

A Critical Evaluation of the Treatment of Deleted Files in Microsoft Windows Operation Systems

Gregory H. Carlton
University of Hawaii
gcarlton@hawaii.edu

Abstract

Recent discourse regarding security vulnerabilities within Microsoft Windows® operating systems has called for the removal of these vulnerabilities.

Perceived security weaknesses within these operating systems typically involve computer system intrusion or data storage security. This paper calls attention to conflicting normative values regarding the concept of secure data storage.

A perceived security risk is associated with the file management system's policy of allowing deleted data to remain intact. Some argue that lingering traces associated with deleted files should not exist.

An alternative view perceives usefulness from the ability to retrieve accidentally deleted data. This view is also held from within the forensic computer science field.

This presents a dilemma for software designers seeking to provide operating systems that meet the security desires of society. This paper discusses arguments of these conflicting views, and it encourages more research to provide a foundation on which to base suggestions to resolve this quandary.

1. Recent discourse regarding security vulnerabilities

There has been much written in the press and the periodicals targeted to practitioners in the field of information technology recently regarding vulnerabilities within the widely used versions of Microsoft Windows operating systems. Articles in PC World [1], Network World [2], Computer Weekly [3], and ZD Net [4] have highlighted weaknesses within versions of Microsoft Windows operating systems, including, Windows XP, Windows NT, Windows Me, Windows 98, and Windows 95.

The primary reason behind this media attention is the opinion that the Microsoft Corporation, in its haste to dominate the personal computer marketplace, has incorporated sloppy programming techniques in its development of these widely used operating systems [5]. It has been argued that these sloppy programming techniques have yielded security vulnerabilities that make it relatively easy for unauthorized or unwelcomed

programs or people to gain access to networked PCs [6]. The combination of the ease of unauthorized access and the proportion of PCs connected to the Internet running Microsoft Windows has resulted in a large number of security-related problems [7].

1.1. Clean up the code

Critics of Microsoft's programming efforts have called for the corporation to take actions to clean up its proprietary code within its operating systems [8]. The call to clean up the code argues that portions of the numerous lines of programming instructions that make up the computer's operating system is defective, and Microsoft should take immediate action to modify these defective computer programming instructions by replacing the defective code with clean code. Clean code would consist of well written computer programming instructions that perform the desired functions without vulnerabilities for unintended functions.

Microsoft has publicly acknowledged that it is taking steps to clean up its code [9]. Now that critics and Microsoft have agreed that the operating system code should be cleaned, it is appropriate to determine exactly what is considered defective (undesired) as opposed to clean (desired).

1.2. Definition of clean code

Much of the furor regarding cleaning the code is concerned with security issues, or perceived weaknesses within the operating system. Perceived security weaknesses within versions of these operating systems tend to fall into two categories [10]. The first category of perceived security weaknesses involves unauthorized access to computer systems, and it concerns communication and program execution. The second category of perceived security weaknesses involves the data stored within a computer system.

Panko indicates that unauthorized access or intrusion is often associated with viruses, worms, Trojan horses, zombies, and other malicious hacking activities [11]. Intrusion compromises computer security by exposing access to data contained within the system and program execution functions that permit the computer to execute programs without the user's knowledge or permission.

There is little disagreement within our society that a properly written computer operating system will provide techniques to prohibit unauthorized access to the computer system. Efforts to clean up the code pertaining to this category would focus on ensuring that unauthorized people or computer programs are not able to communicate with a targeted computer or execute programs on targeted computers. Due to the general agreement of most members of our society for the solution to this concern, this paper does not discuss this issue further.

However, the second category of security weakness regarding data storage is an area that offers much disagreement concerning the proper direction programmers should take to clean up the code. There are two perspectives to security regarding this issue, and they pose opposite and conflicting views of the solution. It is this issue regarding secure data storage that this paper addresses.

To understand the nature of the conflicting views of security pertaining to data storage, it is necessary to consider the method used within Microsoft Windows, and many other popular operating systems, regarding the storage of data. The following discussion on the file management system provides the foundation for this understanding.

2. The file management system

The Windows Operating System, like many other operating systems, uses a file management system to handle the details associated with data storage and retrieval [12]. For the purpose of clarification, the following discussion on data storage deals solely on the issue of storing data and retrieving data, and it does not consider issues pertaining to access privileges. Access privileges are omitted from consideration here because this issue is better addressed in the first category described above pertaining to unauthorized access.

There are three fundamental concepts relating to the file management system that should be understood before the conflicting views of improving security pertaining to data storage can be addressed. The first concerns the method for assigning data onto a physical storage device, the second deals with the tabular method the file management system utilizes to identify the location of stored data, and the third pertains to the concept of data deletion. Each of these three fundamental concepts is discussed below.

2.1. Assigning data onto a physical storage device

The file management system organizes the physical storage media of a computer system into a logical view to facilitate data storage and retrieval. This logical view formats a physical disk into logical disk partitions [13].

The data storage capacity on physical disk drives represent billions of bytes of data, and for simplicity and efficiency reasons, the file management system organizes the storage capacity into a manageable number of units. Each allocation unit consists of a number of sectors. Each sector represents 512 contiguous bytes on the physical storage medium [14].

The file management system uses an algorithm to determine the number of allocations units (clusters) it can successfully manage, and then it assigns a number of sectors of data to each allocation unit. For example, a FAT 16 file management system is capable of managing 2^{16} , or 65,536 allocation units [15]. Similarly, a FAT 32 file management system is capable of managing 2^{28} , or 268,435,456 allocation units (FAT 32 reserves 4 bits; therefore, $32-4=28$) [16].

To accommodate the large storage capacity of physical storage devices and the maximum number of allocation units bound by the algorithm, the file management system assigns an appropriate number of sectors to every allocation unit. For example, assume a physical disk drive has 1,050,000 sectors (537,600,000 bytes) and is managed by a FAT 16 file management system, then there would be 16 sectors per cluster (and a small amount of wasted space at the end of the physical disk). This is determined by dividing 1,050,000 sectors by 65,536 possible allocation units (clusters) [17].

2.2. The file allocation table

The second fundamental concept concerning the functionality of the file management system deals with the file allocation table. The file allocation table identifies every allocation unit (cluster), and it contains information that indicates whether any specify cluster is available to be used, is in use and is the last cluster of a file, or is in use and contains data that continues to an identified cluster [18].

The file management system uses a table listing a directory structure of files and a file allocation table to identify the physical storage location of logical data files.

Comprehension of the file allocation table, its linkage with the file directory structure, and the concepts of bytes, sectors, and clusters provides the foundation for understanding the process of storing data within a file management system. For example, consider a file named G1.txt listed in the directory structure. As illustrated in Figure 1. File allocation table below, suppose that this file begins in cluster 600 and is 2,475 bytes long, and suppose that there are two sectors assigned to a cluster. Based on the algorithm presented above for a FAT 16 file management system, the file G1.txt would be assigned to three physical clusters. The file allocation table would contain the cluster number in cluster 600 that the file continues to, for example, cluster 604. Cluster 604 would contain a value representing the cluster number that the

file continues to, for example, cluster 605. Finally, cluster 605 would indicate an end of file (EOF) value, as the file would logically terminate within cluster 605.

Now, consider the data that is actually stored within each sector allocated to this file. Cluster 600 contains two sectors (sector 1,200 and sector 1201), and each sector holds 512 bytes of data. Likewise, Cluster 604's two sectors each contain 512 bytes of data for this file. However, the first sector in cluster 605 (sector 1,210) contains only 427 bytes of data for the file, and it contains 85 bytes of slack [19]. There are different types of slack, and the slack space between the end of the logical file and the end of its sector is known as RAM slack [20]. Microsoft Windows 98 and newer versions of Microsoft Windows pad RAM slack with binary zeros to represent null data [21]. The second sector assigned to cluster 605 contains no data for the file, yet this sector is part of an allocation unit assigned to the file. This second sector in cluster 605 (sector 1,211) is referred to as drive slack, and the file management system takes no action to alter the data in drive slack. The portion of storage space known as RAM slack combined with the portion identified as drive slack is referred to as file slack. For completeness defining slack terminology, another type of slack is called volume slack or wasted space, and it represents the portion of physical space that occurs after cluster 65,536 (on a FAT 16 file management system) to the end of the physical disk.

The significance of drive slack will be discussed later in this paper after the following discussion on the third fundamental concept of the file management system is presented.

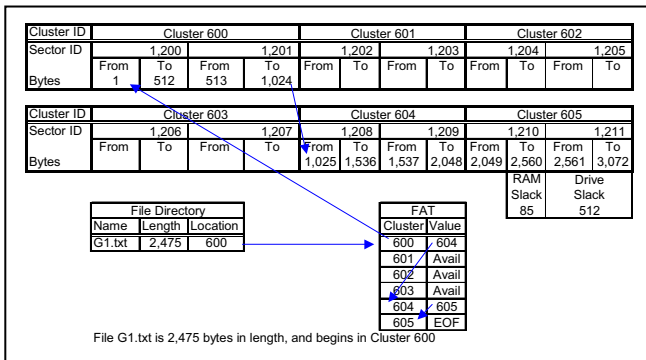


Figure 1. File allocation table

2.3. The concept of data deletion

The file management system generally leaves data intact on the physical storage media after files associated with the data have been deleted logically [22]. For example, the operating system's file management system does not physically erase the data when it deletes files, but rather it updates its file allocation table entry for the first cluster assigned to the file to indicate that the space that was reserved for the deleted file is now available for

future use. Additionally, the file management system alters the file name contained within the directory structure to indicate that the file was deleted. The operating system recognizes the alteration of the file name by the file management system, and it does not identify deleted file names to users; therefore, users do not have access to the data from deleted files. Please refer to Figure 2. Deletion below for an illustration of the deletion process using the G1.txt example discussed above.

During the file deletion process, only the data within the file allocation table and the file directory are modified, as the actual data stored on the physical device is still intact. This deleted data will remain intact on the physical device until it is written over by another logical file [23]. This is an important concept to understand, especially when combined with the understanding of the file management's usage of drive slack identified above. The importance of the combined affects associated with storing files, deleting files, and writing over the deleted data with new files is discussed in the following section on drive slack.

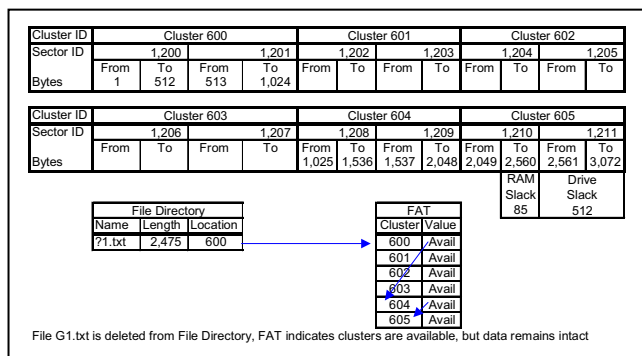


Figure 2. Deletion

2.4. The relevance of drive slack pertaining to writing over deleted data

The primary purpose of this paper is to illustrate differences in perspectives regarding improvements to security within the file management system of Microsoft Windows. This section contains a discussion of a relevant process regarding data storage and security.

To illustrate the cause of concern regarding drive slack, the example regarding the hypothetical G1.txt file is used again. Now, suppose that this 2,475 byte file was deleted. As described above, it is known that the data associated with the file remains intact on the storage media after the file is logically deleted. Now, suppose that another file, G2.txt, is written to the storage medium by the file management system. In this example, let G2.txt represent a 100 byte file that the file management system assigns to cluster 600.

In this example, as illustrated in Figure 3. Writing over existing data, the first sector of cluster 600 (sector 1,200) will contain the 100 bytes of data that represents the

logical contents of G2.txt. The last 412 bytes of sector 1,200 in cluster 600 will contain null data, as it is RAM slack. The second sector assigned to cluster 600, sector 1,201 will not be altered; therefore, the data from bytes 513 to 1024 of G1.txt will still be stored, intact on the physical storage medium even though it is part of a cluster assigned to another file. Likewise, the data from G1.txt that had been stored in clusters 604 and 605 will remain intact.

The fact that data from previously deleted files remains intact on the physical storage medium is a security issue. A number of add-on software products exploit this, including McAfee® Shredder [24] and CyberScrub ES Pro [25]. The determination of whether this represents an instance of good security or an instance of bad security is a normative value, and this is the crux of the problem associated with cleaning up the operating system code. The following sections will discuss two conflicting views relative to security aspects of the file management system's method of handling deleted files and drive slack.

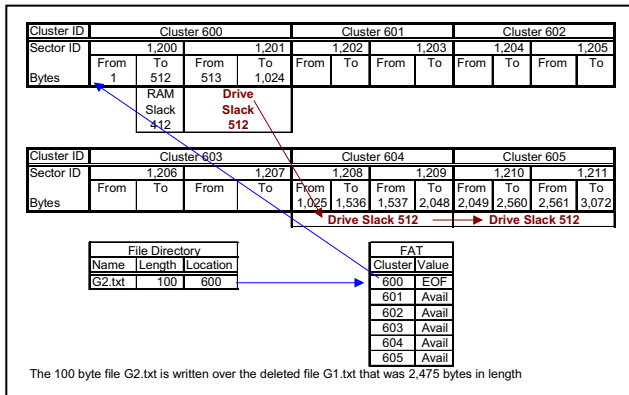


Figure 3. Writing over existing data

2.5. The deletion of user files and system files

The deletion process described above applies to all files managed by the file management system. This includes user files and system files. User files are those that are created by the user, such as word processing documents, electronic spreadsheets, or slide presentations. System files are those that are utilized directly by the operating system and usually they are not of concern to users of the system. Examples of system files include operating system program files, device driver files, and numerous temporary files that are created by the operating system to facilitate its tasks. Temporary files include printer spool files, working copies of files, such as copies of Internet Web pages, and swap files used by the operating system to store virtual RAM [26].

Temporary files are particularly interesting when they are considered in the context of deleted files. This is due to several factors. First, the operating system typically creates temporary files without the knowledge of the user.

Second, the operating system uses temporary files for many of its functions. Third, temporary data files can contain user data that users never intended to save, such as a memo typed into a word processing application and printed without saving. Fourth, temporary files are managed by the file management system; therefore, the data contained within them remain intact on storage media until overwritten by other data.

Consider the example of a memo entered into a word processing application and not saved by the user. Suppose the user instructs the word processing program to print this memo. The operating system will generate a temporary printer spool file that contains the memo of user data, and it will delete the temporary printer spool file once the data has been successfully transmitted to the printer's buffer. However, the deleted temporary printer spool file is managed by the file management system; therefore, the actual data contained within this file will remain intact on the storage medium until the space is written over. This example illustrates how user data that was never saved by the user can remain on the computer's storage media.

Now, consider further the example of the memo the user did not intend to save. Suppose that this memo was rather lengthy, such that the data associated with it occupied several clusters of data. If this data is written over by a relatively small file (i.e., one that occupies less than one sector), then the bulk of the contents of this memo will become drive slack and remain intact on the storage medium even after the beginning cluster is assigned to another file.

3. Normative values regarding file deletion methodology

The above sections have identified the basic concepts regarding the file management system's treatment of deleted files. This section considers the concept of applying the value of human judgment to the file deletion technique used by the file management system. Two conflicting values are presented below concerning the method used by the file management system to delete files. One view considers the file management system's technique of allowing logically deleted data to remain on physical storage media after deletion to be a security weakness. The opposing view considers the ability to retrieve deleted data from physical storage media to be an aid to security.

One perspective takes the view that individuals have the right to expect that there is no data leakage within their computer systems. This perspective contends that individuals and society benefit from a right to privacy, and this right extends to computers; therefore, one should expect that unintended data is not accessible on computer systems. This view holds that deleted files should be

fully erased without any traces of their prior existence, and temporary files, like all other files, are to be completely erased when their existence is no longer required. Through this perspective, individuals believe clean code means to eliminate all trace evidence of preexisting data when files are deleted. Under this perspective, the notion of data in slack space, unallocated clusters, or unused sectors is moot.

A second perspective takes the view that the current design of the file management system is desirable for security reasons at both the individual and societal level. Under this current design, the file management system maintains a file allocation table that identifies the space used on storage devices. It updates this table to indicate whether a specific file is active or deleted. Deleted data is not physically erased, but rather its storage location is made available for future use and recorded in the file allocation table. The deleted data remains until it is written over by another file. This perspective contends that individuals benefit from this method, as it provides individuals an opportunity to undelete data, if necessary and if it is recovered prior to another file being written to the location of the deleted file.

Another benefit to individuals from this perspective is that systems operate faster using this technique, as there is no latency associated with physically writing over deleted data to ensure that all traces of its prior existence is eliminated.

Additionally, this view argues that society benefits from this technique, as forensic examiners with legal access to computers can determine much of the activity that has occurred on the computer. This historic activity can be very helpful in legal proceedings to help determine guilt or innocence in criminal matters or damages in civil matters.

These two perspectives present a dilemma for computer scientists with regards to the proper technique to follow in the pursuit of clean code. Does security within the operating system imply that users can expect fundamental privacy regarding no potential for access to data that has been deleted, or does security imply that those with legal access to the computer system should have resources to reconstruct historical usage activity?

Given the two conflicting views, what should a system developer do when instructed to clean up the code associated with the file management system? The system developer has two fundamental options regarding the treatment of data deletion. He or she can utilize the existing techniques and decide to allow deleted data to remain intact on physical storage media, or he or she can develop new techniques, similar to those used to wipe the contents of RAM slack, to remove deleted data from the entire file slack. Either choice will be considered to be a security weakness by those viewing security from the opposing perspective.

The merits of each of the two views are discussed in the following sections. After considering strengths of each argument, a clearer understanding of the dilemma facing developers will be recognized.

3.1. Perceived file system security weaknesses

Some consider the techniques used by the file management subsystem within the operating system to be a security weakness [27]. This view is based on the value judgment that computer users have a right to privacy. This right to privacy extends to the treatment of deleted files, and it centers on the idea that the computer system should not contain data that has been deleted by the user.

This belief can be associated with individuals who are concerned about the sensitive data they discard, and they believe it should not be retrievable by others. Three categories of users that share these concerns are easily identifiable.

The first category consists of professional individuals that deal with data that is regulated by law, for example, medical record administrators, attorneys, and employees of the government working with classified data. Others that can be included in this category are corporate employees that deal with trade secrets or intellectual property.

The second category of individuals that perceive a security weakness with the file management system's treatment of deleted files are those concerned with the civil rights of individuals. The concept behind this belief is that individuals have a right to expect privacy with the data maintained on their computers. This view supports the concept that others, including government, corporate, or other individuals, should not have the ability to obtain information from previously deleted data on their computers.

A third category of individuals that share this perception can be considered to be part of the criminal population of society. Clearly it is in the interests of these people to ensure that their tracks are covered to reduce the risk of conviction for their crimes. Again, as a normative value, this group's reasons can be considered invalid, and the desires of these individuals will not be considered further in this paper.

Even with the invalidation of the third category above, the remaining two categories do present valid issues regarding undesirable consequences from data remaining in deleted files.

Additional elements of consideration pertaining to these concerns raise questions regarding system performance and data access. The file management system will have to perform additional tasks if it is modified to nullify deleted files similar to the current method it uses to handle RAM slack when data is written to storage devices. These additional tasks will require

additional processing instructions and additional physical data writes (i.e., I/O functions), and it is reasonable to expect that these additional tasks will degrade the overall system performance from the users' perspective. The resulting question is whether the increased perception of security is worth the cost of performance degradation from the overhead associated with nullifying deleted data. This leads to the second question regarding performance and data access.

The second question regarding system performance and data access addresses the concept of data access. In this context, security concerns regarding data access are focused on unauthorized access. As was stated in the above section discussing the definition of clean code, unauthorized access, or intrusion, is an aspect of computer security that is separate from the file management system. Although this paper does not address the subject of improving the operating system to prohibit unauthorized access, the concern raised in this section regarding the file management system's treatment of deleted files as being a security risk becomes less of a concern when the issue of unauthorized access is considered an operating system access issue instead of a file management system issue.

The concern over access to the data maintained on storage media from deleted files remains an issue, however, when consideration is given to the realization that physical access to the storage media can expose previously deleted data. For example, some computer systems are passed on to new users when replaced by their original users. Some other computer systems are lost through theft. In either case, the new users, whether with legal access or not, can use a variety of techniques to read the data stored on the physical storage devices. In these situations, the data obtained from deleted files represents a security risk.

It must also be noted that in the situation described above where a new user recovers data from the physical storage device, much of the security risk is not necessarily from the deleted files. The new users will also be able to obtain data from files that are saved on the storage media that have not been deleted (i.e., those files that the file management system considers to be valid). Realizing that both current and deleted files are accessible to users with physical access to storage media leads one to again consider this more of a physical possession issue rather than a weakness of the file management system.

This becomes a little more evident when considering the hypothetical situation of what would likely happen if the proposed modifications were made to nullify file slack to reduce the security risks associated with the file management system. In this hypothetical situation where someone obtains a physical storage device that had all of the deleted files nullified, the one in possession of this device would still have access to all of the data stored within currently active files. As described in this hypothetical situation, the security risk is not primarily

within the file management system, but rather in the physical access to storage media (i.e., an access issue).

Therefore, the real security risk directly linked to the treatment of deleted files by the file management system is limited to the following two conditions. These two conditions are that it applies only the content of data contained within deleted files, and then only when someone without authorized access to the contents of the data contained within the deleted files obtains access to the physical storage device. This clearer understanding of the real security risk posed by the file management system's policy of not wiping drive slack provides a much smaller actual security risk that it naively thought.

3.2. Perceived file system security benefits

Just as some consider the techniques used by the file management subsystem within the operating system to be a security weakness, others consider this a security benefit. This view is based on the value judgment that it is useful to have the ability to reconstruct events that have occurred on a computer system.

Those that possess this view include computer users that accidentally delete data files and wish to retrieve them. For example, as shown below in Figure 4. Recycle bin, users can utilize the Windows Recycle Bin to perform this function [28]. Software or system analysts and system administrators may also consider the ability to retrieve data from deleted files valuable. Also, computer forensic examiners largely depend on the ability to analyze deleted data, including all forms of slack, to help determine the activities that have occurred on a computer system.

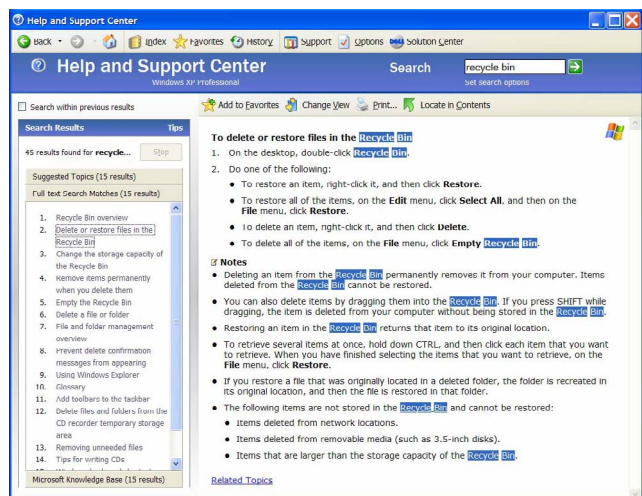


Figure 4. Recycle bin

Security is one of the primary objectives supporting the field of computer forensic science. For example, if a law enforcement agency suspects that a computer user committed a crime, the computer system's storage devices

are obtained for a forensic examination. A forensic examiner will acquire the data directly from the physical devices and analyze it by meticulously evaluating deleted files, including swap files, printer spool files, temporary Internet Web files, unallocated space, unused space, and all forms of slack [29].

After the analysis, the forensic examiner will report on his or her findings by providing written reports and court testimony, if necessary. The purpose is to identify and exhibit evidence that is beneficial to the conviction of a perpetrator. Clearly, the view held by those within the forensic computer science field and law enforcement is that the ability to uncover evidence from deleted files is a valuable security tool. Guidance Software, Inc. indicates, "Computer forensic investigators throughout the world utilize EnCase for the seizure, analysis and court presentation of computer evidence" [30].

Overall, there is merit to the perception that the data associated with deleted files have value from a security standpoint. This view is held by individuals that perceive usefulness from the ability to retrieve accidentally deleted data and from the law enforcement community, specifically within the forensic computer science field. Much information is gleaned from an analysis of operating system artifacts that would not be available if the authors of operating systems "cleaned up" their code to eliminate the lingering traces of deleted files.

4. The dilemma facing security oriented developers

Two conflicting perspectives were presented above concerning security and the file management system. Both positions hold merit, and those that subscribe to each perspective argue that societal values pertaining to security are better served by adopting their view of the proper technique to use within the file management system.

This presents a dilemma for conscientious software designers seeking to provide well-written operating systems that meet the security desires of society regarding the file management system. Which method should be taken? The answer is a normative value judgment based on societal factors.

Perhaps the best solution is to conduct more research on these issues to develop an accurate understanding of societal benefits relating to computer data storage and deletion. The findings of solid, empirical research should illuminate a better perspective than the existing anecdotal arguments on this sensitive topic.

5. References

- [1] Berger, M. (5 Feb. 2002). Microsoft Takes a Break to Clean Its Code. *PC World*. Retrieved May 4, 2004 from <http://www.pcworld.com/resource/printable/article/0,aid,82804,00.asp>

- [2] Berger, M. (5 Feb. 2002). Microsoft stops writing, starts cleaning its code. *Network World*. Retrieved May 4, 2004 from <http://www.nwfusion.com/news/2002/0205mscode.html>

- [3] Evers, J. (29 May 2003). Microsoft creates group to clean its coding act. *Computer Weekly*. Retrieved May 4, 2004 from <http://www.computerweekly.com/Article122161.htm>

- [4] Oltsik, J. (24 Sept. 2003). Microsoft—forget PR, clean up the code. *ZDNet*. Retrieved May 4, 2004 from http://zdnet.com.com/2100-1107_2-5081333.html

- [5] Hunter, R. (13 April 2004). The Structural Failures of Windows. *The Inquirer*. Retrieved May 4, 2004 from <http://www.theinquirer.net/?article=15305>

- [6] Poulsen, K. (24 Nov. 2003). Microsoft Newsletter # 164. *Security Focus*. Retrieved May 4, 2004 from <http://cert.uni-stuttgart.de/archive/focus-ms/2003/11/msg00131.html>

- [7] Nyquest, G. (25 Sept. 2003). MS Windows as National Security Risk. *Greg Nyquest Blogs*. Retrieved May 4, 2004 from <http://homepage.mac.com/machiavel/iblog/B1072909446/C538124065/E611147278/>

- [8] Perera, R. (17 Jan. 2002). Gates calls for 'trustworthy computing'. *Network World*. Retrieved May 4, 2004 from <http://www.nwfusion.com/news/2002/0117gates.html>

- [9] Evers, J. (29 May 2003). Microsoft creates new group to clean its coding act. *Network World*. Retrieved May 4, 2004 from <http://www.nwfusion.com/news/2003/0529microcreat.html>

- [10] Roberts, P. (21 Feb. 2003) Microsoft Looks to Make Software More Secure. *PC World*. Retrieved May 4, 2004 from <http://www.pcworld.com/resource/printable/article/0,aid,109472,00.asp>

- [11] Panko, R.R. (2004). *Business Data Networks and Telecommunications* (5th ed.). New Jersey: Prentice Hall.

- [12] Dew Associates Corporation. (2002). Operating Systems and their File Systems. Retrieved May 4, 2004 from http://www.dewassoc.com/kbase/hard_drives/file_systems_2.htm

- [13] System Medic Incorporated. (2003). Software Education 2, File System. Retrieved May 4, 2004 from <http://www.systemsmedic.com/SoftwareEdu2.htm>

- [14] Guidance Software, Inc. (2003). *EnCase Introduction to Computer Forensics*. (p, 145). Pasadena, California: Guidance Software, Inc.

- [15] Guidance Software, Inc. (2003). *EnCase Introduction to Computer Forensics*. (p. 151). Pasadena, California: Guidance Software, Inc.
- [16] Guidance Software, Inc. (2003). *EnCase Introduction to Computer Forensics*. (p. 152). Pasadena, California: Guidance Software, Inc.
- [17] Kozierok, C.M. (2001). NTFS Clusters and Cluster Sizes. *The PC Guide*. Retrieved May 4, 2004 from <http://www.pcguid.com/ref/hdd/file/ntfs/archCluster-c.html>
- [18] Kozierok, C.M. (2001). File Allocation Tables. *The PC Guide*. Retrieved May 4, 2004 from <http://www.pcguid.com/ref/hdd/file/fatFATs-c.html>
- [19] Kozierok, C.M. (2001). File Chaining and FAT Cluster Allocation, *The PC Guide*. Retrieved May 4, 2004 from <http://www.pcguid.com/ref/hdd/file/clustChaining-c.html>
- [20] Kozierok, C.M. (2001). FAT Partition Efficiency: Slack. *The PC Guide*. Retrieved May 4, 2004 from <http://www.pcguid.com/ref/hdd/file/partSlack-c.html>
- [21] Bean, Mike. (2003, June). Lesson 8 Basic Searching. *EnCase Intermediate Analysis and Reporting*. Course conducted by Guidance Software, Inc., Pasadena, California.
- [22] Kozierok, C.M. (2001). File Deletion and Undeletion. *The PC Guide*. Retrieved May 4, 2004 from <http://www.pcguid.com/ref/hdd/file/clustDeletion-c.html>
- [23] Guidance Software, Inc. (2003). *EnCase Intermediate* (rev. 4.02) (p. 321). Pasadena, California: Guidance Software, Inc.
- [24] DirectDeals.com. (2004). *McAfee Internet Security 2004* (ver. 6.0). Retrieved May 4, 2004 from http://www.directdeals.com/Item_MIS60E001RAA.aspx
- [25] WUGNet Publications, Inc. (2004). *CyberScrub ES Pro*. Retrieved May 4, 2004 from <http://www.wugnet.com/shareware/spow.asp?ID=544>
- [26] Guidance Software, Inc. (2003). *EnCase Intermediate* (rev. 4.02) (p. 227). Pasadena California: Guidance Software, Inc.
- [27] New Technologies Armor, Inc. (2004). *Classified Data Identification & Data Elimination Guidelines, Security Risk Assessment*. Retrieved May 4, 2004 from <http://www.forensics-intl.com/riskscan.html>
- [28] Microsoft Corporation. (2001). Help and Support Center, Microsoft XP Professional (ver. 5.1) [Computer Software]. Redmond, Washington: Microsoft Corporation.
- [29] Guidance Software, Inc. (2004). *EnCase Forensic Edition User Manual* (ver. 4.18, rev. 1). (pp. 99-111). Pasadena, California: Guidance Software, Inc.
- [30] Guidance Software, Inc. (2003). *EnCase Legal Journal*. (p. 51). Pasadena, California: Guidance Software, Inc.