Data Wiping System with Fully Automated, Hidden and Remote Destruction Capabilities

GEORGE PECHERLE, CORNELIA GYŐRÖDI, ROBERT GYŐRÖDI, BOGDAN ANDRONIC
Department of Computer Science
Faculty of Electrical Engineering and Information Technology, University of Oradea
Str. Universitatii 1, 410087, Oradea
ROMANIA
gpecherle@uoradea.ro, cgyorodi@uoradea.ro, rgyorodi@uoradea.ro, andronic_bogdan@yahoo.com

Abstract: - In this article, we will describe a method to securely erase sensitive data in fully automated and hidden mode and with remote data destruction capabilities. Compared to other similar technologies, our method has two main advantages. The first one is the ability to run in a fully automated mode, in other words the system is configured once and the computer is protected without requiring any user intervention. The second advantage is the ability to run in a so-called hidden mode, in which the system looks like a different software, for the main purpose of confusing other users. Also, our system can be useful to prevent data loss in stolen laptops, by triggering remote wiping of sensitive data. This is done by overwriting the encryption key of an encrypted volume, that makes the data completely unrecoverable. Some tests and results that show data is not recoverable are also presented at the end of the paper. We will describe the structure and functionality of our system, and some of the most important technologies and algorithms that we have used.

Key-Words: - security, data wiping, data recovery, automation, scheduling, patterns, overwrite data

1 Introduction

Deleting a file using the operating system functions is not a secure operation. When a file is deleted, the operating system marks the disk areas previously occupied by the file as available for new data. Therefore, the old information is still on the hard drive, until new files happen to be saved in exactly the same locations. This information can be easily recovered by any basic software recovery tool [3].

Files can be deleted by:

1. The Windows user: for example, when the user deliberately removes one or more files that he no longer needs;

2. The Windows operating system or installed applications: during their normal operation, most applications create and remove temporary data, without the user's knowledge or approval.

In addition to the free disk space that can store a lot of previously deleted data, almost all applications save information on the hard drive that is meant to improve the user's experience. For example, web browsers save web pages, images and videos for quick access in the future (the web browser's cache). They also store a list of previously visited websites to enable the user to locate them faster (the web browser's history). A side effect of storing all this information is that it offers anyone a real portrait of the user's activity on the computer. And this is not always desirable [20].

To permanently wipe all traces left by Windows or applications, two important steps must be followed:

1. Finding out what information (files, registry keys, etc.) contain activity traces.

2. Securely erasing this information by using repeated overwrite operations to make it completely unrecoverable [3].

Normal users do not know where to look and find this information and even if they knew, securely erasing it in a continuous way would be a difficult and time consuming task. That's why, it is necessary that a software application (initially configured what to do) will do this job automatically and permanently. Almost all users have the experience and knowledge of running an antivirus on their system. The main advantage of a powerful antivirus is to let it do its job in the background and automatically eliminate threats (viruses, spyware, etc.). Our software acts on the same principles, the only difference is that it won't eliminate viruses or spyware, but it will eliminate sensitive data from the computer, such as traces left by other applications.

2 Advantages over similar technologies

There are a lot of data wiping software products that can destroy the traces left behind by the operating system or other applications. However, our research [1] indicates they have two main problems:

- Not completely automated: the user has to configure and run the wiping process periodically. Between these wiping processes, private data is saved and in danger of being discovered.
- Not completely hidden: even if some products hide the application while wiping data, there are still ways to determine that some hidden processes are running. Also, such hidden processes can be detected as possible threats by anti-spyware software [2].

The system we propose can address both of the problems above. The purpose is to develop a wiping system that ensures that at any moment, no sensitive data is left behind (the system is always clean), without requiring any user intervention. The model we propose will lead to a new concept in data wiping: “just install and let it do its job”. The main idea is to detect changes at specific locations (files/folders/registry keys), considered to be private locations, and erase new or updated data immediately [21].

Our idea of developing a fully automated wiping system came from a similar solution for a different problem, data backup. Genie Timeline from Genie-Soft [6] makes backup copies of user’s data in an automated way, following the same principle, “set up and forget about it”. We thought that something like this would apply very well for secure data wiping, not only data backup.

3 Implementation of our system in a software application

In order to show its advantages, we have implemented our fully automated and hidden system in a software application, that we have developed in Visual Studio 2008 using the .NET Framework and C# language. We tried to design it as simple as possible, so that it does not consume a lot of system resources and to be as fast as possible when running in the background to avoid slowing down the entire system [8].

The erasing mechanism is based on a high level erasing. This is done by opening the file that needs to be erased, replacing everything with random data and then saving it. This process is repeated for a number of times before the file is finally deleted [9].

We have chosen this method of high level wiping and not the low level wiping method (hard drive sector based) because of the nature of the file system: the fragmentation problem. Thus, the problem is avoided and the result is basically the same as low level wiping.

3.1 Secure data overwriting methods

These methods are based on the fact that writing new data in areas where previous data exists, will make old data become inaccessible.

Also, overwriting data is a cheap way to destroy information, because it can be implemented very easy in software applications, like we did with our current solution.

The easiest way to overwrite data is to use the same pattern, usually zeroes. This method will at least prevent reading the data using standard operating system functions. However, to prevent data recovery using more advanced methods, it is recommended that specific patterns are used. For example, writing zero and one is more efficient than writing just using zero.

Subsequent studies have shown that there are methods to overwrite data even after they have been overwritten using the above methods. For example, Peter Gutmann showed that a technology called “magnetic force microscopy” can recover data and he developed special overwriting patterns to stop this from happening. These patterns have become known as the Gutmann overwrite method [10].

Governmental agencies also came with their own solution: the United States Department of Defense (US DoD) has developed a proprietary standard for the secure deletion of sensitive information, called DoD 5220.22-M [9]. Also, the same US DoD states, in November 2007, that, securely overwriting data is not an accepted and secure method and they recommend the physical destruction of the storage media or passing it through a demagnetization process, called degaussing. However, this should be applied only if the data to be erased is highly confidential. For data having a normal degree of confidentiality, overwriting data is accepted.

Peter Gutmann was a researcher at the University of Auckland, New Zealand. He is especially known for his studies and research work in the data security field. One of his most important papers [10] has been published in the Sixth USENIX Security symposium, from San Jose, California. His work and results are applied at a very large scale.

Practically, his paper presents an algorithm to securely erase data from a hard drive beyond forensic recovery. For this purpose, 35 passes are used. Peter Gutmann selected these passes for several hard drive writing encoding mechanisms (MFM/RLL), being a method that can be applied without knowing the encoding mechanism of the hard drive that is erased. However, a user who knows the type of encoding that is used can select only the steps that are specific to that mechanism.

Why are such an algorithm and an overwrite method of 35 passes needed to securely erase data beyond recovery? Why isn’t a single overwrite enough? Peter

Gutmann says that a single overwrite is not enough to ensure data is unrecoverable. He claims that a standard method to recover overwritten data is to capture the analog signal obtained from the write head before decoding. Although this signal will be very close to a signal called "ideal digital signal", the difference is the one that matters. By calculating the ideal digital signal and substracting it from the analog signal, it is possible to ignore the last information that was written, to amplify the remained signal and view the data that existed before [10].

Also, Peter Gutmann says that after overwriting with random data, data recovery is still possible. The permittivity of the medium changes at the same time with the magnetic field frequency. This means that a lower frequency field will be able to go deeper into the magnetic parts of the hard drive, than one of a higher frequency.

The patterns used to overwrite data, as part of the Gutmann's algorithm, will apply alternative magnetic fields with variable frequencies and phases. The first 4 passes are generated randomly, followed by passes 5-31 executed randomly and then the last 4 passes also generated randomly. Passes 5-31 have been designed for all encoding mechanisms (both RLL and MFM), so they can be used in all cases. The final result will be that data cannot be recovered, even by advanced data recovery techniques [10].

There have been some criticism brought to the Gutmann theory by which forensic agencies could recover data even after it has been overwritten using the Gutmann method. It is known that the delete function in Windows just marks the file as being deleted. However, after data has been overwritten, it cannot be recovered using software recovery tools, because the operating system returns only the current content, not the old content. However, Peter Gutmann says this is not true and there are special recovery methods, such as MFM (Magnetic Force Microscopes) that, combined with image analysis tools, can detect the old values of the bits from a magnetic storage (such as a hard drive).

The truth is this issue hasn't been proved and there is no clear evidence of forensic agencies being able to recover data after it has been overwritten. A contradictory issue is that the US Government doesn't approve data overwriting as a recommended method for highly confidential data and they recommend physical destruction.

Also, data recovery companies cannot recover data that has been overwritten (or at least they don't make this public). These companies mainly do data recovery from hard drives that are physically damaged and that suffered several types of shocks (mechanical, caused by water, fire, etc.).

An interesting study was made public by the National Bureau of Economic Research (NBER) from the United States [14]. They have responded to Gutmann's theory analyzing all his paper references.

Peter Gutmann says that, when overwriting bits, "the effect is closer to value 0.95 when a zero is overwritten by one, and closer to value 1.05 when a one is overwritten by a one" [10]. None of the references show examples of data that has been recovered, they only describe experiments where STM (scanning tunneling microscopes) have been used to examine individual bits of data, without any special meaning. NBER says that MFM microscopy or STM is generally used to test and improve the quality of read/write heads of hard drives.

Another aspect is that no copy of the following paper has been located: „Detection of Digital Information from Erased Magnetic Disks", written by Venugopal Veeravalli [11]. On his Internet page, it is shown that the paper has not been published and that it is only in theoretical stage and that there are no experiments to prove those theories.

Also, NBER interrogated several data recovery companies and they all claim they can't recover data that has been overwritten. Although we don't know this for sure, there is no clear evidence to prove that data cannot be recovered. Therefore, it seems that physical destruction remains the best choice for highly confidential data. However, data overwriting remains the most trusted method to securely erase confidential data, also being the most accessible, that's why we are also using it in this paper.

3.2 Our application structure

The following will describe the structure and the main functionalities of our application.

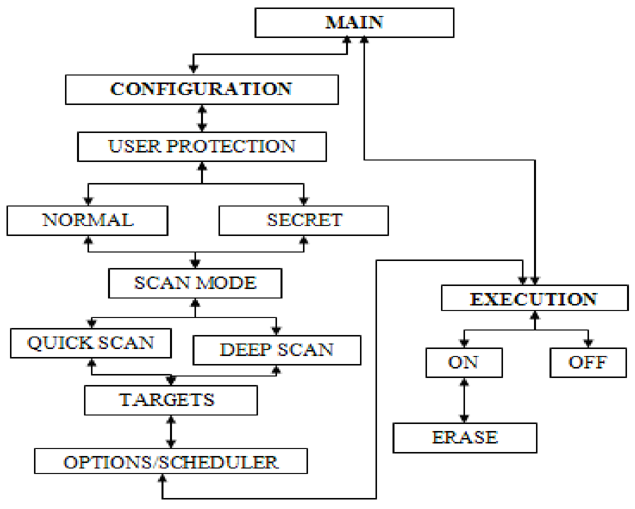


Fig. 1. Our application structure

When the Main Application starts, the user will be asked if he wants to configure the application or run directly with pre-configured options. This way, we can identify two main modules:

- The Configuration Module
- The Execution Module

1. The Configuration Module: when we designed the configuration module, we started from the idea that it has to be as simple as possible. That's why we divided it in 3 steps:

STEP 1 - The selection of the running mode: There are two categories of running modes.

The User Protection defines how the user is protected against someone who wants to see what this program is used for. This mode can have two values:

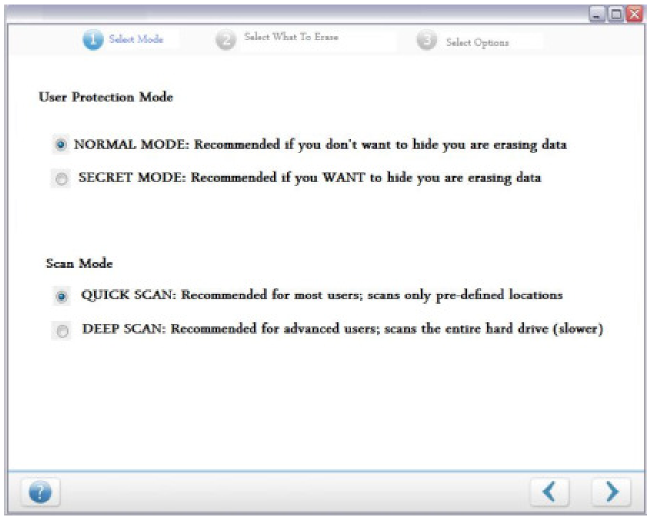
a) Normal Mode: a mode in which the program runs with an interface of a real data wiping system. This mode is recommended if you don't want to hide you are erasing data.

b) Secret Mode: a mode in which the program runs with an interface of a fake data wiping system. One example is to run it with an interface of an antivirus application. Everyone would think you are removing viruses and no one would suspect that you are actually erasing your secret data. Therefore, this mode is recommended if you want to hide you are erasing data.

The Scan mode defines what domains are scanned for sensitive data. This mode can have two values:

a) Quick Scan: a mode in which the program scans only pre-defined locations. These locations can be altered in the next step (Select What To Erase). The data is wiped immediately without user confirmation because the locations are considered to contain only private data. This mode is recommended for most users as it is a good compromise between performance and security.

b) Deep Scan: this will search globally (on selected disk drives) and determine new/updated data. Because not all data is sensitive, we can apply certain filters to select only sensitive data. These filters can be: files that contain specific keywords, files of a certain type (MS Word, JPEG, etc.). The user can choose to trust these filters or he can select the user confirmation mode. The confirmation mode can also be optimized by choosing to erase immediately those files that are located in the pre-defined paths. Confirmation may seem like a non-automated way, however this is only until the system "learns" what data is sensitive or not. Little by little, the system will be able to take its own decisions and erase data without user confirmation. This is similar to the way anti-spam enabled applications work, such as Mozilla Thunderbird's Junk Mail feature [4]. They will ask for confirmation only until they "learn" what spam (in user's own opinion) is.



User Protection Mode

[✓ NORMAL MODE: Recommended if you don't want to hide you are erasing data]

SECRET MODE: Recommended if you WANT to hide you are erasing data

Scan Mode

[✓ QUICK SCAN: Recommended for most users; scans only pre-defined locations]

DEEP SCAN: Recommended for advanced users; scans the entire hard drive (slower)

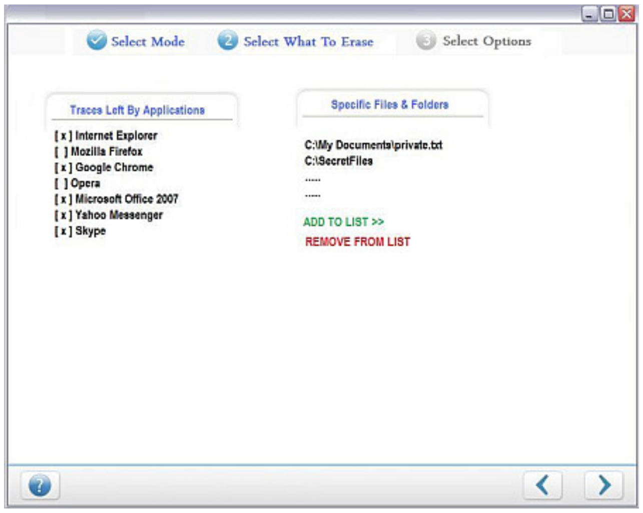
Fig. 2. Select Mode window

STEP 2 - The selection of what to erase: At this step, the user can select three types of sensitive areas. The first two are available both in the Quick Scan and Deep Scan modes. The third one is only available in the Deep Scan mode:

a) Traces Left By Applications: the user can select which application(s) he wants to erase usage traces for (Internet Explorer, Mozilla Firefox, Google Chrome, Opera, Microsoft Office, Yahoo Messenger, Skype, etc.). To define these options, we use a sensitive area definition language, called XSAD (eXtended Sensitive Area Definition), developed by us, as an extension to the OSAD language that we proposed at [2]. This new version of our language is based on XML and has a cleaner structure and better support for web based wiping systems. We use this language to define what sensitive areas (files/folders/registry keys) should be monitored for sensitive data. These sensitive areas can be places where applications leave their traces (such as the web browser's cache, history, cookies) or user defined sensitive areas.

b) Specific Files and Folders: the user can define individual files or folders to target. File masks, such as *.txt, are allowed to be used. At the implementation level, we use the same XSAD structure to save these files/folders locations.

c) File Filtering Rules (available only in the Deep Scan mode): it is a tool described above at step 1 that allows the definition of filtering rules based on file types or file contents. Based on these criteria, a file can be categorized as being sensitive or not. For example, someone could define a rule that all Microsoft Word files containing the term "financial plan" should be considered private.



Traces Left By Applications

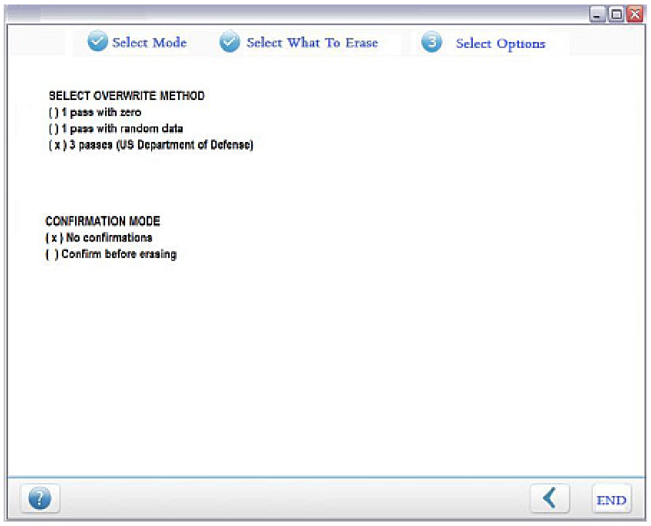
[x] Internet Explorer
[ ] Mozilla Firefox
[x] Google Chrome
[ ] Opera
[x] Microsoft Office 2007
[x] Yahoo Messenger
[x] Skype

Specific Files & Folders

C:\My Documents\private.txt
C:\SecretFiles
......
......

Fig. 3. Select What To Erase window (Quick Scan)

STEP 3 - The selection of options: At this step, the user can select various options, such as the overwrite method, whether he wants to confirm files before erasing or let it run in automated mode (default), the scheduling options (by default, the scanning process takes place continuously in the background, however the user can select a less frequent scanning, for performance reasons).



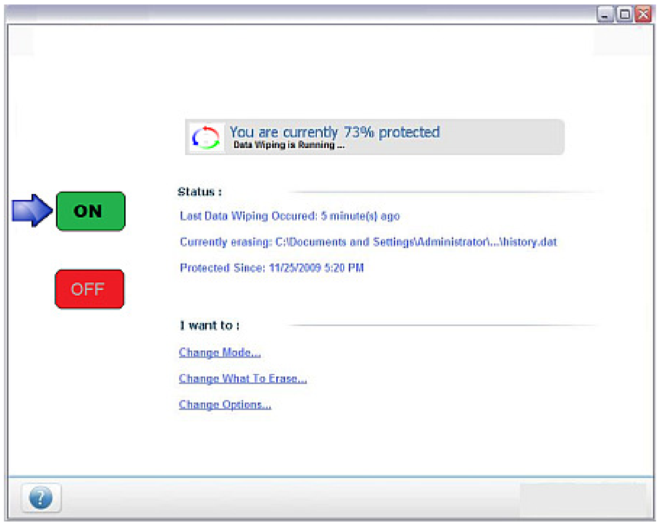
SELECT OVERWRITE METHOD
( ) 1 pass with zero
( ) 1 pass with random data
(x) 3 passes (US Department of Defense)

CONFIRMATION MODE
(x) No confirmations
( ) Confirm before erasing

Fig. 4. Select Options window

2. The Execution Module: after following the steps above, the system is configured and ready to be run in an automated way. The last window is the main interface of the program that is displayed when the program is running. From this window, the user can switch protection ON or OFF, he can view the status of the protection (what is being erased, when the last wiping process was run, etc.), or he can go back to the configuration module to change settings. A progress of the wiping operation is also displayed.

The execution module can also be accessed by double clicking its system tray icon. This icon will change its appearance based on the protection status (green – protection ON, red – protection OFF).


You are currently 73% protected
Data Wiping is Running ...

Status :

Last Data Wiping Occured: 5 minute(s) ago

Currently erasing: C:\Documents and Settings\Administrator\...\history.dat

Protected Since: 11/25/2009 5:20 PM

I want to :

Fig. 5. The Execution Module

When the erasing process starts, the following code is executed:

```
data = DateTime.Now.ToShortDateString ();
label4.Text = data;
int nrFiles = 0;
int nrFilesDeleted = 0;
for(int i=0;i<listBox1.Items.Count;i++)
{
      System.IO.DirectoryInfo dir = new
      System.IO.DirectoryInfo
      (listBox1.Items[i].ToString());
      foreach (System.IO.FileInfo f in
      dir.GetFiles ( "*.*" ))
        {
          nrFiles++;
        }
      }
      progressBar1.Maximum = nrFiles;
      for ( int i = 0; i <
      listBox1.Items.Count; i++)
      {
      System.IO.DirectoryInfo dir = new
      System.IO.DirectoryInfo (
      listBox1.Items[i].ToString () );
      foreach ( System.IO.FileInfo f in
      dir.GetFiles ( "*.*" ) )
        {
      nrFilesDeleted++;
      progressBar1.Value =
      nrFilesDeleted;
      int sleep = nrFiles / 1000;
      System.Threading.Thread.Sleep
      (sleep);
      overwriteFile(f.FullName);
```

```
    File.Delete (f.FullName);
    }
    }
```

Fig. 6. Scanning and erasing files

First, we save and display the current date and time to a label, as a start point for the erasing process. We then take the list of files/folders that have to erased, from a list box. We initialize the progress bar to take the number of files as a maximum value and then we start going through all files/folders. For folders, we get all files inside them (*.*) and erase these files one by one, by using the overwriteFile function to overwrite its contents and then delete the file using File.Delete. After erasing each file, we wait for a number of milliseconds (number of files divided by 1000), otherwise the progress would advance too fast.

The code from above is executed in two distinct situations:

- When the user manually launches the wiping process, on demand, or
- When the protection is switched on (see Fig. 5); in this case, a timer is being used to repeat this process at a specified time frame, to ensure a continuous protection.

Here is a basic version of the ovewriteFile function, used to overwrite file contents:

```
private void overwriteFile (string
fileName)
{
    FileInfo f = new FileInfo (
    fileName);
    long bytes = f.Length;
    StreamWriter write = new
    StreamWriter ( f.FullName );
    for ( long i = 0; i < bytes; i++ )
    {
        if(trecere==0)
        write.Write ( 0 );
        if (trecere==1)
        {
          //generate random data
          write.Write ( randData );
        }
    }
    write.Close ();
}
```

Fig. 7. Overwriting file contents

If the value of variable “trecere” (according to user’s selection) is 0, it means that zero has to be written over the entire file contents. If the value of “trecere” is 1, random data will be used for the overwriting process.

We haven’t included the random generation process here, as it can be any algorithm. However, for increased security, we recommend an algorithm such as ISAAC CSPRNG (cryptographically secure pseudo random number generator). We have shown the efficiency of this method, compared to other methods, in a previous research paper [15].

At that time, we applied the ISAAC CSPRNG using Borland Delphi, however this can be easily transformed into other programming languages, such as Visual C#. The code below is just a usage example, it actually writes the random values to a Memo (text box) in the main form:

```
var
   i: integer;
   x: TIsaac;
begin
   x.Create;
   for i:=1 to 20 do
Memo1.Lines.Add(IntToHex(x.val, 8));
   x.reSeed;
// reseed exactly as in Create()
   for i:=1 to 20 do
Memo1.Lines.Add(IntToHex(x.val, 8));
   // get the same values
end;
```

Fig. 8. ISAAC random number generation (Borland Delphi code sample) [15]

3.3 Integration of unused disk areas wiping as real time deletion

A very useful option to integrate is wiping the free unused areas of the hard drive, as a continuous process. The only disadvantage of this feature is that it consumes system resources, that’s why it is indicated to be scheduled to be run only when the computer is not used or when it is determined to be idle (no mouse move, no keyboard activity, etc.).

Wiping the free (unused) disk areas is needed because there can also be data that has not been erased using our solution (has not been securely wiped) and that data is located in areas marked as deleted information on the hard drive.

The easiest and most convenient way of wiping free space is to create large temporary files, with random data, until there is no free space left on the drive. This process actually overwrites the entire free space with random information, making the previously stored data unrecoverable. After we fill in all available disk space, we finally remove the big temporary files, to free up disk space. This process (to create temporary files and then remove them), can be repeated several times, depending on the number of passes the user has chosen.

4 Hidden System Design

Our second purpose was to prevent other users from discovering there is a data wiping system installed on a computer. For this, we designed our wiping system to look like an anti-virus software. An anti-virus software is something that everyone has, so it is not suspicious at all.

Here is how some data wiping tasks are "transformed" into anti-virus tasks:

- Graphical user interface: all texts and images are updated to be related to an antivirus
- Scan process: when the system is searching for sensitive data, we rename this to something related to anti-virus scanning
- Wiping process: when the system is wiping sensitive data, we rename this to something related to virus removal. We will not show the names of all files that are removed, because there are usually thousands of files that are wiped, and it is usually more difficult to have so many infected files.
- Update process: when XSAD files are updated, we inform the user that virus definition files are updated
- Virus names: we can use a public virus list [5] and assign virus names randomly to files that we wipe

An option to switch to the normal (unhidden) mode, where everything looks like in the reality, is also offered to the user. This can be done from the Configuration module, in the Select Mode window (see Fig. 2).

5 Sensitive Areas Configuration Files. The XSAD File

The configuration module has a panel where the user can select the applications he wants to erase sensitive traces from. Each of these applications (Internet Explorer, Mozilla Firefox, etc.) store their history traces in different places. In order to prevent hard coding these locations in our product, we developed a file structure that uses a definition language which we called XSAD (eXtended Sensitive Area Definition) [3]. This is actually pure XML language, customized for our own needs [7]. We tried to make it as simple as possible.

Here is an example of an XSAD file for Internet Explorer that shows the most important elements that can be used:

```
<?xml version="1.0" encoding="utf-8"?>

<SensitiveAreas>
<SensitiveArea name="Internet Explorer">
<Location name="Cache">
<Item detection="dynamic" type="folder">
<Detection>
<RegistryKey>
HKEY_CURRENT_USER\Software\Microsoft
\Windows\CurrentVersion\Explorer\Shell Folders
</RegistryKey>
<RegistryValue>
Cache
</RegistryValue>
</Detection>
</Item>
</Location>

<Location name="Most Recently Used">
<Item detection="static" type="regkey">
HKCU\Software\Microsoft\Windows\CurrentVersion
\Explorer\ComDlg32\LastVisitedMRU
</Item>
<Item detection="static" type="regkey">
HKCU\Software\Microsoft\Windows\CurrentVersion
\Explorer\ComDlg32\OpenSaveMRU
</Item>
</Location>
</SensitiveArea>

<FilesFolders>
<Item type="folder">
C:\Documents and Settings\Administrator
\My Documents\Private Data
</Item>
<Item type="files">
C:\Documents and Settings\Administrator
\My Documents\Company Plans\*.doc
</Item>
<Item type="file">
C:\Documents and Settings\Administrator
\My Documents\Company Plans\SecretEmail.eml
</Item>
</FilesFolders>

</SensitiveAreas>
```

Fig. 9. An XSAD file for Internet Explorer

The XSAD file has two types of elements inside the root element <SensitiveAreas>. These two elements can be included in a single XSAD file (like in Fig. 9 above) or split in two distinct files for a cleaner structure:

- one or more <SensitiveArea> elements which contain what needs to be erased for a specific application. For example, a <SensitiveArea> element can be for Internet Explorer.
- one <FilesFolders> element that contains the files and/or folders defined by the user.

The <SensitiveArea> element has a name attribute containing the name of the application as it appears in the list. One or more nested elements, called <Location>, each with a separate name attribute, represent the various types of locations that can be erased from an application. For example, Internet Explorer has Cookies, Cache and History that have to be erased.

For each <Location>, we can have one or more <Item> elements which can be detected in two ways (this is specified in the detection attribute):

- static detection, in which the location is specified as it is (for example C:\My Secret Data\December.txt).
- dynamic detection, in which the location is determined following some rules (for example, the location of the Cookies folder for Internet Explorer is the value of a specific registry key).

If the detection is dynamic, there is a nested element called <Detection> containing other sub-elements that specify how the detection is done. In the example above, the Cache location is determined from a registry value. If the detection is static, the path can be specified directly as the value of the <Item> element.

Also, each <Item> can be of several types, and this is specified in the type attribute, which can be file, folder, regkey or regvalue.

The <FilesFolders> element can have several <Item> nested elements that describe what should be erased. In the type attribute, we specify what type of <Item> we refer to, and then in the value of the element, the path to the <Item> is provided, as in Fig. 9 above.

For increased security, we are using the following protection methods for XSAD files:

- the name of the XSAD file is not a descriptive one. It can be a random name and with a random or confusing extension (for example A5FB2U3C.DAT).
- the contents of the XSAD file are encrypted using an encryption method that can be specified by the user.

6 Proposed Updates System

Almost all applications change the way the store their activity traces, when new versions come out. For example, the might change the location (folders, registry keys) where they store this information. Also, new features can be developed in these applications and this involves new sensitive areas to be targeted.

This means that XSAD files also have to be updated, so that the wiping software can keep track of these changes and erase sensitive data correctly and completely. We have thought of two possible sources XSAD files may come from:

1. From the developers of the software. This usually involves having a dedicated team, who is constantly analyzing 3rd party applications and their changes. Their main task is to download these software, install them on as multiple machine configurations as possible and detect what sensitive data they store on the computer and where. For this task, they can use specialized software that detect changes in their computer, when the analyzed software is being run. This software will monitor changes to files and registry on the local system. It is recommended that one person is doing the task of detecting sensitive areas, and another person (preferably, a developer) is implementing the analysis results into XSAD files.
2. From the users community. This might be one of the most valuable sources of XSAD files updates, because developers might develop XSAD files for software which are not so popular among users. For this purpose, we propose a Web 2.0 website, like a wiki, in which users can write detailed pages on what software they recommend adding and which are the sensitive areas they propose. We can have one wiki page for each software. The great thing about this wiki is that any user can contribute even for software pages that have been opened by other users. This is something like free collaboration. The information from this wiki site can then be used by developers to develop or update the existing XSAD files and deploy them.

After XSAD files have been developed or updated, the next step is to deploy them. We can use a web server in which we upload all these files and manage them using a database (a MySQL database would be enough for this purpose). Each XSAD file should have a signature used to determine whether the user already has the latest version of the XSAD file. If the signature is different and newer than what the user has on his computer, the XSAD file will be updated. Otherwise, it will not be updated.

7 Remote Wiping

Another interesting feature we propose for the wiping system is to remotely wipe data. This is especially useful for mobile computers, such as laptops, that can be easily stolen [18].

In case a laptop is stolen, the owner may not want sensitive data from it to fall into the wrong hands. When this happens (the laptop has been stolen), the owner can issue a remote wiping over the Internet.

There are some preliminary steps the user has to perform (when he still has the laptop), to prepare the system for remote wiping:

1. Download and install an encryption software, that can create encrypted volumes, like TrueCrypt [16].
2. Create an encrypted volume and store all private data inside that volume [19].
3. Install our wiping software with the remote wiping feature activated.

The owner of the laptop will have access to a secure Internet location where, by default, an option to trigger the remote wiping, will be disabled. Our wiping software will constantly monitor (considering that Internet access is available), the value of the option. If it is disabled, nothing will be done.

Considering the laptop is stolen, the following should happen:

1. The owner must login to the secure Internet location and activate the option to trigger the remote wiping, as soon as possible.
2. When the laptop is turned on by the thief (considering that Internet location is also available), our wiping software will connect to the web server, detect that the remote wiping option is activated and instantly wipe the encryption key of the encrypted volume. Wiping the encryption key of an encrypted volume will make that volume's data completely unrecoverable.
3. Our wiping software will report back to the web server that the wiping of the encryption key has been performed successfully, or give an error code if it didn't succeed.

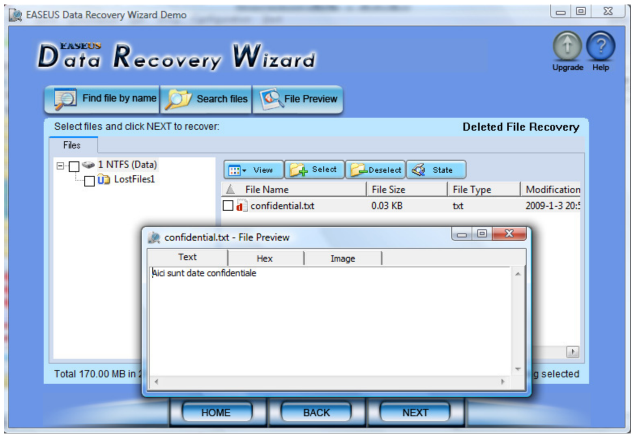
The disadvantage of this remote wipe system is that the computer has to be connected to the Internet after it has been stolen. However, we are looking for other ways to solve this problem and propose a solution that does not depend on an Internet connection.

9 Tests and results

We have performed a simple experiment to test the recovery of data before and after using our wiping solution.

The experiment was performed on a random PC (the configuration is not essential in this case, however for accuracy, we have used a system with Intel Core2 Duo - 2 GHz, 160 GB HDD, 2 GB RAM, Windows Vista Business 32-bit operating system). The steps performed are mentioned below:

1. We created a text file on an NTFS partition (E:), called confidential.txt.
2. We deleted the file using the Shift-Delete key combination, so that it does not end in the Recycle Bin.
3. We scanned partition E using a data recovery software tool. Any such tool can be used, we have used EASEUS Data Recovery Wizard [17].
4. The scanning results were that the confidential.txt file was recovered with its contents intact, as shown in Figure 10 from below.



Select files and click NEXT to recover: Deleted File Recovery

File Name	File Size	File Type	Modification
confidential.txt	0.03 KB	txt	2009-1-3 20:5

Aici sunt date confidentiale

Total 170.00 MB in 2

g selected

Fig. 10. File being recovered before using our solution

5. The next step was that we ran an unused disk space wiping. This process created large temporary files that overwrote all previously deleted data, including the data from the confidential.txt file.
6. We scanned partition E again using EASEUS Data Recovery Wizard and this time, the confidential.txt file was not recoverable.

The only problem was that wiping unused disk space usually takes a lot of time (in our case, it took about 4-5 hours, as we had over 60 GB free space), because the entire free space has to be overwritten.

A much quicker method would have been to directly erase the file using our wiping solution (a method that usually takes a few seconds). However, this would have worked if the file hadn't been deleted first, using Windows functions. If the file was deleted using Windows, we do not know where the contents of file are now, so we have to wipe the entire free space, to make sure that all previously deleted data (including this file) is erased beyond forensic recovery.

10 Conclusions and Future Work

There are a lot of data wiping solutions in the market and we wanted to bring out something unique, because we have determined that users find these solutions difficult to use and not providing continuous and complete protection. Our solution is mainly designed for home users however it can be integrated very well in a business environment, in small or large companies who want continuous data protection and to keep competitors away from their private information.

Also, many companies (the United States of America have such confidential data laws) have to get rid of old confidential documents after some time. In order to comply with these laws, they have to use secure data wiping software.

As future work, we want to bring more options to the user and to improve our user interface. Also, we want to think of a way to make our product work in a local network environment (not just on a local computer). In addition, we want to wipe other sensitive areas of the computer, such as the Windows Shadow Copies (also called "Previous Versions" of files), feature that is present in the newer versions of the Microsoft operating system (Windows Vista and Windows 7). This feature keeps old versions of files for backup purposes. Normal wiping just erases the original file, but not its shadow copies. This requires a special wiping technique that will be researched and proposed in a future work.

References:

[1] „Privacy Software Review 2009 – TopTenREVIEWS" -

[2] Spyware Removers, by CNET Download.com

[3] „Detection of Confidential Data Using the Open Sensitive Area Definition (OSAD) Language" – George Pecherle, Cornelia Gyorodi, Robert Gyorodi – IEEE ICCP 2009 – Cluj Napoca, Romania

[4] Mozilla Thunderbird Junk Mail - http://www.mozilla-europe.org/

[5] VirusList.com – http://www.viruslist.com

[6] Genie Timeline by Genie-Soft – http://www.genie-soft.com/products/genie_timeline/default.html

[7] XML Path Language (XPath) Version 1.0 - http://www.w3.org/TR/xpath

[8] MSDN - http://msdn.microsoft.com/

[9] National Industrial Security Program Operating Manual, reissued February 28, 2006

[10] "Secure Deletion of Data from Magnetic and Solid-State Memory", Peter Gutmann, Department of Computer Science, University of Auckland. Published in the Sixth USENIX Security Symposium Proceedings, San Jose, California, July 22-25, 1996

[11] The IEEE Computer Society – "Remembrance of Data Passed: A Study of Disk Sanitization Practices" - SIMSON L.GARFINKE, ABHI SHELAT (Massachusetts Institute of Technology) – January/February 2003

[12] "How to Forget a Secret", G. Di Crescenzo et al., Symposium Theoretical Aspects in Computer Science (STACS 99), Lecture Notes in Computer Science, Springer-Verlag, Berlin, 1999, pp. 500–509

[13] NIST Special Publication 800-88: "Guidelines for Media Sanitization", Recommendations of the National Institute of Standards and Technology, Richard Kissel, Matthew Scholl, Steven Skolochenko, Xing Li, September 2006

[14] Can Intelligence Agencies Read Overwritten Data?, a refutation of Gutmann's claims - http://www.nber.org/sys-admin/overwritten-data-guttman.html

[15] Robert Győrödi, Cornelia Győrödi, George Pecherle, Livia Bandici, "Secure Data Wiping Using the ISAAC CSPRNG" 13th IGTE Symposium 2008, September 21-24, 2008, Graz, University of Technology, Austria, pag. 278-281, ISBN 3-902465-07.

[16] TrueCrypt – free open-source, on-the-fly encryption

[17] Data Recovery Wizard - http://www.easeus.com/

[18] "A Study on Information Security Management System Model for Small and Medium Enterprises" - Wan-Soo Lee, Sang-Soo Jang - 8th WSEAS International Conference on Information Security and Privacy (ISP '09), pp. 84-88, ISBN 978-960-474-143-4, ISSN 1790-5117

[19] "Guide for Designing Cyber Security Exercises" - VICTOR-VALERIU PATRICIU, ADRIAN CONSTANTIN FURTUNA - 8th WSEAS International Conference on Information Security and Privacy (ISP '09), pp. 84-88, ISBN 978-960-474-143-4, ISSN 1790-5117

[20] "Controlling Your Personal Information Disclosure" - NORJIHAN ABDUL GHANI, ZAILANI MOHAMED SIDEK - Proceedings of the 7th WSEAS International Conference on INFORMATION SECURITY and PRIVACY (ISP '08), pp. 23-28, ISBN 978-960-474-048-2, ISSN 1790-5117

[21] "A New Cryptographic Algorithm for the Real Time Applications" - AHMED H. OMARI AND BASIL M. AL-KASASBEH, RAFA E. AL-QUTAISH AND MOHAMMAD I. MUHAIRAT - Proceedings of the 7th WSEAS International Conference on INFORMATION SECURITY and PRIVACY (ISP '08), pp. 33-39, ISBN 978-960-474-048-2, ISSN 1790-5117