GPU를 이용한 암호화 효율성 연구

변진영, 이기영

인천대학교 정보통신공학과

A Study on Efficiency of Cryptography Using GPU

Jin-yeong Byeon, Ki Young Lee

Dept. of Info & Telecom Engineering, University of Incheon

E-mail : bgyjjang@hotmail.com

요 약

1970년대 라디오 주파수를 사용하여 컴퓨터 통신 네트워크가 구축된 이후 눈부신 발전을 거듭하여 Personal Computer 뿐만 아니라 Mobile이나 Tablet PC등에서도 인터넷이 가능하다. 이렇게 다양한 매체를 통해 인터넷을 사용함에 따라 보안에 대한 중요성이 높아지고 있다. 하지만 최근 현대 캐피탈이나 농협, 네이트와 같은 해킹 사례를 보면 평문 데이터 사용에 의해 피해가 더욱 확대 되었다. 평문 데이터 사용함에 따라 보안 위협이 커지는데 평문 데이터를 사용하는 이유를 암호화를 사용했을 때보다 QoS 하락 때문이라고 볼 수있다. 이를 해결하기 위해 고정된 인프라에서 잉여 자원인 GPU를 사용하여 암호화를 할 때 QoS 하락을 줄일 수 있을 것이다. 또한 CPU보다는 멀티코어를 사용한 병렬 처리를 활용하여 CPU보다 상대적으로 효율적인 암호화가 가능하다고 생각한다.

본 논문에서는 CPU를 이용한 암호화 처리 속도와 GPU를 이용한 암호화 처리 속도를 비교하여 GPU를 이용한 암호화 처리 가능성을 검토하였다.

키워드

Graphics Processing Unit, Crypto Graphics, CUDA, AES

Ⅰ. 서 론

현재 IT 업계에서는 모바일 기기의 발달과 클라우드 컴퓨팅의 발달로 유선 네트워크뿐만 아니라 무선 네트워크 역시 크게 발달하며 널리 확대되고 있다. 그뿐만 아니라 네트워크를 사용하는 단말기 역시 발달하여 Personal Computer 뿐만 아니라 Mobile이나 Tablet PC등에서도 인터넷이 가능하다. 이렇게 다양한 매체를 통해 인터넷을 사용함에 따라 보안에 대한 중요성이 높아지고 있다. 하지만 최근 현대 캐피탈이나 농협, 네이트와 같은 해킹 사례를 보면 평문 데이터 사용에 의해 피해가 더욱 확대 되었다. 평문 데이터 사용함에 따라 보안 위협이 커지는데 평문 데이터를 사용하는 이유를 암호화를 사용했을 때보다 QoS 하락 때문이라고 볼 수있다. 이를 해결하기 위해 고정된 인프라에서 잉여 자원인 GPU를 사용하여 암호화를 할 때 QoS 하락을 줄일 수 있을 것이다. 또한 CPU보다는 멀티코어를 사용한 병렬 처리를 활용하여 CPU보다 상대적으로 효율적인 암호화가 가능하다고 생각한다.

또한 그래픽 카드의 프로세서는 게임의 인기에 힘입어 비약적인 발전을 거듭해 오고 있으며 이미 GPU의 부동 소수점 연산 능력은 CPU의 성능을 뛰어넘고 있다. 하지만 그래픽 처리를 하지 않는 동안은 GPU의 연산 능력은 대부분 활용하지 않고 있다.

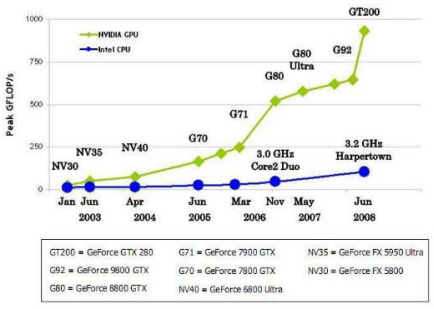

그림 1 GPU 발달 과정[1]

본 논문에서는 GPU의 유휴자원을 암호화 처리에 사용하여 CPU를 이용한 암호화 처리와 비교하여 GPU를 이용한 암호화 처리 가능성을 검토하였다.

II. 관련 연구

2.1. GPU의 구조

GPU는 그래픽 데이터를 전문으로 처리하기 위한 프로세서로 CPU와는 달리 ALU가 없고 대신 스트림 프로세서, 텍스쳐, 셰이더 유닛을 가지고 있으며 부동 소수점 연산에 최적화 되어 있다.

GPU는 호스트 컴퓨터로부터 받은 데이터를 프레임 버퍼에 넣는 과정에서 필요한 일련의 32비트 부동소수점 연산을 빠르게 수행하기 위한 병렬 구조를 가지고 있다.[2]

그림 2 GPU 구조[2]

2.2. GPGPU(General-pupose computing on GPU)

컴퓨터 그래픽스를 위한 계산만 다루는 GPU를 사용하여 CPU가 전통적으로 취급했던 응용ㅇ 프로그램들의 계산을 수행하는 기술이다. 이는 프로그램 가능한 부분과 고정도 연산을 그래픽 파이프라인에 연결하는 것이로, 이를 통하여 소프트웨어 개발자들이 그래픽이 아닌 데이터에 스트림 프로세싱을 사용할 수 있게 된다.

GPU가 사용 가능한 계산 자원은 다음과 같이 다양하게 지원하고 있다.[3]

a. 프로그램 가능한 프로세서 :
꼭지점, 프리미티브, 프래그먼트 파이프라인을 통해 데이터 흐름에 커널을 배풀 수 있다. 여기서 커널이란 2중 for문과 같은 형태를 가지며 블록들로 구성된 그리드이며 블록은 다시 여러 개의 쓰레드로 구성된다.

b. 비트맵 변환기 :
프래그먼트를 만들고 꼭지점 당 상수를 넣을 수 있다.

c. 텍스쳐 유닛 :
읽기 전용 메모리

d. 프레임 버퍼 :
쓰기 전용 메모리

2.3. CUDA(Compute Unified Driver Architecture)

GPU를 이용한 프로그래밍에서 비전문가에게 너무 제한적이어서 GPGPU방식에 커널을 이용하는 단편적인 프로그래밍을 해왔다. 이를 극복하고자 CUDA라는 도구를 발표하게 되었다. CUDA란 GPU에서 수행하는 병렬처리 알고리즘을 산업 표준 언어를 사용하여 작성할 수 있도록 하는 범용 병렬 컴퓨팅 아키텍처로 C언어의 확장으로 구현되어 있으며 개발툴 및 라이브러리(CUBLAS: 선형대수, CUFFT: 퓨리에 변환)를 무상배포하고 있다.

CUDA는 GPU의 메모리를 온칩(on-chip)과 오프칩(off-chip)으로 나누워 관리한다. 메모리의 관리는 프로그램의 실행 방식을 결정하는 그리드, 블록, 쓰레드의 구성에 크게 영향을 미친다.

CUDA 프로그램은 모든 쓰레드에 동일한 프로그램이 할당되며, 실행 시 쓰레드의 고유번호가 파라미터로 사용되어 수행하는 내용이 달라지는 방식이며 CUDA 프로그래밍에서 가장 핵심적인 부분에 해당한다. [4][6]

2.4. AES (Advanced Encryption Standard) Algorithm

1990년대 들어 컴퓨터 처리 능력의 향상에 따라 DES 암호화 알고리즘의 해독 가능성이 높아지고 1998년을 기점으로 DES는 표준 기한ㄴ이 만료됨에 따라 미국 NIST(표준기술 연구소)에서 새로운 블록 암호인 AES를 공모하여 최종적으로 벨기에에서 만든 Rijndael 알고리즘이 AES로 채택되었다.

AES 알고리즘은 initial round, 9 rounds, final round로 총 11과정을 거친다.

initial round에서는 state라고 불리는 평문과 Cipher Key(128, 192, 256 bits)를 XOR연산을 수행하여 Add round key 과정을 실행한다.

initial round에서 나온 결과를 가지고 9번의 round를 거치게 되는데 각 round는 SubBytes, ShiftRows, MixColumns, AddRoundKey 과정을 포함하고 있다.

SubBytes과정은 DES알고리즘의 치환 개념과 같다. AddRoundKey에서 구해진 state table의 값과 S-box라고 하는 테이블을 이용하여 state table을 치환한다.

그 결과를 한 칸씩 옮기는 ShiftRows과정을 거

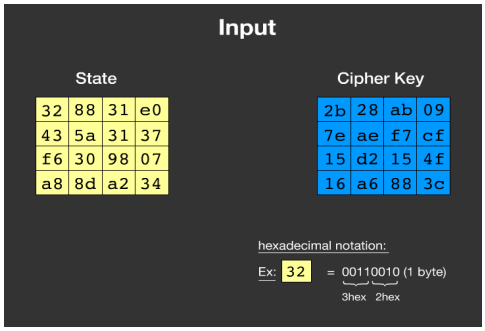

그림 3 AES input state

친다. 첫 번째 줄은 0칸, 두 번째 줄은 왼쪽으로 1칸 Shift시킨다. 맨 처음 칸은 맨 마지막으로 옮기게 된다. 세 번째 줄은 2칸, 네 번째 줄은 3칸 옮기게 된다.

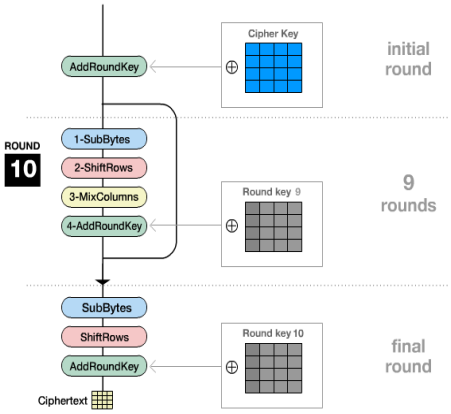

그림 4 AES 진행 과정

MixColumns 과정은 state의 column이 GF()에서의 다항식 a(x)가 고정된 다항식 c(x)에 대해 $a(x) \oplus c(x) \bmod (x^4+1)$ 연산을 행한다.

그 결과에 다시 Round key를 XOR하면 1 Round가 끝난다. Round key는 키 스케줄에 의해 생성된 키와 state의 EXOR를 한 결과이다.

이를 9번 반복한 후 마지막 Final round를 진행한다. final round는 보통 라운드에서 MixColumns 과정이 빠진 round 이다.

복호화 과정은 키 스케줄만 역으로 하여 AES 알고리즘을 실행한다.

III. 실험 구현 및 분석

CUDA는 C언어의 확장으로 설계 되었으며 작성된 커널은 그래픽 카드의 호스트 PC로부터 호출된다. CUDA 소스파일을 컴파일한 후 기존 C 컴파일러에게 전달되어 빌드 된다.

표 1 실험 환경

운영체제	CPU	GPU
Linux	Intel i7 2600	NVidia GeForce GTX 460

병렬처리의 효율을 가장 높게 하기 위해서는 AES 암호화 알고리즘의 Add Round Key, SubBytes, MixColumns 과정을 계산 할 때 State의 전체를 멀티 쓰레드로 동시에 계산하는 것이 바람직하다. AES 128bits의 경우 16bytes에 대해 16개의 쓰레드로 처리하여 한 블록의 암호화에 모두 참여하는 방식으로 구현한다.

실험은 AES 128bits, 192bits, 256bits 3가지 버전에 대해서 실험하였고 CPU로 처리 했을 경우와 속도 비교하였다.

GPU처리 속도를 측정하기 위해서 Timer를 사용하였고 순수하게 GPU내에서 처리되는 속도만을 계산 하였다.

표 2 실험 결과

	AES 128	AES 192	AES 256
CPU	3.1 Gbps	2.2 Gbps	1.4 Gbps
GPU	4.7 Gbps	4.5 Gbps	4.4 Gbps

IV. 결론

GPU의 처리 능력이 CPU보다 뛰어나짐에 따라 GPGPU 관련 연구는 많은 범위로 확대되고 있다. 그래픽 카드의 전문적인 지식 없어도 CUDA 라이브러리가 배포됨에 따라 많은 사람들이 큰 어려움 없이 사용할 수 있게 되었고 비록 NVidia 계열의 그래픽 카드에서 밖에 사용할 수 없다는 제약이 따르지만 병렬 연산이 필요한 많은 분야에서 환영하고 있는 추세다.

GPU의 사용은 암호화 분야에서 CPU의 점유를 낮춰 QoS 하락이라는 큰 문제를 줄여 많은 곳에서 암호화를 수행 할 수 있는 부가적인 장점을 가져온다고 생각한다. 현재 빈번히 일어나는 개인정보 유출과 같은 해킹 사례는 사회 문제로까지 확대되고 있는데 이에 대한 해결책이 될 수 있을 것이다.

본 논문에서는 암호화 분야에서 GPU의 병렬처리 기능을 사용하여 활용할 수 있는 가능성을 살펴보았고 그 처리 속도는 CPU보다 효과적이라고 보인다. 추후 연구로 오라클 적용 모듈을 제작하여 활용 가능성을 살펴보겠다.

참고문헌

[1] http://ixbtlabs.com/articles3/video/cuda-1-p1.html
[2] http://www.geforce.com/Hardware/GPUs/geforce-gtx-460/architecture
[3] http://gpgpu.org/
[4] http://www.nvidia.com/object/cuda_home_new.html
[5] http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
[6] 염용진, 조용국, "GPU용 연산 라이브러리 CUDA를 이용한 블록암호 고속 구현", 情報保護學會論文誌, 2008. 6