Optimasi Algoritma Enkripsi AES Menggunakan OpenCL Pada GPU

Alif Raditya Rochman
Teknik Informatika
Institut Teknologi Bandung
Bandung, Indonesia
13511013@std.stei.itb.ac.id

***Abstract*—Pemanfaatan GPU untuk general purpose computing menjadi semakin marak setelah peningkatan performa yang ada dari arsitektur GPU itu sendiri. Pemanfaatan ini dapat meningkatkan performa dari algoritma yang sering dipergunakan, terutama yang pada data yang cukup besar. Algoritma AES sebagai salah satu algoritma kriptografi yang saat ini popular digunakan oleh berbagai pihak berpotensi untuk ditingkatkan performanya dengan memanfaatkan komputasi paralel dengan GPU. Hasilnya yaitu peningkatan yang cukup signifikan apabila dibandingkan dengan menggunakan CPU saja.**

Keywords—cryptografy; AES; OpenCL; paralel; GPU;

I. PENDAHULUAN

Unutuk melakukan komputasi umum, komputer cenderung memanfaatkan *Central Processing Unit* (CPU). Perkembangan CPU saat ini mengarah kepada multi-core sehingga membuka peluang yang lebih baik untuk memanfaatkan teknologi komputasi paralel. Tetapi dalam segi jumlah cores, CPU bukanlah hardware yang paling banyak memiliki core unit.

Graphic Processor Unit (GPU) merupakan hardware yang umum digunakan untuk melakukan proses rendering dan proses analisa citra dalam komputer. Walaupun memiliki fungsi utama rendering, GPU memiliki unit komputasinya sendiri yang berbeda dengan unit komputasi CPU. GPU cenderung memiliki jumlah core yang jauh lebih banyak dibandingkan CPU.

Pemanfaatan GPU sebagai unit komputasi untuk aplikasi umum yang biasanya menggunakan CPU menjadi marak digunakan. Hal ini dikenal sebagai *general purpose computing on graphics processing unit* (GPGPU). Teknologi ini sering digunakan pada aplikasi yang membutuhkan optimasi dengan memanfaatkan komputasi paralel.

Teknologi enkripsi simetris yang popular digunakan yaitu AES. Penggunannya sering dilakukan pada sistem yang memiliki sistem yang memiliki GPU disamping CPU (terutama pada personal computer). Sehingga apabila ada peningkatan performa yang signifikan hal ini tentunya dapat membantu dari para pengguna teknologi ini.

Algoritma yang akan dibandingkan disini yaitu dibatasi pada algoritma AES dengan mode enkripsi CTR dengan 256 key. Pemanfaatan komputasi paralel dari GPU dan CPU dilakukan dengan menggunakan OpenCL.

Salah satu hal menarik yang akan kita analisis dalam makalah ini yaitu bagaimana perbandingan performa CPU dan GPU pada algoritma AES pada jumlah input dari kecil sampai besar. Terutama berapa pesar peningkatan yang terjadi ketika jumlah input terbilang cukup besar.

II. DASAR TEORI

A. Enkripsi AES

Setelah algoritma DES dianggap tidak aman, banyak algoritma kriptografi baru yang muncul ke permukaan. Salah satunya yaitu algoritma AES yang diperkenalkan oleh Vincent Rijmen dan Joan Daemen dengan salah satu nama lainnya yaitu algoritma Rijdndael. AES menjadi algoritma kriptografi yang paling popular sejak dijadikan sebagai standard oleh pemerintahan amerika serikat.

Algoritma AES merupakan algoritma yang menggunakan key simetri, yang artinya untuk melakukan enkripsi dan dekripsi diperlukan *key* yang sama. Berdasarkan ukuran blocksize yang digunakan, terdapat tiga jenis algoritma AES, yaitu AES-128, AES-192, dan AES-256. AES menggunakan teknik substitusi dan permutasi pada sebuah matriks berukuran 4x4 untuk mengubah *plaintext* menjadi *ciphertext*, Terdapat 14 babak dari kombinasi operasi matriks untuk AES-256.[1]

Secara garis besar, algoritma ini memiliki proses sebagai berikut :

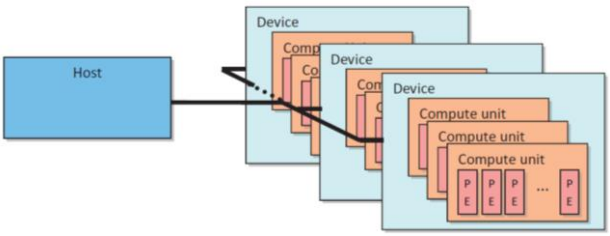
1. Ekspansi Kunci
2. Babak Awal
   a. *AddRoundKey* : melakukan XOR antara state awal (plainteks) dengan cipher key
3. Proses utama setiap babak
   a. *SubBytes* : substitusi byte dengan menggunakan tabel substitusi (S-box)
   b. *ShiftRows* : pergeseran baris-baris array state secara sirkular.
   c. *MixColumns* : mengacak data di masing-masing kolom array state
   d. AddRoundKey
4. Babak Akhir
   a. SubBytes
   b. ShiftRows

c. AddRoundKey

Proses enkripsi dan dekripsi pada algoritma ini tidak terlalu berbeda proses utamanya. Proses *SubBytes* pada dekripsi digantikan dengan *Inverse SubBytes*.

B. *OpenCL*

OpenCL (Open Computing Language) merupakan sebuah *framework* yang digunakan untuk mengembangkan aplikasi yang dieksekusi secara paralel. Spesifikasi dari OpenCL didefinisikan menjadi 4 bagian : Model platform, Model eksekusi, Model memori, dan model pemrograman. Model platform OpenCL mendefinisikan peran dari host dan device serta menyediakan abstraksi model *hardware* pada *device*. Pada model platform, terdapat sebuah host yang mengkoordinasikan eksekusi antara satu atau lebih *device*. Sebuah aplikasi OpenCL diimplementasi sebagai *host code* dan *kernel code*.[2]



Gambar 1. Model platform pada OpenCL

Terdapat dua jenis memori pada OpenCL : *Host Memory* dan *Device Memory*. *Device Memory* terbagi menjadi empat address space yaitu :

1. *Global memory*

Global memory dapat dilihat oleh seluruh *compute units* pada *device*. Setiap kali terdapat transfer data dari *host* dan *device*, data akan disimpan dalam *global memory*. Setiap data yang akan dikirimkan dari *device* ke *host* juga harus disimpan ke global memory.

2. *Local memory*

Local memory merupakan sebuah memory yang memiliki *address space* yang unik terhadap setiap *compute device*. *Local memory* dimodelkan untuk dapat di-*share* pada *workgroup* yang sama. Sehingga akses ke memori ini membutuhkan *latency* yang sedikit tetapi memiliki *bandwidth* yang lebih besar dari *global memory*.

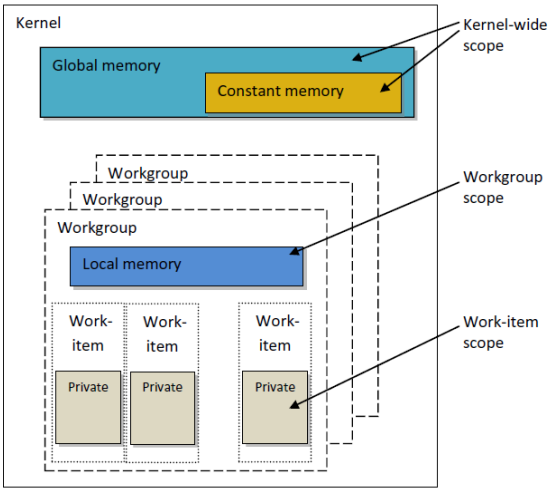
3. *Constant memory*

Constant memory merupakan bagian dari *global memory* yang bernilai konstan. Walaupun begitu, *constant memory* tidak selalu didesain untuk setiap data yang memiliki attribut *read-only*.

4. Private Memory

Private memory merupakan memory yang unik terhadap setiap *work-item*. Variabel lokal dan argumen non-pointer pada kernel merupakan private secara default.

Setiap instance dari kernel yang berjalan pada *compute unit* disebut *work-item*. *Work-item* dipetakan pada sebuah ruang indeks berdimensi n yang disebut *NDRange*. GPU menjadwalkan *works-item* pada processing elements sampai semua *work-item* telah di proses. [3]



Gambar 2. Model memori pada OpenCL

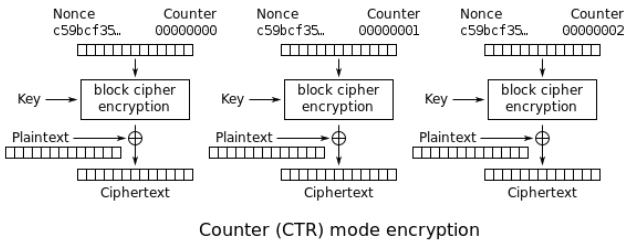
Work-group ditugaskan pada *compute unit*. Seluruh *work-item* pada *work-group* hanya bisa diproses oleh *processing elements* pada sebuah *compute unit* yang telah ditentukan. Sebuah *processing elements* hanya dapat memproses satu *work-item* pada satu waktu, tetapi sebuah *compute unit* dapat memproses lebih dari satu *work-group*.

III. STRATEGI ALGORIMA PARALEL

Algoritma AES bersifat sekuensial secara alami dikarenakan setiap proses pada tahapannya bergantung pada tahapan sebelumnya, sehingga sulit untuk melakukan optimasi secara keseluruhan. [4]

Sehingga untuk melakukan optimasi yang baik, yang perlu dilakukan optimasi yaitu pada proses pembetukan block cipher secara parallel. Pada makalah ini kita akan membuat mode enkripsi counter (CTR) secara paralel.[5]

Pada dasarnya, CTR mengubah blok cipher menjadi stream cipher. Ini menghasilkan keystream blok berikutnya dengan mengenkripsi nilai dari "counter". Counter dapat berupa fungsi yang menghasilkan urutan yang dijamin tidak berulang untuk waktu yang lama, walaupun sebenarnya increment counter merupakan alternatif yang paling sederhana dan populer



Gambar 3. Mode enrkipsi CTR

IV. HASIL PENGUJIAN DAN ANALISIS

Setelah implementasi telah dilakukan dengan baik, waktu eksekusi akan dicatat menggunakan OpenCL API yang telah disediakan. Sehingga waktu yang dicatat berasal dari waktu eksekusi pada *device*. Pengujian dilakukan dengan menguji program yang dibuat dengan beberapa ukuran data uji dari 2^{-16} sampai 2^{9} MiB.

A. Lingkungan Pengujian

Dalam pengujian ini digunakan perangkat keras dan perangkat lunak dengan spesifikasi sebagai berikut :

1. Perangkat keras
   a. CPU : Intel Core i5 4460
   b. GPU : AMD Radeon R9 270X
   c. RAM : DDR3 8GB
2. Perangkat lunak
   a. Sistem Operasi : Windows 8.1
   b. Versi AMD APP SDK : 2.9.1
   c. Versi OpenCL : 1.2
3. Informasi perangkat GPU
   a. Ukuran *local memory* : 32 KB
   b. Ukuran *global memory* : 2 GB
   c. Jumlah *work-group* maksimal : 256
   d. Jumlah *compute-unit* : 20
   e. *Clock frequency* : 1080 Mhz
4. Informasi perangkat CPU
   a. Ukuran *local memory* : 32 KB
   b. Ukuran *global memory* : 8 GB
   c. Jumlah *work-group* maksimal : 1024
   d. Jumlah *compute-unit* : 4
   e. *Clock frequency* : 3200 Mhz

B. Hasil Pengujian

Pengujian dilakukan dengan mencatat waktu eksekusi dari proses enkripsi dan proses dekripsi pada AES. Pengujian dilakukan dengan menggunakan GPU sebagai *device* pada OpenCL lalu menggunakan CPU sebagai *device* pada OpenCL. Sehingga didapatkan dua buah tabel yaitu tabel 1 dan tabel 2 seperti dibawah ini.

Ukuran (MiB)	Waktu Enkripsi (s)	Waktu Dekripsi (s)
0.00001526	0.0027	0.0027
0.00003052	0.0028	0.0027
0.00006104	0.0026	0.0029
0.00012207	0.0027	0.0027
0.00024414	0.0027	0.0027
0.00048828	0.0027	0.0026
0.00097656	0.0027	0.0027
0.00195313	0.0027	0.0027
0.00390625	0.0027	0.0027
0.00781250	0.0027	0.0027
0.01562500	0.0027	0.0027
0.03125000	0.0027	0.0027
0.06250000	0.0029	0.0027
0.12500000	0.0031	0.0031
0.25000000	0.0053	0.0054
0.50000000	0.0082	0.008
1.00000000	0.0132	0.0132
2.00000000	0.0245	0.0252
4.00000000	0.0463	0.0463
8.00000000	0.09	0.09
16.0000000	0.1772	0.1772
32.0000000	0.3517	0.3516
64.0000000	0.7005	0.7006
128.000000	1.3993	1.399
256.000000	2.7968	2.7973
512.000000	5.7639	6.7569

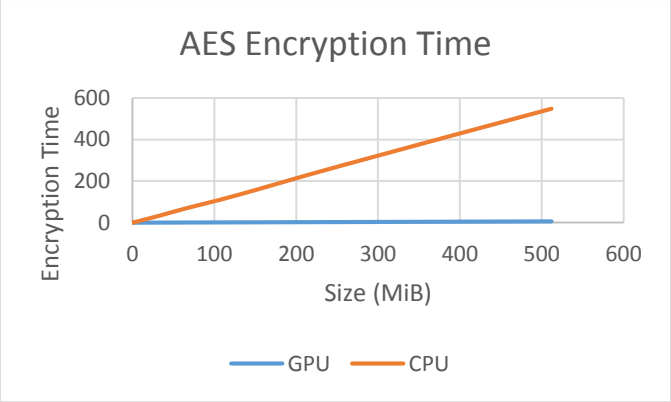
Tabel 1. Performa AES pada GPU.

Ukuran (MiB)	Waktu Enkripsi (s)	Waktu Dekripsi (s)
0.00001526	0.0132	0.0105
0.00003052	0.0106	0.011
0.00006104	0.011	0.0104
0.00012207	0.0074	0.0058
0.00024414	0.0059	0.0059
0.00048828	0.0063	0.0061
0.00097656	0.0072	0.007
0.00195313	0.0085	0.0084
0.00390625	0.0117	0.0115
0.00781250	0.0167	0.0156
0.01562500	0.0221	0.018
0.03125000	0.035	0.0341
0.06250000	0.0661	0.0649
0.12500000	0.1283	0.125
0.25000000	0.2524	0.2589
0.50000000	0.5334	0.5115
1.00000000	1.0102	1.0113
2.00000000	1.9886	1.9872
4.00000000	3.9484	3.9782
8.00000000	8.1258	8.2453

16.0000000	16.375	16.4554
32.0000000	32.644	34.8402
64.0000000	67.5085	66.2877
128.000000	132.5742	150.1931
256.000000	275.3811	268.5858
512.000000	548.1794	565.8992

Tabel 2. Performa AES pada CPU.

Dari data yang diambil, dapat dilihat bahwa pada setiap ujicoba, waktu enkripsi dan waktu dekripsi dengan menggunakan GPU memiliki waktu eksekusi yang lebih cepat dibandingkan dengan menggunakan CPU. Besar peningkatan yang terjadi dapat dilihat seperti pada gambar 4 dibawah.



Gambar 4. Perbandingan waktu enkripsi pada GPU dan CPU

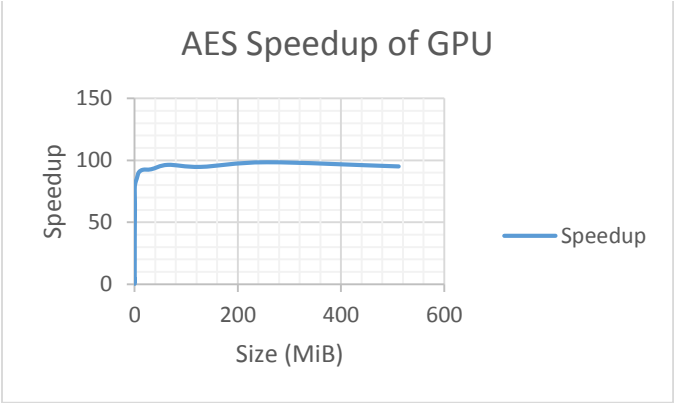
Waktu enkripsi pada CPU dan GPU berbanding lurus dengan ukuran data uji. Walaupun begitu perbandingan antara keduanya cukup besar, dan terlihat semakin besar untuk ukuran data yang lebih besar.

Hasil yang sama juga dapat dilihat pada waktu dekripsi yang ada. Hal ini dikarenakan perbedaan antara proses enkripsi dan dekripsi tidaklah terlalu jauh, walaupun proses dekripsi membutuhkan waktu yang relatif lebih lama dibandingkan dengan waktu enkripsi.

Untuk melihat peningkatan performa yang ada, kita akan menghitung *speedup* yang dihasilkan dengan menggunakan rumus :

$$S = \frac{T_{old}}{T_{new}}$$

Dimana S adalah *speedup* yang dihasilkan, T_{old} merupakan waktu enkripsi pada CPU, dan T_{new} merupakan waktu enkripsi pada GPU. Hasilnya dapat dilihat pada gambar 5.



Gambar 5. Perbandingan waktu enkripsi pada GPU dan CPU

Speedup yang dihasilkan mencapai nilai 90. Nilai ini merupakan nilai yang besar. *Speedup* dengan nilai 90 dicapai mulai data uji berukuran 1 MiB. Walaupun begitu, speedup yang dihasilkan tidak mengalami peningkatan lebih lanjut untuk data uji yang lebih besar.

C. *Analisis*

Perbedaan performa yang cukup besar disebabkan oleh GPU yang digunakan memiliki jumlah *core* yang jauh lebih banyak dibandingkan CPU. Proses setiap babak pada algoritma AES dengan memanfaatkan CTR dapat meningkatkan optimasi yang dapat dihasilkan oleh *core* yang lebih banyak. Walaupun memiliki *clock frequency* yang lebih lemah, tetapi karena jenis komputasi yang diperlukan tidak terlalu berat, maka performa yang dihasilkan menjadi cukup besar pada GPU.

V. KESIMPULAN

Pemanfaatan komputasi paralel dengan menggunakan GPU menghasilkan peningkatan performa yang signifikan terhadap algoritma kriptografi AES. Proses enkripsi dan proses dekripsi akan menjadi lebih cepat dibandingkan dengan menggunakan CPU. *Speedup* yang dihasilkan dapat mencapai nilai 90 pada data uji yang cukup besar.

DAFTAR PUSTAKA

[1] Katz, Jonathan; Lindell, Yehuda. (2014). Introduction to Modern Cryptography, Second Edition. London : CRC Press.

[2] Advanced Micro *Devices*, Incorporated. (2014). *AMD Accelerated Parallel Processing OpenCL™ User Guide*. Sunnyvale, CA: Advanced Micro *Devices*, Incorporated.

[3] Khronos OpenCL, Working Group. (2014). *OpenCL 2.0 API Specification*. Document Revision: 26, Editor: Aaftab Munshi.

[4] Deguang Le; Jinyi Chang; Xingdou Gou; Ankang Zhang; Conglan Lu; , "Parallel AES algorithm for fast Data Encryption on GPU," Computer Engineering and Technology (ICCET), 2010 2nd International Conference on , vol.6, no., pp.V6-1-V6-6, 16-18 April 2010

[5] Nhat-Phuong Tran, Myungho Lee, Sugwon Hong, Seung-Jae Lee, "Parallel Execution of AES-CTR Algorithm Using Extended Block Size", CSE, 2011, 2013 IEEE 16th International Conference on Computational Science and Engineering, 2013 IEEE 16th International Conference on Computational Science and Engineering 2011, pp. 191-198,doi:10.1109/CSE.2011.43