International Journal of Control Theory and Applications

ISSN : 0974–5572

© International Science Press

Volume 9 • Number 42 • 2016

Parallelizable Modes of Operation of a Block Cipher for Disk Encryption

Manoj Kumar[a] and Bappa Mondal[a]

[a]Computer Science and Engineering Department Delhi Technological University, New Delhi

E-mail: mkumarg@dce.ac.in, bappamondal7@gmail.com

Abstract : Secured storage of Data in secondary storage, like hard disk and flash memory, has become a challenge now a days. Owing to the increase in size and use of data, data-security and encryption has become extremely important. For Disk Encryption, we generally use AES Cipher with different modes of operations. In case of CBC modes of operation, it has a dependency with previous block during encryption. Therefore, all disks have to be encrypted for the first time and only one after another block encryption occurs. Therefore, Parallelization cannot be archived.

This paper proposes a method to parallelize AES with CBC modes. AES with proposed CBC has been implemented in C language and performance has been measured using Intel Core *i*7 (2.4GHZ), 8GB of RAM. This parallel approach exhibits improved performance over sequential approach of 1.3 has been achieved.

Keywords: *AES, Parallelization, Initialization Vector, Modes of Operation, Disk Encryption.*

1. INTRODUCTION

With the development of mobile and internet technology, E-commerce and M-commerce services are now becoming part of everyone's life. However, for such commerce services, security is a major concern, which has two basic requirements: integrity and confidentiality. For such services we need an environment on mobile devices which provides data confidentiality and preserves integrity with modest computational costs is urgently required, especially in wireless/mobile environment which have limitation in computational capability [1]. Users are expecting to secure data transmission and storage on wireless mobile devices, which require efficient cryptographic algorithms[2].

CBC mode of block cipher is used as default in many secure communication protocols, but we see that one bit alteration in certain ciphertext block will only produce errors in two plaintext blocks, while decrypting the message. So, CBC mode can not support any mechanism for the integrity verification either. Our aim is to find out a mechanism for the verification of the integrity of the message, and also find an efficient mechanism that does not delay the encryption process. With this in view the Massachusetts Institute of Technology designed a new cipher mode, which is based on CBC mode, to solve this problem.

With CBC cipher block, we have solved malleability attack. It re-creates a problem when encrypting/ decrypting a particular block. As blocks are accessed non-sequentially, it is required that the whole disk be encrypted / decrypted. Disk access cannot depend on the contents of their preceding/succeeding sectors

To avoid such re-work of encryption, CBC algorithm requires Initialization Vector (IV)[2,3]. Each sector need to be independent of each other and is, somehow, random in nature and unpredictable. Therefore, each block requires a separate IV to support random block read/write. The usual methods for generating IVs are predictable sequences of numbers, for example, time stamp or sector number and permit certain attack such as a watermarking attack. Watermarking attack is an attack on disk encryption methods where the presence of a specially crafted piece of data can be detected by an attacker without knowing the encryption key. If these IVs are predictable by an attacker (and the file system reliably starts the content at the same offset to the start of each sector, and files are likely to be largely contiguous), then there is a chosen plaintext attack which can reveal the existence of encrypted data.

With the proposed paper, we shall see how we can generate IV for each individual block and how we can achieve parallelization on it. With the advancement of Multi-processor, we can achieve significant amount of faster encryption as well as reduce power consumption.

2. RELATED WORK

2.1. Disk Encryption at Block Level

I In this section, we shall see one case of disk encryption and the problem exists within it and then how to solve problem. For case study, we have taken Disk Encryption Technology used in Android OS[4]. Android is a mobile operating system developed by Google and has the largest installed base of all operating systems of any kind.

Disk encryption methods aim to provide three distinct properties:

1. The data on the disk should remain confidential.
2. Data read/write should be fast after encryption, no matter where on the disk the data is stored.
3. The encryption method should not waste disk space.

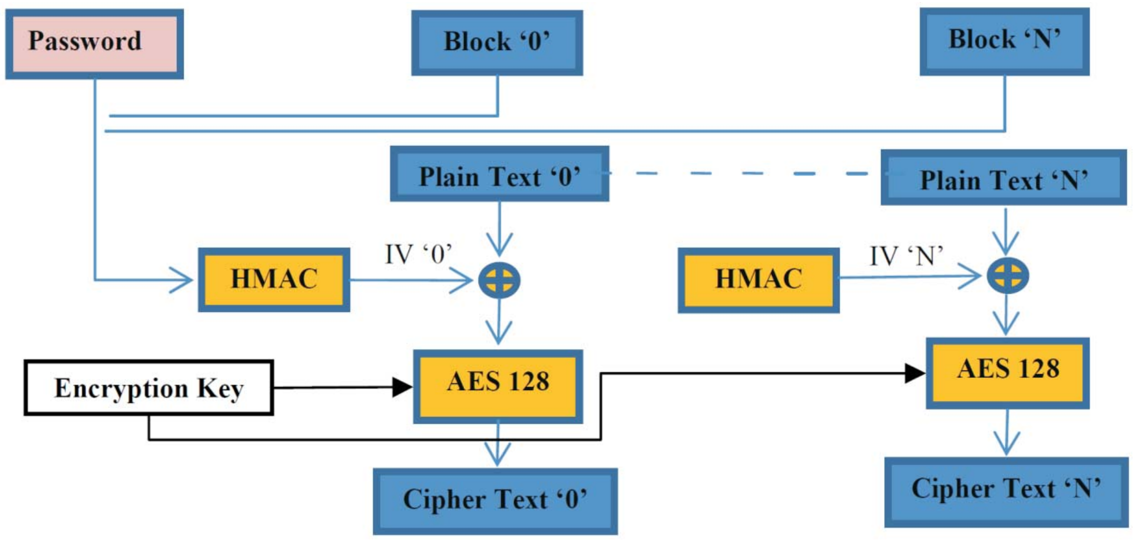


Figure 1 : Parallelizable modes of cipher block

Disk/eMMC storage area is divided into blocks. A block is the smallest readable or writable unit which can be addressed, typically size of 4KB. Disk Encryption uses same key for encrypting each blocks with AES algorithm (Symmetric Algorithm)[5]. Assume two separate blocks having same plain text, as we use same disk encryption key for all blocks, cipher text on those blocks will be same. This expose to Malleability attack, where plaintext is known to the adversary, it is possible to change every 2nd plaintext block to a value chosen by the attacker, while the blocks in between are changed to random values, which does not ensure integrity of the encrypted data.

With proposed parallelizable modes of block cipher, we pass block number that we are going to encrypt and keys (supplied by user password) and pass it to HMAC method (keyed-hash message authentication code) to produce a 16 bytes unique random number, that can be used as IV per block. Since IVs in this case is independent in all blocks, parallelization can be achieved and well suited to operate on a multi-processor machine where blocks can be encrypted in parallel.

2.2. Maintaining the Integrity of the encrypted data

Integrity is important property of encryption and should not corrupt data. Data has been encrypted and decrypted for initial 2 blocks and verified if matches with original plain text.

Integrity checking block has been made for CBC and proposed parallelizable mode block cipher.

```
check = memcmp((char*)plain_input,(char*)plain_output, newLen);
if (0! = check) {
printf("\n\nError:Integrity fails in CBC mode\n");
}
```

3. OVERVIEW OF THE OFB MODES OF OPERATIONS

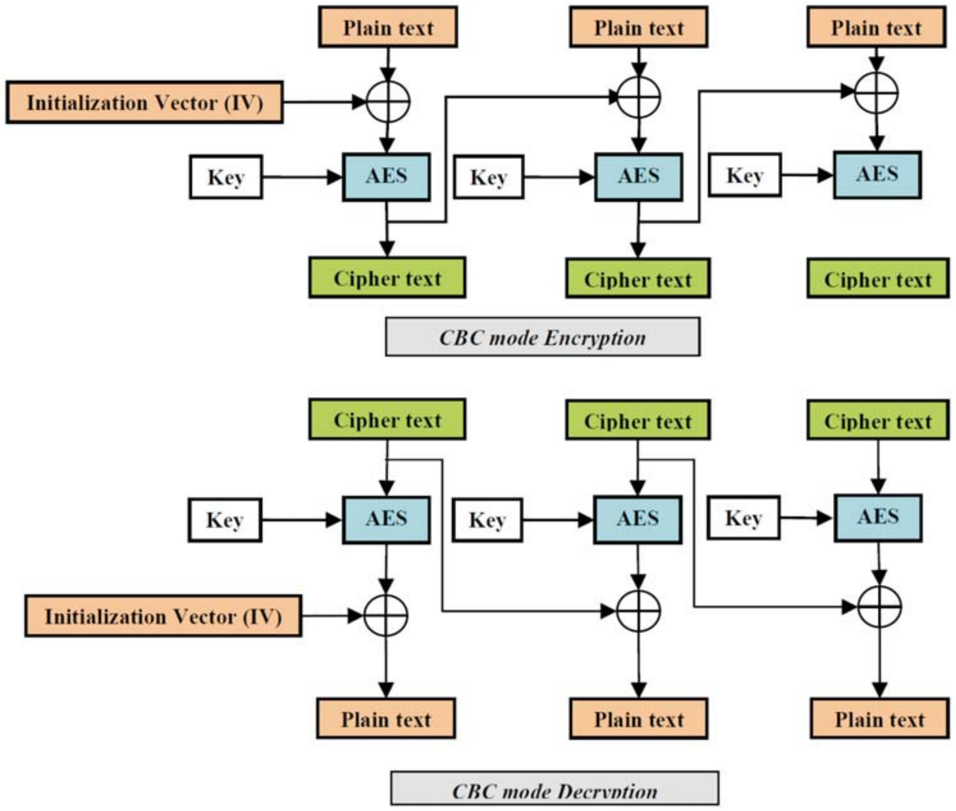


Figure 2 : Cipher Block Chaining

IIn cryptography, a mode of operation is an algorithm that uses a block cipher to encrypt messages of arbitrary length in a way that provides confidentiality or authenticity. A block cipher by itself is only suitable for the secure cryptographic transformation (encryption or decryption) of one fixed-length group of bits called a block[6,7]. A mode of operation describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block[8].

Each block of plaintext is XORed with the previous cipher text block before being encrypted. This way, each cipher text block depends on all plaintext blocks processed up to that point. Therefore Encryption is not parallelizable, where random read is possible.

Its main drawbacks are that encryption is sequential (*i.e.*, it cannot be parallelized), and that the message must be padded to a multiple of the cipher block size. One way to handle this last issue is through the method known as Cipher Text Stealing. Note that a one-bit change in a plaintext or IV affects all following cipher text blocks.

4. RESULTS

Disk Encryption and Parallelizable modes of Block Cipher Performance Analysis have been done with CBC modes of block cipher. Measurement has been made with amount of data to encrypt and time required completing encryption. With Demo program, we have run and tested it on Intel Core *i*7 (2.4GHZ), 8GB of RAM and Windows 7 Enterprise N version.

Program has been tested to establish and to evaluate amount of performance that can be achieved with parallelization.

Approximately, 30% higher performance in average and large data size and bigger block size (1 ~ 4 KB) will give best results for disk encryption.

To verify the encryption ability of the proposed method, simulation is done to do encryption along with CBC. The results thus obtained from the simulation show that the proposed method used for disk encryption, which proposes the use of independent IV per block.

Table 1
Performance Evaluation Result

S.No.	*Block Size*	*Time taken in CBC (Sec)*	*Time taken in Parallel Block Cipher (Sec)*
1.	100 MB	6.675	5.01
2.	200 MB	13.245	10.28
3.	300 MB	19.675	15.595
4.	400 MB	31.68	20.4
5.	500 MB	34.58	26.085
6.	600 MB	39.965	31.1
7.	700 MB	46.97	36.607
8.	800 MB	53.541	41.371
9.	900 MB	64.186	48.796
10.	1GB	68.801	52.311
11.	2 GB	135.515	107.182
12.	3 GB	327.286	154.279
13.	4 GB	335.606	220.531
14.	5 GB	374.791	269.629

From the table 1, we can see that in case of Parallelizable modes performance has been in terms of time taken to encrypt. It can be observed with the graph given below that with increase of size of data, time differences also increases; therefore, it can be an ideal solution and can provide significant output for disk encryption for longer disk size for several GB or TB. With above proposed method, we can see average 30% higher encryption speed as compared to CBC.

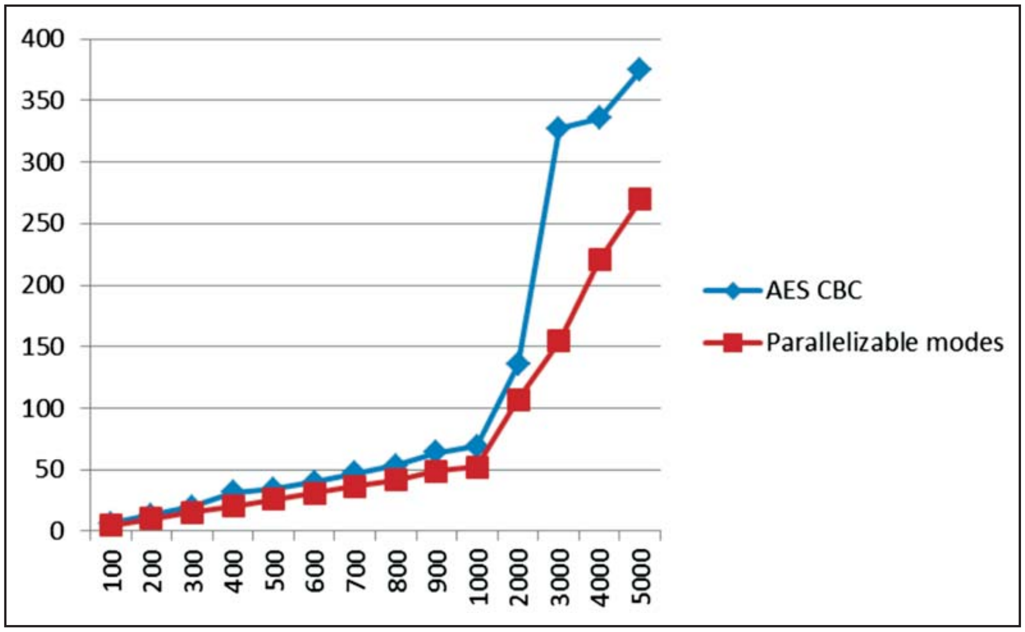
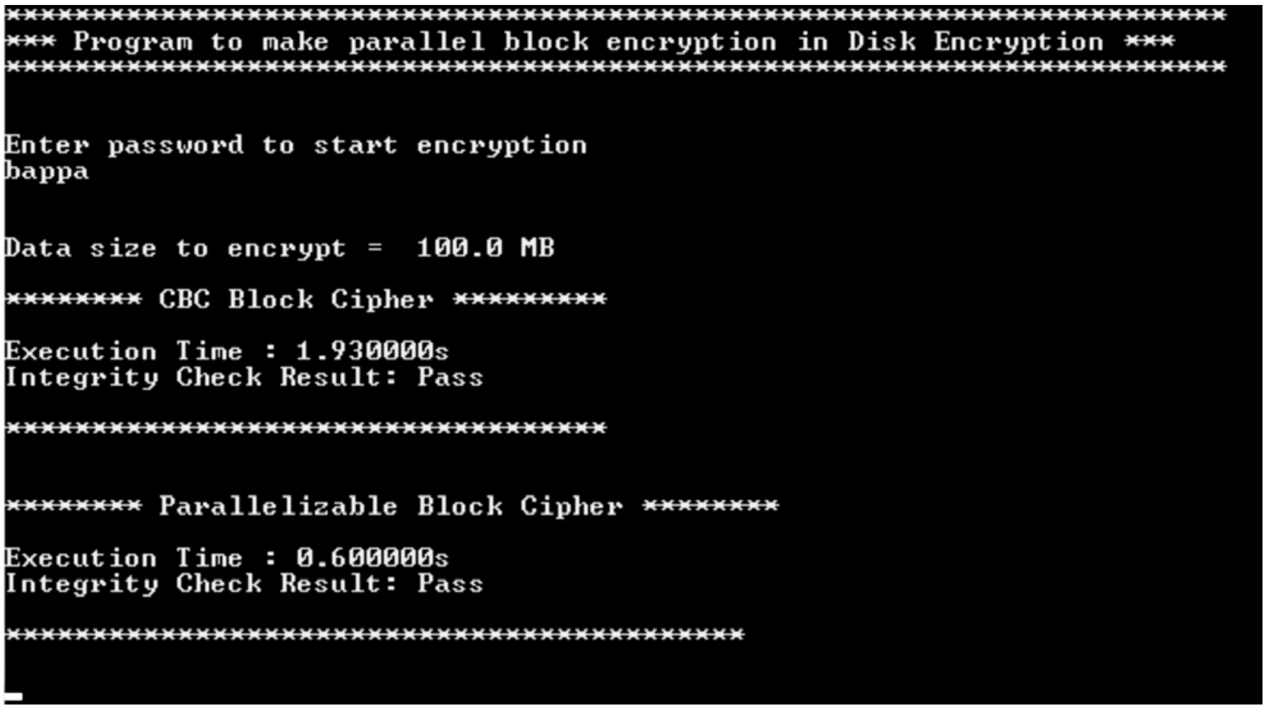


Figure 3: Performance Graph of CBC and Custom penalization mode



```
************************************************************************
*** Program to make parallel block encryption in Disk Encryption ***
************************************************************************


Enter password to start encryption
bappa


Data size to encrypt =  100.0 MB

******** CBC Block Cipher *********

Execution Time : 1.930000s
Integrity Check Result: Pass

***********************************


******** Parallelizable Block Cipher ********

Execution Time : 0.600000s
Integrity Check Result: Pass

*******************************************


_
```

Figure 4: Output to show execution time and integrity check result

5. CONCLUSION AND FUTURE WORK

In this paper, AES and Block Cipher has been used for disk encryption. For reference, we have taken Disk Encryption methodology from Android Disk Encryption and simulation has been done in Windows 7 machine. With output, we can see approximately, 30% higher performance than CBC block cipher mode.

This may form the future work on Disk Encryption and block cipher. The method to be followed for Disk Encryption is as follows:

Bigger encryption blocks size (Currently used 512 bytes) to give encryption results. Here, 512 bytes encryption block has been taken as reference used in Android Disk Encryption. For bigger encryption block, there will be a single IV for single block.Therefore, there could be security hole for water marking attacks. Other side, for small encryption block, need to generate IV for small block means more IV, therefore, more HMAC call, where more HMAC is expensive method in terms of execution time.

REFERENCES

[1] Like Chen, Runtong Zhang, "A Fast Encryption Mode for Block Cipher with Integrity Authentication" IEEE International Conference on Service Operations and Logistics, and Informatics, 2008. IEEE/SOLI 2008.

[2] Razvi Doomun, Jayramsingh Doma, Sundeep Tengur "AES-CBC Software Execution Optimization", International Symposium on Information Technology, 2008. ITSim 2008.

[3] M.Vaidehi, Dr. B.Justus Rabi "Design and Analysis of AES-CBC Mode for High Security Applications", 2nd International Conference on Current Trends in Engineering and Technology (ICCTET), 2014.

[4] Zhaohui Wang, Rahul Murmuria, Angelos Stavrou "Implementing and Optimizing an Encryption File system on Android", IEEE 13th International Conference on Mobile Data Management (MDM), 2012.

[5] Akshay Desai, Krishna Ankalgi, Harish Yamanur, Siddalingesh S. Navalgund: "Parallelization of AES algorithm for Disk Encryption Using CBC and ICBC modes", Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT),2013.

[6] L. Martin, "XTS: A Mode of AES for Encrypting Hard Disks", IEEE Security & Privacy, 2010.

[7] Android Security Bulletin : https://source.android.com/security/encryption/

[8] Block disk encryption : https://bappamondalblog.wordpress/2015/12/26/android-disk-encryption