Integrity Assurance Technique using Determinant Approach

J. A. Ghaeb
Department of Electrical and Computer Engineering,
The Hashemite University,
Zarqa 13115, Jordan

Abstract

Data integrity is an important aspect of storage security and reliability which are prerequisite for most computer systems and network applications. This paper proposes a new technique for improving the detection of data integrity violations. The method is based on check determinant approach. Simulation results show that the new method outperforms the traditional methods such as Hamming code method and RAID method.

Index Terms. Data integrity, Error detection, Security, Storage integrity.

1. Introduction

The elemental concern for data storing is maintaining the data integrity. Data integrity is insuring that the data retrieved is the same as the data stored. These principles are also implemented for data transfer through the space. There are a variety of security measures that can be taken to detect and remedy data security infringements. Depending on their objectives, integrity assurance mechanisms can be classified in several levels: those that take predictions steps to prevent specific types of data corruption; those that carry out integrity checks on data to measure the data assurance and those that are able to detect and correct the data violation.

Some integrity systems ensure the data integrity by applying the avoiding mechanism which performs deterring so as to avoid the possibility of integrity violation. Anti-virus protection is most possibly the best-known form of security and no layered security strategy is complete without it [1]. However, virus protection software works primarily by looking for known virus signatures, therefore virus checkers need to be constantly updated as new variations appear**.** Read only storages are used to guarantee data integrity and avoid violations due to user errors [2, 3]. Journaling file system can recover the data from a system crash by constantly writing the data to directories and bitmaps in a serial log on a disk, before the original disk log is updated [4]. In the case of a system failure, the data on the disk can be restored to its pre-crashed configuration by examining the disk log. Journaling file system can recover data from a system crash but can not protect them from hardware errors or malicious intrusions. Another form of avoiding mechanism is the hashing function [5, 6]. Many algorithms are written to develop different hash functions which convert the data into hash codes. Cryptographic file systems convert the data into coded file system [7]. Cryptographic file systems are developed to ensure the data confidentiality, with some degree of data assurance. Although the data can be modified with this method, it is not possible to do it in a predictable manner as the encryption key is not known. Integrity violation due to hardware errors can not be prevented by using encryption.

Most of the data integrity techniques carry out detection of data corruption depending on the redundant information. These techniques append excess information about the data to use them in detecting the possibility of violation. A checksum is an error detection mechanism that is created by summing up all the bytes or words in a data word to create a checksum value, which is appended to the stored or transmitted data word [8, 9]. The checksum of the retrieved data word is recomputed and compared with the received checksum value. If the computed and received checksum are identical, it is unlikely that the data word suffered an error during transmission or storing. However, there are many types of error which can not be detected by this method. The first one is when the data is reordered. Second type is when zero values are inserted in the data or deleted from it. The third type of errors occurs if multiple errors amount to zero.

Hamming codes methods can be used as an error detection mechanism [10]. It depends on parity bit. A code word is generated by adding the original data bits and the check bits. The check bits are obtained by XORing the weights of bits that have a level one in the original data. The new check bits of the retrieved code word are recomputed and XORed with the received check bits. If the result is zero, the data has no corruption during the transmission or storing.
A certain degree of integrity assurance can be provided by the Redundancy Array of Independent Disk method (RAID) [11]. It is a set of physical disk drives operate independently and examined by the operating system as a single logical drive. The RAID method consists of different levels and most of them are designed to operate in fail-stop drives. Raid-1 uses the mirroring technique to detect or recover the corrupted data [11]. In Raid-3, a single redundant disk is added to the original disk drives. This simple parity bit is used to correct the disk failure in the same level. RAID-6 adds two extra parity disk drives to the original drives of data [12]. The parities of these two disks are determined in different ways to provide extra bit availability for data check. The disadvantage of RAID method is that it can only operate on binary of data, which is inefficient in terms of storage space. Also it is designed to operate only in fail-stop disk.

2. Traditional Data Integrity Methods

Most of the integrity assurance mechanisms generate and store redundant information to be used for integrity checking. In sub-sections below, the common assurance techniques are discussed in details.

2.1. Hamming Code Method

Hamming code method is an error detection mechanism, at which error check bits are computed and inserted to the original data bits to form a Hamming code word. These error check bits help in detecting data integrity violation [10, 13 and 14]. Also Hamming code method can be used as a forward error correction to correct a single bit of error per unit data.

2.1.1. Power of Two (POT) Method. In this section a Hamming rule called (POT) is discussed. For a practical application, it is necessary to design a suitable code word having a sufficient ratio of parity bits to original data bits and a standard character length. In Fig.1, the code word size is assumed n-bits. The sequence of the non-power of 2 designates the original data bits (d) and the sequence of power of 2 defines the parity bits (p). Before transmitting the data through the communication system, the binary representation of their bits setting to one are XORed together to generate the parity bits.

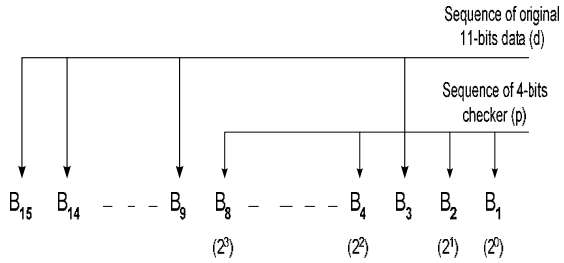

Figure.1 Sequence of (d) and (p) in the code word, n=15.

The computed parity bits are inserted into the original data bits to be sent along with the original data. To determine the data integrity assurance, the parity bits are recomputed at the receiving end and they are XORed with the received parity bits. The zero result indicates that there is no data violation during the transmission process.
POT method has many shortages. For example, some data errors such as malicious intrusions are difficult to detect by this method. If the original data bits are reordered or the parity bits are modified in the way to give zero result for XORing of appended and recomputed parity bits, the POT method can not detect the data violation. Also the word for POT method is represented in binary form and the parity bits take counted size in the code word. For instant, each 11-bit data needs extra 4-bits for the parity bits. This will increase the frame length of the transmitted data and increase the data transmission cost. The complete deficiency of POT is the so called nil weight of level one. On the other hand, if the original data is zero or has one weight of level one, there is no XORing result to produce no parity bits and the POT will fail.
To examine the accuracy of the POT method many series of data stream are generated randomly: starts with data size 10,000 to data size 105,000. The data error rate is taken to be 0.05. Table 1 shows the number of errors that can be detected by POT versus the actual one for various number of data size. It can be seen that the number of errors that were detected by POT does not match the exact number of errors. It is clear that this deficiency is due to the weaknesses mentioned earlier about POT.

Table 1 POT detection for data-error rate=0.05

No. of data	*Actual No. of errors*	*No. of errors detected by POT*	*No. of defects*
10000	493	477	16
14000	647	626	21
21000	999	974	25
33000	1583	1545	38
45000	2198	2135	63
57000	2826	2765	61
69000	3342	3264	78
81000	3913	3840	73
93000	4432	4325	107
105000	5161	5025	136

2.2. Redundancy Array Independent Disk (RAID) Method

For practical storage devices, it is preferred to use multiple-parallel components in the form of arrays or disks. By using the multiple disks, a redundancy can be added to the data to improve the data integrity of the storage devices. The standard scheme that is used for this purpose is the so called redundancy array independent disk method (RAID) [11, 15]. Many schemes are developed in RAID method (RAID0 to RAID6).

The data integrity assurance is achieved in RAID1 scheme by using the duplicating technique of data before processing them. The RAID1 method maintains two or more copies of the original data. At retrieval stage, the actual data is compared with the duplicated copy to ensure the integrity. Many weaknesses are accompanied with this technique. If the original copy and the duplicated copy of data have the same errors, the RAID1 technique can not detect the damage Moreover, if a malicious person inserts or drops the same value for the original and duplicated coppices, RAID1 can not detect the integrity violation.

Parity check is used in RAID3, RAID4 and RAID5 to ensure the integrity of the data array [12, 16]. In RAID3 scheme, a single redundant disk is added to the original disk of data bits. The redundant disk contains the parity bits of each block in the original disks, and it is used to detect if there is any error in the data bits. At retrieval stage, the series of the data of each block are XORed with their corresponding parity bit. It is expected a zero result, if there is no data violation. This method is incapable of many data corruptions. If the practical data stream and a parity bit are reordered, the RAID3 will not detect this type of data corruption. Also if any extra error bit is added in a way to keep the same parity bit level (“0” for 1-XOR result and “1” for 0-XOR result), RAID3 can not detect this corruption.

Simulation of a large number of data was performed to test the performance of the RAID technique. Many series of data streams are generated randomly starting from data size 10,000 to data size 90,800. The data error is taken to be 0.05 and the results are shown in Table 2. The deficiency of this method is clear and the number of errors detected by this method is not the exact number of errors. This deficiency occurs due to the weaknesses mentioned earlier.

Table 2 RAID detection for data-error rate=0.05

No. of data	*Actual No. of errors*	*No. of errors detected by RAID*	*No. of defects*
10000	495	484	11
14000	707	669	38
20400	983	940	43
32400	1613	1538	75
44400	2166	2151	15
56400	2868	2739	129
60400	3022	2885	137
66800	3214	3184	30
78800	3792	3668	124
90800	4395	4333	62

3. Determinant Technique

For some data corruption, the traditional techniques discussed earlier are inefficient and can not detect all the data corruption. Therefore, it is necessary to try a technique that is able to solve all the problems associated with the techniques mentioned earlier such as POT and RAID. In this paper we developed a new technique called Clockwise Array Rotation (CAR). This proposed technique is based on the determinate factor in measuring the data integrity assurance and involves the following steps:

Before transmitting the series of data, it is divided into N-matrices, where N is given by:

$$N = \frac{total\ \ number\ of\ \ data}{d \times d} \qquad (1)$$

where (d×d) is the number of elements per a matrix. The determinant of each matrix is computed directly from the data matrix and appended with the data. At retrieving stage, it is compared with the determinant of the retrieved data to measure the integrity assurance.

But it is observed that there are many defects with this method. The determinant is zero if any row is proportional to another row; the same is true for columns. Also the determinant does not change if some rows or some columns are interchanged. In addition, the determinant is zero if any single row or column has zero values only. Therefore, in this paper an improvement is added to eliminate these defects. For each matrix of data, a clockwise-rotated matrix (*Ac*) is generated. Then a clockwise rotation for the two halves of the matrix *Ac* is made to formalize the original data matrix into a new form. The determinants of both; original and clockwise-rotated matrices are computed and appended with each matrix before transmission or storing the data. At the retrieving or receiving stage, both determinants are recomputed again in a similar way and compared with the attached values. If there is no difference, the data of that particular matrix has no errors during the transmission or storing. The power of the CAR technique comes from its ability to work out all the deficiencies accompanied with other methods, as we are going to see in the simulated results.

4. Results and discussions

The performances of the proposed technique; CAR together with both methods: POT and RAID are investigated for different data runs. Figure.2 shows the detection accuracies of the CAR and the two traditional methods; POT and RAID, for data size 45,000. The error order in the original data is generated randomly and counted at different rates. At very low error rate, the detection accuracy of RAID is near to 100%. This is because the error rate is still small and there is no an opportunity to have its deficiencies. As the error rate is increased the RAID detection accuracy is decreased, due to its failings mentioned earlier in this paper. Also, due to the deficiencies referred to the POT, its detection accuracy is less than 100% and oscillates between 97% and 98%.

The detecting factor that is used in the CAR technique has worked out all the defects accompanied the traditional methods. Therefore, the CAR detection accuracy sticks at 100% for all error rates. The power of the CAR appears clearly as it is giving by the continuous 100% accuracy, but it is less than 100% for POT and RAID.

Figure.3 shows the detection accuracies for CAR, POT and RAID at different sizes of data. The error order is generated randomly at 0.05 rate. For POT and RAID, as the data size is increased their accuracies are still less than 100%. The CAR accuracy is constant and kept out at 100%. This shows the power of CAR technique compared to the other traditional methods.

Due to the random insertion of data errors in the simulation, it is normal that the POT and RAID deficiencies are oscillating and depend on the random insertion of the errors in that data run. Therefore, the accuracy for some larger data size is smaller than that of smaller data size as shown in Fig.3.

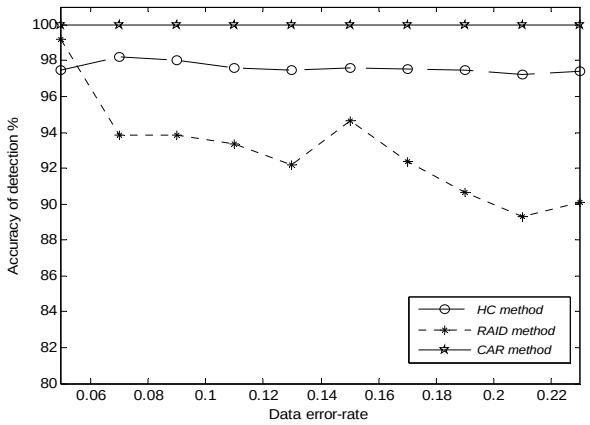

Figure.2 Detection accuracy of CAR, POT and RAID for a data size 45,000.

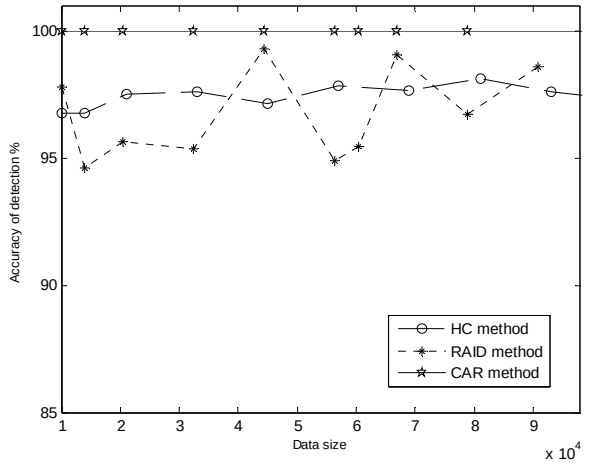

Figure.3 Detection accuracy of CAR, POT and RAID for various data size and error rate=0.05.

5. Conclusions

The data could be changed in unauthorized manner during the storing or transit processes. To assure the data integrity it is necessary to detect any error inserted

to the data, irrespective of its type and how it is interjected in. In this paper, it has established a high performance integrity technique through the solving of all the problems associated with the traditional methods. Before data transit, it divided into N-matrices. For each matrix, a clockwise rotated matrix is generated. Two determinant factors are computed; for original and clockwise rotated matrices and then both are appended to the data for integrity check. The new approach invades all the weaknesses that go along with other data integrity methods. The simulation results show that the new technique worked out all the problems in data integrity assurance methods.

6. References

[1] J. L. Aron, M. O'leary, A. R. Gove, S. Azadegan and M. C. Schneider," The Benefits of a Notification Process in Addressing the Worsening Computer Virus Problems: Results of a Suevey and a Simulation Model." Computers & Security, Elsevier, Vol.21, No.2, 2002.

[2] K. Fu, M. F. Kaashoek and D. Mazieres , " Fast and Secure Distributed Read-only File System." Proceedings of the fourth symposium on operating systems desine and implementation, san Diego, CA, Oct.2000.

[3] S. Quinlan and S. Dorward, "A New Approach to Archival Storage." ." Proceedings of first USENIX conference on file and storage technology, Monterey, CA, jan.2002.

[4] B. Carrier," An Investigator's Guide to File System Internals." First conference in computer security, incident handling and response, Hawaii, United States, June 2002.

[5] I. Damgrad I " A Design Principle for Hash Functions." G Brassard, Crypto, Vol.435, 1989.

[6] M. Seltzer and O. Yigit, " A New Hashing Package for UNIX." Proceedings of the Winter USENIX technical conference, Dallas, TX, Jan, 1991.

[7] C. P. Wright, M. Martino and E. Zadoc, "NCryptfs: A Secure and Convenient Crypto Graphic File System." Proceedings of the annual USENIX technical conference, san Antonio, TX, June.2003.

[8] McShane-Inc. "Calculating the Checksum. Communications-Protocol.", http://mcshaneinc.com/ html/Library.CommProtocol. html accessed Dec.2005.

[9] R. Braden, D. Borman and C. Partridge," Computing the Internet Checksum." Network Working Group Request for Comments (RFC),1071, Sept. 1988.

[10] S. J. Chung, "Network Architecture: Hamming Codes and Cyclic Redundancy for Transmission Error Correction." ACM**,** Vol.33, Issue.4, Dec.2001.

[11] W. Toshihiro, S. Takahisa, Y. Hiroyuki , K. Koji," OS Environment Backup Technology (RAID-1, Standby Disk Series, Pre-install Image on *HDD)*" NEC Technical Journal, VOL.56, NO.6, 2003.

[12] W. Stallings," *Computer Organization and Architecture*." 7 [Ed], Prentice Hall, 2006, pp.186.

[13] U. K. Kumar, and B. S Umashankar B, "Improved Hamming Code for Error Detection and Correction." IEEE 2nd International Symposium on Wireless Pervasive Computing, 5-7 Feb. 2007.

[14] C. Chi-Shiang and C. Chin-Chen "An Efficient Image Authentication Method Based on Hamming code." Elsevier Pattern Recognition, Vol.40, No.2, 2007, pp.681-690.

[15] A. Thomasian, C. Han and G. Fu," A performance Tool for RAID Disk Arrays." Proc. Quantitative Evaluation of System, QEST'04, Netherlands, 2004, pp: 8-17.

[16] R. Y. Hou and Y. N.Patt,"Comparing Rebuild Algorithms for Mirrored and RAID5 Disk Arrays. "International Conference on Management of Data, 1993 (ACM SIGMOD 1993).