Improving Disk Sector Integrity Using 3-dimension Hashing Scheme

Zoe L. Jiang, Lucas C.K. Hui, K.P. Chow, S.M. Yiu and Pierre K.Y. Lai
Department of Computer Science
The University of Hong Kong, Hong Kong
ljiang, hui, chow, smyiu,kylai@cs.hku.hk

Abstract

To keep the evidence that a stored hard disk does not modify its content, the intuitive scheme is to calculate a hash value of the data in all the sectors in a specific order. However, since one or more sectors, with some probability, may become a bad sector after some time, this scheme fails to prove the integrity of all other sectors that are still good. In this paper, we suggest a scheme which calculates three hash values for each sector, in a three dimensional manner, such that the integrity proof of a sector depends only on the sectors in any one of the three dimensions, in stead of all sectors in the hard disk. Our analysis shows that this new scheme can greatly reduce the effect of bad sector formation in proving the integrity of the disk sectors.

1. Introduction

With the rapid development of electronic commerce and Internet technology, network and computer crimes become more and more common. There is a great need to collect and analyze data during the transactions. In this paper, we just focus on a particular problem in data investigation: Given a hard disk obtained on a certain date for digital evidence collection purpose, after some time, say one month, we want to prove the content of this hard disk is the same as before.

The straight-forward scheme is to calculate a hash value of all data in all sectors (the smallest individually-addressable unit of information stored on a hard disk) in a specific order. Then the hash value is digitally signed and stored in somewhere for future use. After some time, the disk will be taken out to re-calculate the hash value again. If the newly calculated value is the same as the pre-stored hash value, it means that the hard disk content is not modified. Otherwise, if the newly calculated value is different from the pre-stored one, we know that at least one sector of the disk is modified, but we cannot prove any other facts. Since one (or more) sector in a hard disk tends to become a bad sector, with some probability, this scheme will fail to prove that all other sectors are still good. In other words, we can say that it depends on the all other sectors in the hard disk to check whether a sector is modified or not.

In our proposed scheme, rather than to generate a single hash value for the entire hard disk, three kinds of hash values are generated in three geometric dimensions, on which the probability to check the integrity of a certain sector depends. As a result, even if some disk sectors become bad, we can still prove that a lot of other unaffected sectors are not modified with higher probability than the above scheme.

1.1. Related Works

Given rise to the need for computer forensics, there are several existing research and tools [1] [2], among which ENCASE [1] is most popular in computer forensic investigation technology. It provides intuitive GUI, superior analytics, enhanced email/Internet support and a powerful scripting engine. DESK [2] (Digital Evidence Search Kit) is another tool which more focuses on Chinese language encoding. Both systems, together with many others, use the straight-forward hash value checking for a hard disk. Thus also face the challenge that even a one-bit change to a data item (such as a known file or a disk) [3], the hash value for that item will be radically altered. Kornblum [4] described a scheme, called Context Triggered Piecewise Hash (CTPH) to identify modified versions of known files even if data has been inserted, modified, or deleted in the new files. Although this work is originally designed for files, it can also be adopted to a hard disk (for a sector becoming bad can be considered as a part of a file being modified). However, the CTPH scheme required an $O(n \log n)$ running time where n is the data size, which is a very high overhead if when it is applied to a hard disk with huge size such as 120G bytes.

1.2. Outline

We now briefly describe the organization of this paper. Section 2 introduces preliminaries. Section 3 elaborates the details of our proposed schemes. Section 4 contains the

probability analysis and comparison. Section 5 concludes the paper.

2. Preliminaries

This section introduces hard disk geometry structure and operation, and cryptographic hashing algorithms.

2.1. Geometry Structure and Operation of Hard Disk

Every hard disk contains one or more flat disks that are used to actually hold the data in the drive. These disks are called *platters*, each of which use two *heads* to record and read data, one for the top of the platter and one for the bottom. Standard consumer hard disks, the type probably in your PC right now, usually have between one and five platters in them. Each platter is broken into *tracks*–tens of thousands of them–which are tightly-packed concentric circles. These are similar in structure to the annual rings of a tree. Data is accessed by moving the heads from the inner to the outer part of the disk. A *cylinder* is basically the set of all tracks that all the heads are currently located at. Since a track holds too much information to be suitable as the smallest unit of storage on a disk, each one is further broken down into *sectors*. Today's hard disks can have thousands of sectors in a single track. A sector is normally the smallest individually-addressable unit of information stored on a hard disk, and normally holds 512 bytes of information. [5]

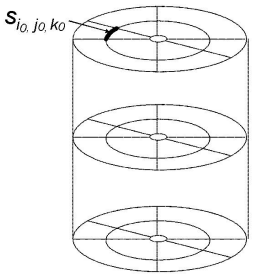

Figure 1. Hard disk structure

For the ease of discussion, in this paper, we will represent a disk sector by two methods. The first is just to treat all sectors as an ordered list. (See description in scheme 1.) The second method is to treat the hard disk as an entry in a three-dimension object. So we address by three indices. (See description in scheme 2.)

In the 3-dimension way of addressing an individual disk sector, s_{i_0,j_0,k_0}, is traditionally done by referring to CHS, which uses an ordered triple giving the Cylinder number, the Head number and the Sector number. See Figure 1 for reference. Although due to the 8.4 GB limit of the Int 13h interface, modern drives are no longer specified in terms of classical CHS mode, but rather in terms of their total number of sectors and addressed using LBA (Logical Block Addressing) in logical level, in physical level most hard disks actually organize their sectors using the cylinder, head and sector structure. So if we can access to the integrated disk controller, which is responsible for automatically translating LBA into physical geometry, it is still possible to match the ordered triple dimensions to the physical hard disk characteristics. [5] So all hash values stored will have a physical meaning (such as a hash value of all sectors in one track of the physical hard disk). This physical meaning of hash value, may provide more chances for future research discoveries.

2.2. Cryptographic Hashing Algorithm

In computer forensic area, examiners often need to understand and analyze a large number of data which seem arbitrary to them. Cryptographic hash functions are often used by forensic examiners for data integrity check. [9] In 2006, The National Software Reference Library (NSRL) is provided to identify known files by comparing several kinds of hash algorithms, based on two fundamental properties - collision resistance and being a one-way function. The first, being collision resistant, means that two different messages should not hash to the same value. The second property that good hash algorithms have is that they are pre-image resistant, i.e., it is computationally infeasible for a message to be constructed that matches a given hash. As a result, both the MD5 and SHA-1 passed the examination as the cryptographic hash algorithms. The National Institute of Standards and Technology (NIST) also plans to add additional file signatures generated by other hash algorithms in the future, including those identified in FIPS PUB 180-2 (SHA-256, SHA-384, SHA-512). [7]

3. Proposed Scheme

For ease of discussion, we first describe the most straight-forward scheme by using the hash value on a whole hard disk (denoted as Scheme 1), for later comparison. Then our proposed scheme (denoted as Scheme 2) by using the hash values on three dimensions is described in details.

3.1. Scheme 1: Hash on A Whole Hard Disk

Let a hard disk consists of sectors $\{ s_1, s_2, \cdots, s_N \}$ where N is the total number of the sectors in the whole hard disk. Assume each sector has an independent probability of being a bad sector after some time: p ($0 < p < 1$ and obviously p is very small). We keep the evidence that a stored

hard disk does not modify its content, as the hash value

$$v = Hash(s_1 \| s_2 \| \cdots \| s_N)$$

of all sectors in the hard disk under a one-way hash function $Hash(\cdot)$ mentioned in subsection 2.2, where " $\|$ " denotes concatenation. Then any one sector that becomes a bad sector will make the hash value v changed. So we cannot check the disk integrity. That is to say, the rule to check whether a certain sector s_{i_0} is modified or not depends on whether v is changed or not, or the fact that all other sectors in the hard disk become bad or not. Only when no any other sector becomes bad, we can declare that s_{i_0} is not modified.

Let P_1 be the probability to prove the sector s_{i_0} is not modified. P_1 is equivalent to the probability that the hash value v is not changed after some time, and sequentially to the probability that all other $N-1$ sectors do not become bad sectors, i.e. $(1-p)^{N-1}$. So,

$$P_1 = (1-p)^{N-1}.$$

Obviously, although $(1-p)$ is very close to 1 since p is a small value, $(1-p)^{N-1}$ can be greatly decreased when N is growing more and more. So this scheme may fail to judge the disk integrity with a high probability of $1-P_1$ which is $1-(1-p)^{N-1}$. The concrete analysis will be studied in Section 4.

3.2. Scheme 2: Hash on Three Dimensions

Suppose we have the same hard disk as the above scheme, but with X cylinders, Y heads and Z sectors per track. The total number of the sectors in the hard disk N equals to XYZ. Then let the sectors in the hard disk represented by another form using an ordered three-dimension array (i,j,k), i.e. a sector is denoted by

$$s_{i,j,k} \ \ where \ \ (1 \le i \le X, 1 \le j \le Y, 1 \le k \le Z).$$

Also assume each sector has an independent probability of being a bad sector after some time: $p(0 < p < 1)$ and usually p is very small. We calculate the following **three kinds of hash values**:

The first kind of hash value, denoted as a **cylinder-hash-value**, is the hash value of all sectors with the same and fixed head j and sector k, and with cylinders from 1 to X. Therefore

$$v_{c_{j,k}} = Hash(s_{1,j,k} \| s_{2,j,k} \| \cdots \| s_{X,j,k}).$$

For example, in Figure 2, area A_c includes all the sectors from s_{1,j_0,k_0} to s_{X,j_0,k_0} for computing hash value $v_{c_{j_0,k_0}}$.

Similarly, the second kind of hash value, denoted as a **head-hash-value**, is the hash value of all sectors with the same and fixed cylinder i and sector k, and with heads from 1 to Y. Therefore

$$v_{h_{i,k}} = Hash(s_{i,1,k} \| s_{i,2,k} \| \cdots \| s_{i,Y,k}).$$

See area A_h in Figure 3 for reference.

The third kind of hash value, denoted as a **sector-hash-value**, is the hash value of all sectors with the same and fixed cylinder i and head j, and with sectors from 1 to Z. Therefore

$$v_{s_{i,j}} = Hash(s_{i,j,1} \| s_{i,j,2} \| \cdots \| s_{i,j,Z}).$$

See area A_s in Figure 4 for reference.

It's easy to compute that there are YZ cylinder-hash-values, XZ head-hash-values, XY sector-hash-values, and totally $YZ+XZ+XY$ hash values for the whole hard disk. From a logical point of view, our designed scheme to store more hash values is equivalent to designing a lot of chains of sectors, and for each chain of sectors we will store the hash value of all the bits in this chain sectors. Hopefully the calculation of many hash values will increase the chance of a sector being able to prove its data integrity after some time, even when some other sectors are becoming bad sectors. We can also say that using an ordered triple to represent a sector is just a design choice to achieve this goal.

Assume the sector s_{i_0,j_0,k_0} is not a bad sector after some time. Let p_c, p_h and p_s denote the probabilities that $v_{c_{j,k}}$, $v_{h_{i,k}}$ and $v_{s_{i,j}}$ are unchanged respectively. In other words, each of them represents the probability that, besides s_{i_0,j_0,k_0}, all other sectors in A_c or A_s or A_h do not become bad. Then we can obtain the followings:

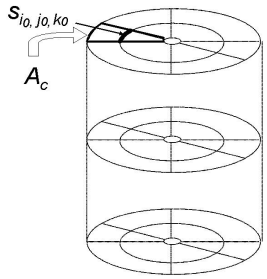

Figure 2. Area A_c in hard disk

$$\begin{aligned} p_c &= \prod_{i=1,i\neq i_0}^{X} (prob(s_{i,j_0,k_0} \text{ is not a bad sector})) \\ &= (1-p)^{X-1}, \end{aligned}$$

$$\begin{aligned} p_h &= \prod_{j=1,j\neq j_0}^{Y} (prob(s_{i_0,j,k_0} \text{ is not a bad sector})) \\ &= (1-p)^{Y-1}, \end{aligned}$$

$$\begin{aligned} p_s &= \prod_{k=1,k\neq k_0}^{Z} (prob(s_{i_0,j_0,k} \text{ is not a bad sector})) \\ &= (1-p)^{Z-1}. \end{aligned}$$

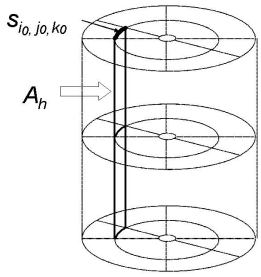

Figure 3. Area A_h in hard disk

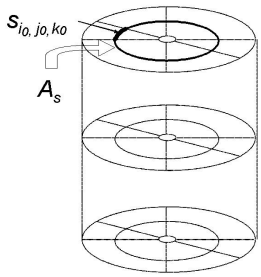

Figure 4. Area A_s in hard disk

Then we can deduce the following results:

- $p_c \equiv$ probability of all sectors in A_c are still good $\equiv$ probability of $v_{c_{j,k}}$ is not modified,
- $1 - p_c \equiv$ probability of at least one sector in A_c becomes bad $\equiv$ probability of $v_{c_{j,k}}$ is modified,
- $p_h \equiv$ probability of all sectors in A_h are still good $\equiv$ probability of $v_{h_{i,k}}$ is not modified,
- $1 - p_h \equiv$ probability of at least one sector in A_h becomes bad $\equiv$ probability of $v_{h_{i,k}}$ is modified,
- $p_s \equiv$ probability of all sectors in A_s are still good $\equiv$ probability of $v_{s_{i,j}}$ is not modified,
- $1 - p_s \equiv$ probability of at least one sector in A_s becomes bad $\equiv$ probability of $v_{s_{i,j}}$ is modified,
- $(1 - p_c)(1 - p_h)(1 - p_s) \equiv$ probability of $v_{c_{j,k}}$, $v_{h_{i,k}}$ and $v_{s_{i,j}}$ are modified,
- $1 - (1 - p_c)(1 - p_h)(1 - p_s) \equiv$ probability of at least one of $v_{c_{j,k}}$, $v_{h_{i,k}}$ and $v_{s_{i,j}}$ is not modified.

Given the sector s_{i_0,j_0,k_0}, as long as there are no bad sectors in A_c, $v_{c_{j,k}}$ will keep unchanged, we can prove that the sector s_{i_0,j_0,k_0} is not modified. Similarly, as long as there are no bad sectors in A_h or in A_s, $v_{h_{i,k}}$ or $v_{s_{i,j}}$ will keep unchanged. Then we can prove the sector s_{i_0,j_0,k_0} is not modified as well.

Now define P_2 as the probability of proving the sector is not modified, provided that there is a small probability p that any sector will become a bad sector. So P_2 is exactly the probability that at least one of $v_{c_{j,k}}$, $v_{h_{i,k}}$ and $v_{s_{i,j}}$ is not becoming a bad sector, i.e.

$$\begin{aligned} P_2 &= 1 - (1 - p_c)(1 - p_h)(1 - p_s) \\ &= 1 - [1 - (1 - p)^{X-1}][1 - (1 - p)^{Y-1}][1 - (1 - p)^{Z-1}] \end{aligned}$$

Unlike the first scheme where the probability to prove sector s_{i_0,j_0,k_0} does not change its content depends on all other sectors in the whole hard disk, we compute the probability depending on all other sectors only located in any of the three dimensions, as the figures 2, 3, and 4 show, the three geometry parts.

4. Probability Analysis and Comparison

For scheme 1, the probability to prove that the sector s_{i_0} is not modified is equivalent to the probability that all other $N - 1$ sectors in the whole hard disk do not become bad, i.e. $P_1 = (1 - p)^{N-1}$. However, in our proposed scheme 2, the probability to prove the integrity of sector s_{i_0,j_0,k_0}, i.e. $P_2 = 1-(1-(1-p)^{X-1})(1-(1-p)^{Y-1})(1-(1-p)^{Z-1})$ is equivalent to the probability that all other sectors in at least one of the three areas A_c, A_h and A_s do not become bad. Since the number of sectors that P_2 depends on in scheme 2 is far smaller than that P_1 depends on, it's obvious that P_2 should be higher than P_1. That is, using scheme 2, we can keep the evidence that a certain sector in a hard disk does not modify its content with a higher probability than using scheme 1.

To provide a concrete evaluation of the proof probability and compare the above two schemes, we refer to the hard disk parameters from [5], where the total number of cylinders X, heads Y, and sectors Z range from $10,000$ to $99,999$, from 1 to 10, and from $1,000$ to $9,999$, respectively. Since we could not find a reference pointing out the independent probability for a certain sector being a bad one, we arbitrarily select several values based on intuition. Recall that p is an independent probability for one sector being a bad sector after some time, $1 - P_1$ and $1 - P_2$ are the probabilities for scheme 1 and 2 that fail to judge the disk integrity, respectively.

In Table 1, we choose to compare the two schemes when the capacity of a hard disk is fixed with the values of p decreasing. Let $X = 16,000$, $Y = 6$, $Z = 2,600$, and $N = XYZ = 2.496e8$. Since each sector normally holds 512 bytes of data, the capacity is $512N \approx 120G$.

Table 1. Comparison with different p (capacity $\approx 120G$)

p	$1 - P_1$	$1 - P_2$
$1.0000e-05$	$1.0000e-00$	$1.8965e-07$
$1.0000e-10$	$2.4651e-02$	$2.0791e-22$
$1.0000e-12$	$2.4956e-04$	$2.0789e-28$
$1.0000e-15$	$2.4940e-07$	$2.0741e-37$

From Table 1, we can see that $1 - P_2$ is several orders of magnitude smaller than $1 - P_1$. In other words, scheme 2 is always better than scheme 1, with the smaller probability $(1 - P_2)$ that fails to prove that a sector is modified or not. Also, as p decreases, we can achieve even better results. In Table 2, we compare the two schemes when p is fixed with the capacity of hard disk increasing. Let $p = 10e - 10$. The relevant parameters are listed in Table 3. Table 2 shows that scheme 2 is always better with lower probability of failure.

Table 2. Comparison with different hard disk capacity $p = 10e - 10$

Capacity	$1 - P_1$	$1 - P_2$
$60G$	$1.1454e-02$	$8.6357e-23$
$120G$	$2.4651e-02$	$2.0791e-22$
$180G$	$3.5360e-02$	$2.9988e-22$

Table 3. Hard disk characteristics for table 2 (sector size $= 512$ bytes)

Capacity	X	Y	Z	N
$60G$	12000	4	2400	$1.152e8$
$120G$	16000	6	2600	$2.496e8$
$180G$	20000	6	3000	$3.6e8$

5. Conclusion

In this paper, we have introduced a three-dimension hashing scheme to keep the evidence that the contents of a stored hard disk have not been modified. We showed that our scheme is efficient and much better than the commonly used straight-forward one. It takes only one complete scan of disk sectors to compute all cylinder-hash-values, head-hash-values, and sector-hash-values. The amount of computation is only three times more computational resources than the straight-forward scheme. This is much better than the CTPH scheme [4].

Further research directions include the followings. It is a practice to digitally sign the hash value of a hard disk. There may be a lot of hash values to be signed in our scheme. One may try to apply the Merkle hash tree [6], which was previously applied to reduce the digital signature resource for multimedia data [8], to reduce the computational resources for digitally signing the hash values. The three-dimension scheme is to match the physical characteristics of a hard disk. In fact, this is not necessary. We can try to extend the scheme to an n-dimension scheme in order to minimize the $1 - P_2$ value.

6. Acknowledgment

The work described in this paper was partially supported by two grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. HKU 7136/04E and HKU 7132/06E).

References

[1] Encase. *http://www.guidancesoftware.com/products /ef_index.asp*, 2007.

[2] K. P. Chow, C. F. Chong, K. Y. Lai, L. C. K. Hui, K. H. Pun, W. W. Tsang, and H. W. Chan. Digital evidence search kit. In *Proceedings of the First International Workshop on Systematic Approaches to Digital Forensic Engineering (SADFE'05)*, pages 187–194, 2005.

[3] J. M. Foster and V. T. Liu. Catch me, if you can. *http://www.blackhat.com/presentations/bh-usa-05/bh-us-05-foster-liu-update.pdf*, 2005.

[4] J. Kornblum. Identifying almost identical files using context triggered piecewise hashing. In *Proceedings of the 6th Annual Digital Forensic Research Workshop (DFRWS '06)*, volume 3, pages 91–97, 2006.

[5] C. M. Kozierok. *The PC Guide*. http://www.pcguide.com/ref/hdd/index.htm, 2001.

[6] R. C. Merkle. A certified digital signature. In *Proceedings on Advances in cryptology (Crypto '89)*, pages 218–238, 1989.

[7] M. Steve. Unique file identification in the national software reference library. *http://www.nsrl.nist.gov/documents/analysis/draft-060530.pdf*, 2006.

[8] M. Q. Wang, S. M. Yiu, L. C. K. Hui, C. F. Chong, K. P. Chow, W. W. Tsang, H. W. Chan, and K. H. Pun. A hybrid approach for authenticating mpeg-2 streaming data. In *Proceedings of Multimedia Content Analysis and Mining (MCAM '07)*, pages 203–212, 2007.

[9] D. White. Nist national software reference library. national institute of standards and technology. *http://www.nsrl.nist.gov/*, 2005.