I. J. Computer Network and Information Security, 2020, 3, 33-43
Published Online June 2020 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2020.03.04

Detecting Hidden Information in FAT

Kyryl Shekhanin, Alexandr Kuznetsov and Victor Krasnobayev
V. N. Karazin Kharkiv National University, Svobody sq., 4, Kharkiv, 61022, Ukraine
E-mail: {kyryl.shekhanin and kuznetsov}@karazin.ua, v.a.krasnobaev@gmail.com

Oleksii Smirnov
Central Ukrainian National Technical University, 8 University Ave, Kropivnitskiy, 25006, Ukraine
E-mail: dr.smirnovoa@gmail.com

Received: 24 March 2020; Accepted: 30 March 2020; Published: 08 June 2020

***Abstract*—**Various steganographic methods are used to hide information. Some of them allow you to reliably hide the fact of storage and transmission of information data. This paper analysis the methods of technical steganography that are based on hiding information messages into the structure of the FAT file system by reordering particular clusters of specially selected files (cover files). These methods allow you to reliably hide information in the file system structure, while redundancy is not explicitly entered anywhere. This means that the hidden information is not explicitly contained in the service fields or individual clusters of the file system, the size of the data stored on the physical storage medium does not change. Such steganographic systems are very difficult to detect, it is almost impossible to identify the fact of hiding information by traditional methods. The steganographic analysis technique based on the study of file system properties was developed. In particular, we analyzed the fragmentation of various files stored on a physical medium, and examine the statistical properties of various types, sizes and uses of files. Identification of anomalous properties may indicate a possible reordering of clusters of individual files, i.e. this will detect hidden information. The study of these principles is important for a better understanding of the design and counteraction of steganographic systems based on the methods of reordering clusters of cover files in the structure of the FAT. Thus, this article substantiates new approaches to steganoanalysis of cluster file systems for information hidingю. They are based on a statistical analysis of file systems of various data carriers, as well as an assessment of the fragmentation level of both individual files and the entire file system.

***Index Terms*—**Hidden Information, File System, File Allocation Table, cover files, Steganographic Analysis Technique.

I. Introduction

The development of information technology and, above all, global telecommunication networks has led to the emergence of new information services, which greatly expanded the scope and scope of the use of computing. Modern Internet technologies are used in electronic commerce and for the provision of administrative and banking services, management of complex industrial facilities, transport and energy systems, and much more [1-6]. However, the widespread computerization also leads to the emergence of new threats and risks of information and cybersecurity. In particular, new technologies have appeared, when through malicious active actions hackers can disrupt the correct functioning or completely destroy the elements of the information infrastructure [7, 8]. This inevitably leads to significant financial and material losses, and also reduces the effectiveness of the information services provided. Consequently, conducting proactive research on new technologies and techniques for protecting the information space is an urgent and important scientific task.

Among the new computer technologies for information protection, it should be noted the steganographic techniques of data hiding, which can be used to seamlessly transmit informational messages [9-12]. Unlike cryptography, steganographic techniques hide the fact of the existence of informational messages, and this is potentially preferable in a number of practically important applications.

In the papers [13-17] describe methods for hiding information in cluster file systems. They use reordering of clusters of cover files, which increases a level of fragmentation. Such methods allow the hidden storage and transmission of information using media with a specific file system. In most cases, these are FAT file systems. Using standard flash drives, you can hide about 10 MB of information, and using PC drives you can hide up to 10 times as much. But it is necessary to determine how much information can be transferred without the risk of disclosure of hidden information. Therefore, the purpose of the article is to investigate possible directions for the stegano-analysis of cluster file steganographic systems, to substantiate practical recommendations for their application. Whereas hiding information is about mixing clusters in the cover files the article investigates various properties of file systems, in particular, the level of fragmentation of cover files provides recommendations on the use of methods of hiding information in the structure of the FAT file system.

II. Literature Review

A. Description of the file system of the FAT family

Disk file systems are stream-oriented. Files in stream-oriented file systems are represented by a sequence of bytes, often providing functions such as recording, data modification, reading, and access to data. The most common modern stream-oriented file systems are the File Allocation Table (FAT) used in moving storage devices (USB flash devices) and the New Technology File System (NTFS). In view of the full openness of the file system specification, only the FAT cluster file system family is considered in this work.

The FAT structure consists of [18, 19]:

- the boot sector;
- two copies the File Allocation Table;
- the section of metadata files;
- the data files.

The boot sector is located at the beginning of the disk partition with the FAT file system. It is required to initialize the storage device. It also contains information about the settings of this file system.

File Allocation Table is intended to indicate file clusters. The entire data area is divided into clusters - blocks whose size is specified when formatting a disk. Each file and directory are occupied by one or more clusters. Thus, cluster chains are formed. In the file allocation table, each cluster is marked in a special way. The size of the label in bits for each cluster is specified in the file system name. For example, for the FAT16 file system, the tag size will be 16 bits. There are three types of labels for clusters in total:

- free cluster - a cluster that can store new files and directories.
- busy cluster - the index indicates the next cluster in the chain. If the cluster chain ends, the cluster is marked with a special index.
- BAD block - cluster with access errors. Indicated by the formatting of the disk, which would exclude further access to it.

Damage to the file allocation table completely destroys the structure of the file system, so two copies of the table are always stored on the disk.

The section of metadata files - a section of data that is placed immediately after the last FAT table. The FAT directory is a regular file marked with a special attribute. The data (content) of such a file in any version of FAT is a chain of 32-byte file records (directory notices). A directory cannot contain two files with the same name. If a disk scanner detects an artificially created pair of files with the same name in one directory, one of them is renamed.

The root directory is the disk area where the root directory and child folders information are located. In FAT32 and above, the size of the root directory is unlimited, since the root directory is the same file that can be resized. Unlike FAT16, where there can be no more than 512 files and subfolders since the root directory has a fixed size in clusters.

The advantage of the FAT file system is its ease of implementation and maximum distribution. There is a large amount of open documentation for this file system, as opposed to proprietary (closed) file systems.

B. Hiding Information in the Structure of the File System

The simplest methods for hiding information in the structure of file systems are discussed in [20, 21]. These methods use free clusters (or service data fields) to hide information, but this method is unreliable in terms of information confidentiality.

Other methods, such as [13-17], are based on the use of cover files and the hiding of information by intermixing files, that is, changing the relative positions of clusters of several covering files relative to each other.

The basic method [13, 17] uses mixing clusters of cover files in an appropriate queue with respect to clusters of another cover file. In each cluster of cover files $log_2 p = m$ information bits can be hidden. Where p is the number of cover files.

The modified method [16, 17] uses reordering of cover file clusters in an appropriate queue with respect to clusters of another cover file and clusters within a single file. The modification concerns the ordering of the cluster order not only in the FAT table but also the order within the cluster chain of the given cover file. Unlike the method [13, 14], this method allows increasing the hidden information by one bit per cluster, using the same values of setting parameters. In the first step, clustering should be done as a basic method. The next step is to mix the clusters within the cluster chains of each cover file.

Comparing these methods, it can be argued that with increasing computational complexity and fragmentation level, more bandwidth can be achieved approximately twice. The modified method was implemented programmatically, experimental researches were conducted on its effectiveness. In Figures 1, 2 the results of the comparative analysis of the volume of embedded data by basic and modified methods are given [17].

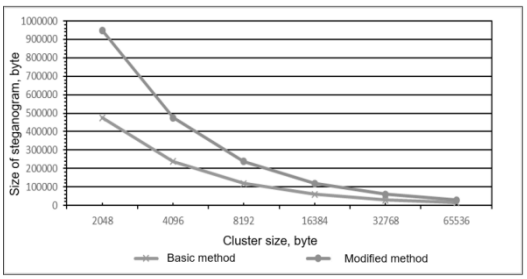

Fig.1. The size of the hidden message depends on the size of the cluster

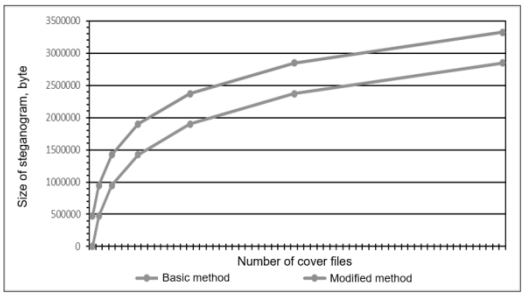

Fig.2. The dependence of the size of the hidden message on the number of cover files

In Fig. 1 shows the dependence of the maximum size of hidden information on the size of clusters of cover files. You can clearly see the advantage of double the bandwidth of an organized quilting channel. In Fig. 2 shows the dependence of the size of the hidden message on the number of roof files.

Comparing the results, it can be argued that with equal input parameters, the modified method allows hiding the information message is twice the size than the basic method. But the main advantage of the modified method is the ability to use only one cover file to hide information, unlike the basic method. Concluding in comparison of both methods, we can say that:

- the size of the hidden message depends directly on the number of clusters of the cover file;
- the size of the hidden message is inversely related to the size of the file system cluster;
- the modified method allows you to hide twice as many messages, with the same setting parameters.

Thus, it is advisable to use both methods for hiding information in the file system structure. The choice of the hiding method depends on: the availability of time to hide and remove the message; availability of computing resources; method input parameters (message size and structure, number and size of cover files).

III. Detecting Hidden Information in the File System Structure

In order to detect hidden information in the file system structure, it is necessary to analyze and detect some deviations from the standard operation of the file system. Because message hiding methods change the structural fields of the file system, detecting them will be to find the anomalous values of the structure. For example: for detecting hidden information in reserved sectors is to search for any information values other than zero; detecting hidden information in higher quartets of free cluster pointers in FAT tables is to search for any information values other than those specified in the specification; and so, on

So, detection is to compare the file system structure with the hidden information with the file system structure specified in the specification. We will assume that steganographic methods for which some differences have been identified, are *unstable for detection*.

The methods described in Section 3 use a special sequence of file clusters, which is a normal operation of the file system. Therefore, it is extremely difficult to detect hiding an information message with simple algorithms. For this purpose, it is necessary to analyze the level of fragmentation of data and, taking into account the obtained results, to compare them with some "standard" properties inherent in file systems without hidden information.

The general scheme for detecting embedded information can be represented by the following steps.

A. The stage of collecting and summarizing the source data

At this stage, it is necessary to determine the application area of use of the storage medium and the type of file system, to designate the most likely applications, services and/or services implemented using them. For example, the medium (or its separate volume) can be used as the main system partition of the operating system, or as a storage of multimedia files, or, for example, for storing files of a distributed database, etc. Each application has its own specificity, which is expressed in the prevailing types of files, their sizes, level of fragmentation, etc. In addition, the implemented services and services also have a significant impact on the statistics of data stored on physical media. For example, the distribution of clusters on the media used to maintain the health of the mail server will differ significantly from the distribution of clusters of multimedia services or online streaming platforms.

The result of this stage is the classification of various situations of using data carriers and file systems by types, classes or other gradations.

B. The stage of statistical analysis and generalization of the data

At this stage, it is necessary to collect statistics on the characteristics of media and file systems for specific source data (for a specific typical situation introduced in the first stage). For example, if a suspicious medium (with possible hidden messages) is used in an educational environment (as a physical data medium in a classroom), then statistics should be collected for all such accessible media.

For such studies, on the one hand, the mass character is needed (to ensure the required accuracy and reliability of the research results), and on the other hand, caution and accuracy are necessary so that carriers that are not used in the selected typical situation are not included in the statistical sample.

The result of this stage is a set of typical characteristics both in terms of the fragmentation level of the file system and in the individual parameters of various files (their size, fragmentation, frequency of overwriting, etc.). All collected data is used hereinafter as reference characteristics showing a typical distribution for a given source data.

C. The stage of analysis and comparison with the reference characteristics

At this stage, the suspicious medium (with possible hidden messages) is analyzed according to the characteristics entered in the second stage (the level of fragmentation of the file system and individual file characteristics). The data obtained are compared with the reference, i.e. with typical characteristics for a given source data. The convergence of the analyzed data can be estimated in various ways, for example, using statistical criteria of agreement.

The result of this step is the assessment (qualitative and/or quantitative) of the convergence of the reference data with the characteristics of the suspicious carrier.

D. Decision-making stage

At this final stage, a decision is made about the presence or absence of hidden data in a suspicious medium (with possible hidden messages). Decision making can be carried out according to various rules and criteria. For example, if a large number of empirical observations are collected at the second stage, then the decision-making process can be automated. For this, for example, statistical consent criteria can be used. If empirical observations are not enough, or, for example, if the statistical criteria for agreement do not provide acceptable accuracy, then additional experts in steganographic analysis must be involved to make a decision.

Thus, this article develops the technology of steganographic analysis, which is based on a statistical study of various physical media and file systems in typical application areas and situations.

The main indicator in conducting statistical studies is the level of fragmentation. This article presents the results of sample statistical studies of various media and file systems used in the educational environment (as a physical data carrier in the classroom). These results can be interpreted as a brief demonstration of the second stage (statistical analysis and generalization of the obtained data) of the general scheme for detecting embedded information.

IV. Experimental Researches of File System Fragmentation

A. Fragmentation Score

In information technology, fragmentation is the division of something into many small, fragmented fragments [22, 23]. A significant level of fragmentation adversely affects the capabilities of the file system and the storage media [23, 24]:

- increases the amount of movement of the read head, which in turn reduces the bandwidth;
- reduces the shelf life of the storage medium;
- on certain file systems (unlike FAT), it is not possible to write data to fragmented areas, even though the total free space satisfies the necessary conditions.

To reduce the level of fragmentation, use the reverse process to fragmentation - defragmentation [19-25]. This is an artificially created process, the essence of which is to move the data files one by one.

In general, the level of fragmentation increases with multi-threaded work with the file system. That is, the level of fragmentation can increase at [25]:

- operating system;
- processes that modify or delete data from the file system;
- copying files from one storage medium to another;
- downloading data over the internet.

Separate file system fragmentation is the inability of the file system to place related data sequentially (continuously). This phenomenon is inherent in file systems for storing data that allow direct modification of data [23]. That is, for example, the file system has the required number of free clusters, but these clusters are disjointed, as shown in Figure 3.

Also separately distinguish data fragmentation - the level of fragmentation of the sequence of clusters of one file, that is, the greater the level of fragmentation, the farther away from the adjacent clusters of the same file. For example, Figure 4 shows the fragmentation of different files, which shows different cases of alternating individual file clusters A and B.

To formally evaluate the fragmentation level of an individual file, we introduce a quantitative indicator F_A, which will be calculated using pseudocode (see Fig. 5). The figure shows:

- A – is the designation of a file of a certain length with the appropriate cluster chain;
- cluster - indicates the cluster index in the FAT table;
- fragmentation - the value of the fragmentation index (the initial value is 1, since each file contains at least one fragment).

Thus, the fragmentation index F_A is calculated as follows: if the number of the current cluster differs from the number of the previous cluster by more than 1 then we increase the level of fragmentation by one (since the file clusters can be placed in the opposite direction, the absolute difference of the numbers must be estimated). So, in fact, metric F_A determines the number of individual fragments for file A in the sequential record of the file system cluster chain.

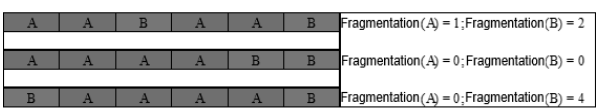

Fig.3. Data fragmentation

```
fragmentationLevel = 1;
//typeof clustarI === 'number'
fileA = [clustar1, cluster2, ..., clusterN];
for(int i=0; i<fileA.length; i++){
    if(fileA[i] - fileA[i-1] > 1){
        fragmentationLevel++;
    }
}
return fragmentationLevel;
```

Fig.4. Pseudocode finding the level of fragmentation

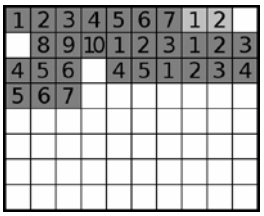

Fig.5. File system fragmentation

The metric F_A entered allows you to quantify the fragmentation level of individual files. Based on the assumption that the fragmentation is uniform, the value of this metric for different files should be approximately the same. Therefore, fragmentation studies of the file system and a F_A score for each file can provide a criterion (rule) for detecting steganographic hiding: if the value of F_A for certain files is significantly different from the median for all files of the fragmentation, then we will assume that there is a hiding information.

It should be noted that metric F_A gives the absolute value of fragmentation, which will directly depend on the total number of clusters in a particular file. Therefore, it is advisable to use a normalized indicator to reliably detect the hiding of an information message:

$$\tilde{F}_A = \frac{F_A}{n_A} \quad (1)$$

where n_A is the number of clusters in file A.

Then to detect the fact of concealment we will compare the value of $\tilde{F}_A$ with the average value of $\tilde{F}$, which is calculated as

$$\tilde{F} = \frac{1}{N}\sum_{i=1}^{N}\tilde{F}_A \quad (2)$$

where N is the total number of files in the file system.

Experimental researches of file systems on desktops were conducted in order to determine the average values of fragmentation.

B. Experiment Conditions

30 computers were used to analyze the fragmentation level from the training laboratories of V. N. Karazin Kharkiv National University, which contains system volume "C" with Windows 7 installed, volume to store other "D" information. Also analyzed are flash drives, conventionally labeled as volume - "F". Preliminary defragmentation was performed 7 - 14 days before the results were obtained. The results were obtained by the Auslogics Disk Defrag (www.auslogics.com).

The results (Figure 6) of Auslogics Disk Defrag are presented as tables with the specified data for each fragmented file: number of fragments, file size, name, and file type. To analyze the results of Auslogics Disk Defrag, a software implementation of the automatic calculation of the necessary data was created, namely the following dependencies:

- the number of files from the fragmentation level;
- the number of files from the specific fragmentation level;
- the number of fragments per file type (.log, .doc, .exe).

The repository with the program code and analysis results are located at «https://github.com/Shekhanin Kyryl/ausdiskdefrag-parser». You can get file system snapshots and do test studies use this link. The reproducibility and repeatability of the results of independent studies confirm their reliability and adequacy.

The dependency of the number of files on the fragmentation level indicates how many files in the file system contain a certain number of fragments. This result will allow you to determine the optimal parameters for methods of hiding information: the number of cover files, the maximum permissible size of the steganogram.

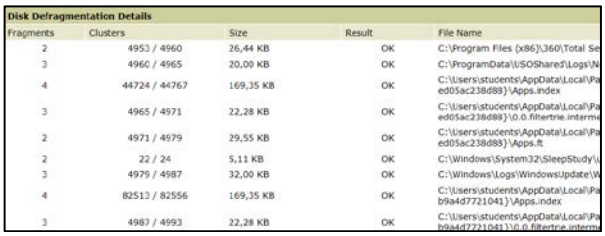

Fig.6. An example of submitting data to Auslogics Disk Defrag

The dependency of the number of files on the specific fragmentation level indicates how many files with the specified fragmentation level. This result will determine how much the fragmentation level depends on the file size.

Depending on the number of fragments per file type, it will be possible to analyze how much the fragmentation level depends on the file type, that is, how the file is used by the file system.

The following are the results of the analysis of file systems, volumes "C" and "D", according to the three indicators mentioned above.

C. The results of experimental studies

Analyzing system volume "C" (Table 1), it can be argued that fragmentation is an inevitable result of the operating system. The operating system currently accesses the file system for writing and reading data, resulting in a large number of fragmented files.

Table 1. Type Sizes for Camera-Ready Papers

Num fragments	Num files	Num fragments	Num files
2	6223	19	8
3	2196	20	8
4	985	21	6
5	525	22	10
6	278	23	3
7	152	24	4
8	102	25	4
9	79	26	7
10	54	27	7
11	49	28	2
12	51	29	4
13	38	30	6
14	26	31	7
15	24	32	5
16	21	33	2
17	25	34	3
18	9	…	…

Analyzing Figure 7, which shows the corresponding histogram, it can be argued that fragmentation is inversely dependent on the number of files.

Analyzing the average number of files with the appropriate fragmentation level, we can say that the fragmentation level of the system volume "C" is such that there is at least one file with the fragmentation level 13. As shown in Table 2 and Figure 8.

Specific fragmentation is the level of fragmentation per file cluster defined by formula (1). The size of the file is calculated by the amount of occupied space in the clusters, with rounding to a larger whole (as required by the specification of the file system, since a not fully occupied cluster is considered occupied).

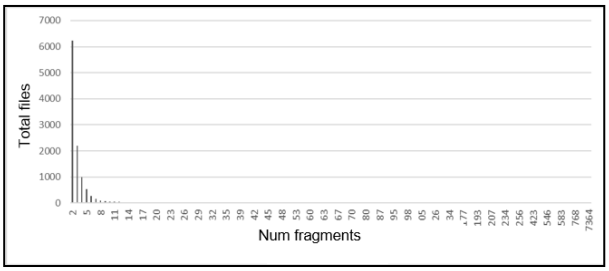

Fig.7. Total number of files with the appropriate fragmentation level at "C"

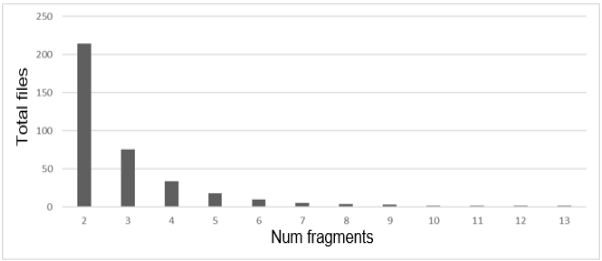

Fig.8. Average number of files with the appropriate fragmentation level at "C"

Analysis of the total number of files from the specific fragmentation level indicates that, in the file system (Fig. 9 and Table 3) the most common files are with small fragmentation ($\tilde{F}_A < 0{,}01$), or with 100% specific fragmentation. But most often, these files are 2 clusters in size and have a fragmentation level of 2. The dependence of the fragmentation level on the file size is tracked and It has an inverse relation. But there are also anomalous values.

Table 2. Average number of files with the appropriate fragmentation level at "C"

Num fragments	Num files	Num fragments	Num files
2	214,5862069	8	3,517241379
3	75,72413793	9	2,724137931
4	33,96551724	10	1,862068966
5	18,10344828	11	1,689655172
6	9,586206897	12	1,75862069
7	5,24137931	13	1,310344828

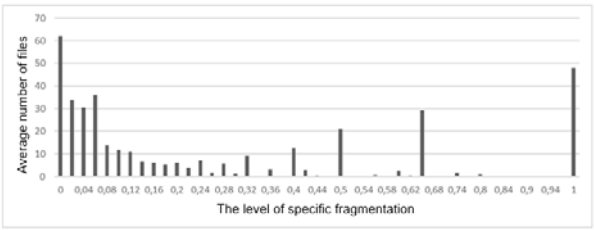

Fig.9. Average number of files with a specific fragmentation level of "C"

Analysis of the average number of files with the appropriate level of specific fragmentation is consistent with the results indicated above and are listed in Table 4 and Figure 10. Analysis of the fragmentation level by file type is a prerequisite for a complete analysis of the file system. The results are shown in Table 5 and Figure 11.

To conclude, it should be noted that files with a digital type or without type are commonly used by the operating system or by certain programs for internal use only. Those types are not standardized.

By analyzing standard file types, you can determine which file operations lead to more fragmentation:

- ZIP: The file contains archived data in a concise format. Depending on the original size, such data may have a considerable archiving time. Thus, over time, the next free cluster may become occupied with other data, leading to increased fragmentation. Also, usually the data is transmitted over the network in archived form, and when downloading data over the Internet, file fragmentation is inevitable.
- XML: A file containing a dataset for applications, including browsers (web pages). This data is downloaded along with the html page through the network.
- LOG: A file containing the information generated about the performance of certain services,

programs, services, or operating system. This file is generated all the time the program is running. Therefore, writing time to this file may be too high, which increases the likelihood of fragmentation.

Table 3. Total number of files with the corresponding specific fragmentation level on system volume "C"

Specific fragmentation	Number of Files	Specific fragmentation	Number of Files
0	1801	0,44	13
0,02	977	0,46	1
0,04	891	0,5	609
0,06	1041	0,52	2
0,08	401	0,54	8
0,1	339	0,56	20
0,12	318	0,58	1
0,14	191	0,6	69
0,16	176	0,62	12
0,18	159	0,66	852
0,2	177	0,68	4
0,22	111	0,7	9
0,24	208	0,74	47
0,26	50	0,78	2
0,28	162	0,8	31
0,3	37	0,82	10
0,32	265	0,84	10
0,34	7	0,86	3
0,36	92	0,9	3
0,38	2	0,92	1
0,4	360	0,94	1
0,42	87	1	1391

Table 4. The average number of files with the corresponding specific fragmentation level on system volume "C"

Specific fragmentation	Number of Files	Specific fragmentation	Number of Files
0	62,10344828	0,44	0,448275862
0,02	33,68965517	0,46	0,034482759
0,04	30,72413793	0,5	21
0,06	35,89655172	0,52	0,068965517
0,08	13,82758621	0,54	0,275862069
0,1	11,68965517	0,56	0,689655172
0,12	10,96551724	0,58	0,034482759
0,14	6,586206897	0,6	2,379310345
0,16	6,068965517	0,62	0,413793103
0,18	5,482758621	0,66	29,37931034
0,2	6,103448276	0,68	0,137931034
0,22	3,827586207	0,7	0,310344828
0,24	7,172413793	0,74	1,620689655
0,26	1,724137931	0,78	0,068965517
0,28	5,586206897	0,8	1,068965517
0,3	1,275862069	0,82	0,344827586
0,32	9,137931034	0,84	0,344827586
0,34	0,24137931	0,86	0,103448276
0,36	3,172413793	0,9	0,103448276
0,38	0,068965517	0,92	0,034482759
0,4	12,4137931	0,94	0,034482759
0,42	3	1	47,96551724

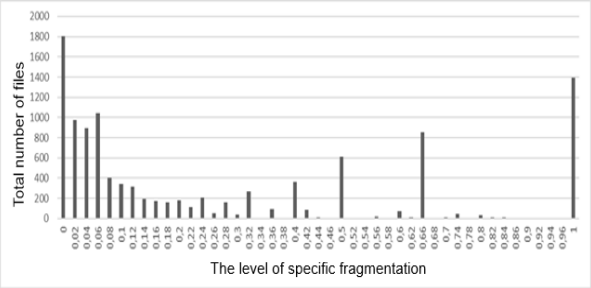

Fig.10. Total number of files with a specific fragmentation level of "C"

Table 5. 20 File Types with the most Fragmentation level in System Volume C

Type of file	Fragmentation level		
	Min	Avg	Max
79	11	255,7143	768
ZIP	2	94,07143	670
XML	2	37,84375	7364
VDM	2	36,46154	578
7C	2	25,6	103
LOG	2	20,57759	3240
CACHE	2	19,05882	177
BIN	2	13,625	539
83	2	10,5	25
EDB	2	9,136364	59
INDEX	2	8,822581	196
CAB	2	8,6875	39
EXE	2	8,052632	130
DAT	2	7,923469	423
INI	2	7,342105	31
DLL	2	6,716216	281
PDB	2	6,571429	26
1	2	6,096774	19
7	2	6	8
TTF	2	5,933333	32

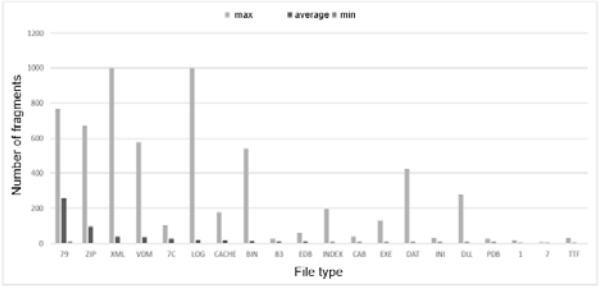

Fig.11. 20 file types with the highest level of fragmentation

Thus, the operation of the operating system leads to fragmentation inevitably. Files without a standard type (system files) are the most fragmented. Regarding standard types, it can say that these are files that:

- they are recorded over time, so other data may take up the next cluster (ZIP, LOG, CACHE);
- downloaded over the Internet (XML, INI, DLL, ZIP).

Analyzing information volume "D", table 6, it can be argued that fragmentation is an inevitable result of multithreaded file system access. This is mostly achieved by user requests. For example, a user can simultaneously:

- run a process that creates log files;
- download files over the Internet;
- edit documents.

The simultaneous implementation of such actions increases the level of fragmentation.

In general, the fragmentation level of "D" has the same dependency as the system volume "C", but the fragmentation level of each file is less. This is due to the fact that the fragmentation in the "D" is mostly caused by the user and some programs, but not the operating system.

Table 6. Total number of files with appropriate fragmentation level on "D"

Num fragments	Num files
2	193
3	89
4	56
5	40
6	26
7	20
8	9
9	5
10	6
11	7
12	2

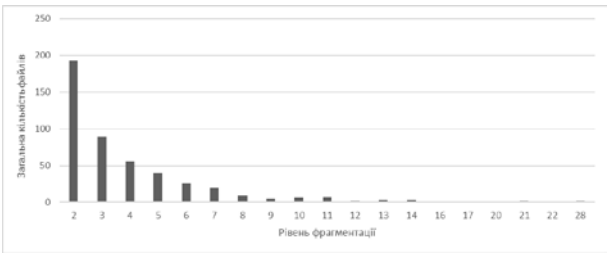

Fig.12. Total number of files with appropriate fragmentation level on "D"

The result of the fragmentation level analysis based on the file type is shown in Table 7, Figure 13. Analyzing the result, it can be argued that in the absence of an operating system impact on volume "D", the level of fragmentation will be minimal. However, there are some types of files that have a high level of fragmentation:

- DB is a file with a special structure for storing a database. Database work is usually performed on one such file over time, which directly results in increased fragmentation.

Table 7. File types with the level of fragmentation in the system volume "D"

Type of file	Fragmentation level		
	Min	Avg	Min
DB	2	10,71875	28
IDB	2	7,565217	16
PDB	2	5,272727	13
VDI	2	4,5	7
IPCH	2	3,966667	7
PCH	2	3,909091	6
INDEX	2	3,842105	22
ILK	2	3,7	10
DAT	2	3,315789	5
UNDEFINED	2	2,970588	14
SUO	2	2,833333	5
OBJ	2	2,611111	5
TLOG	2	2,388889	3
RESX	2	2,1	3
EXE	2	2	2
CACHE	2	2	2

Fig.13. – File types with the level of fragmentation in the system volume "D"

Analyzing the flash drive - "F", you can say that the fragmentation of such storage media is minimal, this is due to the mode of use of flash drives. For the most part, they are only used to store data on them. Thus, it is not recommended to use a flash drive for hidden transmission of information.

V. Discussion of Results

In this paper describes the steganographic techniques to hide information on the structure of the file system FAT. A comparative analysis method as the conclusion modified method allows you to hide twice more information than basic. A method of steganoanalysis to detect hidden data is also proposed. The file system fragmentation level is estimated:

- system disk C is the most fragmented since the continuous operation of the operating system leads to an increase in the level of fragmentation;
- data disk D has less fragmentation and depends on user actions and/or third-party programs;
- Flash drives have a minimal level of fragmentation because drives are mostly used for storing and transmitting information, not for processing.

Thus, system disk C is more likely to be used to hide the message, since it has the maximum number of fragmented files. It is necessary to counteract this:

- conduct defragmentation as often as possible;
- allocate disk space by destination - operating system, third-party programs, data storage;
- delete unnecessary files and programs.

The obtained experimental data can be interpreted as the result of the second stage of the general detection scheme of embedded information. In particular, the obtained average results (with appropriate processing) can be used as reference characteristics for the subsequent detection of hidden messages. For example, Figure 8 shows a graph of the average number of files by the level of fragmentation. When normalizing to the total number of files and approximating the results, this dependence can be used for benchmarking with a suspicious medium. Similarly, you can use processed (normalized, smoothed by approximation, etc.) results of other tables. These studies appear to be the most relevant area of our future research.. In addition, interesting in our opinion is the imitation of the process of hiding information messages with the subsequent attempt to deceive the detection system of stegano-messages.

VI. Conclusions

This paper presents the results of an analysis of the FAT file system, the results of the fragmentation level depend on the size, type, or usage of the file. These results will provide further guidance on the input parameters of steganographic methods, on the one hand. And in the future, they may formulate recommendations to counteract the methods described. Understanding these principles is important for a greater understanding of the de-sign and counteraction of steganographic systems based on methods of blending clusters of cover files in

the structure of the FAT file system. In particular, this work proposes a general scheme for detecting embedded information based on statistical studies of file cluster information carriers. The experimental results presented in the article can be interpreted as the main stage of this scheme. The statistical analysis and generalization of the obtained data can be used to form reference templates for detecting hidden information in cluster file systems.

REFERENCES

[1] D. Johnson and M. Ketel, "IoT: Application Protocols and Security," International Journal of Computer Network and Information Security, vol. 11, no. 4, pp. 1–8, Apr. 2019.

[2] M. Zaliskyi, R. Odarchenko, S. Gnatyuk, Yu. Petrova. A.Chaplits, Method of traffic monitoring for DDoS attacks detection in e-health systems and networks. CEUR Workshop Proceedings, Vol. 2255, pp. 193-204, 2018.

[3] J. A. Ojeniyi, E. O. Edward, and S. M. Abdulhamid, "Security Risk Analysis in Online Banking Transactions: Using Diamond Bank as a Case Study," International Journal of Education and Management Engineering, vol. 9, no. 2, pp. 1–14, Mar. 2019.

[4] Gnatyuk S., Akhmetova J., Sydorenko V., Polishchuk Yu., Petryk V. Quantitative Evaluation Method for Mass Media Manipulative Influence on Public Opinion, CEUR Workshop Proceedings, Vol. 2362, pp. 71-83, 2019.

[5] T. K. Fataliyev and S. A. Mehdiyev, "Analysis and New Approaches to the Solution of Problems of Operation of Oil and Gas Complex as Cyber-Physical System," International Journal of Information Technology and Computer Science, vol. 10, no. 11, pp. 67–76, Nov. 2018.

[6] S. Gnatyuk, M. Aleksander, P. Vorona, Yu. Polishchuk, J. Akhmetova, Network-centric Approach to Destructive Manipulative Influence Evaluation in Social Media, CEUR Workshop Proceedings, Vol. 2392, pp. 273-285, 2019.

[7] Z. Hassan, R. Odarchenko, S. Gnatyuk, A. Zaman, M. Shah, Detection of Distributed Denial of Service Attacks Using Snort Rules in Cloud Computing & Remote Control Systems, Proceedings of the 2018 IEEE 5th International Conference on Methods and Systems of Navigation and Motion Control, October 16-18, 2018. Kyiv, Ukraine, pp. 283-288.

[8] S. Gnatyuk, Critical Aviation Information Systems Cybersecurity, Meeting Security Challenges Through Data Analytics and Decision Support, NATO Science for Peace and Security Series, D: Information and Communication Security. IOS Press Ebooks, Vol.47, №3, pp.308-316, 2016.

[9] O. Hosam, "Attacking Image Watermarking and Steganography - A Survey," *International Journal of Information Technology and Computer Science*, vol. 11, no. 3, pp. 23–37, Mar. 2019.

[10] H. Ogras, "An Efficient Steganography Technique for Images using Chaotic Bitstream," *International Journal of Computer Network and Information Security*, vol. 11, no. 2, pp. 21–27, Feb. 2019.

[11] S. Dogan, "A New Approach for Data Hiding based on Pixel Pairs and Chaotic Map," *International Journal of Computer Network and Information Security*, vol. 10, no. 1, pp. 1–9, Jan. 2018.

[12] G. P. Rajkumar and V. S. Malemath, "Video Steganography: Secure Data Hiding Technique," *International Journal of Computer Network and Information Security*, vol. 9, no. 9, pp. 38–45, Sep. 2017.

[13] H.Khan, M.Javed, S.A.Khayam, F.Mirza. "Designing a cluster-based covert channel to evade disk investigation and forensics". *Computers & Security*, Volume 30, Issue 1, January 2011. On-line]. Internet: https://www.sciencedirect.com/science/article/pii/ S016740481000088X.

[14] H.Khan, M.Javed, S.A.Khayam, F.Mirza. "Designing a cluster-based covert channel to evade disk investigation and forensics". *Computers & Security*, Volume 30, Issue 1, January 2011. [On-line]. Internet: https://www.sciencedirect.com/science/article/pii/ S016740481000088X.

[15] H.Khan, M.Javed, S.A.Khayam, F.Mirza. "Evading Disk Investigation and Forensics using a Cluster-Based Covert Channel". *National University of Science & Technology* (NUST), Islamabad 44000, Pakistan. [On-line]. Internet: https://www.sigsac.org/ccs/CCS2009/pd/abstract_17.pdf.

[16] N.Morkevičius, G.Petraitis, A.Venčkauskas, J.Čeponis. "Covert Channel for Cluster-based File Systems Using Multiple Cover Files". *Information Technology and Control*, 2013, Vol.42, No.3. pp. 32. [On-line]. Internet: http://itc.ktu.lt/index.php/ITC/article/view/3328.

[17] A. Kuznetsov, K. Shekhanin, A. Kolhatin, I. Mikheev and I. Belozertsev, "Hiding data in the structure of the FAT family file system," *2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies (DESSERT)*, Kyiv, Ukraine, 2018, pp. 337-342. DOI: 10.1109/DESSERT.2018.8409155.

[18] K.Yu. Shekhanin, A.O. Kolhatin, E.E. Demenko, A. A. Kuznetsov. "On Hiding Data Into the Structure of the FAT Family File System." *Telecommunications and Radio Engineering*, Volume 78, 2019, Issue 11, pp. 973-985. DOI: 10.1615/TelecomRadEng.v78.i11.5.

[19] Description of the FAT32 File System: Microsoft Knowledge Base Article 154997. http://support.microsoft.com/kb/154997/.

[20] Overview of FAT, HPFS, and NTFS File Systems: Microsoft Knowledge Base Article 100108. http://support.microsoft.com/kb/100108/.

[21] S. f. Liu, S. Pei, X. y. Huang and L. Tian, "File hiding based on FAT file system," *2009 IEEE International Symposium on IT in Medicine & Education, Jinan*, 2009, pp. 1198-1201.

[22] J. Davis, J. MacLean and D. Dampier, "Methods of Information Hiding and Detection in File Systems," *2010 Fifth IEEE International Workshop on Systematic Approaches to Digital Forensic Engineering*, Oakland, CA, 2010, pp. 66-69.

[23] "Partitioning, Partition Sizes and Drive Lettering", The PC Guide. April 17, 2001. Retrieved 2018-09-20. [On-line]. Internet: http://www.pcguide.com/ref/hdd/file/part.htm.

[24] "Switches: Sector copy". Symantec. 2001-01-14. Retrieved 2018-09-20, [On-line]. Internet: https://support.symantec.com/en_US/article.TECH107956 .html.

[25] D. Samanta, "Classic Data Structures", Prentice Hall India Pvt., Aug 1. 2004 p. 480, pp.76-127.

Authors' Profiles

Kyryl Shekhanin

Graduate student of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University, Ukraine. Areas of scientific interests: steganography, information hiding and steganographic analysis.

Email: kyryl.shekhanin@karazin.ua

Alexandr Kuznetsov

Doctor of Sciences (Engineering), Full Professor, Academician of the Academy of Applied Radioelectronics Sciences, Professor of the Department security information systems and technologies of the V. N. Karazin Kharkiv National University, Ukraine. Areas of scientific interests: applied cryptology and coding theory.

Email: kuznetsov@karazin.ua

Victor Krasnobayev

Doctor of Sciences (Engineering), Full Professor, a Professor of the Department of Electronics and Control Systems of V.N. Karazin Kharkiv National University, Ukraine. Areas of scientific interests: the theory and practice of creating computer systems and components in a residue numeral system.

Email: v.a.krasnobaev@gmail.com

Oleksii Smirnov

Doctor of Sciences (Engineering), Full Professor. Head of Cybersecurity & Software Academic Department Central Ukrainian National Technical University, Ukraine. Areas of scientific interests: applied cryptography and coding, security information systems and technologies.

Email: dr.SmirnovOA@gmail.com

How to cite this paper: Kyryl Shekhanin, Alexandr Kuznetsov, Victor Krasnobayev, Oleksii Smirnov, "Detecting Hidden Information in FAT", International Journal of Computer Network and Information Security(IJCNIS), Vol.12, No.3, pp.33-43, 2020. DOI: 10.5815/ijcnis.2020.03.04