

## GPU 메모리 접근 시간 부채널 특성 분석

정승호<sup>01</sup> 윤명국<sup>2</sup> 구건재<sup>1</sup><sup>1</sup> 고려대학교 컴퓨터학과<sup>2</sup> 이화여자대학교 소프트웨어학부

minttmdgh@korea.ac.kr, myungkuk.yoon@ewha.ac.kr, gunjaekoo@korea.ac.kr

## Analyzing Characteristics of Memory Timing Side-Channels in GPU

Seungho Jung<sup>01</sup> Myung Kuk Yoon<sup>2</sup> Gunjae Koo<sup>1</sup><sup>1</sup>Department of Computer Science and Engineering, Korea University<sup>2</sup>Department of Computer Science and Engineering, Ewha Womans University

## 요약

최근 연구에서 GPU에서 수행하는 AES와 RSA와 같은 암호 알고리즘에서 GPU의 고유한 메모리 접근 구조에 따른 메모리 접근 시간의 차이를 이용하여 공격자가 암호키를 복원할 수 있음이 밝혀졌다. 이는 GPU 캐시 구조 및 메모리 접근 방식에 따라 공격자가 역연산을 통해 GPU의 메모리 요청 개수와 이에 따른 암호화 커널의 수행시간의 관계를 쉽게 알아낼 수 있기 때문이다. 본 연구에서는 변환 테이블을 사용하는 암호화 알고리즘에서 GPU의 메모리 접근 크기 및 변환 테이블의 접근 시간에 따른 GPU의 메모리 접근 시간 부채널의 특성을 GPU 구조 시뮬레이터를 통해서 분석하였다. 이러한 분석을 통하여 메모리 접근 크기 및 접근 시간에 따라 GPU 메모리의 부채널 취약성이 변화함을 밝혀냈으며 이러한 GPU 메모리 부채널의 취약점을 dummy 메모리 요청의 추가로 보완할 수 있음을 보여주었다.

## 1. 서론

Graphics processing unit (GPU)은 주로 그래픽이나 비디오 어플리케이션을 가속하기 위해서 개발되었으나 현재는 대규모 병렬 처리를 요구하는 기계 학습이나 빅데이터 어플리케이션과 같이 고성능 어플리케이션을 가속하는 데에 널리 쓰이고 있다. 이는 GPU가 수백 개의 간단한 코어를 가지고 있으며, 이렇게 많은 수의 코어를 사용하여 수백/수천개의 스레드(thread)를 동시에 실행시킬 수 있는 능력을 가지고 있기 때문이다. 이러한 GPU의 고성능 처리 능력을 사용하기 위하여 최근에는 AES와 RSA와 같은 널리 쓰이는 암호화 어플리케이션을 GPU용으로 개발하여 실행하고 있다. 즉 GPU에서 암호화 어플리케이션을 수행하여 대량의 평문이나 암호문 데이터를 병렬적으로 처리함으로써 암호화/복호화 과정의 처리량을 크게 향상시킬 수 있다.

그렇지만 최근에 GPU가 가지고 있는 고유의 메모리 접근(memory coalescing)특성으로 인하여 AES의 암호키의 값에 따라서 암호화 커널의 수행시간이 달라지고 이를 이용하여 AES의 암호키를 탈취할 수 있음이 밝혀졌다 [1]. 암호화/복호화의 처리 시간을 줄이고 GPU의 병렬처리방식을 이용하기 위해서 AES와 같은 암호화 방식은 변환 테이블을 사용한다. AES는 여러 단계의 암호화 과정을 가지고 있으며 각 단계마다 암호키와 비선형적인 데이터 처리를 통하여 결정된 결과값을 사용하여 변환 테이블에서 해당되는 결과값을 구하게 된다. GPU는 수십개의 스레드를 모은 warp라는 단위로 명령어를 수행하며 같은 warp안에 있는 스레드들은 모두 같은 명령어를

수행한다. 암호화 커널의 변환 테이블에서 데이터를 인덱스로 읽어오는 부분은 load 명령어로 GPU에서 수행이 되며, 이러한 메모리 명령어를 효율적으로 처리하기 위해서 GPU에서는 하나의 warp안의 여러 스레드에서 생성된 주소값이 동일한 캐시 라인의 데이터 영역에 있을 경우 이에 해당하는 메모리 요청들을 병합하여 하나의 메모리 요청으로 만들게 된다. 이를 GPU의 메모리 접근 과정이라고 부른다. 즉 AES 커널에서 하나의 warp에서 생성된 메모리 요청의 수는 메모리 접근 과정을 거치게 되면서 암호키에 따라 달라지게 되며, 이에 따라 암호키에 따라서 암호화 과정은 다른 수행 시간을 보이게 된다. 공격자는 비교적 간단한 AES의 마지막 단계를 역변환 테이블을 사용하여 마지막 단계의 메모리 요청 개수를 계산할 수 있으며, 메모리 요청 개수에 따라 수행시간이 비례하여 나타난다는 점을 이용하여 암호키를 유추할 수 있게 된다 [3]. 요약하자면 GPU에서는 메모리 접근 과정에 따른 메모리 요청의 개수가 암호키에 따라 달라지게 되고 이는 메모리 요청의 개수에 비례한 메모리 접근 시간으로 나타나게 되므로 보안에 취약하게 된다. 이번 연구에서는 AES와 같은 변환 테이블을 이용하는 암호화 커널에서 메모리 접근 단위의 크기와 변환 테이블이 GPU의 메모리 계층 구조에서 저장되어 있는 단계, 즉 데이터 접근 시간에 따른 GPU의 메모리 접근 시간 부채널의 특성을 분석하였다. 다시 말해서 앞에서 설명한 요소들에 따라 GPU의 메모리 접근 시간 부채널을 이용한 공격 취약성이 어떻게 변화하는지를 GPU 구조 시뮬레이터를 통해서 분석하였다. 또한 앞에서 설명한 GPU의 공격 취약점을 보완하기 위하여 dummy 메모리 요청을 추가함으로써 메모리 요청 개수에 따른 메모리 접근 시간 부채널 특성이 어떻게 변할 수 있는지를 보여주었다.

\* 이 성과는 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임(No. 2019-0-00533, 컴퓨터 프로세서의 구조적 보안 취약점 검증 및 공격 탐지대응)

## 2. 실험 환경

GPU의 메모리 접근 시간 부채널의 특성을 분석하기 위해서 이번

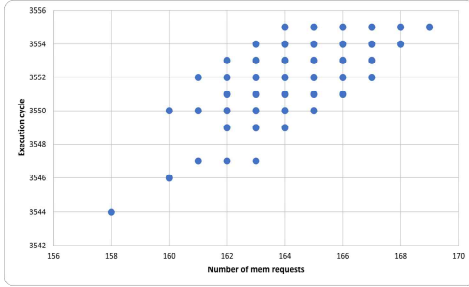


그림 1 점합 크기 128B, L1 ~ DRAM 사용

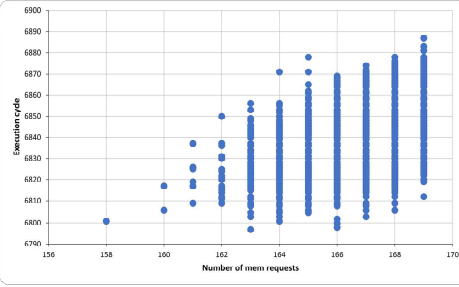


그림 2 점합 크기 128B, L2 ~ DRAM 사용

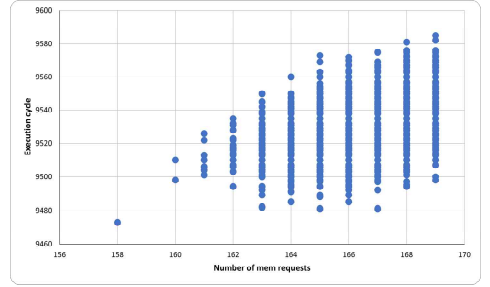


그림 3 점합 크기 128B, DRAM 사용

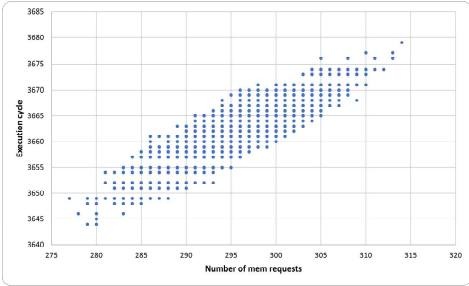


그림 4 점합 크기 64B, L1 ~ DRAM 사용

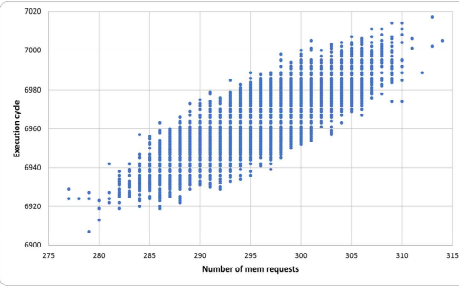


그림 5 점합 크기 64B, L2 ~ DRAM 사용

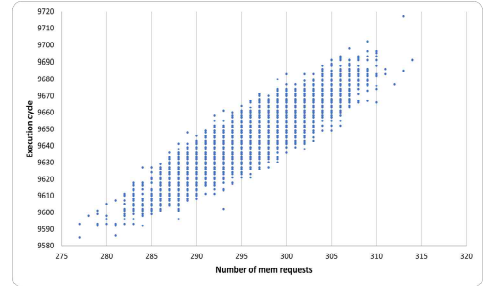


그림 6 점합 크기 64B, DRAM 사용

연구에서는 GPU 구조 시뮬레이터인 GPGPU-Sim을 사용하였다 [4]. 표1은 실험에 사용한 GPGPU-Sim의 코어 및 메모리 계층 구조의 설정을 정리한 것이다. 메모리 점합 크기에 따른 메모리 부채널의 특성을 살펴보기 위해서 메모리 점합 크기는 64 byte와 128 byte의 두 가지 크기를 설정하여 메모리 요청 개수에 따른 AES 마지막 단계의 수행시간을 비교하였다. 또한 변환 테이블의 접근 시간에 따른 부채널의 특성 및 취약점을 분석하기 위해 변환 테이블 데이터가 저장되는 위치를 L1 캐시, L2 캐시, DRAM으로 다르게 설정하여 메모리 요청 개수에 따른 수행시간을 측정하여 비교하였다. 부채널의 특성 및 공격 취약성을 비교하기 위해서 이 연구에서는 메모리 요청 개수에 따른 수행 시간의 상관 관계값을 피어슨 상관 계수를 이용하여 계산하고 비교하였다 [2].

표 1 실험에 사용한 GPU configuration

GPU 구성	
Core clock	1400MHz
Warp size	32 threads/warp
L1 Cache	128B cache block, 4-way, 32 sets
L2 Cache	128B cache block, 8-way, 64 sets * 12
DRAM	6 GDDR5 memory controllers, 4 Bank-group/MC, 16 DRAM-banks $T_{CL}=12, T_{RP}=12, T_{RAS}=28, T_{RCD}=12,$ $T_{RRD}=6$

### 3. 메모리 계층과 점합 크기에 따른 비례 관계 변화

이 장에서는 변환 테이블이 저장되어 있는 메모리 계층과 메모리 점합 크기에 따른 비례 관계를 살펴볼 것이다. GPU에서 공격자가 실행하는 부채널 공격은 AES의 마지막 라운드에선 인덱스를 이용하여 테이블 연산하는 과정이 한 번이기 때문에 상대적으로

다른 라운드보다 공격에 취약한 것을 노린다. 따라서, 주 공격 지점인 마지막 라운드에서의 메모리 요청 개수와 마지막 라운드의 실행 사이클을 측정해보면 그림1에서 그림6과 같이 나타난다. 메모리 점합 크기가 128 byte인 그림1~그림3 중에서 그림2와 그림3을 보게 된다면 비례하게 보이는 것은 하지만 점들이 산발적인 형태를 취하고 있어서 직관적으로 비례한 것을 확인하기 어렵다. 비례관계를 나타내는 피어슨 상관 계수를 이용하여 두 그림의 피어슨 계수를 측정해보면 약 0.43정도 나오게 되고 이는 정비례한 관계라고 볼 수는 없지만 약간의 비례 관계를 가진다고 볼 수 있다.

이와는 조금 다르게 모든 메모리 계층을 사용한 그래프인 그림1을 보게 된다면 어느정도 비례하게 증가하는 것을 볼 수 있고 상관 계수가 약 0.74가량으로 높게 나오는 것을 볼 수 있다. 그 이유는 테이블에 접근할 수 있는 캐시 라인이 최대 8개로 작기 때문에 실행 사이클의 다양성이 적어지고 이로 인하여 점들의 겹침이 많아져 샘플의 개수가 많아지더라도 각 메모리 요청 별 실행 사이클의 분화가 많아지지 않기 때문이다.

메모리 점합 크기를 128 byte에서 64 byte로 변경하여 메모리 점합을 실행할 경우 메모리 요청이 증가하게 되며 그에 따라 비례관계를 나타내는 피어슨 계수도 0.82~0.89까지 증가하게 된다. 이는 그림4~그림6을 통하여 볼 수 있다시피 거의 정비례함을 나타내게 된다. 이러한 높은 정비례함이 나타나는 이유는 기존의 128 byte 메모리 점합의 크기에서는 테이블에 접근하기 위한 메모리 요청이 1~8개가 필요했다면 메모리 점합 크기가 64 byte로 낮아짐에 따라 필요한 메모리 요청 개수가 1~16개까지 더욱 세분화하여 더 높은 비례관계를 보이게 된다. 이러한 비례관계의 차이는 공격의 난이도에 영향을 줄 수 있다.

부채널 공격의 난이도를 결정하는 것에는 비례관계와 메모리 지연 시간 (latency) 두가지가 있다. 상위 메모리 계층을 사용할수록 메모리로 인한 메모리 지연시간은 줄어들기 때문에 메모리 요청 개수와 실행 시간 사이의 연관성을 찾기 더 어려워지고 이로 인하여 공격의 난이도는 상승하게 될 것이라고 예상된다.

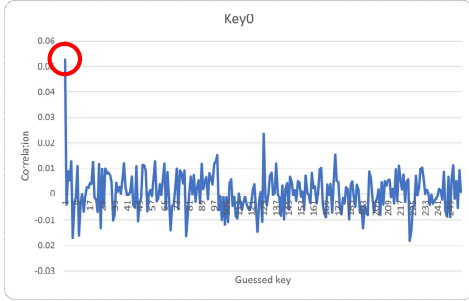


그림 7 접합 크기 128B, DRAM 사용

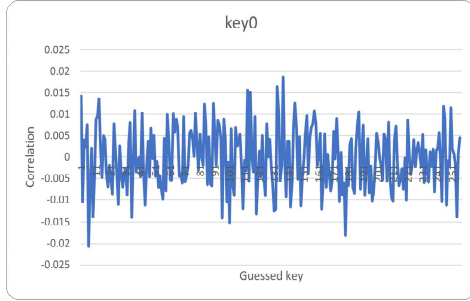


그림 8 접합 크기 128B, L2 ~ DRAM 사용

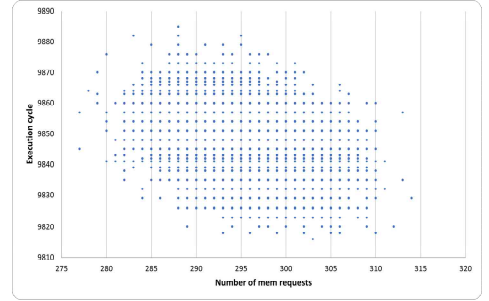


그림 9 접합 크기 128B, DRAM 사용, Dummy 메모리 요청이 있을 경우

AES의 키 16 byte 중에서 첫 번째 키 바이트 key0에 대하여 공격을 실행할 때 그림3의 조건인 128 byte 접합 크기, DRAM을 사용한 상태에서의 공격에 성공하기 위해서는 AES를 실행하는 단위인 샘플이 약 2만회가 필요하고 그림7과 같이 공격에 성공하게 된다. 그러나 그림2, 3의 조건인 128 byte 접합 크기에 L2 캐시 이상의 메모리 계층을 사용하는 경우 공격을 실행하는 것에 5만 회 이상의 샘플이 필요하였다. DRAM만 사용하여 공격이 가능했던 샘플 개수에서 L2~DRAM 계층을 사용한 조건을 공격하게 되면 그림8과 같이 정상적인 공격이 진행되지 않는 것을 볼 수 있다.

비례관계는 접합 크기가 변화함에 따라 큰 차이를 보이며 이는 공격의 난이도에 영향을 줄 수 있다. 접합 크기가 변화하여 메모리 요청이 변화하게 되는 경우 이는 메모리 지연시간에 영향을 주게 된다. 따라서, 메모리 지연시간의 변화는 비례 관계의 변화로 나타나게 되는데 접합 크기가 작아지는 경우, 비례 관계가 증가하는 것을 볼 수 있다. 표2는 AES의 마지막 라운드의 GPU 커널에서 메모리 요청 개수와 실행시간 사이의 관계를 정량적으로 나타낸 것이다. 둘 사이의 관계는 요소 사이의 비례관계를 나타내는 피어슨 상관 계수를 이용하여 나타냈다. 위에서 언급한 것처럼 메모리 접합 크기가 작아지게 되면 메모리 계층별 비례관계가 증가하는 것을 확인할 수 있다. 이렇게 비례 관계가 증가하는 경우 공격의 난이도는 낮아지게 되는데 비례 관계가 매우 뚜렷하게 나타나는 그림6의 64 byte 접합 크기의 경우 그림3의 조건과 비교하면 절반의 샘플을 이용하여 키 탈취가 가능했다.

표 2 접합크기별 요청 개수와 실행시간 사이의 비례관계

메모리계층 접합크기	L1 Cache	L2 Cache	DRAM
128B Width	0.7384	0.4349	0.4288
64B Width	0.8897	0.8217	0.8933

부채널공격을 막기위해서 보편적이고 간단한 방법은 공격자가 정확한 메모리 요청 개수를 알아내지 못하게 하는 것이다. 만일 AES를 실행하는데 필요한 메모리 요청 개수가 있다면 여기에 메모리 요청 개수를 임의로 추가하여 비례관계를 깨트릴 수 있다. AES에서 사용하는 테이블의 경우 1 KB이므로 GPU의 캐시 8개의 블록에 모두 담길 수 있다. 따라서, AES를 실행할 때 실제 메모리 요청 개수에 추가적인 메모리 요청 개수를 더해서 모든 메모리 요청을 8개로 보이게 한다면 공격자는 비례관계를 찾아낼 수 없게 된다. 여기서 가짜로 추가하는 메모리 요청을 dummy 메모리 요청이라고 한다.

그림9는 접합 크기 128 byte, DRAM의 경우에서 dummy 메모리 요청을 사용하였을 때, AES의 마지막 라운드의 메모리 요청 개수와 실행 사이클의 그래프를 나타낸 것이다. 그림에서 볼 수 있듯이 전체 메모리 요청 개수에서 dummy 메모리 요청 개수를 뺀 실제 메모리 요청 개수와 실행 사이클 간에 비례관계는 전혀 찾아볼 수 없고, 따라서 해당 조건에서는 공격이 실행되지 않는다.

#### 4. 결론

본 논문에서는 AES의 마지막 라운드를 실행하는 동안 발생하는 메모리 요청 개수와 GPU 실행 시간의 비례관계를 메모리 계층 별로 분석하였다. 메모리 계층이 높으면 메모리 지연시간이 줄어들어 공격이 더 어려워지고 메모리 접합 크기가 낮아지게 되면 그로 인한 메모리 지연이 증가하여 메모리 요청 개수와 실행 시간 사이의 비례관계가 증가하게 되어 공격이 더 쉽게 가능해지는 것을 보였다. 또한, dummy 메모리 요청을 사용할 시 AES의 마지막 라운드에서 메모리 요청 개수와 실행시간 사이의 비례 관계가 사라져 공격을 할 수 없음을 보였다.

이를 활용하여 메모리 요청 개수와 실행시간 사이의 비례관계를 사라지게 하면서 효율이 낮아지지 않는 GPU 메모리 계층 구조 및 동작 방식을 연구하여 신뢰성을 보장할 수 있게 만드는 것이 향후에 수행할 연구 과제이다.

#### 참고 문헌

[1] Z. H. Jiang, Y. Fei and D. Kaeli, "A complete key recovery timing attack on a GPU," *2016 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 394-405, 2016

[2] K. Pearson, "Notes on regression and inheritance in the case of two parents", *Proc.the Royal Society of London*, vol.58, pp.240-242, 1895

[3] G. Kadam, D. Zhang and A. Jog, "RCoal: Mitigating GPU Timing Attack via Subwarp-Based Randomized Coalescing Techniques," *2018 IEEE International Symposium on High Performance Computer Architecture (HPCA)*, pp. 156-167, 2018

[4] A. Bakhoda, G. L. Yuan, W. W. L. Fung, H. Wong and T. M. Aamodt, "Analyzing CUDA workloads using a detailed GPU simulator," *2009 IEEE International Symposium on Performance Analysis of Systems and Software*, pp. 163-174, 2009