ÇANKAYA UNIVERSITY
FACULTY OF ENGINEERING
COMPUTER ENGINEERING DEPARTMENT

Project Report
Version 1

CENG 407
Innovative System Design and Development I

201912
A CUDA Based Disk and File Encryption

Fatih Taşdemir
201511058
Muhammed Bera Erkaya
201511022
Emre Yüncü
201511068
Doğa Üçüncü
201411063

Advisor: *Asist. Prof. Dr. Hüseyin Temuçin*

Table of Contents

Abstract

The files on the computer system are kept in plain form, and third parties who access the disk via physical or network can read this data. A file encryption library will be developed within the scope of the project and will enable encrypt/decrypt operations of encrypted files in a certain format. Since the files are large-scale data, CUDA will be used to make this process faster and encryption will be performed on the GPU.

Key words:

File encryption, CUDA, encryption, decryption, block encryption, AES algorithm, CPU, GPU, cache.

Özet:

Bilgisayar sistemindeki dosyalar düz olarak tutulur ve diske fiziksel veya ağ üzerinden erişen üçüncü kişiler bu verileri okuyabilir. Proje kapsamında bir dosya şifreleme kütüphanesi geliştirilecek ve şifrelenmiş dosyaların şifreleme / şifre çözme işlemlerinin belirli bir formatta yapılmasına olanak sağlanacaktır. Dosyalar büyük ölçekli veriler olduğundan, bu işlemi hızlandırmak için CUDA kullanılacak ve GPU'da şifreleme yapılacaktır.

Anahtar Kelimeler:

Dosya şifreleme, CUDA, şifreleme, şifre çözme, blok şifreleme, AES algoritması, CPU, GPU, ön bellek.

Introduction

Nowadays, many files are shared on the internet. They must be protected and not in the hands of third parties. Our project aims to protect the data by encrypting large-scale files. Because our files are large, we use Cuda to achieve our goal faster.

CUDA is a parallel computing platform and a programming model that makes GPU usage for general purpose computing simple and neat. The developers still work on C, C++, Fortran, and expand the list of supported languages. Moreover, they incorporate extensions of these languages in the form of a few basic keywords.

To encrypt, symmetric encryption will be applied. Symmetric encryption is the most preferred encryption model today. Because symmetric encryption is hard to break it. But for users it is easy to use. And we use AES algorithm. AES algorithm is one of most using algorithm of symmetric encryption. Because it is more safe than other algorithms.

We are developing CUDA based disk and file encryption library. When users added this library to their project, users can encrypt their files which they want. With CUDA encryption will be faster and with symmetric encryption it will be safer.

Scope of Project

With this library people can encrypt their files. Because we use CUDA for GPU's CPU. And it work faster than pc's cpu. And this library compatible with C/C++. We will use symmetric encryption. And use AES algorithm for encryption. AES (Advanced Encryption Standard) is a standard for encryption of electronic data. AES, adopted by the US government, is also used internationally encryption (crypto) standard. It replaces DES (Data Encryption Standard). The encryption algorithm defined by AES is a symmetric-key algorithm in which the keys used for both encryption and decryption are related. The encryption and decryption keys for AES are the same.

1. Literature Review

1.1 Project Related Sections

Our work in our project is divided into two important sections. One is Cuda and the other is Encryption.

1.1.1 CUDA

- The main task of the Graphics Processing Unit (GPU) is to display the images generated on the computer.
- Due to the insufficient CPU computing problems, the parallel structure of the GPU has begun to be utilized.
- CUDA is a parallel computing architecture introduced by NVIDIA in 2006 to take advantage of the computing power of the GPU.
- It can run on Linux, Windows and Mac Osx platforms.
- CUDA can support programming languages such as C, C ++, Python.
- The CUDA interface allows faster data readings than the GPU compared to the CPU.

  GPU differs from CPU in that it has a Single Instruction Multiple Data (SIMD) architecture. CPU calculations are performed in series. GPU calculations are performed in parallel.

1.1.2 CUDA Work Flow

- Applications developed in CUDA architecture do not only work on the GPU. First, it must be copied to the memory on the graphics card via the main memory controlled by the CPU.
- The data in the GPU memory is executed by the CUDA threads and the calculation is completed in parallel. Then sent back to the main memory to finish the process.
- The pieces of code running on the CPU are different from the pieces of code running on the GPU.

The "Host" can be considered a CPU. The "Device" can be considered a GPU. Serial codes are executed on the CPU. Parallel pieces of code called "kernel" are executed on the GPU.

1.1.2.1 CUDA Programming Architecture

- Kernel
    - The part of a code developed in CUDA that will run on the GPU side is called "kernel".
    - GPU creates a kernel copy for each element of the dataset and is called a "thread".
    - The kernel code is invoked by the Host and executed on the Device. Kernel code is considered "global".
- Thread
    - Thread is the smallest thing in CUDA architecture. In blocks, they can be 1D, 2D or 3D.
    - They run the same piece of code simultaneously.
    - The threads are arranged in blocks and grouped. Threads in different Blocks do not work together.
    - Each thread has its own ID in the block. These indices; "threadIdx.x", "threadIdx.y" and "threadIdx.z".
- Block
    - "Block" structure consists of parallel threads. They are unique in the "grid". They can be 1D, 2D or 3D within the grid.
    - The blocks are arranged in grid and grouped.. Each Block has its own index in the Grid. These indices are "blockIdx.x", "blockIdx.y" and "blockIdx.z".
    - They are dimensioned according to the number of rows and coulumns, such as “blockDim.x”, “blockDim.y”, blockDim.z”.
- Grid
    - Grid is a structure that "blocks" come together. Each "kernel" call creates a "grid".
    - Grid dimensions can be expressed as “gridDim.x”, “gridDim.y” and “gridDim.z”. [1]

1.2.2 Encryption

Encryption is an encryption process to prevent data from being read by people and other computers and to prevent the original content from being accessed.

Each Encrypt operation is performed according to a specific algorithm and the encrypted data can be made readable with a simple solver. Encrypting data appears to be pointless until decrypted and is nonfunctional.

1.2.2.1 Symmetric Encryption

Symmetric encryption is an encryption scheme in which the same key is used to both encrypt and decrypt messages. This type of information coding method has been used frequently in the past decade to ensure confidential communication between states and armies. Nowadays, symmetric key algorithms are widely applied to improve data security in various computer systems.

Symmetric encryption schemes are based on a single key shared by two or more users. The same key is used to encrypt and decrypt plain text. The encryption process consists of creating a ciphertext by passing a plaintext through an encryption algorithm called cipher.[2]

If the encryption scheme is strong enough, the only way for a person to read or access information in the ciphertext is to use the key to decrypt it. The decryption process consists essentially of converting the encrypted text back to plain text.

The security of symmetric cryptosystems is based on how difficult it is to estimate the key corresponding to the system by applying brute force. For example, it takes billions of years to estimate a 128-bit key using ordinary computer hardware. The longer the password key, the harder it is to break it. 256-bit keys are generally considered very secure and theoretically resistant to brute force attacks by quantum computers. Symmetric encryption has advantages over itself. Encryption and decryption are quick, easy to implement with hardware. Confidentiality of communication between the parties is ensured. The integrity of the data is ensured. The original text cannot be changed unless the encrypted text is decoded.[3]

1.2.2.2 AES Encryption Algorithm

The more popular and widely adopted symmetric encryption algorithm likely to be encountered nowadays is the Advanced Encryption Standard (AES). It is found at least six time faster than triple DES.

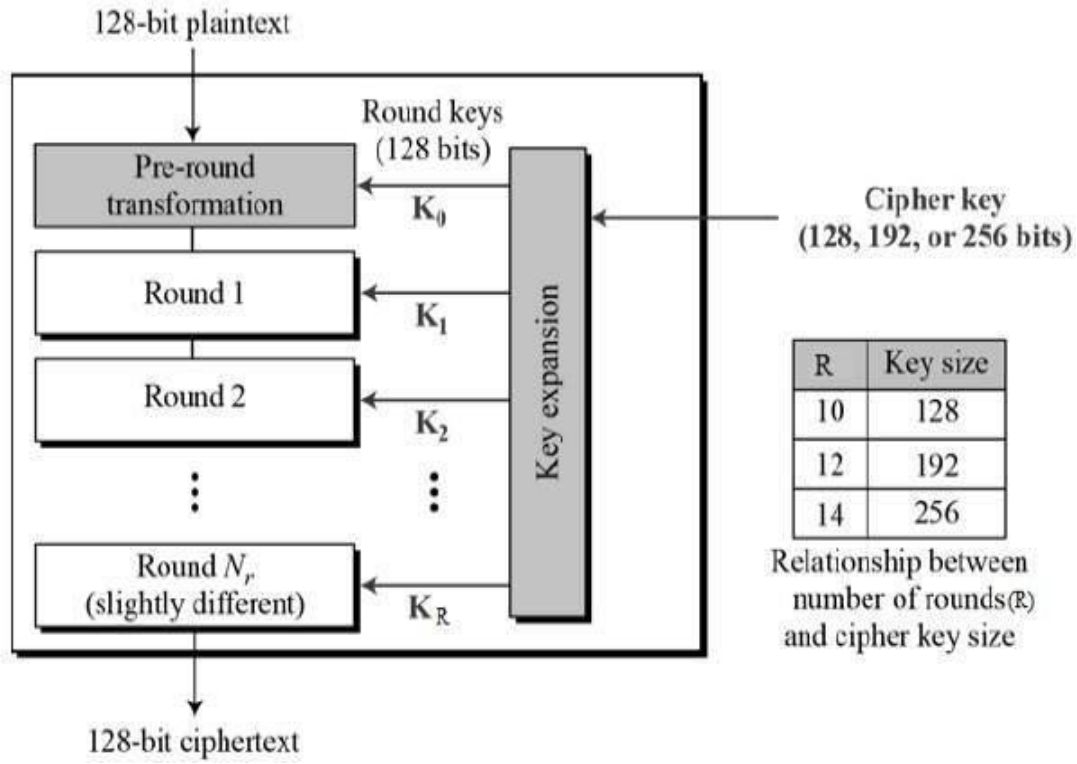
A replacement for DES was needed as its key size was too small. With increasing computing power, it was considered vulnerable against exhaustive key search attack. Triple DES was designed to overcome this drawback but it was found slow.[4]

1.2.2.2.1 Operation of AES

AES is an iterative rather than Feistel cipher. It is based on ‘substitution–permutation network’. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix.

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.[5]



R	Key size
10	128
12	192
14	256

Relationship between number of rounds(R) and cipher key size

2. Software Requirements Specification

2.1 Purpose

The files on the computer system are kept in plain form, and third parties who access the disk via physical or network can read this data. A file encryption library will be developed within the scope of the project and will enable encrypt/decrypt operations of encrypted files in a certain format. Since the files are large-scale data, CUDA will be used to make this process faster and encryption will be performed on the GPU.

2.1.1 Glossary

- CUDA- Compute Unified Device Architecture
- AES- Advanced Encryption Standard(Encryption algorithm)
- ENC-Encryption

- DEC- Decrption
- CPU- Central Processing Unit
- GPU- Graphics Processing Unit

2.2 Overall Description

The SRS document will include performance and interface requirements, operations and functions, software and system features, and interface information for this project.

2.2.1 Product Perspective

This product will be tool and library. Users can use this for encrypt their files and disks. And this system work with CUDA because this system use GPU's CPU.

2.2.2 Operations

After importing our library into the program, we will open the file with the fopen () function and convert the text into encrypted text with AES encryption. We can do this in 256, 512 or 1024 blocks. We will determine the efficiency of the encryption process in a direct proportion. We're thinking of assigning the blocks to be encrypted as the key of the next block during the encryption process. The encrypted file can be decrypted at any time with the help of these keys.

Our library encrypt a file which you want.When user add this library, the system encrypts the file with CUDA using symmetric encryption using CPU of the GPU. Users have a key for encrypt and decrypt. User select file for encryption. Then with our library system encrypt this file automatically. When user want decrypt this file users need their symmetric key. Otherwise users can not access their file. We use AES algorithm for symmetric encryption. And then transfer to otherside, then decrypts on the otherside.

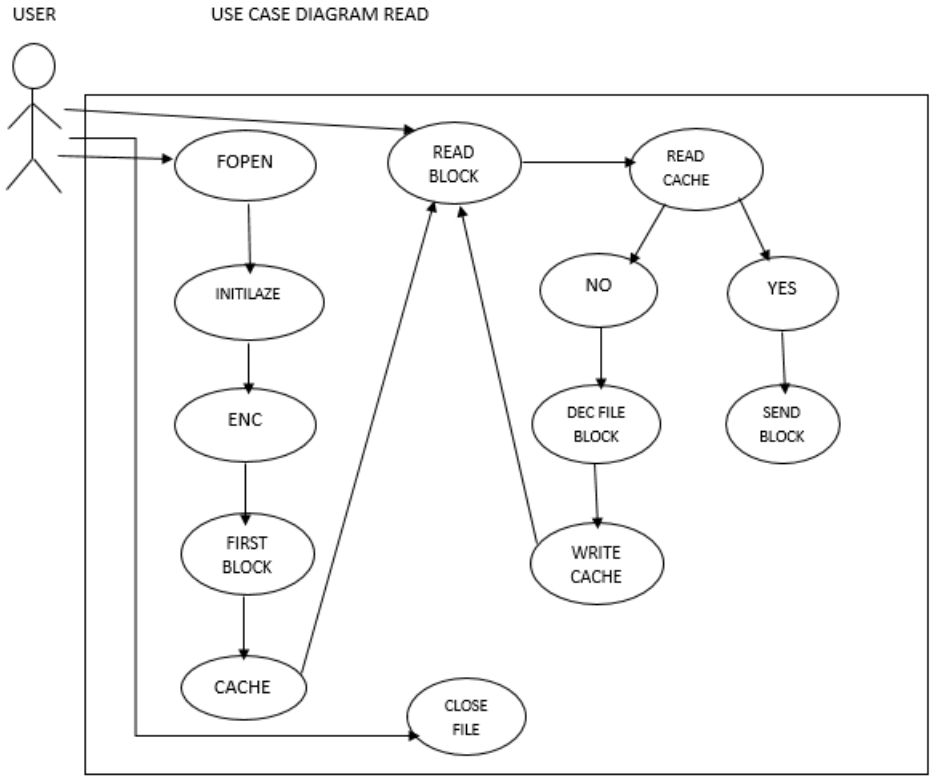
2.2.3 System Adaption Requirements

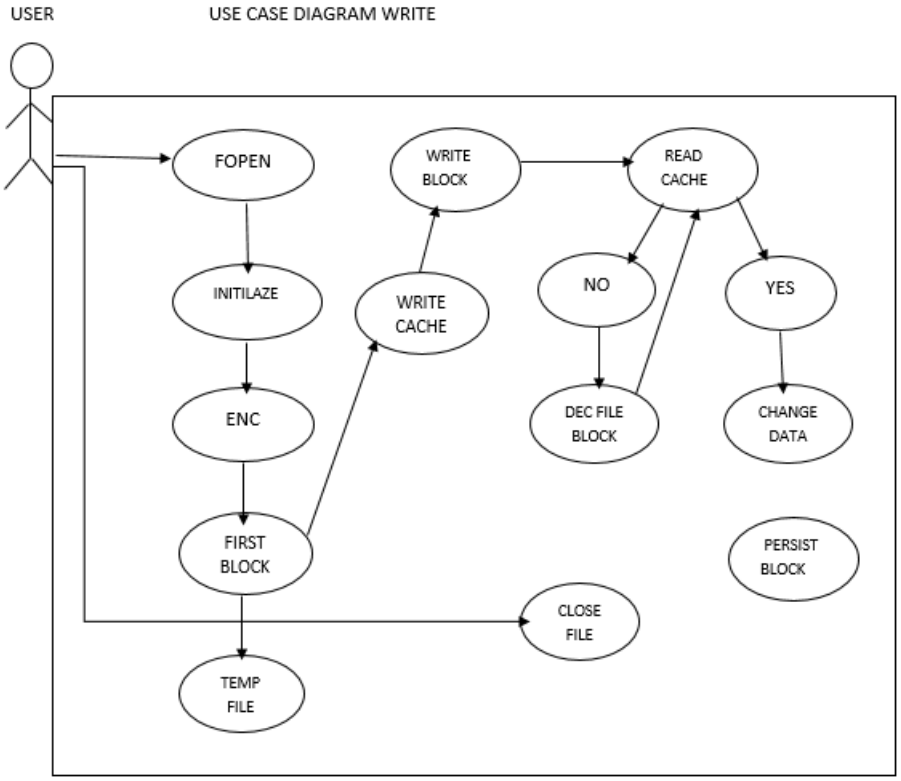
We decided to use the GPU because we could not get the desired efficiency due to the CPU's lack of cores in this project, where we will make encryption using the library. We will develop our library using the Multiple Instruction Multiple Data (MIMD) structure of the GPU. We will use linux systems as operating system. This system is a C/C++ library and this library needs to be called from users for encryption. In future works there might be phyton support this library. We adapt our symmetric encryption system to c/c++ language in first.

2.2.4 Product Funcitons

In this project, we can split the file to be encrypted into 256, 512 or 1024 blocks and perform the encryption process gradually. We will decide the number of blocks to be directly proportional to the efficiency we will receive in encryption operations. During the encryption process, the blocks to be encrypted can be the key of the next block, or we can focus on different scenarios. With this library, there will be encryption and decryption on file. The other function of this library is in system “a” it’s encrypted and then in system b it is decrypted. At first system “a" encrypt their file and system “a” has key for encryption. Then in system “b” has a key for decryption. Without this key system “b” can not decrypt this file. These keys can be128,192 or 256-bit long. User try to decrypt file. If user type incorrect this key, system will lock this file for security in our system. This work with symmetric encryption. We use AES algorithm, because nowadays this algorithm is more safe than other algorithms. In system “a” and system “b” there are keys. This symmetric encryption works compatible with CUDA.

2.2.5 Use Case Diagram





2.3 Specification of Requirements

2.3.1.1 Software Interfaces

Any software can access to this system which use NVidia GPU. In first we will use it for c/c++. But then we adapt it for other languages. This library can be used in visual studio, DEV-C.

2.3.1.2 Performance Requirements

Since we will run our project over GPU, the number of cores in the GPU will be important for us. The NVIDIA graphics card is critical, as we will also use the CUDA language. In NVIDIA graphics cards, the number of cores increases as the models increase. For these reasons, the NVIDIA graphics card models and the number of cores are important elements for our project. On encryption side we choose symmetric encryption and AES algorithm. Symmetric encryption is less likely to occur. And AES is algorithm of symmetric encryption. We choose AES algorithm, because this algorithm is one of safest algorithm in the world. But AES algorithm is slower than DES algorithm. Because AES works with higher bits.

2.3.2 Software System Attributes

2.3.2.1 Portability

At the end of the project we will add our own file structure. This file structure will have a structure that has its own header like pdf. This file structure will add mobility to us. In this way, we will provide greatly convenience in running our file.

2.3.2.2 Performance

Since we will use the AES algorithm on the encryption side, it will be more secure. AES algorithm already included in many encryption packages is the first publicly available encryption algorithm approved by the NSA(National Security Agency) to encrypt confidential information. We use CUDA to make the system encrypt faster.

2.3.2.3 Adaptability

Our library will be compile and execute on any linux operating system. If compiled and executed using CUDA, the yield will be higher. The reason for this is the efficiency to be obtained by running through the GPU. We plan to adapt this library C/C++ in first. But then we can adapt other languages like java, phyton.

2.3.2.4 Safety Requirements

This is encryption library. Most important thing is you must not loose your key and don't give your key to other people. We will use one of most secure encryption algorithm. There are 2128 different keys in AES 128-bit encryption, and it takes a great deal of time and cost to decrypt it. Suppose that a person uses all the technologies currently available as hardware to decode a 128-bit password, it takes approximately 100 years to crack the password. It means at 256-bit it is very long time. In our project user have key for encryption and decryption. User enter key for decryption the file. But then if user enter this key wrongly for 3 times file lock itself. After that users need another key like "Puc code" in phones. If user enter this key too, file will be deleted by system for safety. With this safety system, system will be protected from brute force. A brute force attack is an attempt to crack a password or username or find a hidden web page, or find the key used to encrypt a message, using a trial and error approach and hoping, eventually, to guess correctly.[6]

3.Software Design Descriptions

3.1.1 Purpose

- Information and communication technologies are evolving and businesses are restructuring on the basis of new technology and they are trying to achieve more successful activities, services and products. There are also a number of problems associated with the fact that information is important to companies. These are the

concerns about data security. A number of measures need to be taken to ensure that these concerns are eliminated. [7]

- Your data can be seized at the border, taken from you in the street, or burgled from your house and copied in seconds. Unfortunately, locking your device with passwords, PINs, or gestures may not protect your data if the device itself is seized. It’s relatively easy to bypass such locks because your data is stored in an easily-readable form within the device. An adversary would just need to access the storage directly in order to copy or examine your data without your password.[8]

- If you use encryption, your adversary needs both your device and your password to unscramble the encrypted data. Therefore, it's safest to encrypt all of your data, not just a few folders. Most smartphones and computers offer complete, full-disk encryption as an option.

- While encrypt and decrypt operations are performed on small files, CPU power may be insufficient for larger files, while processing power may be insufficient in terms of speed. Therefore, the GPU can be used to speed up this processing time.

- CUDA technology allows parallel programming because the CPU is suitable for parallel programming due to its architecture. This parallelism allows us to perform many operations at the same time and gives us great speeds when encrypt and decrypt large files.

3.1.2 Glossary

- CUDA- Compute Unified Device Architecture
- AES- Advanced Encryption Standard(Encryption algorithm)
- ENC-Encryption
- DEC- Decrption
- CPU- Central Processing Unit
- GPU- Graphics Processing Unit

3.1.3 Overview of Document

More detailed information of the rest of the content is clarified in the below sections. Section 2.1. is the "Design Approach" which contains information about the development methodology of the project.

Section 2.2 contains information about which technologies and which encryption algorithms our project will have and how to use them.

Section 2.3 contains the activity diagram of the project that represents the workflows of activities and actions.

3.2 Architecture Design

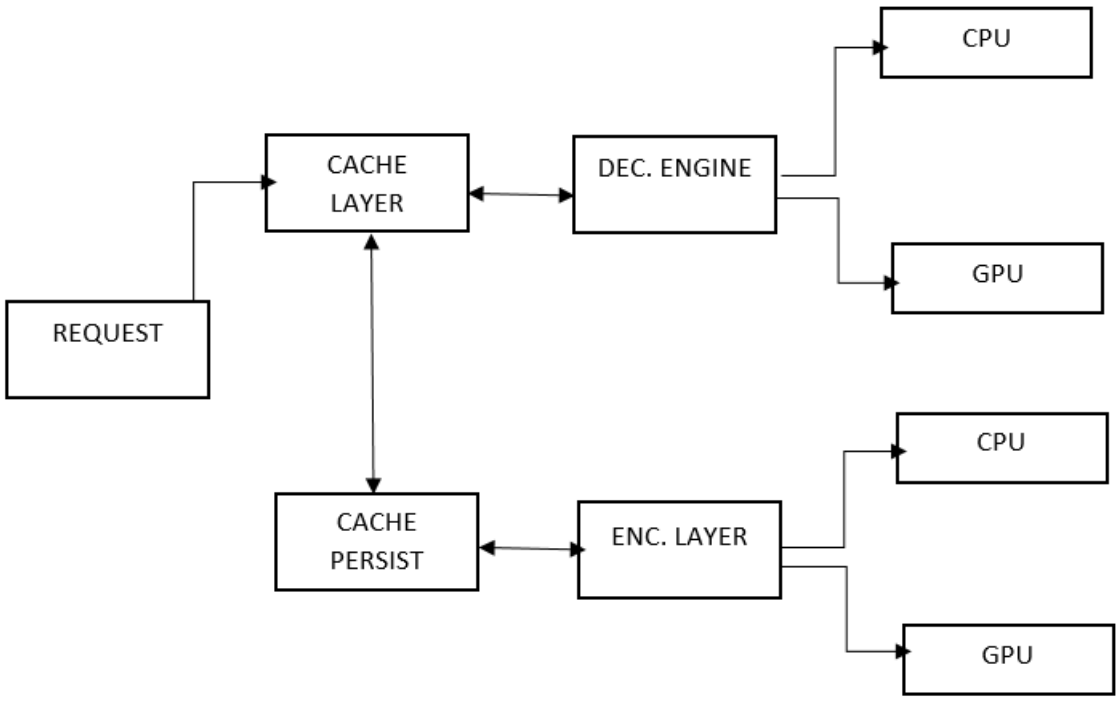
3.2.1.1 Design Approach

This figure allows us to illustrate the overall life cycle of the system.

Cache is the memory page, area of the operating system. Normally all data is encrypted in the file, we decrypt it and put it in the cache. Every request to the system comes to the cache layer first. Read requests from the cache layer must first look at the cache. If the requested read request is found in the cache, the request is answered through the request. If it cannot be found, the blocks containing the beginning and end of it are retrieved from decrypt. For example, we will read the byte array in block 2. At the beginning and end of block 2, if 4 blocks are grouped, we send them to the GPU, and then decrypt and cache them. We need to balance the cache while doing these operations, so we need to inflate the cache. Because at the end of all these files will not be in the cache. The search is constantly cycling through the cachet, so read operations proceed from top to bottom in a linear manner.

Persist's task is to send a request for writing, look at the cache layer, encrypt the 4-block blocks, open a temporary temp file, and make changes to the temp file. The read will be provided from the main file as the user makes the changes. When the last operations are finished, it will take what the user reads and encrypt the temp file and write the blocks asynchronously at run time. Thus, the user will be able to encrypt all of the related things again because of a page he has changed, but they will continue asynchronously while the user is running.

3.2.1.2 Encryption/ Decryption Engine Layer Design

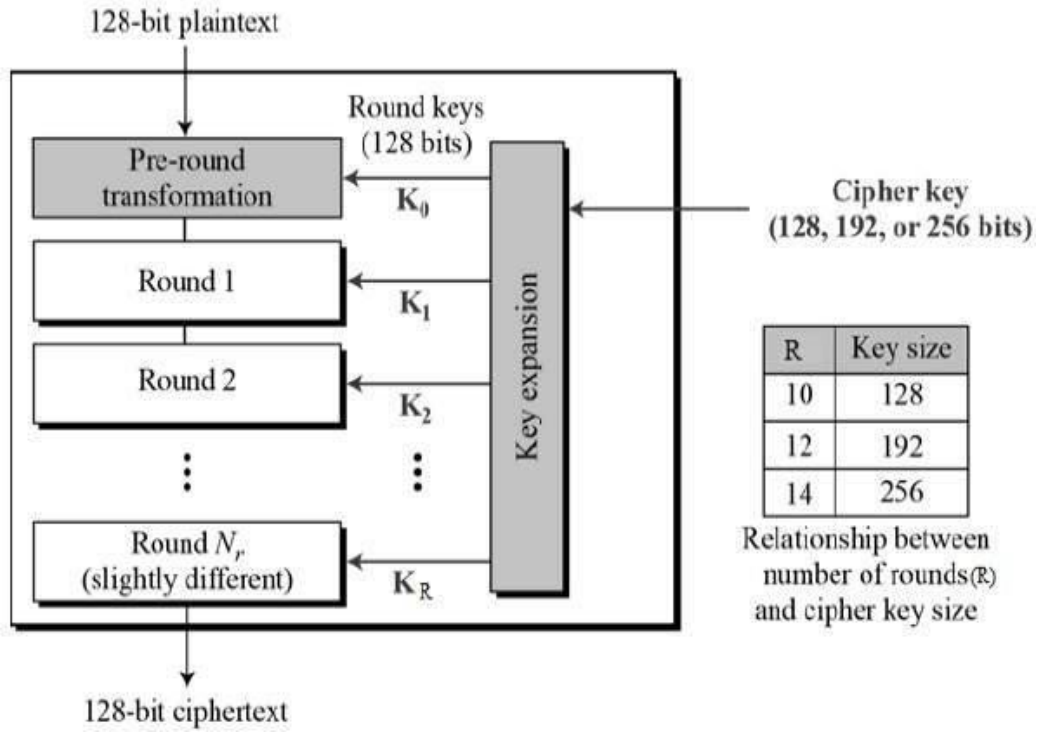


3.2.1.3 Encryption Layer

- AES is an iterative rather than Feistel cipher. It is based on 'substitution–permutation network'. It comprises of a series of linked operations, some of which involve replacing inputs by specific outputs (substitutions) and others involve shuffling bits around (permutations).

- Interestingly, AES performs all its computations on bytes rather than bits. Hence, AES treats the 128 bits of a plaintext block as 16 bytes. These 16 bytes are arranged in four columns and four rows for processing as a matrix −

Unlike DES, the number of rounds in AES is variable and depends on the length of the key. AES uses 10 rounds for 128-bit keys, 12 rounds for 192-bit keys and 14 rounds for 256-bit keys. Each of these rounds uses a different 128-bit round key, which is calculated from the original AES key.[9]



R	Key size
10	128
12	192
14	256

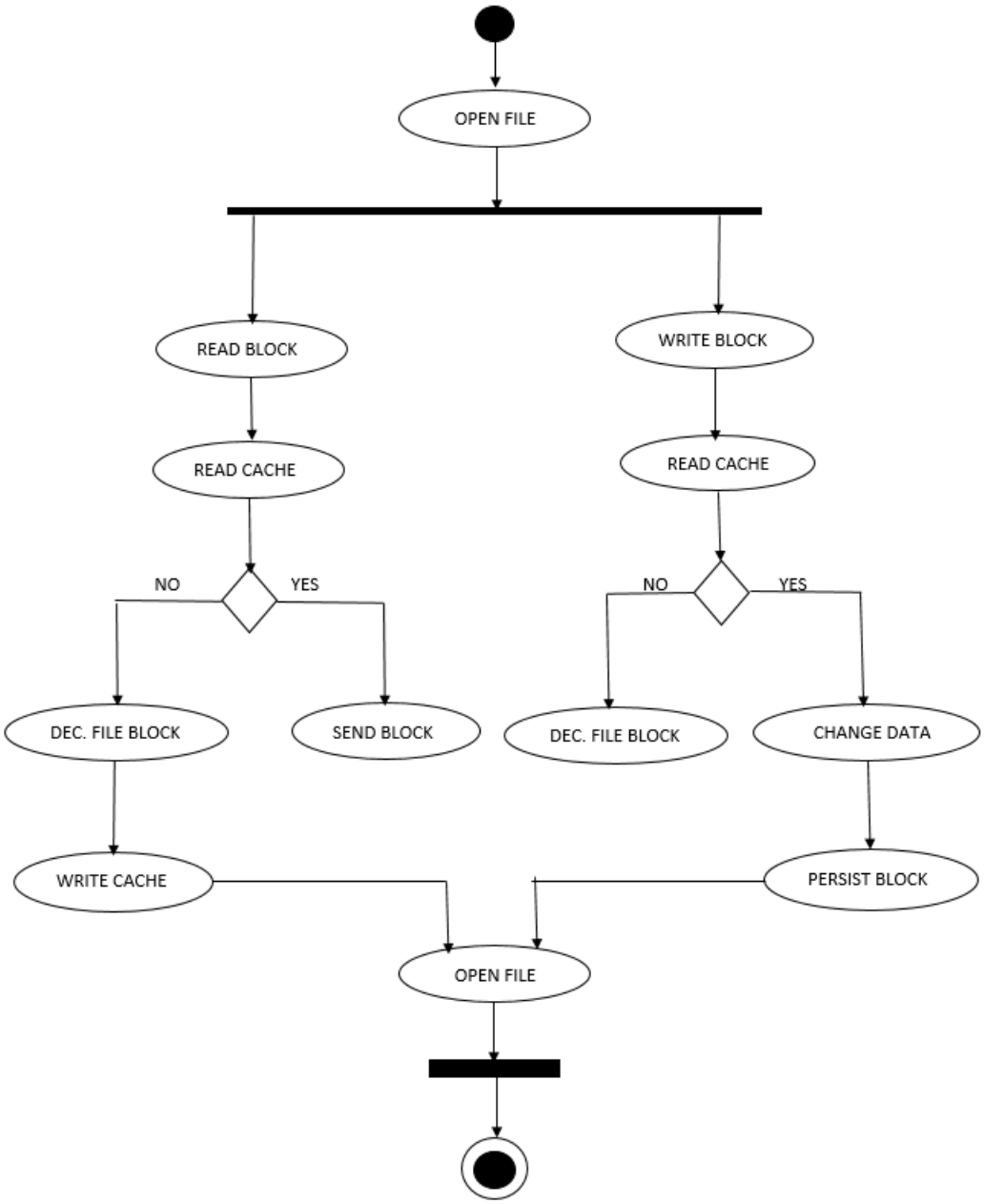
Relationship between number of rounds(R) and cipher key size

3.2.2 Architecture Design

The purpose of the system is file encryption. In doing so, we plan to encrypt files of 512, 1024, or 2048 bits using the AES encryption algorithm, which is known to be the most commonly used symmetric and block encryption. Using CUDA as an add-on for NVIDIA's C programming language for GPU / CPU, we increase the speed of file encryption and reduce the cost. If the files are small in size, we benefit from the CPU and the GPU if they are large. Using fopen (), files are initialized first. Files are saved to cache after they are encrypted. To read files, read block takes blocks from the cache. The files here are encrypted so we can

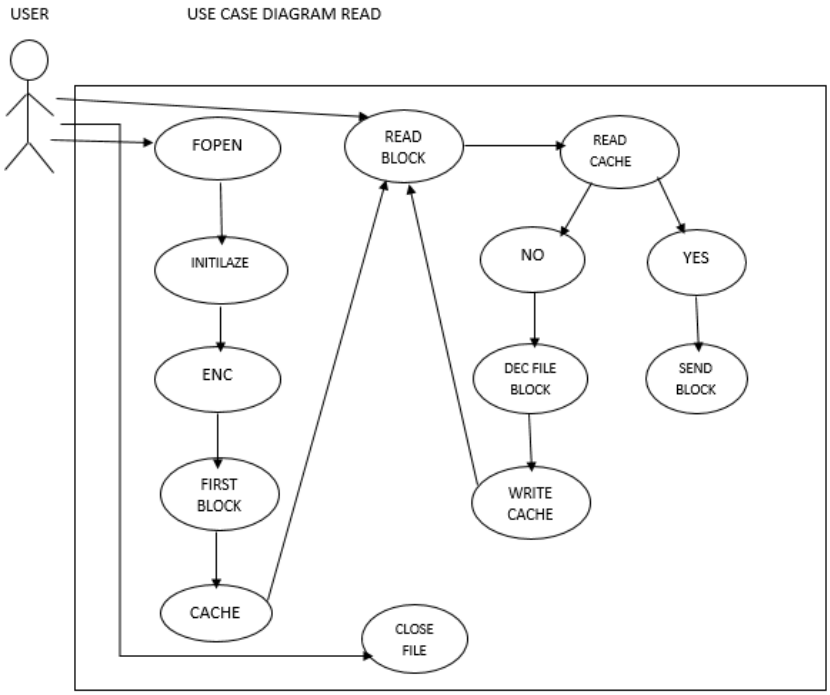
perform decryption and access them. To write files, the write block takes blocks from the cache. The Decryption process is applied, and then the persist block updates the file

3.2.3 Activity Diagram

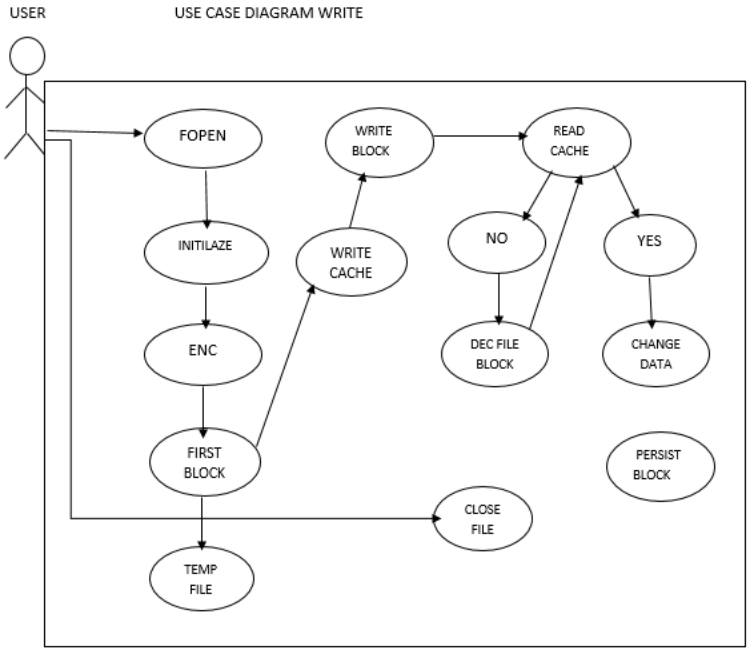


3.3 Use Case Realization

3.3.1 Use Case Read Diagram



3.3.2 Use Case Write Diagram



4.Future Plan

In the next stages of our project, we aim to create a library about encryption. We aim to develop the implementation leg of our project by using C programming language in linux operating system.

In this project we aim to encrypt user-supported files using CUDA technology by using AES. CUDA technology will benefit us in terms of speed thanks to the advantage of parallel operation in encryption of large files. This way, we will be able to encrypt the file in larger blocks faster.

If our development process proceeds quickly, we will open the library that we intend to create in future stages for the development and use of people. In addition, we will be able to make the encrypt file available on operating systems with our own extension.

5. References

[1] https://nezihesozen.github.io/mydoc/cuda2

[2] https://www.binance.vision/security/what-is-symmetric-key-cryptography

[3] https://www.garykessler.net/library/crypto.html

[4] https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm

[5] https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm

[6] https://www.kaspersky.com/resource-center/definitions/brute-force-attack

[7] https://www.oksbdc.org/why-is-technology-important-in-business/

[8] https://mainstreetpractitioner.org/tech-topics/keeping-your-data-safe/

[9] https://www.tutorialspoint.com/cryptography/advanced_encryption_standard.htm