Gutmann method

From Wikipedia, the free encyclopedia

The **Gutmann method** is an algorithm for securely shredding the contents of computer hard drives, such as files. Devised by Peter Gutmann and Colin Plumb, it does so by writing a series of 35 patterns over the shredded region.

The selection of patterns assumes that the user doesn't know the encoding mechanism used by the drive, and so includes patterns designed specifically for three different types of drives. A user who knows which type of encoding the drive uses can choose only those patterns intended for their drive. A drive with a different encoding mechanism would need different patterns. Most of the patterns in the Gutmann method were designed for older MFM/RLL encoded disks. Today, however, any relatively modern drive no longer uses these old encoding techniques, making most of the patterns in Gutmann's method superfluous. Indeed, Gutmann himself mentions in a follow up to his original 1996 paper that, "For any modern PRML/EPRML drive, a few passes of random scrubbing is the best you can do."

Contents

Technical overview

One standard way to recover data that has been overwritten on a hard drive is to capture the analog signal which is read by the drive head prior to being decoded. This analog signal will be close to an ideal digital signal, but the differences are what is important. By calculating the ideal digital signal and then subtracting it from the actual analog signal it is possible to ignore that last information written, amplify the remaining signal and see what was written before.

For example:

```
Analog signal:        +11.1  -8.9  +9.1 -11.1 +10.9  -9.1
Ideal Digital signal: +10.0 -10.0 +10.0 -10.0 +10.0 -10.0
Difference:            +1.1  +1.1  -0.9  -1.1  +0.9  +0.9
Previous signal:      +11    +11   -9   -11    +9    +9
```

This can then be done again to see the previous data written:

```
Recovered signal:     +11    +11   -9   -11    +9    +9
Ideal Digital signal: +10.0 +10.0 -10.0 -10.0 +10.0 +10.0
Difference:            +1    +1    +1    -1    -1    -1
Previous signal:      +10   +10   -10   -10   -10   -10
```

In 1996, when this method was developed, it was possible to use a digital oscilloscope to recover eight levels of overwrites, without damaging the drive. Since then higher disk densities have probably *reduced* the number of overwrites necessary to completely erase data.

However, overwriting the disk repeatedly with random data will not always work. The permittivity of a medium changes with the frequency of the magnetic field. This means that a lower frequency field will penetrate deeper into the magnetic material on the drive than a high frequency one. So a low frequency signal will still be detectable even after it has been overwritten hundreds of times by a high frequency signal.

The patterns used are designed to apply alternating magnetic fields of various frequencies and various phases to the drive surface and thereby approximate degaussing the material below the surface of the drive.

Method

An overwrite session consists of a lead-in of four random write patterns, followed by patterns 5-31, executed in a random order, and a lead-out of four more random patterns.

Each of patterns 5-31 was designed with a specific magnetic media encoding scheme in mind, which each pattern targets. The drive is written to for all the passes even though the table below only shows the bit patterns for the passes that are specifically targeted at each encoding scheme. The end result should obscure any data on the drive so that only the most advanced physical scanning (e.g. using a magnetic force microscope) of the drive is likely to be able to recover any data.

The series of patterns is as follows:

Gutmann overwrite method

Pass	Data Written		Pattern written to disk for targeted encoding scheme		
	In Binary notation	In Hex notation	(1,7) RLL	(2,7) RLL	MFM
1	(Random)	(Random)			
2	(Random)	(Random)			
3	(Random)	(Random)			
4	(Random)	(Random)			
5	01010101 01010101 01010101	55 55 55	100...		000 1000...
6	10101010 10101010 10101010	AA AA AA	00 100...		0 1000...
7	10010010 01001001 00100100	92 49 24		00 100000...	0 100...
8	01001001 00100100 10010010	49 24 92		*0*000 100000...	*1*00 100...
9	00100100 10010010 01001001	24 92 49		100000...	00 100...
10	00000000 00000000 00000000	00 00 00	101000...	1000...	
11	00010001 00010001 00010001	11 11 11	0 100000...		
12	00100010 00100010 00100010	22 22 22	*00*000 100000...		
13	00110011 00110011 00110011	33 33 33	10...	1000000...	
14	01000100 01000100 01000100	44 44 44	*0*00 100000...		
15	01010101 01010101 01010101	55 55 55	100...		000 1000...
16	01100110 01100110 01100110	66 66 66	*0*000 100000...	0*0*0000 10000000...	
17	01110111 01110111 01110111	77 77 77	100010...		
18	10001000 10001000 10001000	88 88 88	00 100000...		
19	10011001 10011001 10011001	99 99 99	0 100000...	00 10000000...	
20	10101010 10101010 10101010	AA AA AA	00 100...		0 1000...
21	10111011 10111011 10111011	BB BB BB	00 101000...		
22	11001100 11001100 11001100	CC CC CC	0 10...	0000 10000000...	
23	11011101 11011101 11011101	DD DD DD	0 101000...		
24	11101110 11101110 11101110	EE EE EE	0 100010...		
25	11111111 11111111 11111111	FF FF FF	0 100...	000 100000...	
26	10010010 01001001 00100100	92 49 24		00 100000...	0 100...
27	01001001 00100100 10010010	49 24 92		0*0*00 100000...	*1*00 100...
28	00100100 10010010 01001001	24 92 49		100000...	00 100...
29	01101101 10110110 11011011	6D B6 DB		0 100...	
30	10110110 11011011 01101101	B6 DB 6D		100...	
31	11011011 01101101 10110110	DB 6D B6		00 100...	
32	(Random)	(Random)			

33	(Random)	(Random)			
34	(Random)	(Random)			
35	(Random)	(Random)			

Encoded bits shown in italics are what should be present in the ideal pattern, although due to the encoding the complementary bit is actually present at the start of the track.

Criticism

Some have criticized Gutmann's claim that intelligence agencies are likely to be able to read overwritten data.[1]

The delete function in most operating systems simply marks the space occupied by the file as reusable (removes the pointer to the file, without immediately removing any of its contents). At this point the file can be fairly easily recovered by numerous recovery applications. However, once the space is overwritten with other data, there is no known easy way to recover it. It cannot be done with software alone since the storage device only returns its current contents via its normal interface. Gutmann claims that intelligence agencies have sophisticated tools, among these magnetic force microscopes, that, together with image analysis, can detect the previous values of bits on the affected area of the media (for example hard disk).

This has not been proven one way or the other, and there is no published evidence as to intelligence agencies' current ability to recover files whose sectors have been overwritten, although published Government security procedures clearly consider an overwritten disk to still be sensitive.[2]

Companies specializing in recovery from damaged media (for example Ibas) cannot recover completely overwritten files. These companies specialize in the recovery of information from media that has been damaged by fire, water or otherwise. No private data recovery company claims that it can reconstruct completely overwritten data.

Gutmann himself has responded to some of these criticisms and also criticized how his algorithm has been abused in an epilogue to his original paper, in which he states:

> “In the time since this paper was published, some people have treated the 35-pass overwrite technique described in it more as a kind of voodoo incantation to banish evil spirits than the result of a technical analysis of drive encoding techniques. As a result, they advocate applying the voodoo to PRML and EPRML drives even though it will have no more effect than a simple scrubbing with random data. In fact performing the full 35-pass overwrite is pointless for any drive since it targets a blend of scenarios involving all types of (normally-used) encoding technology, which covers everything back to 30+-year-old MFM methods (if you don't understand that statement, re-read the paper). If you're using a drive which uses encoding technology X, you only need to perform the passes specific to X, and you never need to perform all 35 passes. For any modern PRML/EPRML drive, a few passes of random scrubbing is the best you can do. As the paper says, "A good scrubbing with random data will do about as well as can be expected". This was true in 1996, and is still true now.”

Bad sectors on the disk may be silently suppressed by the drive controller so they may be not be overwritten.

Software implementations

- The GNU Core Utilities shred program.
- E3 Security Kit The first complete and fully functional implementation of Gutmann for Windows.
- Eraser Free open-source software that uses the Gutmann method.
- Disk Utility Software provided with Mac OS X uses Gutmann on a per disk basis only.
- TuneUp Utilities Has the Gutmann method as an option.
- Darik's Boot and Nuke (DBAN) Another free open-source wipe utility that supports Gutmann.
- Window Washer by Webroot Software supports the Gutmann algorithm.

See also

- Data remanence
- Data recovery
- Computer forensics
- Shredding

External links

- Secure Deletion of Data from Magnetic and Solid-State Memory, Gutmann's original paper
- Can Intelligence Agencies Read Overwritten Data?, a refutation of Gutmann's claims.
- Recovering Unrecoverable Data, the need for drive-independent data recovery.
- A Guide to Understanding Data Remanence in Automated Information Systems

Notes

1. ^ Can Intelligence Agencies Read Overwritten Data? A response to Gutmann..
2. ^ Clearing and Declassifying Electronic Data Storage Devices.