A Parallel Implementation of the Advanced Encryption Standard (AES)

Gurupungav Narayanan, 16CO114
Computer Engineering
National Institute of Technology
Surathkal, Karnataka 575025

Nihal Haneef, 16CO128
Computer Engineering
National Institute of Technology
Surathkal, Karnataka 575025

Rishika Narayanan, 16CO241
Computer Engineering
National Institute of Technology
Surathkal, Karnataka 575025

I. Problem Statement

On cloud environments, where private data of a large number of users need to be protected, the Standard Advanced Encryption Standard (AES) algorithm is inefficient. In such large environments, the encryption must be done as fast as possible. A sequential AES algorithm for a massive number of users would take too long for such time sensitive applications.

Implement a parallelised Advanced Encryption Standard that would be efficient for large use cases

II. Execution Plan

The primary paper we are implementing is Fei et.al 's *Practical parallel AES algorithms on cloud for massive users and their performance evaluation*. The paper discusses six algorithms, three GPU based and three CPU based, for parallising AES. The most efficient algorithm (GCS, which stands for Coalescent and Sliced GPU) coalesces all the user data and slices them into equal length. Every 16B in this coalesced sliced is assigned to a GPU thread for encryption. The two other GPU variations are GNCS (Coalesced but not sliced GPU) and GNC (Unocoalesced and Unsliced GPU). The three CPU algorithms are the direct CPU equivalents of the GPU algorithms, using CPU core threads instead of the GPU threads. The CPU algorithms are expected to be slower because of the limited number of cores and threads in the CPU.

TABLE I
Six parallel algorithms

Name	Parallel Scope	Coalesced?	Sliced?
GCS	User data on GPU	Yes	Yes
GCNS	User data on GPU	Yes	No
GNC	One user data after another on GPU	No	No
CCS	User data on CPU	Yes	Yes
CCNS	User data on CPU	Yes	No
CNC	One user data after another on CPU	No	No

We intend to implement these six algorithms. The three GPU algorithms would be implemented in CUDA C/C++ using Thrust libraries if necessary. The CPU algorithms would be implemented using OpenMP. Apart from these, we would also require programs for generating inputs and keys, along with frameworks for analyzing the performance of the implementations.

TABLE II
Execution Plan

Frameworks	CUDA, OpenMP in C++
Inputs	Message Text and Keys of N users in separate files
Outputs	Cipher Text for every Message
Metrics	Speedup (w.r.t sequential) per algorithm for different N

We intend to run the analysis on at least two different platforms to ensure generalization.

III. Proposed Timeline

Date	Milestone
October 26	Project Proposal
October 29	Data Generator, GCS, CCS, Sequential
November 5	GCNS, CCNS, GNC, CNC
November 10	Complete Analysis
November 15	Final Project Report

IV. Work Distribution

We intend to split the algorithms equally per person (2 per person). The algorithms are similar in structure so it shouldn't be hard to convert it from one library to another.

V. Current Progress

We have initialised our github repository https://github.com/gurupunskill/parallel-aes. It has basic incomplete encryption code for AES. We need to improve on this and create a more solid importable package.