

Erasing a stored information pattern on a storage medium.

The invention relates to a method of erasing, in a document data processing device, a stored information pattern on a rewritable data carrier that is accessible by a data processing facility of said device, the document data processing device having

- primary processes for processing document data, wherein data may be stored on said data carrier, and
- secondary processes for erasing stored data, through overwriting a selected storage area of said carrier by a shredding pattern.

BACKGROUND OF THE INVENTION

Prior art has recognized that magnetic hard disks and other more or less similar storage facilities are increasingly vulnerable to reading of confidential data that is stored thereon. A first type of attack may occur through a hacker "entering" the facility via a data network. Such interference may be countered by relatively simple operations, such as encrypting the data before storing them on the internal hard disc, deleting the entry of the particular data file in the file administration of the carrier, or – preferably – overwriting the data.

A more serious type of attack may occur after physical removal of such a disk from a Personal Computer, from a Digital Access Controller associated to a printing facility, or other. Deleting the file administration of the carrier will then be to no avail.

Reformatting the carrier or over-writing by a so-called shredding pattern provides improved security, but there are techniques for detecting bit patterns that have been overwritten. A single "data shredding" run is therefore insufficient when deletion of certain data is really important.

Overwriting data with a plurality of shredding runs involving multiple different bit patterns, further to be called: shredding patterns, is generally considered as an effective policy, wherein the construction of the shredding patterns should be appropriate to the intended degree of security.

The data carrier can be based on various different writing/storing technologies, such as magnetic, magneto-optical, optical such as in a rewritable CD, and other. Usually, the geometrical storage organization is based on a kind of track, that may be a

cylinder, a spiral, a straight line, or other. The storage physics is based on some kind of remanence property of the storage substrate.

In particular, United States Published Patent Application 2002/0,181,134 to Bunker et al. discloses the application of user-selectable shredding patterns. The present inventors have recognized that this prior art technology may offer enough protection against infringers, but that the long time required by the overall shredding will keep the data processing proper, *e.g.* printing, stalled for an often highly needed time interval and may therefore severely degrade system performance.

Furthermore, an overall discussion of shredding operations and other related items is given in P. Gutmann, "Secure Deletion of Data from Magnetic and Solid-State Memory", Sixth USENIX Security Symposium Proceedings, San José, CA USA, July 22-25, 1996, downloadable at <http://www.usenix.org/publications/library/proceedings/sec96/gutmann.html>.

SUMMARY OF THE INVENTION

In consequence, amongst other things, it is an object of the present invention to share the data storage facility between the data processing proper on the one hand and the shredding operation on the other by effecting the latter operation at least partially in the background of the former operation.

Now therefore, according to one of its aspects, the invention is characterized running said primary and secondary processes asynchronously, i.e. starting a primary process irrespective of the completion of any pending secondary processes.

The invention also relates to an apparatus being arranged for implementing the method of the invention. Further advantageous aspects of the invention are recited in dependent Claims.

BRIEF DESCRIPTION OF THE DRAWING

These and further features, aspects and advantages of the invention will be discussed more in detail hereinafter with reference to the disclosure of preferred embodiments of the invention, and in particular with reference to the appended Figures that illustrate:

Figure 1, a system block diagram pertaining to the present invention;

Figure 2, an operation diagram without shredding being applied;

Figure 3, an operation diagram with synchronous shredding;

Figure 4, an operation diagram with asynchronous shredding;

Figure 5, an operation diagram with mixed mode shredding;

Figure 6, a flow diagram of a mixed (a)-synchronous shredding operation;

Figure 7, a Table of shredding patterns;

Figure 8A and 8B, user interface windows for controlling data erasure according to the invention;

Figure 9, a user interface window for shredding process supervising.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS

In the following, the term "shredding" will be used for erasure of stored data patterns on a hard disc or other re-writable storage media.

Figure 1 illustrates a system block diagram pertaining to the present invention. The Figure is generally symbolically organized, so that various respective blocks will represent operations rather than identifiable hardware modules. Elements 10 represent the relevant software processes of the controller of a device in which the invention is practised, such as a printer. The processes 10 make use of job-related files 12, stored in the internal disc of the device. In the course of the processes 10, some of the job-related files may no longer be needed, and in the context of the present invention, such files should be effectively erased. For that purpose, an *erase file* process is issued for such a file, as indicated by reference number 14. Erasure takes place in the form of over-writing the relevant data storage locations on the device disc, further to be called: shredding. During the shredding, the original data are over-written with various bit patterns, ranging from simple sequences of all zeroes to complex patterns of zeroes and ones.

The *erase file* process 14 is controlled by a set of user-defined or default parameters 16. Such parameters are for instance:

- synchronous/asynchronous/combined shredding
- number and selection of shredding patterns
- shredding priority.

The terms used here will be explained in the description below.

In reaction to the issuing of an *erase file* process 14, a shredding process 18 is activated (symbolized by arrow 20) for the specified file, and in accordance with the erasure parameters 16. Actual bit patterns for the shredding process are specified in a list 22, accessible for the process 18.

Before further describing the actual operation of the shredding process, we will now first explain several different shredding techniques as used herein, with reference

to Figures 2, 3, 4 and 5. This example is directed to a printing application, but may, *mutatis mutandis*, also be applied to other document processing operations such as scanning and copying.

In printing, a print file is received via a network and stored in the system disc of the printing facility or printer. The file may be encrypted in the case of secure data transport, or it may be encrypted directly after receipt. Print files wait in a print queue until they can be processed. When the printer is ready to process a print job, it reads the print file from disc, if necessary decrypts it and processes the print file by rasterizing it, and subsequently prints the rasterized file. In a standard situation, the file would thereupon be ready and need no longer be used at the printing facility.

Figure 2 illustrates a an operation diagram without shredding being applied. As an example, print jobs are symbolized by sequences of three print sheets, but it will be clear that in fact print jobs may comprise any number of sheets. As shown, a first print job Job1 is processed first and is followed by a second print job Job2.

Figure 3 illustrates an operation diagram with so-called "synchronous" shredding as is known from the prior art. For simplicity, the same configuration of jobs as in Figure 2 has been assumed. Now, at the end of executing Job1, the associated file is considered out-of-use. Thereupon, the shredding of Job1 is effected immediately. Subsequently, the execution of Job2 is effected. Of course, Job2 may become due for shredding as well. The requirement for shredding may be determined on the basis of an intended security level that can be non-uniform over the jobs. In an exemplary embodiment, only files protected by a PIN-code would be shredded (to be explained later). The information shredded can be the original information, and will preferably also include any intermediate or temporary data files created in the process, data rendered out-of-date through updating, such generally as considered appropriate vis-à-vis the intended security level. Generally, the actual operating system will know where to find such intermediates, even if the application in question is based on running third party software. Evidently, there may occur an unwanted delay in the processing of Job2, due to the preference given to the immediate or synchronous shredding of Job1.

By contrast, Figure 4 illustrates an operation diagram with so-called "asynchronous" shredding according to the present invention. As soon as Job1 is finished and the associated files are up for shredding, the shredding of Job1 is actually started, provided that the execution of Job2 can proceed for some time interval without storage access. If such access becomes necessary, it can take preference over the shredding operation, as based on some preference criterium. This renders the

shredding “asynchronous” regarding the decision “to-be-shredded”. The shredding is triggered automatically on the basis of certain criteria. Subsequent to the interrupting, the shredding can proceed again. The procedure is executed likewise regarding the shredding of Job2 during execution of Job3. Thus, the shredding can occur in a time-distributed manner.

In a related and particularly advantageous embodiment of the present invention, shown in Figure 5, the shredding occurs in a mixed manner: For example, the first few, preferable only the first) shredding run would be immediate, automatic and synchronous, i.e. before the next job is started. Further shredding runs would then be asynchronous. In this embodiment, the file can no longer be read by the system (although it would still be possible to reconstruct the stored data pattern using sophisticated analysis applied to the physical disc). Still, the most time-consuming part of shredding, the multiple overwriting, is done in the background and will not hinder the processing of the succeeding jobs.

Now, returning to Figure 1, and in connection to the above-described mixed shredding mode, the shredding process 18 first removes the files-for-shredding from the file administration 12 by renaming them and moving the pointers to another storage location 24 (operation symbolized by arrow 26), that is dedicated to synchronous shredding. The data locations of the files are immediately overwritten with one or a limited number of shredding patterns.

After finishing the synchronous scanning operation, the shredding process 18 returns a “done” message to the *erase file* process 14 (symbolized by arrow 32), such that the *erase file* process 14 may signal the relevant primary process 10 that it may resume operation, and moves the pointers to the (shredded) files on to a next storage location 28 (operation symbolized by arrow 30), that is dedicated to asynchronous shredding. The data locations of the files are now further overwritten in a background process, not further hampering the primary processes for the actual document data processing 10.

During the shredding process, status information of the process is communicated by the shredding process 18 to a display process 36 for informing an operator of the security situation of the system, as will further be explained with reference to Figure 9.

As an alternative to the mixed mode shredding explained above, also fully asynchronous scanning may be effected for less important information. This mode can easily be done with the same composition of processes and structures as shown in

Figure 1, with the exception that files for shredding are removed directly to the storage location 28 dedicated to asynchronous shredding. This is symbolized by hatched arrow 34 in Figure 1.

A particularly relevant application of the asynchronous shredding occurs with a shared printer facility, because such a printer may be shared among various different persons and groups. In such a case, security is all the more important. In particular, an advantageous organization of the present invention would be applied to printers using a mailbox concept as is featured by several printers and digital copiers, e.g. those marketed by Océ: therein, all yet unprinted files reside in the mailbox, and, moreover, printed files will remain there for some time, until they are actively (or automatically, after a predetermined time interval of, say, 24 hours) removed. The mailbox is implemented as non-volatile memory, such as a hard disc. Therefore, if a user deletes a file from the mailbox, it is not only deleted from the file administration, but also shredded. Also in this case, it is advisable to execute the first shredding run in a synchronous manner, i.e. immediately after the delete command. Of course, for optimal security all job files should be stored in encrypted form and only be decrypted when the data are needed for printing, while decrypted data are kept in volatile memory only.

Further, the invention can be applied in a digital copying and a scanning environment. Digital copying and digital scanning are notoriously data intensive. This raises the need for effective shredding.

Note that in a situation like Figure 4 or 5, the effective shredding operation of Job1 may extend beyond the start of Job3, so that various jobs can be simultaneously in the course of being shredded. This will require determining some prioritizing among the various shredding runs, as will be discussed infra.

The above shredding procedure started from the level of the individual file or bottom up. Another manner is to start on the system level: freeze the overall operations, and determine on the basis of file system operations whether deleted files should be shredded. This procedure will need an approach based on sector analysis. Usually, the operating system will sufficiently know the organization on the sector level.

Figure 6 illustrates a flow diagram of a mixed (a)-synchronous shredding operation. In block 40, the operation starts, and the necessary hardware and software facilities get claimed. This operation will for example occur at power-up. In block 42, the system determines that a particular file or storage area is no longer needed. The organization of a pertinent waiting loop has not been shown. In block 43, the system determines whether shredding is necessary. If not, the system goes to block 52. If

shredding is necessary, the system goes to block 44, to determine whether synchronous shredding is necessary. If not, the system goes to block 48. If synchronous shredding is necessary, the system goes to block 46, for executing the synchronous shredding. Thereupon, in block 48, the asynchronous shredding is executed.

Starting of the asynchronous shredding is triggered when an interval occurs in between the normal data processing operations. Such interval can be detected through only the start thereof, or through finding that a particular interval is predicted to have at least a certain length. In this latter case the shredding operation can on this lower level have a temporary precedence over the normal data processing. After a certain amount of shredding, the system may interrogate the standard data processing as to its storage access requirements. Such background processing is, however, a basic mechanism of the particular operating system, and the precise implementation is not part of the present invention.

If shredding operations are in progress on more than one file, some priority organization is maintained. A first solution therefor is a first-come-first-take basis, so that the files are treated according to the sequence in which they were found shreddable, usually apart from the synchronous part of the shredding operation. A second approach is through letting the *ranking number* of the shredding pattern (cf. Figure 7) determine the priority. Still another approach is through assigning a priority level to all asynchronous shredding operations that pertain to a particular file. In a printer application, print files would take preference over logging files.

Now, as long as the shredding must proceed, the loop of blocks 48 and 50 ("ready?") revolves. If shredding is ready, the system proceeds to block 52, wherein the storage area in question is released for new system usage. Thereupon, the system reverts to block 42.

For simplicity, the interaction between the shredding and the other data processing operations has not been given in detail, but various priority-controlled operations may be executed. Such details are considered well within the knowledge of the skilled person.

Furthermore, a brief delay could be introduced immediately after the finishing of a document data processing job, before a shredding process is allowed to start. Otherwise, if a new document data processing job starts, although it has priority over shredding processes, if a shredding process has claimed the disc, it takes some time for the priority job to access its required storage locations on the disc.

Shredding, and more in particular the further shredding runs after a synchronous

initial shredding run, may also be effected off-line, e.g., just before shut-down of the system or at night or during quiet hours as a batch process. In fact, the storage sections used in print processing may in this case be used again after the initial shredding run, and only their use will be logged. Then, in the batch shredding at night, all files that have been logged as being used in the print processes during the day will be erased properly by shredding, leaving a "clean" system at the end.

The level of effectiveness of data shredding depends on the number and content of the shredding patterns used. Every additional shredding run makes retrieval of the original bit pattern on the storage medium more difficult, the more so when the shredding patterns used differ from run to run. It is therefore within the scope of the present invention to offer a settable security level to the users.

In a first embodiment, the printer driver window 70 shown on the user's PC contains a security level setter in the form of a slide switch, a series of radio buttons or any other appropriate means. The printer driver window has a series of tabs and one of them (69), here designated "Erase Data", wherein a security level can be selected by clicking the appropriate button, as shown in figure 8a. Every security level relates to a predefined combination of shredding procedures and bit patterns as further explained below in relation to figure 7.

The list of options in Figure 8a includes:

- "highest", in which the maximum number of shredding patterns (refer to Fig. 7) is used, in a synchronous manner
 - "high", in which an initial synchronous shredding run is followed by an asynchronous process using a selection of, e.g. 15, shredding patterns
 - "medium", in which only an asynchronous shredding process is applied, with a limited number of shredding patterns
 - "custom", in which a user may select his own choice of number and selections of shredding runs, including applying them as synchronous or asynchronous runs.
- Of course, other options would be within the scope of the invention.

In the basic screen of the printer driver, shown in figure 8b, there is a tick box 71 for defining the particular print job as a security job. When a user does so, a forced dialogue will appear (not shown) for the user to enter a PIN code, that he will be asked for when it comes to printing the job. Other possible forms of security jobs involve other security tokens that are suitable for identifying or otherwise authorizing a user, such as a fingerprint, iris scan or code card. Defining a print job as a security job automatically enforces erasure by shredding of the print data in the printer after finishing the job.

In an alternative embodiment, the selection of the security level is reserved for a super user (also known as "key operator") or the system administrator, who may enforce a security level for all users, using a selection window similar to the one shown in figure 8a.

Figure 7 illustrates a Table of shredding patterns as proposed by Gutmann in his article mentioned in the introduction, which can be used in the present invention. The left hand column shows respective data passes or shredding runs. The second column shows the various shredding patterns. Certain thereof are random as generated by a random pattern generator not shown. Others are produced by a sequence of more elementary patterns, produced by appropriately repeating bit patterns as shown in the column in a hexadecimal "shorthand" notation ("0x" signalling hexadecimal notation) . For example, the seventh row would produce a repetition of the bit pattern 0x92, 0x49, 0x24, or "10010010 01001001 00100100". As shown, up to 35 shredding runs can be needed. An optimum case is attained through executing 35 successive runs with various different patterns as shown. To attain a balance between possible performance degradation and the needed security, the number of erasure patterns can be selected between 0 and 35. At 35 runs, the scheme is used as shown. When fewer runs are applied, a predetermined selection is made.

Figure 9 shows an information window 100 that can be projected on the local device display upon actuation of a key on the operating panel (not shown). It shows the actual security-related situation of the internal hard disc of the device, as entries for "data in use" (excluding system files not related to document data), "data in erasure process", and free ("clean") disc space in bytes. A "fuel gauge" type display item 110 is added to give a quick impression of the situation. In an alternative embodiment, only entries for "data in use" including data scheduled for shredding, but not yet completely erased, are shown. It would of course also be possible to show the information specified in numbers of files. With this indication on the device display, a user can instantly see if the data that should be removed are indeed gone. The necessary information to make the display can easily be delivered by the operating system resident in the device controller and can be refreshed regularly, to give a dynamic situation display. The display window can also be shown on the workstation of the system administrator.

The present invention may be used advantageously for scanning and copying in a very similar way. In scanning, on a dedicated scanner device or on a multifunctional device, a user must first enter his name or other identification code, and may then scan documents. The scan data are then stored on the system disc of the device in

connection with the id of the user. The user may then return to his workstation, contact the scanner and fetch his scan file. According to the present invention, the scan data are then removed from the disc and the used disc sectors are erased by shredding.

In a copy operation, a scanner scans the original document, stores the scan data on the internal disc of the copier, whereafter the printer prints the copies from the disc. Shredding of the used disc space may be done automatically or in reaction to a setting made by the operator while setting the copy job parameters.

In all cases mentioned, a shredding procedure effected wholly or at least partly as a background process according to the present invention, will only slightly disturb new print, copy or scan jobs.

Now, the present invention has hereabove been disclosed with reference to preferred embodiments thereof. Persons skilled in the art will recognize that numerous modifications and changes may be made thereto without exceeding the scope of the appended Claims. In consequence, the embodiments should be considered as being illustrative, and no restriction should be construed from those embodiments, other than as have been recited in the Claims.

CLAIMS

1. A method of erasing, in a document data processing device, a stored information pattern on a rewritable data carrier that is accessible by a data processing facility of said device,
the document data processing device having
 - primary processes for processing document data, wherein data may be stored on said data carrier, and
 - secondary processes for erasing stored data, through overwriting a selected storage area of said carrier by a shredding pattern,
said method being characterized by running said primary and secondary processes asynchronously, i.e. starting a primary process irrespective of the completion of any pending secondary processes.
2. The method of claim 1, characterized by scheduling primary processes and any secondary processes in such a way that at least an end phase of any secondary process is run asynchronously, such as not to interfere with the primary processes.
3. The method of claims 1 or 2, characterized by at least a part of any secondary process is assigned a lower priority than the primary processes.
4. A method as claimed in any one of the preceding claims, wherein said overwriting in a secondary process is made up from a sequence of shredding runs with shredding patterns that each at least substantially cover the whole of said selected storage area.
5. A method as claimed in Claims 3 or 4, wherein an initial shredding run of a secondary process vis-à-vis a storage area used in a first primary process is effected synchronously, i.e. by preventing starting of a next primary process following said first primary process until completion of said initial shredding run.
6. A method as claimed in any one of the preceding claims, wherein after completion of a primary process, the relevant stored data files are directly removed from

the system administration table, and a secondary process for erasing said data files is run asynchronously.

7. A method as claimed in any one of the preceding claims, wherein said overwriting is implemented on the basis of an intended security level vis-à-vis said selected storage area.

8. A method as claimed in Claim 7, wherein said overwriting is automatically effected for a security job, that is a print job that can only be printed after entering a security token, such as a password, at the printer location.

9. A method as claimed in any one of the preceding claims, wherein said data processing device is a printer.

10. A method as claimed in any one of the preceding claims, wherein said data processing device is a digital copier.

11. A method as claimed in any one of the preceding claims, wherein said data processing device is a scanner.

12. A method as claimed in any one of the preceding claims, wherein said selected storage area includes storage of temporary, intermediate, and other data used by an actual operating system during a particular primary process.

13. A method as claimed in any one of the preceding claims, wherein a storage area on said rewritable data carrier is automatically designated for said overwriting by a shredding pattern upon an out-of-use information being present with respect to said storage area.

14. A method as claimed in any one of the preceding claims, wherein said overwriting is terminated as based on a predetermined sufficient overwriting activity on a storage area of predetermined sufficient size.

15. A method as claimed in any one of the claim12, wherein said overwriting is scheduled by respective priority levels among various deletable files, such priority level

being controlled by any of the first-come-first-take principle, or the identity or character of a particular file.

16. A method as claimed in any one of the claim 4, wherein said overwriting is scheduled by respective priority levels among various deletable files, such priority level being controlled by the ranking of a particular shredding run among a plurality of shredding runs applicable to a particular file.

17. A method as claimed in any one of the preceding claims, whilst introducing a brief delay after a primary process, before starting the shredding for a particular file.

18. A method as claimed in any one of the preceding claims, further including displaying overall progress of secondary processes in the device, to give an indication of an internal data security situation.

19. A document processing device comprising
 a data processing facility and
 a rewritable data carrier accessible by said data processing facility,
 said data processing facility running
 - primary processes for processing document data, wherein data may be stored on said data carrier, and
 - secondary processes for erasing data stored in said data carrier, through overwriting a selected storage area of said carrier by a shredding pattern,
 characterized in that
 the data processing facility is adapted to run said primary and secondary processes asynchronously, i.e. starting a primary process irrespective of the completion of any pending secondary processes.

20. A device as claimed in claim 19, including
 a scheduler for scheduling primary processes and any secondary processes in such a way that at least an end phase of any secondary process does not interfere with the primary processes.

21. A device as claimed in claims 19 or 20, wherein
 a secondary process includes effecting a sequence of shredding runs with shredding patterns that each at least substantially cover the whole of said selected

storage area.

22. A device as claimed in claim 21, wherein
a secondary process includes effecting an initial shredding run synchronously,
i.e. by preventing starting of a next primary process until completion of said initial
shredding run, and effecting further shredding runs asynchronously.
23. A device as claimed in any one of the claims 19 to 22, wherein
the data processing facility checks a file of document data for presence of a
predefined added information item, and wherein the data processing facility
automatically applies a second process to a file having said added information item after
having applied a first process to said file.
24. A device as claimed in any one of the claims 19 to 23, including
means for monitoring the overall progress of secondary processes in the device,
and
means for displaying said progress to an operator, to give an indication of an
internal data security situation.
25. A device as claimed in any one of the claims 19 to 24, wherein said device is a
printer, a copier or a scanner.