in *Proceedings of International Workshop on Computational Forensics(IWCF)*, 2007.

[10] V. Roussev and S. L. Garfinkel, "File fragment classificaiton - the case for specialized approaches," in *Proceedings of Systematic Approaches to Digital Forensics Engineering*, 2009.

[11] A. Pal, H. T. Sencar, and N. Memon, "Detecting file fragmentation point using sequential hypothesis testing," *Digital Investigation*, vol. 5, no. S1, pp. S2–S13, 2008.

[12] J. Garcia and T. Holleboom, "Retention of micro-fragments in cluster slack - a first model," in *Proceedings of IEEE Workshop on Information Forensics and Security*, 2009.

[13] J. Uspensky, *Introduction to Mathematical Probability*. McGraw-Hill New York, 1937.