



(19) 대한민국특허청(KR)

(12) 등록특허공보(B1)

(45) 공고일자 2023년12월27일

(11) 등록번호 10-2617678

(24) 등록일자 2023년12월20일

(51) 국제특허분류(Int. Cl.)
G06F 3/06 (2006.01) **G06F 12/14** (2006.01)
 (52) CPC특허분류
G06F 3/062 (2013.01)
G06F 12/14 (2013.01)
 (21) 출원번호 10-2023-0040139
 (22) 출원일자 2023년03월28일
 심사청구일자 2023년03월28일
 (30) 우선권주장
 1020220190281 2022년12월30일 대한민국(KR)
 (56) 선행기술조사문헌
 KR1020190031683 A
 US20190220283 A1
 KR1020030034463 A
 KR1020130070501 A

(73) 특허권자
고려대학교 산학협력단
 서울특별시 성북구 안암로 145, 고려대학교 (안암
 동5가)
 (72) 발명자
구건재
 경기도 남양주시 불암로 337, 5103동 2004호
이종민
 경기도 수원시 팔달구 효원로93번길 33, 107동
 505호
정승호
 경기도 부천시 신흥로 40, 1105호
 (74) 대리인
김준석, 박민욱

전체 청구항 수 : 총 16 항

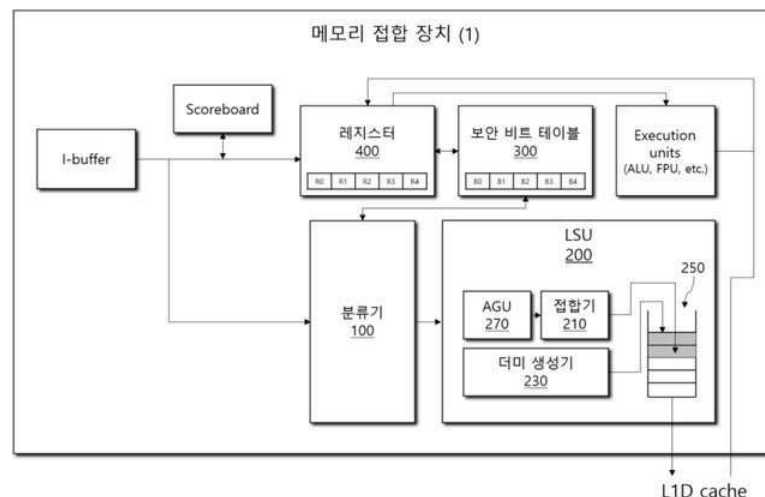
심사관 : 이진

(54) 발명의 명칭 메모리 읽기 명령어 분류에 따른 선별적 메모리 접합 장치 및 방법

(57) 요약

본 발명의 메모리 접합 장치는 GPU에서의 선별적 메모리 접합 장치에 관한 것으로, 메모리 읽기 명령어를 외부 공격에 대한 취약 여부에 따라, ND-load 혹은 D-load로 분류하는 분류기, 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 LSU(load/store unit), 생성된 메모리 요청을 접합하는 접합기 및 메모리 읽기 명령어가 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성기를 포함할 수 있다.

대표도 - 도1



(52) CPC특허분류

G06F 3/0604 (2013.01)

G06F 3/0659 (2013.01)

G06F 2212/1016 (2013.01)

G06F 2212/1052 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711152774
과제번호	2019-0-00533-004
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보보호핵심원천기술개발(정진기금)
연구과제명	컴퓨터 프로세서의 구조적 보안 취약점 검증 및 공격 탐지대응
기 여 율	40/100
과제수행기관명	고려대학교산학협력단
연구기간	2022.01.01 ~ 2022.12.31

이 발명을 지원한 국가연구개발사업

과제고유번호	1711167077
과제번호	2021R1C1C1012172
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	개인기초연구(과기정통부)
연구과제명	대규모 그래프 구조의 신경망 처리를 위한 효율적인 프로세서 및 메모리 구조에 대
한 연구	
기 여 율	40/100
과제수행기관명	고려대학교
연구기간	2022.03.01 ~ 2023.02.28

이 발명을 지원한 국가연구개발사업

과제고유번호	1711159611
과제번호	2020-0-01819-003
부처명	과학기술정보통신부
과제관리(전문)기관명	정보통신기획평가원
연구사업명	정보통신방송혁신인재양성
연구과제명	ICT명품인재양성(고려대학교)
기 여 율	20/100
과제수행기관명	고려대학교산학협력단
연구기간	2022.01.01 ~ 2022.12.31

공지예외적용 : 있음

명세서

청구범위

청구항 1

GPU에서의 선별적 메모리 집합 장치에 관한 것으로,

메모리 읽기 명령어가 외부 공격에 취약한 사용자 의존적 명령어이면 ND-load로, 외부 공격에 취약하지 않은 시스템 의존적 명령어이면 D-load로 분류하는 분류기;

상기 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 LSU(load/store unit);

생성된 상기 메모리 요청을 집합하는 집합기; 및

상기 메모리 읽기 명령어가 상기 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성기;

를 포함하는 메모리 집합 장치.

청구항 2

제1항에 있어서,

상기 분류기는,

상기 메모리 읽기 명령어의 의존적 인덱스 포함 여부에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 장치.

청구항 3

제2항에 있어서,

상기 분류기는,

상기 메모리 읽기 명령어가 사용자 의존적 인덱스를 포함하고 있으면 상기 ND-load로 분류하고, 시스템 의존적 인덱스를 포함하고 있으면 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 장치.

청구항 4

제1항에 있어서,

사용자가 복수의 메모리 영역을 직접 지정하고, 상기 메모리 영역을 상기 메모리 읽기 명령어가 포함하는지 여부에 따라 상기 분류기가 상기 ND-load 혹은 상기 D-load로 분류하도록 설정하고,

상기 분류기는,

상기 메모리 읽기 명령어가 상기 사용자가 지정한 상기 메모리 영역을 포함하는 경우, 이를 상기 설정에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 장치.

청구항 5

제1항에 있어서,

레지스터; 및

각 비트들이 상기 레지스터에 할당되고, false로 초기화된 상태의 보안 비트 테이블; 을 더 포함하여,

상기 메모리 요청에 따라 가져온 데이터를 상기 레지스터에 저장할 때, 상기 데이터가 특정 메모리 영역으로부터 가져온 것이면, 상기 레지스터에 할당된 상기 비트의 값을 true로 변경하고,

상기 분류기는,

상기 메모리 요청이 접근하는 상기 레지스터에 할당된 상기 비트의 값이 true인지 false인지에 따라 상기 메모리

리 읽기 명령어를 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 장치.

청구항 6

제1항에 있어서,

상기 더미 생성기는,

상기 더미 메모리 요청 및 상기 집합기에서 집합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되도록 하는 수만큼 더미 메모리 요청을 생성하는 것을 특징으로 하는 메모리 집합 장치.

청구항 7

제6항에 있어서,

상기 더미 생성기는,

상기 메모리 요청들의 기존 주소값들의 범위 안에서 겹치지 않는 주소값을 순차적으로 사용하여 상기 더미 메모리 요청의 주소로 설정하는 것을 특징으로 하는 메모리 집합 장치.

청구항 8

제7항에 있어서,

상기 더미 생성기는,

상기 메모리 요청들의 상기 기존 주소값들의 범위 내에 상기 더미 메모리 요청에 할당할 주소가 충분하지 않을 경우, 상기 기존 주소값들에 인접한 주소를 상기 더미 메모리 요청의 주소로 설정하는 것을 특징으로 하는 메모리 집합 장치.

청구항 9

GPU에서의 선별적 메모리 집합 방법에 관한 것으로,

메모리 읽기 명령어가 외부 공격에 취약한 사용자 의존적 명령어이면 ND-load로, 외부 공격에 취약하지 않은 시스템 의존적 명령어이면 D-load로 분류하는 분류 단계;

상기 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 메모리 요청 생성 단계;

생성된 상기 메모리 요청을 집합하는 집합 단계; 및

상기 메모리 읽기 명령어가 상기 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성 단계;

를 포함하는 메모리 집합 방법.

청구항 10

제9항에 있어서,

상기 분류 단계는,

상기 메모리 읽기 명령어의 의존적 인덱스 포함 여부에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 방법.

청구항 11

제10항에 있어서,

상기 분류 단계는,

상기 메모리 읽기 명령어가 사용자 의존적 인덱스를 포함하고 있으면 상기 ND-load로 분류하고, 시스템 의존적 인덱스를 포함하고 있으면 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 방법.

청구항 12

제9항에 있어서,

사용자가 복수의 메모리 영역을 직접 지정하고, 상기 메모리 영역을 상기 메모리 읽기 명령어가 포함하는지 여부에 따라 상기 분류 단계에서 상기 ND-load 혹은 상기 D-load로 분류하도록 설정하는 단계를 더 포함하고,

상기 분류 단계는,

상기 메모리 읽기 명령어가 상기 사용자가 지정한 상기 메모리 영역을 포함하는 경우, 이를 상기 설정에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 방법.

청구항 13

제9항에 있어서,

레지스터 및

각 비트들이 상기 레지스터에 할당되고, false로 초기화된 상태의 보안 비트 테이블을 더 포함하여,

상기 메모리 요청에 따라 가져온 데이터를 상기 레지스터에 저장할 때, 상기 데이터가 특정 메모리 영역으로부터 가져온 것이면, 상기 레지스터에 할당된 상기 비트의 값을 true로 변경하고,

상기 분류 단계는,

상기 메모리 요청이 접근하는 상기 레지스터에 할당된 상기 비트의 값이 true인지 false인지에 따라 상기 메모리 읽기 명령어를 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 하는 메모리 집합 방법.

청구항 14

제9항에 있어서,

상기 더미 생성 단계는,

상기 더미 메모리 요청 및 상기 집합 단계에서 집합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되도록 하는 수만큼 더미 메모리 요청을 생성하는 것을 특징으로 하는 메모리 집합 방법.

청구항 15

제14항에 있어서,

상기 더미 생성 단계는,

상기 메모리 요청들의 기존 주소값들의 범위 안에서 겹치지 않는 주소값을 순차적으로 사용하여 상기 더미 메모리 요청의 주소로 설정하는 것을 특징으로 하는 메모리 집합 방법.

청구항 16

제15항에 있어서,

상기 더미 생성 단계는,

상기 메모리 요청들의 상기 기존 주소값들의 범위 내에 상기 더미 메모리 요청에 할당할 주소가 충분하지 않을 경우, 상기 기존 주소값들에 인접한 주소를 상기 더미 메모리 요청의 주소로 설정하는 것을 특징으로 하는 메모리 집합 방법.

발명의 설명

기술 분야

[0001]

본 발명은 메모리 집합 장치 및 방법에 관한 것으로, 메모리 요청을 집합하고, 메모리 읽기 명령어의 취약성에 기반하여 선별적으로 더미 메모리 요청을 생성하는 메모리 집합 장치 및 방법이다.

배경 기술

[0002] 메모리 집합이란 GPU에서 워프 단위의 실행 모델로 인하여 발생하는 메모리 명령어의 성능 오버헤드를 줄이기 위한 방법의 하나로, GPU에서 워프에 의해 메모리 읽기 명령을 수행에 따라 생성된 메모리 요청들 중, 일 기준에 따라 메모리 요청 일부 혹은 전부를 묶어 실행해야 할 메모리 요청의 수를 줄여 캐시 접근을 수행하도록 하는 기술이다. GPU는 메모리 집합을 통해, 메모리 요청에 필요한 시간을 단축할 수 있는 등 성능을 향상시킬 수 있다.

[0003] 하지만, 메모리 집합에 따라 보안 상에 허점이 발생할 수 있고, 이를 이용한 집합 유닛 부채널 공격(coalescing unit side-channel attack)과 같은 외부 공격에 취약해질 수 있다. 하나의 예로, GPU에서 동작하는 AES(Advanced Encryption Standard)에 대한 실행 시간(execution time) 및 메모리 요청 개수의 상관관계를 이용하는 상관 기반 AES 공격(correlation based AES attack)의 경우, 공격자는 GPU에서 AES를 실행시킨 후, AES 암호화를 통하여 평문이 암호문으로 나오는데 걸리는 실행 시간과, 메모리 집합 결과로 생성된 메모리 요청의 수 정보를 획득하여 이들 사이의 연관성을 이용하여 AES 암호키 값을 유추할 수 있다.

[0004] 이러한 예와 같이 메모리 집합은 암호화에 걸리는 시간과, 메모리 집합의 결과로 생성된 메모리 요청의 수의 상관관계로 인하여 암호키가 유추될 수 있어, 이를 이용한 공격에 취약하다는 문제가 있었다. 따라서 메모리 집합에 따라 생성된 메모리 요청의 개수와 암호화에 걸리는 시간 사이의 관계를 유추할 수 없도록 하는 기술이 필요하게 되었다.

발명의 내용

해결하려는 과제

[0005] 본 발명의 일 실시예에 따른 선별적 메모리 집합 장치 및 방법은, 외부 공격에 취약한 메모리 읽기 명령어를 선별하고, 이에 따라 생성되는 메모리 요청들은 추가적인 더미 메모리 요청을 생성하여 공격자로 하여금 정확한 메모리 요청의 수를 알아낼 수 없도록 하여 보안을 강화하고자 한다.

[0006] 또한, 외부 공격에 취약하지 않은 메모리 읽기 명령어에 대해서는 추가적인 더미 메모리 요청을 생성하지 않아 불필요하게 성능이 저하되는 것을 방지하고자 한다.

과제의 해결 수단

[0007] 본 발명의 일 실시예에 따른 메모리 집합 장치는, GPU에서의 선별적 메모리 집합 장치에 관한 것으로, 메모리 읽기 명령어가 외부 공격에 취약한 사용자 의존적 명령어이면 ND-load(non-deterministic load)로, 외부 공격에 취약하지 않은 시스템 의존적 명령어이면 D-load(deterministic load)로 분류하는 분류기, 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 LSU(load/store unit), 생성된 메모리 요청을 집합하는 집합기 및 메모리 읽기 명령어가 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성기를 포함할 수 있다.

[0008] 또한, 일 실시예에 따라서, 분류기는 메모리 읽기 명령어의 의존적 인덱스 포함 여부에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 할 수 있다.

[0009] 또한, 일 실시예에 따라서, 분류기는, 메모리 읽기 명령어가 사용자 의존적 인덱스를 포함하고 있으면 상기 ND-load로 분류하고, 시스템 의존적 인덱스를 포함하고 있으면 상기 D-load로 분류하는 것을 특징으로 할 수 있다.

[0010] 또한, 일 실시예에 따라서, 사용자가 복수의 메모리 영역을 직접 지정하고, 메모리 영역을 메모리 읽기 명령어가 포함하는지 여부에 따라 분류기가 ND-load 혹은 D-load로 분류하도록 설정하고, 분류기는, 메모리 읽기 명령어가 사용자가 지정한 메모리 영역을 포함하는 경우, 이를 설정에 따라 ND-load 혹은 D-load로 분류하는 것을 특징으로 할 수 있다.

[0011] 또한, 일 실시예에 따라서, 레지스터 및 각 비트들이 레지스터에 할당되고, false로 초기화된 상태의 보안 비트 테이블을 더 포함하여, 메모리 요청에 따라 가져온 데이터를 레지스터에 저장할 때, 데이터가 특정 메모리 영역으로부터 가져온 것이면, 레지스터에 할당된 비트의 값을 true로 변경하고, 분류기는, 메모리 요청이 접근하는 레지스터에 할당된 비트의 값이 true인지 false인지에 따라 메모리 읽기 명령어를 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 할 수 있다.

[0012] 또한, 일 실시예에 따라서, 더미 생성기는, 더미 메모리 요청 및 집합기에서 집합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되도록 하는 수만큼 더미 메모리 요청을 생성하는 것을 특징으로 할 수 있다.

[0013] 또한, 일 실시예에 따라서, 더미 생성기는, 메모리 요청들의 기존 주소값들의 범위 안에서 겹치지 않는 주소값

을 순차적으로 사용하여 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.

- [0014] 또한, 일 실시예에 따라서, 더미 생성기는, 메모리 요청들의 기존 주소값들의 범위 내에 더미 메모리 요청에 할당할 주소가 충분하지 않을 경우, 기존 주소값들에 인접한 주소를 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.
- [0015] 본 발명의 일 실시예에 따른 메모리 집합 방법은, GPU에서의 선별적 메모리 집합 방법에 관한 것으로, 메모리 읽기 명령어가 외부 공격에 취약한 사용자 의존적 명령어이면 ND-load로, 외부 공격에 취약하지 않은 시스템 의존적 명령어이면 D-load로 분류하는 분류 단계, 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 메모리 요청 생성 단계, 생성된 메모리 요청을 집합하는 집합 단계 및 메모리 읽기 명령어가 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성 단계를 포함할 수 있다.
- [0016] 또한, 일 실시예에 따라서, 분류 단계는, 메모리 읽기 명령어의 의존적 인덱스 포함 여부에 따라 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 할 수 있다.
- [0017] 또한, 일 실시예에 따라서, 분류 단계는, 메모리 읽기 명령어가 사용자 의존적 인덱스를 포함하고 있으면 상기 ND-load로 분류하고, 시스템 의존적 인덱스를 포함하고 있으면 상기 D-load로 분류하는 것을 특징으로 할 수 있다.
- [0018] 또한, 일 실시예에 따라서, 사용자가 복수의 메모리 영역을 직접 지정하고, 메모리 영역을 메모리 읽기 명령어가 포함하는지 여부에 따라 분류 단계에서 ND-load 혹은 D-load로 분류하도록 설정하는 단계를 더 포함하고, 분류 단계는, 메모리 읽기 명령어가 사용자가 지정한 메모리 영역을 포함하는 경우, 이를 설정에 따라 ND-load 혹은 D-load로 분류하는 것을 특징으로 할 수 있다.
- [0019] 또한, 일 실시예에 따라서, 레지스터 및 각 비트들이 레지스터에 할당되고, false로 초기화된 상태의 보안 비트 테이블을 더 포함하여, 메모리 요청에 따라 가져온 데이터를 레지스터에 저장할 때, 데이터가 특정 메모리 영역으로부터 가져온 것이라면, 레지스터에 할당된 비트의 값을 true로 변경하고, 분류 단계는, 메모리 요청이 접근하는 레지스터에 할당된 비트의 값이 true인지 false인지에 따라 메모리 읽기 명령어를 상기 ND-load 혹은 상기 D-load로 분류하는 것을 특징으로 할 수 있다.
- [0020] 또한, 일 실시예에 따라서, 더미 생성 단계는, 더미 메모리 요청 및 집합기에서 집합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되도록 하는 수만큼 더미 메모리 요청을 생성하는 것을 특징으로 할 수 있다.
- [0021] 또한, 일 실시예에 따라서, 더미 생성 단계는, 메모리 요청들의 기존 주소값들의 범위 안에서 겹치지 않는 주소값을 순차적으로 사용하여 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.
- [0022] 또한, 일 실시예에 따라서, 더미 생성 단계는, 메모리 요청들의 기존 주소값들의 범위 내에 더미 메모리 요청에 할당할 주소가 충분하지 않을 경우, 기존 주소값들에 인접한 주소를 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.

발명의 효과

- [0023] 본 발명의 일 실시예에 따른 선별적 메모리 집합 장치 및 방법은, 외부 공격에 취약한 메모리 읽기 명령어를 선별하고, 이에 따라 생성되는 메모리 요청들은 추가적인 더미 메모리 요청을 생성하여 공격자로 하여금 정확한 메모리 요청의 수를 알아낼 수 없도록 하여 보안을 강화할 수 있다.
- [0024] 또한, 외부 공격에 취약하지 않은 메모리 읽기 명령어에 대해서는 추가적인 더미 메모리 요청을 생성하지 않아 불필요하게 성능이 저하되는 것을 방지할 수 있다.

도면의 간단한 설명

- [0025] 도 1은 일 실시예에 따른 본 발명의 메모리 집합 장치에 관한 블록도이다.
- 도 2는 일 실시예에 따른 메모리 읽기 명령어이다
- 도 3은 일 실시예에 따른 메모리 읽기 명령어이다
- 도 4는 일 실시예에 따라 집합기가 메모리 요청을 집합하는 과정을 도시한 블록도이다.
- 도 5는 일 실시예에 따른 LSU의 블록도이다.

도 6은 일 실시예에 따라 각 메모리 요청의 주소값을 도시한 도면이다.

도 7a는 일 실시예에 따른 메모리 집합 장치와 다른 메모리 집합 장치의 실행 시간을 비교한 그래프이다.

도 7b는 일 실시예에 따른 메모리 집합 장치와 다른 메모리 집합 장치의 에너지 소비량을 비교한 그래프이다.

도 8은 일 실시예에 따른 본 발명의 메모리 집합 방법의 순서도이다.

발명을 실시하기 위한 구체적인 내용

- [0026] 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 후술되어 있는 실시예들을 참조하면 명확해질 것이다. 그러나 본 발명은 이하에서 개시되는 실시예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시예들은 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자에게 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다.
- [0027] 본 명세서에서 사용되는 용어에 대해 간략히 설명하고, 본 발명에 대해 구체적으로 설명하기로 한다.
- [0028] 본 발명에서 사용되는 용어는 본 발명에서의 기능을 고려하면서 가능한 현재 널리 사용되는 일반적인 용어들을 선택하였으나, 이는 당 분야에 종사하는 기술자의 의도 또는 관례, 새로운 기술의 출현 등에 따라 달라질 수 있다. 또한, 특정한 경우는 출원인이 임의로 선정한 용어도 있으며, 이 경우 해당되는 발명의 설명 부분에서 상세히 그 의미를 기재할 것이다. 따라서 본 발명에서 사용되는 용어는 단순한 용어의 명칭이 아닌, 그 용어가 가지는 의미와 본 발명의 전반에 걸친 내용을 토대로 정의되어야 한다.
- [0029] 명세서 전체에서 어떤 부분이 어떤 구성요소를 "포함"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라 다른 구성요소를 더 포함할 수 있음을 의미한다. 또한, 명세서에서 사용되는 "부", "모듈", "유닛" 등의 용어는 적어도 하나의 기능 또는 동작을 처리하는 단위를 의미하며, 소프트웨어, FPGA 또는 ASIC과 같은 하드웨어 구성요소, 또는 소프트웨어와 하드웨어의 결합으로 구현될 수 있다. 그렇지만 "부", "모듈", "유닛" 등의 용어가 소프트웨어 또는 하드웨어에 한정되는 의미는 아니다. "부", "모듈", "유닛" 등은 어드레싱할 수 있는 저장 매체에 있도록 구성될 수도 있고 하나 또는 그 이상의 프로세서들을 재생시키도록 구성될 수도 있다. 따라서, 일 예로서 "'부", "모듈", "유닛" 등의 용어는 소프트웨어 구성요소들, 객체지향 소프트웨어 구성요소들, 클래스 구성요소들 및 태스크 구성요소들과 같은 구성요소들과, 프로세스들, 함수들, 속성들, 프로시저들, 서브루틴들, 프로그램 코드의 세그먼트들, 드라이버들, 펌웨어, 마이크로 코드, 회로, 데이터, 데이터베이스, 데이터 구조들, 테이블들, 어레이들 및 변수들을 포함한다.
- [0030] 아래에서는 첨부한 도면을 참고하여 본 발명의 실시예에 대하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 상세히 설명한다. 그리고 도면에서 본 발명을 명확하게 설명하기 위해서 설명과 관계없는 부분은 생략한다. "제1", "제2" 등과 같이 서수를 포함하는 용어는 다양한 구성 요소들을 설명하는데 사용될 수 있지만, 구성 요소들은 용어들에 의해 한정되지는 않는다. 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다. 예를 들어, 본 발명의 권리 범위를 벗어나지 않으면서 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소도 제1 구성요소로 명명될 수 있다. "및/또는"이라는 용어는 복수의 관련된 항목들의 조합 또는 복수의 관련된 항목들 중의 어느 하나의 항목을 포함한다.
- [0031] 본 명세서에 설명된 다양한 시스템, 장치, 저장 매체, 모듈 및 기구는 전용 연산 디바이스, 또는 하나 이상의 전용 연산 디바이스의 하나 이상의 연산 칩에서 구현될 수도 있다.
- [0032] 이하, 본 발명의 선별적 메모리 집합 장치(1) 및 방법에 관하여 설명하도록 한다.
- [0033] 도 1은 일 실시예 따른 본 발명의 메모리 집합 장치(1)에 관한 블록도이다.
- [0034] 본 발명의 메모리 집합 장치(1)은 메모리 읽기 명령어(memory load instruction)에 따라 메모리 요청(memory request)을 생성하고, 메모리 집합(memory coalescing)을 실시할 수 있다. 또한, 메모리 집합 장치(1)은 이에 따라 생성된 메모리 요청을 항상 일정한 수로 유지하기 위해 더미 메모리 요청(dummy memory request)를 추가적으로 생성할 수 있다.
- [0035] 본 발명의 메모리 집합 장치(1)은 공격에 취약한 메모리 읽기 명령어 및 공격에 취약하지 않은 메모리 읽기 명령어를 선별하여 모든 메모리 읽기 명령어에 대해 더미 메모리 요청을 생성하는 것이 아닌, 공격에 취약한 메모리

리 읽기 명령어를 수행하는 경우에만 더미 메모리 요청을 생성하도록 할 수 있다. 여기서 공격에 취약한 메모리 읽기 명령어를 non-deterministic memory load instruction(ND-load)로, 공격에 취약하지 않은 메모리 읽기 명령어를 deterministic memory load instruction(D-load)로 나타낼 수 있다. 또한, 외부 공격에 취약한 메모리 읽기 명령어(ND-load)의 경우, 사용자 의존적 명령어와 혼용될 수 있고, 외부 공격에 취약하지 않은 메모리 읽기 명령어(D-load)는 시스템 의존적 명령어와 혼용될 수 있다. 이에 대해서는 후술할 설명에 의해 명확해질 것이다.

[0036] 메모리 읽기 명령어는 캐쉬(cache) 등의 메모리에 접근하도록 하는 명령어로, GPU의 메모리 계층 구조 중 하나인 전역 메모리(Global Memory)에서 데이터를 읽어와서 레지스터(Register)나 공유 메모리(Shared Memory)로 복사하도록 명령할 수 있다. 메모리 읽기 명령어는 인스트럭션 버퍼(Instruction buffer, I-Buffer) 등에 저장되어 메모리 접근시에 이용될 수 있다.

[0037] 본 발명의 메모리 집합 장치(1)은 GPU의 가장 작은 단위의 코어(core)인 스트리밍 프로세서(streaming processor, SP)들의 집합체인 스트리밍 멀티프로세서(streaming multiprocessor, SM)에서 동작할 수 있다. 또한, 본 발명의 워프(warp)는 스트리밍 멀티프로세서의 기본 실행 단위를 의미한다.

[0038] 본 발명의 메모리 집합 장치(1)은 GPU에서 동작할 수 있는 선별적 메모리 집합 장치(1)에 관한 것으로, 분류기(100), LSU(load/store unit, 200), 집합기(210) 및 더미 생성기(230)를 포함할 수 있다. 일 실시예에 의하면, 집합기(210) 및 더미 생성기(230)는 LSU(200) 내에 포함된 형태로 구성될 수 있다. 또한, 메모리 집합 장치(1)는 인스트럭션 버퍼(I-buffer), 스코어보드(scoreboard) 및 실행 유닛(execution units), 보안 비트 테이블(300) 및 레지스터(400)를 더 포함할 수 있다. 본 발명의 메모리 집합 장치(1)에 포함될 수 있는 인스트럭션 버퍼, 스코어보드 및 실행 유닛 등은 본 발명의 장치의 동작을 위한 장치로, 일반적인 메모리 장치의 해당 구성과 역할이 유사하여 자세한 설명은 생략하도록 한다.

[0039] 분류기(100)는 메모리 읽기 명령어를 공격에 대한 취약 여부에 따라 분류할 수 있다. 여기서 공격은 전술한 상관 기반 AES 공격(correlation based AES attack)과 같은 집합 유닛 부채널 공격(coalescing unit side-channel attack)을 의미할 수 있다. 분류기(100)는 메모리 읽기 명령어를 공격에 취약한 메모리 읽기 명령어(ND-load) 혹은 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류할 수 있다. 본 발명의 메모리 집합 장치(1)은 메모리 읽기 명령어를 공격에 취약하지 혹은 공격에 취약하지 않은지에 따라 더미 메모리 요청을 생성 여부를 결정하여 불필요한 더미 메모리 생성 요청을 방지하여 성능의 저하를 방지할 수 있다.

[0040] 분류기(100)는 메모리 읽기 명령어가 외부 공격에 취약한 사용자 의존적 명령어이면 공격에 취약한 메모리 읽기 명령어(ND-load)로, 외부 공격에 취약하지 않은 시스템 의존적 명령어이면 공격에 취약하지 않은 명령어(D-load)로 분류할 수 있다. 사용자 의존적 명령어는 및 시스템 의존적 명령어는 해당 메모리 읽기 명령어에 포함된 코드 등의 내용을 기반으로 사용자에게 따라 생성되는 메모리 요청의 개수가 변할 수 있는지에 따라 결정될 수 있다. 즉, 사용자 의존적 명령어는 사용자에게 따라 생성되는 메모리 요청의 개수가 변경될 수 있는 메모리 읽기 명령어이고, 시스템 의존적 명령어는 사용자에게 따라 생성되는 메모리 요청의 개수가 변경되지 않고 항상 일정한 메모리 읽기 명령어를 의미할 수 있다.

[0041] 분류기(100)는 다양한 방식에 따라 메모리 읽기 명령어를 분류할 수 있다.

[0042] 일 실시예에 의하면, 분류기(100)는 메모리 읽기 명령어에 포함된 인덱스의 의존성에 따라 취약 여부를 결정할 수 있다.

[0043] 도 2 및 3을 참조하여 메모리 읽기 명령어에 포함된 인덱스의 의존성에 따른 취약 여부 결정 과정에 관하여 설명하도록 한다.

[0044] 도 2는 일 실시예에 따른 메모리 읽기 명령어이다.

[0045] 도 2의 메모리 읽기 명령어는 GPU에서 동작하는 AES에 따른 메모리 읽기 명령어이다. 도 2에 따르면, 메모리 읽기 명령어에 포함된 pt array의 인덱스(index)는 tid로부터 유래될 수 있으며, tid는 워프 내에 존재하는 스레드들에 할당된 thread id 및/혹은 block id를 이용하여 계산될 수 있다. GPU 커널(GPU Kernel)은 일반적으로 스레드에 thread id 및/혹은 block id를 할당하여 array에 접근하며, thread id 및 block id를 이용하여 tid를 계산한다. GPU는 array 형태를 가지는 데이터를 복수의 스레드가 병렬적으로 array 데이터의 요소(element)들을 thread id 및/혹은 block id로 계산된 인덱스를 이용하는 형태를 빈번하게 사용한다. Thread id 및/혹은 block id는 사용자에게 따라 값이 변하는 것이 아니며, 동일한 값으로 유지되는 경우가 많으므로, thread id 및/혹은 block id에서 계산되는 인덱스는 매우 정형적인 형태를 가질 수 있다. 따라서, thread id 및/혹은 block id를

이용하는 것과 같이 사용자에 따라 값이 변하지 않는 인덱스를 사용자 비의존적 인덱스(user-independent index) 혹은 시스템 의존적 인덱스(system-dependent index)로 정의할 수 있다. 이렇게 시스템 의존적 인덱스를 이용하여 array 데이터를 접근할 경우, LSU(200) 및 집합기(210)를 통하여 거의 동일한 수의 메모리 요청을 생성하게 된다. 즉, 사용하는 인덱스 값에 따라 생성하는 메모리 요청 개수가 거의 변하지 않기 때문에 이러한 형태의 메모리 읽기 명령어는 전체 커널의 수행시간에서 거의 고정된 수행 시간을 차지하고 있을 수 있다. 따라서, 공격자는 이러한 형태의 메모리 요청에 대해서는 메모리 요청 개수 및 실행시간의 상관관계를 이용한 공격을 수행하기 어렵다. 분류기(100)는 전술한 바와 같이 시스템 의존적 인덱스를 사용한 유형의 메모리 읽기 명령어를 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류할 수 있다.

[0046] 도 3은 또 다른 실시예에 따른 메모리 읽기 명령어이다.

[0047] 도 3의 메모리 읽기 명령어는 GPU에서 동작하는 AES에 따른 메모리 읽기 명령어이다. 도 3에 따르면, 메모리 읽기 명령어에 포함된 Te0는 인덱스로 pt와 rek를 연산한 결과 값을 사용한다. 따라서, 전술한 공격에 취약하지 않은 메모리 읽기 명령어(D-load)와 달리 Te0는 rek로 사용되는 암호키와 pt에 따라 접근해야 하는 데이터 접근의 형태가 달라지게 된다. 즉, Te0의 데이터를 불러올 때, 실행되는 메모리 읽기 명령어는 인덱스가 사용자의 데이터 값에 따라서 결정되게 된다. 위와 같이 사용자의 데이터 값에 따라 결정되기 때문에 사용자에게 따라 값이 변하는 인덱스를 사용자 의존적 인덱스(user-dependent index)로 정의할 수 있다. 사용자 의존적 인덱스는 사용자가 메모리 값을 변경할 수 있는 메모리 영역을 포함할 수 있으며, 대표적으로 global, local 및 shared 메모리 영역이 이에 해당할 수 있다. 사용자 의존적 인덱스를 이용하여 데이터를 접근할 경우 LSU(200) 및 집합기(210)에서 데이터의 값에 따라 상이한 개수의 메모리 요청을 생성하게 될 수 있다. 상이한 개수의 메모리 요청으로 인하여 메모리 요청 개수 및 실행시간의 상관관계를 이용한 공격에 취약하게 될 수 있다. 분류기(100)는 전술한 바와 같이 사용자 의존적 인덱스를 사용한 유형의 메모리 읽기 명령어를 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류할 수 있다.

[0048] 또 다른 실시예에 따르면, 분류기(100)는 메모리 읽기 명령어에 사용자가 미리 지정한 메모리 영역이 포함되어 있는지에 따라 취약 여부를 결정할 수 있다.

[0049] 실시예에 따르면, 사용자는 복수의 메모리 영역을 사용자가 직접 지정할 수 있고, 또한 해당 메모리 영역을 메모리 읽기 명령어가 포함하는지 여부에 따라 분류기(100)가 공격에 대한 취약 여부를 결정하도록 설정할 수 있다. 분류기(100)는 메모리 읽기 명령어가 앞서 사용자가 지정한 메모리 영역을 포함하는 경우, 이를 사용자 설정에 따라 공격에 취약한 메모리 읽기 명령어(ND-load) 혹은 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류할 수 있다. 예를 들어, 사용자가 값이 변화할 가능성이 높은 어느 메모리 영역을 지정하고, 분류기(100)는 어느 메모리 읽기 명령어가 해당 메모리 영역을 포함하는 경우, 해당 메모리 읽기 명령어를 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류할 수 있다. 또한, 다른 예로, 사용자가 값이 변화할 가능성이 높지 않은 어느 메모리 영역을 지정하고, 분류기(100)는 어느 메모리 읽기 명령어가 해당 메모리 영역을 포함하는 경우, 해당 메모리 읽기 명령어를 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류할 수 있다.

[0050] 사용자가 메모리 영역을 직접 지정하여 이를 포함하는지에 따라 공격 취약성을 결정하는 경우, 앞서 시스템 의존적 인덱스 및 사용자 의존적 인덱스 포함 여부에 따라 취약 여부를 결정하는 것에 비하여, 사용자의 직접 판단에 따른 더 세세한 설정이 가능하다는 효과가 있을 수 있다.

[0051] 다시 도 1을 참조하여 보안 비트 테이블(secure bit table, 300)을 이용하여 공격 취약 여부에 대하여 결정하는 과정에 관하여 설명하도록 한다.

[0052] 실시예에 의하면, 본 발명의 메모리 집합 장치(1)은 레지스터(400) 및 보안 비트 테이블(300)을 더 포함할 수 있다. 각 스트리밍 멀티프로세서 당 한 세트의 보안 비트 테이블(300)이 포함되어 있을 수 있다. 보안 비트 테이블(300)의 각 비트(310)는 false 혹은 true 값을 가질 수 있으며, 모든 비트(310)는 false로 초기화 되어있을 수 있다. 또한, 각 비트(310)는 각 레지스터(400)에 대응되도록 할당되어 있을 수 있다.

[0053] 본 발명의 메모리 집합 장치(1)은 메모리 읽기 명령어를 실행하여 가져온 데이터를 레지스터(400)에 기록하는 경우, 해당 데이터가 특정 메모리 영역으로부터 가져온 것이면, 상기 레지스터(400)에 할당된 상기 비트(310)의 값을 true로 변경할 수 있다. 예를 들어, 해당 데이터가 공격에 취약한 메모리 영역으로부터 가져온 것이면, 해당 데이터를 기록한 레지스터(400)에 할당된 비트(310)의 값을 true로 변경할 수 있다. 여기서 공격에 취약한 메모리 영역은 앞서 언급된 사용자 의존적 인덱스에 포함된 메모리 영역을 의미하는 것일 수 있다.

[0054] 또한, 메모리 집합 장치(1)이 다시 메모리 읽기 명령어를 실행할 때, 할당된 비트(310)의 값이 true인 레지스터

(400)의 데이터를 이용하는 경우, 가져온 데이터를 기록하고자 하는 레지스터(400)에 할당된 비트(310)의 값을 true로 변경할 수 있다. 위와 같은 과정을 통해 메모리 집합 장치(1)은 보안 비트 테이블(300)의 각 비트(310)들의 값을 true로 변경할 수 있다.

- [0055] 분류기(100)는 상기 메모리 요청이 접근하는 상기 레지스터(400)에 할당된 상기 비트(310)의 값이 true인지 false인지에 따라 상기 메모리 읽기 명령어의 공격에 대한 취약 여부를 결정할 수 있다. 예를 들어, 분류기(100)는 할당된 비트(310)의 값이 true인 레지스터(400)에 접근하는 메모리 읽기 명령어를 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류할 수 있다. 반면, 할당된 값이 false인 레지스터에 접근하는 메모리 읽기 명령어는 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류할 수 있다.
- [0056] LSU(200)는 메모리 읽기 명령어에 따라 메모리 요청을 생성할 수 있다. LSU(200)는 GPU에서 메모리를 읽거나 쓰는 작업을 수행하고, 이를 위해 메모리 주소(address)를 계산하고, 적절한 캐시에서 데이터를 읽거나 쓰는 등의 기능을 수행할 수 있다. 즉, LSU(200)는 메모리 읽기 명령어를 전달받고, 메모리 읽기 작업을 처리하기 위해 해당 데이터가 저장된 메모리 주소를 계산하여 이에 기반한 메모리 요청을 생성할 수 있다.
- [0057] LSU(200)는 주소 생성 유닛(address generation unit, AGU(270))을 포함하여, 워프가 메모리 읽기 명령을 실행할 때, AGU(270)는 각 스레드의 요청 주소를 계산할 수 있다.
- [0058] LSU(200)는 워프의 스레드 수에 대응되는 수만큼 메모리 요청을 생성할 수 있으며, 각각의 스레드에 대응하여 각각의 메모리 요청을 생성할 수 있다. 예를 들어, 메모리 읽기 명령어를 수행하기 위한 일 워프의 스레드가 32개인 경우, 해당 워프의 스레드의 개수인 32개만큼의 메모리 요청을 생성할 수 있다.
- [0059] 또한, LSU(200)는 메모리 요청 대기열(250)(memory request queue)을 포함할 수 있다. 메모리 요청 대기열(250)은, LSU(200)에서 생성되어 후술될 집합기(210)에서 집합된 메모리 요청 및 터미 생성기(230)에서 생성된 터미 메모리 요청들을 저장하여 수행될 때까지 대기시킬 수 있다.
- [0060] 일 실시예에 따른 LSU(200)는 집합기(210) 및 터미 생성기(230)를 더 포함할 수 있다.
- [0061] 도 4를 참조하여 집합기(210)에 대하여 설명하도록 한다.
- [0062] 집합기(210)는 메모리 요청을 집합할 수 있다. 메모리 집합은 GPU에서 메모리 액세스 성능을 향상시키기 위한 방법으로, 집합기(210)는 LSU(200)에서 메모리 읽기 명령어를 수행하기 위한 워프의 스레드에 따라 생성된 메모리 요청의 일부 혹은 전부를 집합할 수 있다.
- [0063] 도 4는 일 실시예에 따라 집합기(210)가 메모리 요청을 집합하는 과정을 도시한 도면이다.
- [0064] 도 4의 실시예에서, 단일 워프가 4개의 스레드를 포함하고, 이에 따라 LSU(200)에서 4개의 메모리 요청이 생성되는 것을 가정할 수 있다. 이 실시예에서 스레드 1과 스레드 2의 메모리 요청은 스레드 1과 스레드 2의 주소가 0x10에서 0x1F 사이의 범위에 있으므로 주소 0x1X의 단일 메모리 요청으로 집합될 수 있다. 따라서, 집합기(210)는 워프의 스레드로부터 4개의 메모리 요청 중 일부를 집합하여 총 3개의 메모리 요청을 생성할 수 있다.
- [0065] 집합기(210)는 집합한 메모리 요청을 메모리 요청 대기열(250)에 전송시켜 메모리 요청이 수행될 때까지 대기시킬 수 있다.
- [0066] 메모리 요청을 집합하는 기준은 위의 예시에 한하지 않고 다양한 기준이 적용될 수 있다.
- [0067] 도 5를 참조하여 터미 생성기(230)에 대하여 설명하도록 한다.
- [0068] 도 5는 일 실시예에 따른 터미 생성기(230)를 포함하는 LSU(200)의 블록도이다.
- [0069] 터미 생성기(230)는 터미 메모리 요청(dummy memory request)을 생성할 수 있다. 여기서 터미 메모리 요청은 메모리 요청 주소를 갖는 등 일반적인 메모리 요청과 동일한 형태를 갖춘 반면, 실제 메모리 요청과는 달리 데이터를 가져오지 않는 등 목적이 없는 가짜 메모리 요청으로 정의할 수 있다.
- [0070] 터미 생성기(230)는 LSU(200)에 포함되어, 메모리 읽기 명령을 수행하는 워프에 따라 터미 메모리 요청을 생성할 수 있다. 즉, LSU(200)가 워프에 따라 메모리 요청을 생성하고자 할 때 터미 생성기(230)도 이에 따라 터미 메모리 요청을 생성할 수 있다.
- [0071] 터미 생성기(230)를 통해 생성된 터미 메모리 요청은 LSU(200)에서 생성 및 집합기(210)에서 집합된 메모리 요청과 함께 메모리 요청 대기열(250)에 저장되어 수행 준비될 수 있다.

- [0072] 더미 생성기(230)는 메모리 요청 대기열(250)에 저장되는 메모리 요청의 수가 항상 일정한 개수가 되도록 더미 메모리 요청의 수를 조절하여 생성할 수 있다. 즉, 더미 생성기(230)는 더미 메모리 요청 및 집합기(210)에서 집합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되돌고 하는 수만큼 더미 메모리 요청을 생성할 수 있다. 따라서, 더미 생성기(230)는 집합기(210)에서 집합된 메모리 요청의 수에 맞추어 더미 메모리 요청의 수를 조절하여 생성할 수 있다. 예를 들어, LSU(200)에서 생성되는 메모리 요청의 수가 항상 8개로 일정하게 하기 위한다고 가정하였을 때, 메모리 집합의 결과로 메모리 요청이 5개가 생성된 경우, 더미 생성기(230)는 더미 메모리 요청을 3개 생성하여 총 생성되는 메모리 요청이 8개가 되도록 할 수 있다. 따라서, 메모리 요청 대기열(250)에는 메모리 읽기 명령어의 종류에 관계없이 항상 같은 수의 메모리 요청이 존재할 수 있다. 이를 통해 본 발명의 메모리 집합 장치(1)은 공격자가 항상 같은 수의 메모리 요청 수를 관찰하게 되도록 하며, 실제 메모리 요청의 수를 알 수 없게 하여 실행시간과 메모리 요청 수의 상관관계에 의한 암호키 유추가 불가능하도록 할 수 있다.
- [0073] 더미 생성기(230)는 일반적인 메모리 요청과 비슷한 특성을 갖도록 더미 메모리 요청의 특성을 조정하여 생성할 수 있다. 예를 들어, 더미 메모리 요청이 데이터를 가져오기 위한 대기시간이 일반적인 메모리 요청의 대기시간과 동일 내지 유사하도록 생성할 수 있다. 이는 공격자가 더미 메모리 요청과 실제 메모리 요청을 구분하는 것을 방지하고자 하는 것으로, 만약 더미 메모리 요청의 대기시간 등의 특성이 일반적인 메모리 요청과 크게 다를 경우, 공격자가 이를 구분하여 실제 메모리 요청 수를 유추할 수 있기 때문이다.
- [0074] 일 실시예에 따른 더미 생성기(230)는 메모리 읽기 명령어의 외부 공격에 대한 취약 여부에 따라 더미 메모리 요청을 생성할 수 있다. 즉, 메모리 읽기 명령어가 공격에 취약한 메모리 읽기 명령어(ND-load)인지 공격에 취약하지 않은 메모리 읽기 명령어(D-load)인지에 따라 더미 메모리 요청을 생성하거나 생성하지 않을 수 있다. 여기서 공격에 취약한 메모리 읽기 명령어(ND-load) 및 공격에 취약하지 않은 메모리 읽기 명령어(D-load) 여부는 앞서 분류기(100)에서 분류된 것에 따를 수 있다.
- [0075] 일 실시예에 따른 더미 생성기(230)는 수행되는 메모리 읽기 명령어가 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류된 경우, 더미 메모리 요청을 생성할 수 있다. 공격에 취약한 메모리 읽기 명령어(ND-load)의 경우, LSU(200) 및 집합기(210)를 통해 생성되는 메모리 요청의 수가 상이하게 될 수 있기 때문에, 일반적으로 외부 공격에 취약할 수 있다. 더미 생성기(230)는 이러한 경우에 더미 메모리 요청을 생성함으로써, 항상 일정한 수의 메모리 요청이 생성되도록 할 수 있으며, 이를 통해 외부 공격을 방어할 수 있다.
- [0076] 반대로, 일 실시예에 따른 더미 생성기(230)는 수행되는 메모리 읽기 명령어가 공격에 취약하지 않은 메모리 읽기 명령어(D-load)로 분류된 경우, 더미 메모리 요청을 생성하지 않을 수 있다. 공격에 취약하지 않은 메모리 읽기 명령어(D-load)의 경우, LSU(200) 및 집합기(210)를 통해 생성되는 메모리 요청의 수가 일정하므로, 일반적으로 외부 공격에 취약하지 않을 수 있다. 이러한 경우 오히려 더미 메모리를 생성하게 되면, 불필요한 메모리 요청을 생성하는 것으로 불필요한 성능 저하를 일으킬 수 있다. 따라서, 더미 생성기(230)는 공격에 취약하지 않은 메모리 읽기 명령어(D-load)의 경우에는 더미 메모리를 생성하지 않을 수 있다.
- [0077] 도 6을 참조하여 더미 생성기(230)가 더미 메모리 요청을 생성하는 과정의 일 실시예에 대해 설명하도록 한다.
- [0078] 도 6은 각 메모리 요청의 주소값을 도시한 도면이다.
- [0079] 일 실시예에 따르면, 더미 생성기(230)는 더미 메모리를 생성하는 경우, 더미 메모리 요청의 주소값을 설정하여 생성할 수 있다. 더미 생성기(230)는 더미 메모리 요청의 주소값을 해당 워프에서 수행하고 있는 메모리 읽기 명령어가 접근 가능한 범위에서 찾을 수 있다. 도 x의 실시예를 참고하면, 더미 생성기(230)는 워프에서 수행하고 있는 메모리 읽기 명령어가 접근할 수 있는 0x40 내지 0x90의 주소값 중에서 더미 메모리 요청의 주소값을 찾을 수 있다.
- [0080] 더미 생성기(230)는 기존 주소값들의 범위 안에서 겹치지 않는 주소값을 순차적으로 사용할 수 있다. 또한, 기존 주소값 범위 안에서 사용가능한 주소값이 없다면 근처의 주소값을 사용하여 더미 메모리 요청의 주소로 설정하여 주소값 간 차이를 최대한 줄일 수 있다. 도 X의 실시예를 참고하면, 기존의 집합된 메모리 요청이 0x40, 0x60 및 0x90에 접근하는 경우, 더미 생성기(230)는 접근 가능한 0x40 내지 0x90의 주소값 중에서 누락된 블록(0x50, 0x70 및 0x80)을 찾고, 생성할 더미 메모리 요청의 주소로 설정할 수 있다. 만일, 메모리 주소 범위 안에서 추가로 더미 메모리 요청에 할당할 주소가 충분하지 않은 경우, 0x30 및 0xA0와 같이 인접한 블록의 주소를 할당하여 설정해줄 수 있다. 더미 생성기(230)는 이러한 방법으로 더미 메모리 요청의 주소를 설정하여 기존의 실제 메모리 요청들과 유사한 특성을 갖도록 더미 메모리 요청을 생성할 수 있다.

- [0081] 이하 도 7a 내지 도 7b를 참조하여 본 발명의 메모리 접합 장치(1)의 성능에 대하여 설명하도록 한다.
- [0082] 도 7a는 본 발명의 실시예에 따른 메모리 접합 장치(1)과 다른 메모리 접합 장치(1)의 실행 시간을 비교한 그래프이다.
- [0083] 도 7a에 따르면, 본 발명의 서로 다른 실시예에 따른 메모리 접합 장치(1)인 GhostLeg-ND 및 GhostLeg-Key의 실행 시간은 다른 메모리 접합 장치(1)에 비하여 상당히 감소된 실행 시간을 나타냄을 알 수 있다. 따라서, 본 발명의 메모리 접합 장치(1)은 기존의 메모리 접합 장치(1)에 비하여 속도면에서 향상된 성능을 가지고 있음을 알 수 있다.
- [0084] 도 7b는 본 발명의 실시예에 따른 메모리 접합 장치(1)과 다른 메모리 접합 장치(1)의 에너지 소비량을 비교한 그래프이다.
- [0085] 도 7b에 따르면, 본 발명의 서로 다른 실시예에 따른 메모리 접합 장치(1)인 GhostLeg-ND 및 GhostLeg-Key의 실행 시간은 다른 메모리 접합 장치(1)에 비하여 상당히 감소된 에너지 소비량을 나타냄을 알 수 있다. 따라서, 본 발명의 메모리 접합 장치(1)은 기존의 메모리 접합 장치(1)에 비하여 에너지 소비면에서 향상된 성능을 가지고 있음을 알 수 있다.
- [0086] 도 8을 참조하여 본 발명의 메모리 접합 방법에 관하여 설명하도록 한다.
- [0087] 도 8은 일 실시예에 따른 본 발명의 메모리 접합 방법의 순서도이다.
- [0088] 본 발명의 메모리 접합 방법은 메모리 읽기 명령어를 외부 공격에 대한 취약 여부에 따라 ND-load 혹은 D-load로 분류하는 분류 단계(S110)를 포함할 수 있다. 또한, 메모리 접합 방법은 메모리 읽기 명령어에 따라 메모리 요청을 생성하는 메모리 요청 생성 단계(S120)를 포함할 수 있다. 또한, 메모리 접합 방법은 생성된 메모리 요청을 접합하는 접합 단계(S130)를 포함할 수 있다. 또한, 메모리 접합 방법은 메모리 읽기 명령어가 ND-load로 분류된 경우, 더미 메모리 요청을 생성하는 더미 생성 단계를 포함할 수 있다.
- [0089] 또한, 분류 단계는 메모리 읽기 명령어의 의존적 인덱스 포함 여부에 따라 취약 여부를 결정하는 것을 특징으로 할 수 있다.
- [0090] 또한, 분류 단계는 메모리 읽기 명령어가 사용자 의존적 인덱스를 포함하고 있으면 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류하고, 시스템 의존적 인덱스를 포함하고 있으면 취약하지 않은 메모리 읽기 명령어로 분류하는 것을 특징으로 할 수 있다.
- [0091] 또한, 메모리 접합 방법은 사용자가 복수의 메모리 영역을 직접 지정하고, 메모리 영역을 메모리 읽기 명령어가 포함하는지 여부에 따라 분류 단계에서 공격에 대한 취약 여부를 결정하도록 설정하고, 분류 단계는 메모리 읽기 명령어가 사용자가 지정한 메모리 영역을 포함하는 경우, 이를 설정에 따라 공격에 취약한 메모리 읽기 명령어(ND-load) 혹은 공격에 취약하지 않은 읽기 명령어로 분류하는 것을 특징으로 할 수 있다.
- [0092] 또한, 레지스터(400) 및 각 비트(310)들이 레지스터에 할당되고, false로 초기화된 상태의 보안 비트 테이블(300)을 더 포함하여, 메모리 요청에 따라 가져온 데이터를 레지스터에 저장할 때, 데이터가 특정 메모리 영역으로부터 가져온 것이라면, 레지스터에 할당된 비트(310)의 값을 true로 변경하고, 분류 단계는, 메모리 요청이 접근하는 레지스터에 할당된 비트(310)의 값이 true인지 false인지에 따라 메모리 읽기 명령어의 공격에 대한 취약 여부를 결정하는 것을 특징으로 할 수 있다.
- [0093] 또한, 더미 생성 단계는, 메모리 읽기 명령어가 공격에 취약한 메모리 읽기 명령어(ND-load)로 분류된 경우, 더미 메모리 요청을 생성하는 것을 특징으로 할 수 있다.
- [0094] 또한, 더미 생성 단계는 더미 메모리 요청 및 접합기(210)에서 접합된 메모리 요청의 수의 합이 항상 기설정된 개수가 되도록 하는 수만큼 더미 메모리 요청을 생성하는 것을 특징으로 할 수 있다.
- [0095] 또한, 더미 생성 단계는 메모리 요청들의 기존 주소값들의 범위 안에서 겹치지 않는 주소값을 순차적으로 사용하여 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.
- [0096] 또한, 더미 생성 단계는 메모리 요청들의 기존 주소값들의 범위 내에 더미 메모리 요청에 할당할 주소가 충분하지 않을 경우, 기존 주소값들에 인접한 주소를 더미 메모리 요청의 주소로 설정하는 것을 특징으로 할 수 있다.
- [0097] 본원 발명의 실시예 등과 관련된 기술 분야에서 통상의 지식을 가진 자는 상기 기재의 본질적인 특성에서 벗어나지 않는 범위에서 변형된 형태로 구현될 수 있음을 이해할 수 있을 것이다. 그러므로, 개시된 방법들은 한정

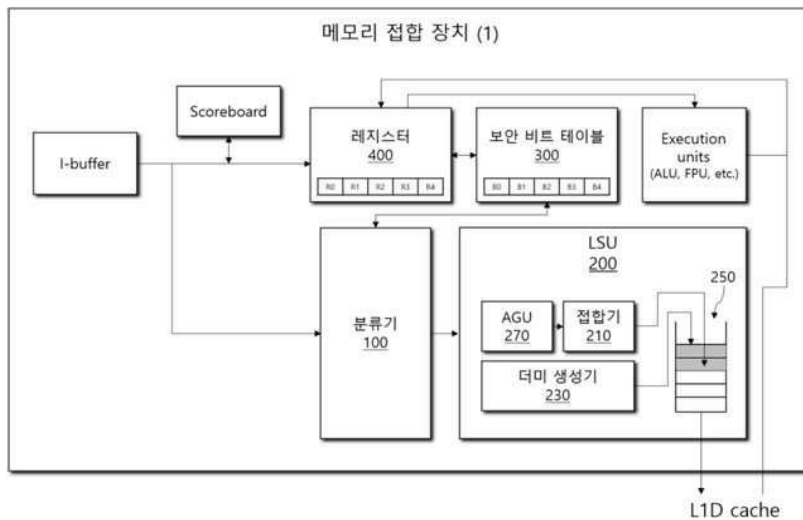
적인 관점이 아닌 설명적 관점에서 고려되어야 한다. 본 발명의 범위는 발명의 상세한 설명이 아닌 특허청구 범위에 나타나며, 그와 동등한 범위 내에 있는 모든 차이점은 본 발명의 범위에 포함되는 것으로 해석되어야 한다.

부호의 설명

- [0098]
- 1: 메모리 접합 장치
 - 100: 분류기
 - 200: LSU
 - 210: 접합기
 - 230: 더미 생성기
 - 250: 메모리 요청 대기열
 - 270: AGU
 - 300: 보안 비트 테이블
 - 310: 비트
 - 400: 레지스터

도면

도면1



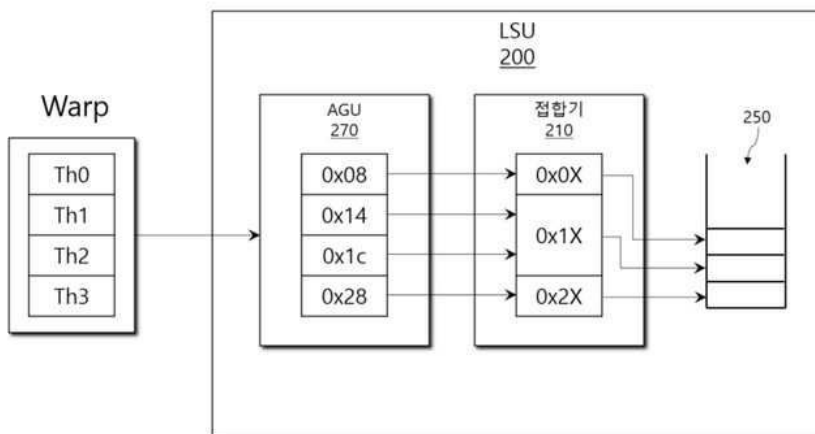
도면2

```
__global__ void AES_encrypt(const uint *pt,uint *rek,...){
uint tid = blockIdx.x * blockDim.x + threadIdx.x;
uint offset = (NUM_THREADS * NUM_BLOCKS + tid) << 2;
s0 = pt[offset + 0] ^ rek[0];
...
t0 = Te0[s0 >> 24] ^ ...;
...
}
```

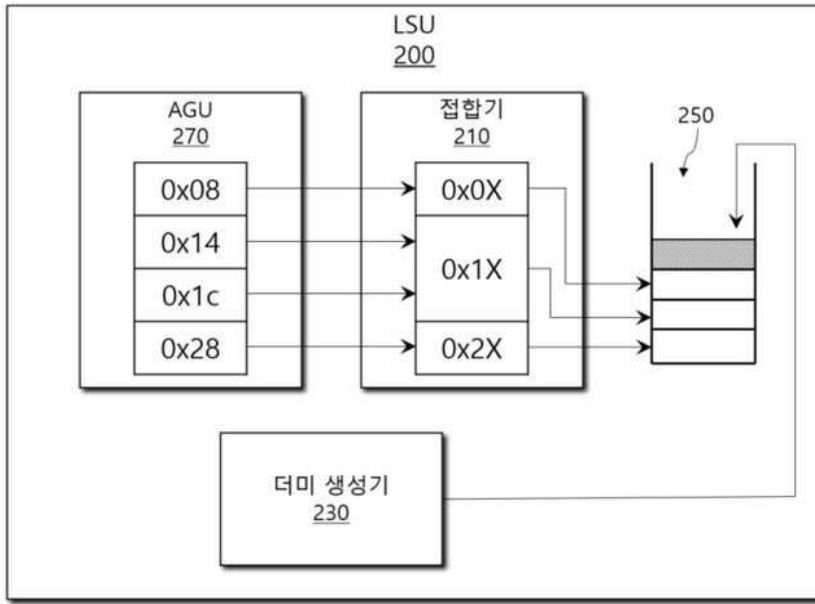
도면3

```
__global__ void AES_encrypt(const uint *pt,uint *rek,...){
uint tid = blockIdx.x * blockDim.x + threadIdx.x;
uint offset = (NUM_THREADS * NUM_BLOCKS + tid) << 2;
s0 = pt[offset + 0] ^ rek[0];
...
t0 = Te0[s0 >> 24] ^ ...;
...
}
```

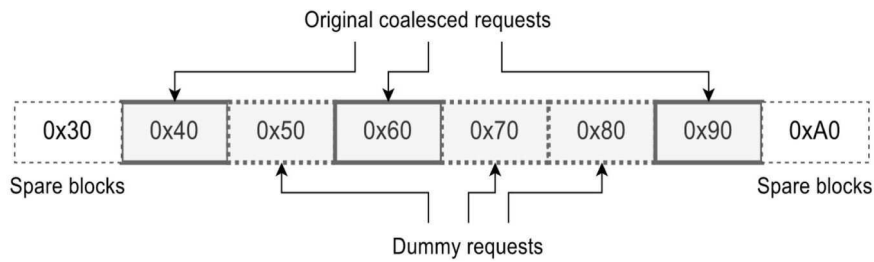
도면4



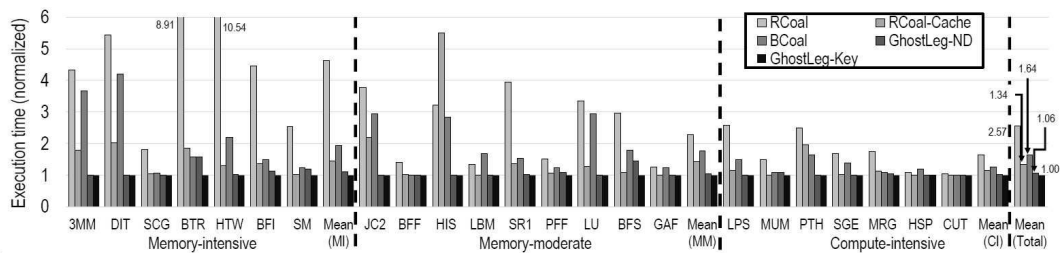
도면5



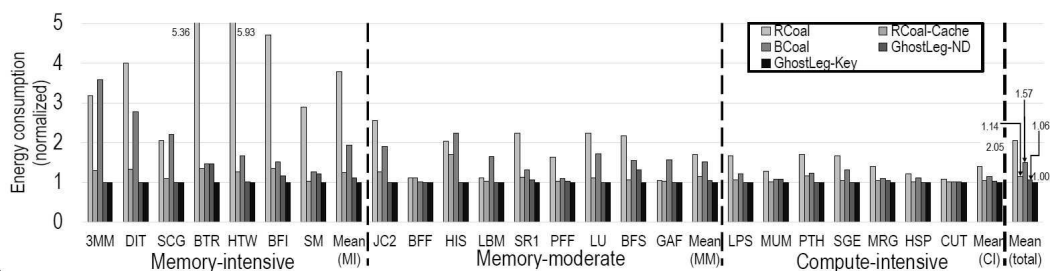
도면6



도면7a



도면7b



도면8

