

L-010

GPGPU を用いた暗号攻撃

Attack against cipher algorithm using CUDA

西川 尚紀†
Naoki Nishikawa

岩井 啓輔†
Keisuke Iwai

黒川 恭一†
Takakazu Kurokawa

1 はじめに

GPU(Graphics Processing Unit)とは、グラフィックス処理に特化した半導体チップである。2005年ころからGPUの並列演算能力を汎用計算(GPGPU:Generally Purposed computation on GPU)に応用する研究が広がりを見せ、成果を挙げた。更に、最近の世代のGPUからは整数・論理演算命令もサポートされたことにより、暗号処理への応用も可能になった。

本稿は、GPUを用いた暗号処理の高速化の一例として、暗号アルゴリズムへのパスワードクラックを実装し、その成果を元に同手法の可能性を検討したものである。攻撃対象とする暗号部分はモジュール化し、複数の暗号に対応できる。今回はDESとAESを攻撃対象とした。

2 パスワードクラック

一般的に行われるパスワードクラックとして、Brute Force Attackや辞書攻撃等があるが、本稿では、最も一般的な方法であるBrute Force Attackを図1に示すようにGPU実装した。

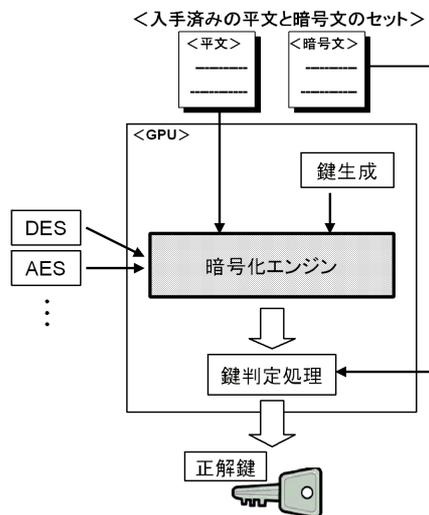


図 1: Concept of Key Crack using GPU

3 DES と AES

3.1 DES

DES(Data Encryption Standard)は1976年に決定された米国政府標準の共通鍵ブロック暗号である。鍵

長は56ビットで、ラウンドと呼ばれる暗号化ステップを繰り返すFeistel構造をとっている。DESでは、ラウンド処理を16回行う。各ラウンド処理では64ビットの平文データが入力され、入力は左半分32ビットと右半分32ビットに2等分され、別々に処理される。

3.2 AES

AES(Advanced Encryption Standard)は、安全性が低下したDESに代わる標準暗号規格として、1997年、米国NISTが世界中から公募し制定された米国政府標準の共通鍵ブロック暗号である。AESは、16バイトのデータを1単位として暗号化/復号を行うアルゴリズムであり、SubBytes, ShiftRows, MixColumns, AddRoundKeyの4つの処理が繰り返し行われることにより暗号化される。

4 GPU 統合開発環境 CUDA

CUDAではGPUを複数のマルチプロセッサから成る並列計算アーキテクチャとして扱う。MPには8個のストリームプロセッサ(SP)、1基の命令ユニット、複数本のレジスタ、1基の共有メモリが内蔵されている。命令ユニットはSPで共有される。

MP内には、SPが使用するレジスタと、SP間で共有する共有メモリがある。レジスタはスレッド毎に割り当てられ、他のスレッドからは参照できない。共有メモリはレジスタ並みの高速アクセスが可能だが、容量が少ない。GMは大容量だが、メモリアクセスに遅延が生じる。共有メモリはグローバルメモリからデータを読み出す際のキャッシュとして用いられることが多い。今回用いたGPUであるNVIDIA Geforce GTX 285ではMP30個及びグローバルメモリ1GBを搭載している。

CUDAでは、CPUからGPUに発行されたジョブは、1~512スレッドから成るスレッドブロックと呼ばれる単位でMPに割り当てられる。更にMP内では、32スレッドから成るワーブと呼ばれる単位で処理が実行される。ワーブ内で条件分岐が発生した場合、速度が低下するので、ワーブ内で条件分岐が発生しないよう、工夫する必要がある。

5 実装

今回実装したパスワードクラックの基本的な実行フローは以下の通りとした。

1. 入手済である1組の平文と暗号文のセットとその他暗号化に必要なデータをGPUに転送
2. 暗号化に使用する鍵をGPU内部で生成
3. 平文を暗号化
4. 鍵判定処理を行う

† 防衛大学校 情報工学科

5. 完成した暗号文と入手済みの平文が一致した場合に限り、用いた鍵のみをリードバック

どのデータを1スレッドで処理させるかは、スレッドによる並列処理の効率を決定するために重要である。DESでは64ビットの平文に対しビット単位で暗号化/復号を行うため、本来は平文を64ビット領域に格納し、1ビット1スレッドで処理させたい。しかし、1ビット1スレッドだとメモリアクセスの際に並列処理できない。そのため、各ビットは1バイト領域に格納し、1スレッドで1バイト領域に格納されたビットを処理させる。AESでは128ビットの平文に対しバイト単位で暗号化/復号を行うため、平文はそのまま128ビット領域に格納し、1バイト1スレッドで処理させた。

SPのパイプラインレイテンシを隠蔽するには最低6ワーブ起動するのが良いとされているため[3]、共有メモリの容量を考慮して、スレッドブロック1つ当たりのスレッド数は512に決定した。同条件だとスレッドブロックが2つ起動し、MPの最大アクティブスレッド数1024が保証され、MPを最大効率で使用できる。

GPUで鍵を内部生成した理由は、CUDAの仕様では1度のジョブ発行で 2^{35} 個の鍵を生成して同鍵空間をチェックできるのに対し、メモリには 2^{23} 個程度しか蓄積できないためであり、またCPU~GPU間のデータ通信を極力低減させたいためでもある。鍵生成はスレッド並列で高速処理するよう工夫し、全体の計算時間に対し鍵生成が占める割合はDESで実験したところ4%~35%から0.1%~1.2%程度となり、GPU内部での鍵生成によるオーバーヘッドが極力回避できていることがわかる。

GPU内部での鍵判定処理も鍵生成と同じ理由からGPU内部で処理した。鍵判定部分の処理時間は全体の実行時間と比較してDESでは0.1%~2.5%程度と、GPU内部での判定処理が負担にならないことがわかった。

CUDAではワーブ(=32スレッド)を1単位として処理を行う。しかし、DESやAESでは、56ビットの転置等、ワーブ内での条件分岐を発生させる処理が存在する。そのため、56ビットの転置表であれば64ビットの共有メモリ領域に多少冗長に格納し、条件分岐を避けるよう工夫した。

6 性能評価

CUDAによる検索速度を評価するためにCPUによる実装と比較する。用いたマシンの仕様を表1に、それぞれ実装したプログラムの実行時間のグラフを図2に示す。グラフの横軸は鍵空間、縦軸は計算時間を対数で表している。ここではDESのみを示す。 2^{35} の鍵空間に対し、GPUでは約7.45時間、CPUでは約67.8時間と、約9.1倍の速度向上を示した。また、我々は、2009年3月に行われたマルチプログラミングコンテストGPU Challenge 2009でAESを実装しTesla-C1060上で 2^{14} 個の平文ブロックに対し2.933msの処理時間を達成している。従って、 2^{35} 個の平文をAES処理するのにかかる時間はおよそ6150sであることが予想できる。一方、 2^{35} 個の鍵空間の場合の暗号処理を除いた時間は1073sであった。従って、暗号化モジュールをAESに載せ換えた場合には、 2^{35} 個の平文に対し、およそ1.9時間でキークラックできると見積もられる。

表 1: Spec of computer

CPU	Corei7 Quad7-920(2.66GHz)
Memory	6GB
OS	CentOS5.2 (カーネル ver2.6.18)
Compiler	gcc ver4.1.2(option -O3)
GPU Accelerator	NVIDIA Geforce GTX 285
GPU Memory	1GB
CUDA Compiler	nvcc ver2.1

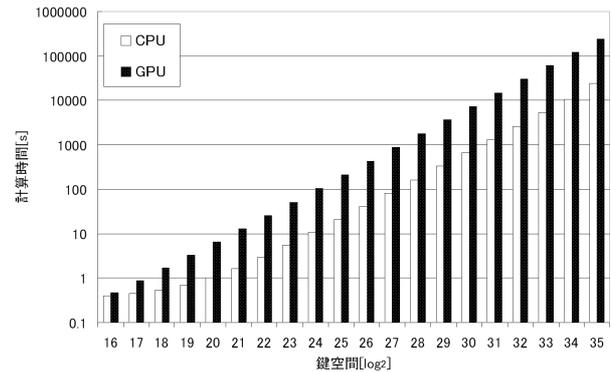


図 2: Comparison of Calculation Time

7 結論

本稿では、GPGPUによるパスワードクラックの可能性の検討を行った。 2^{35} 乗個の鍵空間に対し、DESではGeforce GTX 285を用いて約7.45時間、AESではTesla-C1060を用いて約1.9時間で探索可能であることを示した。また、DESの場合、Intel Core i7-920 2.66GHzに対し、約9倍の速度向上を示した。

本研究の発展としては、複数のGPUを使用してパフォーマンスを向上させることが考えられる。渡辺らは、データ転送に対して計算時間の割合が高い場合には、複数GPUを用いて性能を発揮しやすいという結論を示している[4]。本稿による実装はこの条件に合致しているため、複数GPUで期待通りの性能が引き出せるアプリケーションであると考えられる。

参考文献

- [1] Svetlin A.Manavski, "CUDA COMPATIBLE GPU AS AN EFFICIENT HARDWARE ACCELERATOR FOR AES CRYPTOGRAPHY", ICSPC (2007).
- [2] Sandeep Kumar, Christof Paar, Jan Pelzl, Gerd Pfeiffer, and Manfred Schimmler. "Breaking Ciphers with COPACOBANA - A Cost-Optimized Parallel Code Breaker", CHES 2006, pp.101-118 (2006).
- [3] Vasily Volkov, James W.Demmel, "Benchmarking GPUs to Tune Dense Linear Algebra", SC'08 (2008).
- [4] 渡辺 裕也, 遠藤 敏夫, 松岡 聡, "複数GPUにおけるセルフスケジューリングによる並列数値演算", 情報処理学会ARC研究報告, Vol.2008 No.75, pp85-90 (2008).