Minimally Invasive Data Concealment in NTFS

Xiashen Gu and Hengming Zou
Shanghai Jiao Tong University
{guxiashen, zou}@sjtu.edu.cn

ABSTRACT

Data protection is becoming increasingly important as electronic data permeates our society. Numerous techniques have been proposed to deal with this issue, but most such techniques suffer the drawbacks of low efficiency in time or space, algorithmic complexity, pervasive modification or even outright replacement of existing file systems. This paper describes a minimally invasive data concealment scheme that works on NTFS. The proposed scheme makes a tiny modification to existing NTFS structure and combines light-weight random data scrambling to provide secure data storage with high efficiency and easy operation.

1. INTRODUCTION

With people's increasing reliance on electronic data for social, personal, and business activities, protection of such data has become critical. Many techniques have been proposed to secure data from disclosure or attack or both: separation of privileges, access control, encryption, steganographic and deniable file systems, etc. However, many of these methods suffer the drawbacks of low efficiency in time or space or both, algorithmic complexity, target exposure, pervasive modification or even outright replacement of existing file systems. For example, access control risks exposing the whole system once cracked. An encrypted file or disk volume is itself evidence of the existence of valuable data. Steganographic and deniable file systems often compromise the practicality of the file systems, which in turn, result in large increases in I/O, ineffective storage usage, and even the risk of data loss as hidden files may be overwritten.

This paper proposes a Minimally Invasive Data Concealment (MIDC) scheme that retains the benefits of a steganographic system while avoiding its pitfalls. Our basic idea is to manipulate file system's addressing ability[2] and scramble flat data with random block exchange. Our MIDC scheme can be superimposed on existing file systems and is highly secure and efficient in both time and space. Once data is concealed, it will be invisible to viruses and would-be adversaries that operate on the file level, and very difficult to be re-constructed by attackers who use raw disk scan.

MIDC is designed to meet three key requirements: protecting files/directories from unauthorized access and virus attacks; preventing data loss and corruption by not interfering with the normal function of the file system and OS; and minimizing space overheads and modification to the file systems. Compared to other data protection methods, our scheme has the following advantages: double protection with concealment and scrambling, works with existing file systems, invisible to viruses, faster than encryption, more efficient than steganographic[1] and deniable file systems[3], yet simple in algorithmic design and implementation.

Next we describe the scheme in detail.

2. APPROACH

MIDC is not a file system itself but an add-on module to existing file systems. It does not interfere with the function of the normal file systems or other parts of the OS. A user can manipulate data either through MIDC or though the native functions provided by the OS. Figure 1 illustrates the overview of our MIDC scheme.

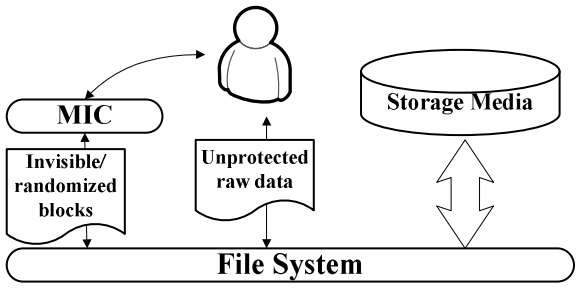

Figure 1: MIDC System Architecture

MIDC uses a two-step approach to protect a file/folder: First it manipulates file metadata to render the file/folder invisible to any OS and user applications. Second MIDC performs a light-weight random data scrambling on the raw data to render it impossible for raw disk readers to reconstruct the meaning of any data she or he might find. This light-weight scrambling differs from traditional encryption in that it only works on part, not all, of the content of a file.

Next we describe our two steps in detail.

2.1 Minimally Invasive Concealment

A file in a NTFS[5] volume usually has two components: Master File Table (MFT) Record containing a set of attributes and directory entry for fast data access. Each file in a NTFS volume has its unique record in MFT which contains the attributes of this file. The most important attribute is $DATA (0x80) which indicates the location of the real data in a file. Besides the MFT, NTFS also maintains an index of all the files on each volume to facilitate fast data access. With this index, NTFS does not need to read the MFT record to get directory information of a file.

In NTFS, a directory can be classified into two types: small or large directory. The former is just a list of directory entries but the latter is implemented as B+ trees. To conceal a file in a NTFS volume, we remove the MFT record of the to-be-concealed file from MFT and delete the relevant node which contains the file's information in the index. A system metadata $Bitmap should also be checked to ensure that the concealed data block won't be overwritten by other users.

For compatibility, NTFS also generates an MS-DOS filename in 8.3 format. In this case, the index of a NTFS volume contains two index entries that point to the same file record in the MFT with one being the NTFS filename and the other being the DOS filename. Thus, we also remove the redundant index entry containing the MS-DOS filename, if it exists, when executing the concealment.

The process to recover concealed files is just the reversal of the concealment: we first create an empty file with the same name of the concealed file, then we restore the file attributes in the MFT record and directory entry.

2.2 Light-weight Random Scrambling

The concealment thus performed will prevent most attackers/viruses from pirating/corrupting protected data through native read and write functions provided by the file system. However, a determined attacker can execute raw disk scan to reconstruct the concealed files by using various data carving techniques[4]. To deal with this problem, we perform lightweight random data scrambling to disturb the patterns of specific file types. This scrambling adds great difficulty to an attacker's effort in reconstructing a concealed file because the file's signature has been broken and (portions of) its data blocks are (pseudo) randomly transposed.

The algorithm is designed to be flexible so that the extent of scrambling can vary according to the sensitivity of the data and the security threat faced by the specific environment. If the threat is grave and physical disk itself can be lost or stolen, the scrambling can degrade into a full-blown encryption. However, if the physical disk is highly secured and raw disk scan is not likely, we can do away with data scrambling, thus improve concealment efficiency. Normally the scrambling is adjusted to achieve a balance between computation penalty and security objective.

2.3 HFL and MIDC Authentication

One important data structure for MIDC is the Hide File List (HFL), which is used by MIDC to locate and reconstruct concealed files. HFL includes such information as the full data path before concealment, the relevant MFT record, and the seeds for generating random scrambling sequence. The total size of this information for one concealed file is at most 2Kb with the MFT record constituting the main part. Thus, the space usage by MIDC is small to negligible.

Needless to say, the HFL must be protected or data can be compromised. Our solution is to store the HFL on a safe removable storage media such as an encrypted flash disk. Only the legal owner of the concealed files with compatible flash disk can pass MIDC's authentication and manipulate concealed files through the GUI and API provided by MIDC. Furthermore, multiple users on a single system have separate access rights since a HFL indicates the only way to manipulate concealed data on the storage media. This authentication also boosts security of MIDC since an attacker has little chance to get a copy of the unique HFL on the flash disk. Figure 2 depicts the MIDC authentication.

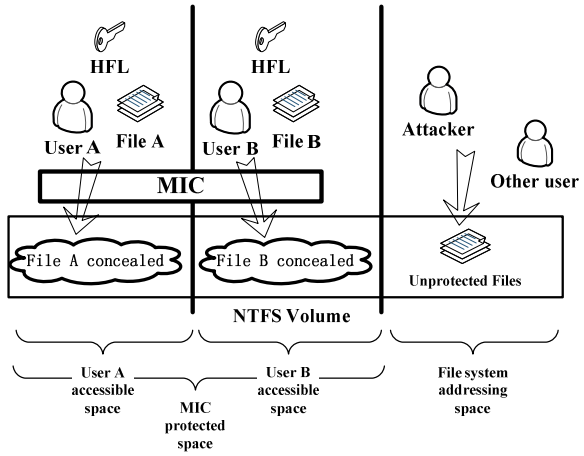

Figure 2: MIDC Authentication

Note that storing HFL on a removable flash disk has another advantage. An attacker who has obtained such a disk will have a hard time to decide which computer and which user the HFL corresponds to, if he is lucky enough to have guessed that the content of the flash disk is a HFL.

3. LIMITATION

The clusters marked as occupied in the $Bitmap by MIDC are not related to any file and thus may be reclaimed by some disk scan utilities such as ScanDisk if it is intentionally reconfigured to retest bad and orphan clusters. However, this approach is officially not recommended due to potential risks. If users refrain from running such utilities or notify MIDC before-hand so that precaution can be taken, then there would be no data loss. In the worst case, data get lost but no disclosure or leaky is possible.

Since HFL provides the only way to manipulate concealed data, loss of HFL will cause loss of protected data. But this is a common problem faced by all data protection schemes and can be mitigated by backing up the HFL.

4. CONCLUSION

This paper introduces a minimally invasive data concealment scheme for NTFS. As a flexible add-on to existing file systems, MIDC offers files/folders concealment with little overhead and minimum modification to the underlying storage structure. By combining concealment with random data scrambling, MIDC achieves excellent data protection.

5. REFERENCES

[1] R. Anderson. The Steganographic File System. In *Proc. of the Information Hiding Workshop, April*, 1998.

[2] Bairavasundaram. Analyzing the effects of disk-pointer corruption. In *DSN With FTCS and DCC*, 2008.

[3] A. Czeskis. Defeating Encrypted and Deniable File Systems: TrueCrypt v5. 1a and the Case of the Tattling OS and Applications. *USENIX HotSec 08'*, 2008.

[4] S.L. Garfinkel. Carving contiguous and fragmented files with fast object validation. *Digital investigation*, 1991.

[5] NTFS.com. NTFS.com. http://www.ntfs.com.