An Adaptive Technique Using Advanced Encryption Standard to Implement Hard Disk Security for Personalized Devices

Minal Moharir and A.V. Suresh

R.V. College of Engineering, Bangalore-59
minalmoharir@yahoo.com,
sureshav@rvce.edu.in

Abstract. The main objective of the paper is to develop an efficient and cost effective method for Hard Disk Drive(HDD) Security. The task is implemented using Full Disk Encryption (FDE) with Advanced Encryption Standards(AES) for data security of Personal Computers(PCS) and Laptops . The focus of this work is to authenticate and protect the content of HDD from illegal use. The paper proposes an adaptive methods for protecting a HDD based on Partial Disk Encryption(PDE) which one of the flavor of FDE. The proposed method is labeled as DiskTrust. FDE encrypts entire content or a single volume on your disk. Symmetric key uses same key for encryption as well for decryption. DiskTrust uses these two technology to build cost effective solution for small scale applications. Finally, the applicability of these methodologies for HDD security will be evaluated on a set of data files with different key sizes.

Keywords: Information Security, Integrity, confidentiality, Authentication, Encryption.

1 Introduction

As of January 2011 the internet connected an estimated 941.7 million computers in more than 450 countries on every continent, even Antarctica (Source: Internet Software Consortium's Internet Domain Survey; www.isc.org/index.pl). The internet is not a single network, but a worldwide collection of loosely connected networks that are accessible by individual computer hosts, in a variety of ways, to anyone with a computer and a network connection[1]. Thus, individuals and organizations can reach any point on the internet without regard to national or geographic boundaries or time of day.

However, along with the convenience and easy access to information come risks. Among them are the risks that valuable information will be lost, stolen, changed, or misused. If information is recorded electronically and is available on networked computers, it is more vulnerable than if the same information is printed on paper and locked in a file cabinet. Intruders do not need to enter an office or home; they may not even be in the same country. They can steal or tamper with information without touching a piece of paper or a photocopier. In this way security of stored information is an important issue. The proposed paper consider the security of Hard Disk Drive which is a fundamental element in computing chain.

The paper organized as follows. Related work, gap & problem is described in Section 2. A view of simulation and experimental design is given in section 3. Simulation results are shown in section 4. Finally the conclusions are drawn section 5.

2 Related Work

The related survey is divided into two parts. The first part is survey about full disk encryption. The second part is survey about advanced encryption standards.

Information security is the process of protecting information. It protects its availability, privacy and integrity[2]. More companies store business and individual information on computer than ever before. Much of the information stored is highly confidential and not for public viewing. Without this information, it would often be very hard for a business to operate[3]. Information security systems need to be implemented to protect this information. There are various ways to implement Information security systems. One of the popular technique is full disk encryption. Full Disk Encryption (FDE) is the safest way to protect digital assets[4], the hard drive is a critical element in the computing chain because it is where sensitive data is stored. Full disk encryption increases the security of information stored on a laptop significantly[5]. It helps to keep business critical data absolutely confidential. Moreover, full disk encryption helps to meet several legislative requirements.

Min Liang and Chao wen Chang (2010 IEEE) described a full disk encryption scheme based on XEN virtual machine which is stored in a security flash disk. XEN is used to encrypt (decrypt) all the data in hard disk and manage the whole system.

Li Jun & Yu Huiping (2010 IEEE) introduced the data encryption technologies of encrypting file system (EFS) and traditional full-disk encryption (FDE), and points out the problems of data encryption of EFS and FDE. Combined with the features of trusted platform module (TPM)[6], this paper constructed a trusted full-disk encryption (TFDE)[7] based on TPM.

The second part of survey covers implementation of Encryption Algorithms. Many encryption algorithms are widely available and used in information security. They can be categorized into Symmetric (private) and Asymmetric (public) keys encryption. In Symmetric keys encryption or secret key encryption, only one key is used to encrypt and decrypt data. The key should be distributed before transmission between entities. Keys play an important role. If weak key is used in algorithm then every one may decrypt the data. Strength of Symmetric key encryption depends on the size of key used. For the same algorithm, encryption using longer key is harder to break than the one done using smaller key[8]. The paper uses Symmetric key cryptography to implement disk security. The related work with respect to performance of various encryption algorithm is as follows

Jyothi Yenuguvanilanka Omar Elkeelany (2008 IEEE), This paper addressed the performance of Rijndael AES Encryption algorithm of key length 128 bits. Two hardware models based on HDL and IP core are used to evaluate the performance of the algorithm. The encryption time and also the performance metrics such as size, speed and memory utilization are evaluated, using these models.

Dazhong Wang & Xiaoni Li (2009 IEEE) presented the design, implementation and performance of a FIPS – approved cryptographic algorithm – Advanced Encryption Standard (AES), which can be used to protect electronic data.

El-Sayed Abdoul-Moaty ElBadawy & all (ICES 2010), This paper proposed a new chaos AES algorithm for data security. The algorithm is based on substituting the Rijndael affine transformation S-box by another one based on chaos theory. The new S-box has a low correlation and exhibits a significant performance improvement with an acceptable complexity addition.

S.Anandi Reddy & M.Arul Kumar M.Tech.,(2011 IEEE) In this paper, concurrent structure independent fault detection schemes for designing high performance and reliable architecture of the AES is presented. For high performance applications, instead of using look-up tables alone for the implementation of S-box and inverse S-box and their parity predictions, logic gate implementations based on composite fields are also utilized.

2.1 Research Gap

- The FDE technology discussed in above survey are encrypting the entire contents of Hard disk Drive. However encryption of the entire HDD is expensive in terms of time and cost. The large scale industries needs this much of tough security, as well they can accommodate big cost. For small industries or institution or personal users the data security is needed for partial data, so need some cost effective security scheme.
- The Symmetric Key Cryptography(SKC) is best for the security of personal devices as no need to share the key.
- Rijndael is most robust & better performance algorithm in available SKC

3 Problem Statement

To develop HDD security technique labeled as DiskTrust. DiskTrust technology uses PDE, creates authorized invisible volume on HD & implements SKC with Rijndael to secure the data stored on secured volume.

The technical objectives of the thesis are:

1. Create Hidden partition
2. Check authentication
3. Store/access data from the hidden volume
4. Execute encryption/decryption algorithm while reading /writing data on Hard Disk Drive.

4 AES for Implementation

The standard AES algorithm is as follows:

KeyExpansion: The keys are derived from the cipher key using Rijndael's key schedule

4.1 Initial Round

- **AddRoundKey:** each byte of the state is combined with the round key using bitwise xor

4.2 Rounds

- SubBytes—a non-linear substitution step where each byte is replaced with another according to a lookup table.
- ShiftRows—a transposition step where each row of the state is shifted cyclically a certain number of steps.
- MixColumns—a mixing operation which operates on the columns of the state, combining the four bytes in each column
- AddRoundKey

4.3 Final Round (no MixColumns)

- SubBytes
- ShiftRows
- AddRoundKey

To improve the performance following modifications are done:

In order to enhance the security and reliability of AES, we bring in three changes. In each iterative round, apart from the usual four above mentioned operations, we also include two new operations: The Arithmetic Operator and The Route Cipher.

We also modify the key schedule so as to increase the number of the AES encryption rounds. For example, for 16 byte key, we generate 336 bit key instead of the usual 176 bit key. By this process, we are able to successfully process 20+1 rounds instead of the previous 10+1 rounds for the 16 byte key. Lets have a look at the modifications and there implications.

4.4 Arithmetic Operation

In this operation, each element of the state is arithmetically added by a number depending on their row number.

The 1st row is added to 1.
The 2nd row is added to 2.

The 3rd row is added to 3.
The 4th row is added to 4.

To retain the symmetric nature of AES, during decryption we have inversed the process by subtracting the corresponding same numbers.

The 1st row is added to 1.
The 2nd row is added to 2.
The 3rd row is added to 3.
The 4th row is added to 4.

4.5 Route Cipher

In a route cipher, the plaintext was first written out in a grid of given dimensions, and then read off in a pattern given in the key. For example, using the same plaintext:

W R I O R F E O E
E E S V E L A N J
A D C E D E T C X

4.6 Extending the Key Schedule

We have also extended the key schedule. We have followed the key schedule process but we haven't stopped at the earlier specifications, rather we continued doing so in order to enable more computing iterative rounds, giving the attacker an even tougher code to break. For example, for 16 byte null key, we generate the following 336 bit extended key which facilitates the proper operation of 20+1rounds, i.e. double the number of rounds earlier.

The diskTrust technique implements AES & improved AES with different key size.

5 Simulation and Design

This section describes design and GUI implementation, some of the important results that were found as part of the implementatton.

5.1 Implementation of Hidden Volume

DiskTrust Security user interfaces are shown below in the screenshots. The user interface is basically a frame work application where user can use the application

5.1.1 Main Application Window

The Main application window holds multiple options such as CreateVolume, Mount and Dismount All.

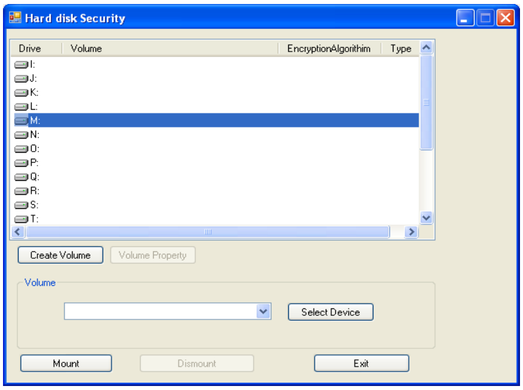
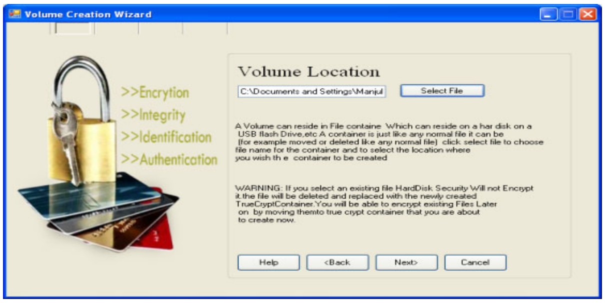


Fig. 1. Screenshot of Main Application Window

This is the first step from methodology which create a hidden volume on the HDD.

5.1.2 Volume Location

Volume Location Window allows the user to select location where the user wants to create the volume.



Volume Location

C:\Documents and Settings\Manjul

A Volume can reside in File containe Which can reside on a har disk on a USB flash Drive,etc A container is just like any normal file it can be (for example moved or deleted like any normal file) click select file to choose file name for the container and to select the location where you wish th e container to be created

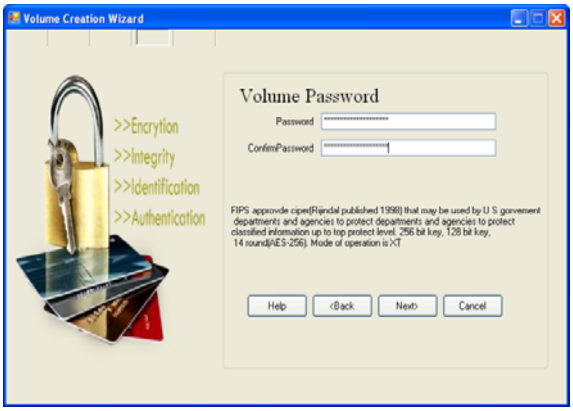
WARNING: If you select an existing file HardDisk Security Will not Encrypt it.the file will be deleted and replaced with the newly created TrueCryptContainer.You will be able to encrypt existing Files Later on by moving themto true crypt container that you are about to create now.

Fig. 2. Screenshot of Volume Location Window

5.2 Volume Password

Volume Password window will allow the user to enter the password and confirm Password.

Password implements user authentication.



Volume Password

Password ********************

ConfimPassword ********************

FIPS approvde ciper(Rijndal published 1998] that may be used by U S gorvement departments and agencies to protect departments and agencies to protect classified information up to top protect level. 256 bit key, 128 bit key, 14 round(AES-256). Mode of operation is XT

Fig. 3. Screenshot of Volume Password Window

5.3 Encrypt or Decrypt Data Using AES and Imprives AES with Different Key Size While Retrieving from Hidden Volume

For our experiment, we use a laptop PentiumV 2.4 GHz CPU, in which performance data is collected. In the experiments, the laptop encrypts a different file size ranges from 321K byte to 7.139Mega Byte. Several performance metrics are collected:

1- encryption time
2- CPU process time
3- CPU clock cycles and battery power.

The encryption time is considered the time that an encryption algorithm takes to produce a cipher text from a plaintext. Encryption time is used to calculate the of an encryption scheme. It indicates the speed of encryption. The CPU process time is the time that a CPU is committed only to the particular process of calculations. It reflects the load of the CPU. The more CPU time is used in the encryption process, the higher is the load of the CPU. The CPU clock cycles are a metric, reflecting the energy consumption of the CPU while operating on encryption operations. Each cycle of CPU will consume a small amount of energy.

6 Simulation Results

The effect of changing key size of AES on power consumption. The performance comparison point is the changing different key sizes for AES algorithm. In case of AES, We consider the three different key sizes possible. In case of AES it can be seen that higher key size leads to clear change in the battery and time consumption. It can be seen that going from 128 bits key to 192 bits causes increase in power and time consumption about 8% and to 256 bit key causes an increase of 16% . The simulation results with different key sizes are as shown in Table1.

Table 1. Time for Different Key Size

AES Key Size	AES 128	AES 192	AES 256
Time in Milliseconds	287	310	330

The graphical representation of the given data is as follows,

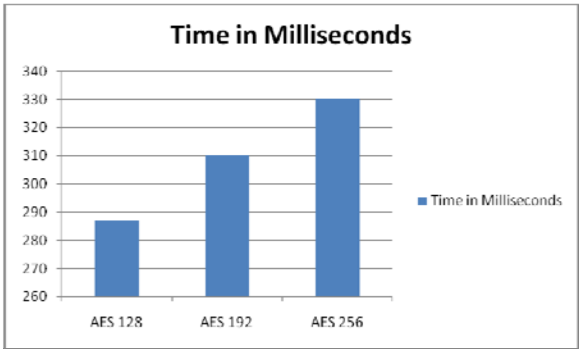


Fig. 4. Time with different key size

The analysis with improved AES is shown in Table3.

Table 2. Time for Different Key Size

Improved AES Key Size	I-AES 128	I-AES 192	I-AES 256
Time in Milliseconds	250	300	330

The graphical representation of the given data is as follows,

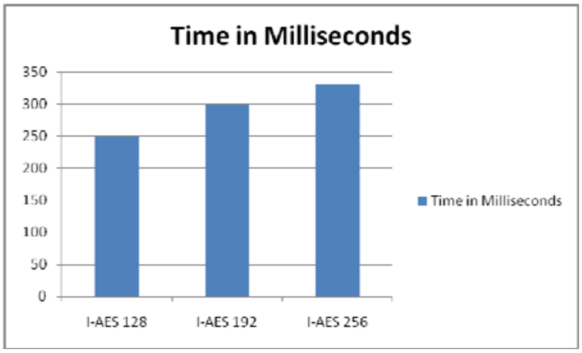


Fig. 5. Time with different key size

7 Conclusion

Proposed DiskTrust model is better suited for disk security of personal devices such PCs/Laptops over existing techniques. Disktrust model is cost effective & userfriendly.

– DiskTrust stores data on hidden volume so the user's information is in accessible or invisible to the unauthorized user.
– DiskTrust provides user's authentication.
– DiskTrust provides confidentiality by encrypting the data stored in invisible & authenticated disk volume.

With Modified Rijndael algorithm, for 128-bits block & 128-bits key encryption time required is 250ms. This gives 023% performance over standard Rijndeal.

References

1. Blömer, J., Krummel, V.: Fault Based Collision Attacks on AES. In: Breveglieri, L., Koren, I., Naccache, D., Seifert, J.-P. (eds.) FDTC 2006. LNCS, vol. 4236, pp. 106–120. Springer, Heidelberg (2006)
2. Blomer, J., Seifert, J.-P.: Fault Based Cryptanalysis of the Advanced Encryption Standard (AES). In: CHESS 2003, pp. 162–181 (2003)
3. Chen, C.-N., Yen, S.-M.: Differential Fault Analysis on AES Key Schedule and Some Countermeasures. In: Safavi-Naini, R., Seberry, J. (eds.) ACISP 2003. LNCS, vol. 2727, pp. 118–129. Springer, Heidelberg (2003)
4. Giraud, C.: DFA on AES. In: Dobbertin, H., Rijmen, V., Sowa, A. (eds.) AES 2005. LNCS, vol. 3373, pp. 27–41. Springer, Heidelberg (2005)
5. Kim, C.H., Quisquater, J.-J.: Faults, Injection Methods, and Fault Attacks. IEEE Design & Test of Computers 24(6), 544–545 (2007)
6. Moradi, A., Shalmani, M.T.M., Salmasizadeh, M.: A Generalized Method of Differential Fault Attack Against AES Cryptosystem. In: Goubin, L., Matsui, M. (eds.) CHES 2006. LNCS, vol. 4249, pp. 91–100. Springer, Heidelberg (2006)
7. Piret, G., Quisquater, J.-J.: A Differential Fault Attack Technique against SPN Structures, with Application to the AES and KHAZAD. In: Walter, C.D., Koç, Ç.K., Paar, C. (eds.) CHES 2003. LNCS, vol. 2779, pp. 77–88. Springer, Heidelberg (2003)
8. FIPS-197, Advanced Encryption Standard (AES), Federal Information Processing Standards Publication 197, November 26 (2001), `http://csrc.nist.gov/publications/`
9. Karri, R., Wu, K., Mishra, P., Kim, Y.: Concurrent Error Detection Schemes for Fault-Based Side-Channel Cryptanalysis of Symmetric Block Ciphers. IEEE Trans. on Computer-Aided Design of Integrated Circuits and Systems 21(12), 1509–1517 (2002)
10. Maistri, P., Vanhauwaert, P., Leveugle, R.: A Novel Double-Data- Rate AES Architecture Resistant against Fault Injection. In: Workshop on Fault Diagnosis and Tolerance in Cryptography, FDTC 2007, pp. 54–61 (August 2007)
11. Monnet, Y., Renaudin, M., Leveugle, R.: Designing Resistant Circuits against Malicious Faults Injection Using Asynchronous Logic. IEEE Trans. on Computers 55(9), 1104–1115 (2006)

12. Wu, K., Karri, R., Kuznetsov, G., Goessel, M.: Low Cost Concurrent Error Detection for the Advanced Encryption Standard. In: International Test Conference, pp. 1242–1248 (2004)
13. Bertoni, G., Breveglieri, L., Koren, I., Maistri, P., Piuri, V.: Error Analysis and Detection Procedures for a Hardware Implementation of the Advanced Encryption Standard. IEEE Trans. on Computers 52(4) (April 2003)
14. Yen, C.H., Wu, B.F.: Simple Error Detection Methods for Hardware Implementation of Advanced Encryption Standard. IEEE Trans. on Computers 55(6), 720–731 (2006)
15. Di Natale, G., Flottes, M.L., Rouzeyre, B.: An On-Line Fault Detection Scheme for SBoxes in Secure Circuits. In: Proc. of 13th IEEE International On-Line Testing Symposium, IOLTS 2007, pp. 57–62 (2007)
16. Mozaffari Kermani, M., Reyhani-Masoleh, A.: Parity-Based Fault Detection Architecture of S-box for Advanced Encryption Standard. In: 21st IEEE Int. Symp. on Defect and Fault-Tolerance in VLSI Systems (DFT 2006), pp. 572–580 (2006)
17. Dusart, P., Letourneux, G., Vivolo, O.: Differential Fault Analysis on A.E.S. In: Zhou, J., Yung, M., Han, Y. (eds.) ACNS 2003. LNCS, vol. 2846, pp. 293–306. Springer, Heidelberg (2003)
18. Bosio, A., Di Natale, G.: LIFTING: a Flexible Open-Source Fault Simulator. In: Proc. of the IEEE Asian Test Symposium, pp. 35–40 (2008)
19. Wolkerstorfer, J., Oswald, E., Lamberger, M.: An ASIC Implementation of the AES SBoxes. In: Preneel, B. (ed.) CT-RSA 2002. LNCS, vol. 2271, pp. 67–78. Springer, Heidelberg (2002)