Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.30, NO.6, Dec. 2020 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2020.30.6.1237

Performance Counter Monitor를 이용한 머신 러닝 기반 캐시 부채널 공격 탐지*

황 종 배,[1†] 배 대 현,[1] 하 재 철[2‡]
[1,2]호서대학교 (학생, 교수)

Machine Learning-Based Detection of Cache Side Channel Attack Using Performance Counter Monitor of CPU*

Jongbae Hwang,[1†] Daehyeon Bae,[1] Jaecheol Ha[2‡]
[1,2]Hoseo University (Student, Professor)

요 약

최근 마이크로 아키텍처의 취약점을 이용하여 내부의 비밀 정보를 노출시키는 캐시 부채널 공격들이 제안되었다. 캐시 부채널 공격 중 Flush+Reload 공격은 높은 해상도와 낮은 노이즈 특성으로 인해 여러 악의적 응용 공격에 활용되고 있다. 본 논문에서는 CPU 캐시 활동을 관측할 수 있는 PCM(Performance Counter Monitor) 기능을 이용하여 캐시 기반 부채널 공격을 찾아낼 수 있는 탐지기를 구현하였다. 특히, Spectre 공격과 AES 암호 연산 중 비밀 키 추출 공격이 발생했을 때를 가정하여 PCM 카운터 값의 변화를 관측하였다. 실험 결과, PCM의 4가지 카운터 특성이 캐시 부채널 공격에 크게 반응함을 확인하였고, SVM(Support Vector Machine), RF(Random Forest), MLP(Multi Level Perceptron)와 같은 머신 러닝 기반 검출기를 통해 높은 정확도로 캐시 부채널 공격을 탐지할 수 있었다.

ABSTRACT

Recently, several cache side channel attacks have been proposed to extract secret information by exploiting design flaws of the microarchitecture. The Flush+Reload attack, one of the cache side channel attack, can be applied to malicious application attacks due to its properties of high resolution and low noise. In this paper, we proposed a detection system, which detects the cache-based attacks using the PCM(Performance Counter Monitor) for monitoring CPU cache activity. Especially, we observed the variation of each counter value of PCM in case of two kinds of attacks, Spectre attack and secret recovering attack during AES encryption. As a result, we found that four hardware counters were sensitive to cache side channel attacks. Our detector based on machine learning including SVM(Support Vector Machine), RF(Random Forest) and MLP(Multi Level Perceptron) can detect the cache side channel attacks with high detection accuracy.

Keywords: Cache-based Side Channel Attack, Spectre Attack, AES, Performance Counter Monitor, SVM, RF, MLP

Received(10. 07. 2020), Modified(11. 11. 2020)
Accepted(11. 11. 2020)
* 이 논문은 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (No. 2020R1F1A10 74358)

† 주저자, hwangjb17@gmail.com
‡ 교신저자, jcha@hoseo.ac.kr(Corresponding author)

I. 서 론

암호 알고리즘을 정보보호용 디바이스에 구현하는 과정에서 발생하는 취약점을 이용한 부채널 공격은 정보보호 제품 개발 및 서비스를 제공함에 있어 심각한 위협이 되고 있다[1-2]. 최근에는 마이크로 프로세서의 설계 과정에서 발생하는 결함을 이용하는 부채널 공격이 등장하였고 이 공격 기술을 응용한 Spectre 공격이나 Meltdown 공격 등이 제안되었다[3-4]. 이러한 마이크로 아키텍처의 취약점을 이용한 공격들은 CPU 캐시 메모리를 대상으로 내부의 비밀 정보를 탈취하기 때문에 일반적으로 캐시 부채널 공격(cache side channel attack)이라고 한다.

캐시 부채널 공격으로는 Flush+Reload 공격[5], Flush+Flush 공격[6], Prime+Probe 공격[7] 등이 있다. 이 중에서 Flush+Reload 공격은 높은 해상도와 낮은 잡음 특성을 가지고 있어 Spectre 및 Meltdown 공격을 비롯해 대부분의 CPU 마이크로 아키텍처 공격에서 기본적으로 사용되고 있다.

Flush+Reload 공격은 기본적으로 공격자(spy)와 공격 대상(victim)간에 메모리를 공유하는 과정에서 비밀 정보가 노출된다. 특히, 이 공격은 캐시 메모리 중에서 LLC(Last Level Cache)를 주로 이용하는데 공격자는 미리 캐시 메모리에 접근하여 메모리를 비우고(flush) 공격 대상이 되는 희생자가 비밀 정보와 관련한 연산을 하도록 기다린다. 이후 재적재(reload)과정을 통해 접근하는 시간을 측정하여 비밀 정보를 탈취한다. 특히 주목할 점은 Flush+Reload 공격이 진행되면 LLC 캐시 미스(cache miss)가 비정상적으로 증가하는 것이다.

이러한 비정상적인 캐시 미스 증가 특성에 기반하여 G. Irazoqui 등은 AES의 비밀 키를 탈취하는데 성공하기도 하였다[8]. 또한, Y. Yoram 등은 RSA 암호 알고리즘 수행 과정을 공격하여 개인 키를 복구하는 공격을 제안하기도 하였다[9].

CPU 마이크로 아키텍처를 대상으로 하는 공격이 발견된 것은 최근인 관계로 알려진 심각성에 비해 대응 방법에 관한 연구는 아직까지 미흡한 실정이다. 특히, CPU 마이크로 아키텍처의 성능 최적화 기술의 취약점을 대상으로 한 공격이기 때문에 칩 설계를 바꾸지 않는 이상 원천적으로 차단할 방법 역시 마땅하지 않다. 따라서 캐시 부채널 공격이 실행되고 있을 때 이를 실시간으로 탐지할 수 있는 대응 방법에 관한 연구가 필요하다.

본 논문에서는 Flush+Reload를 활용한 캐시 부채널 공격을 구별해 낼 수 있는 탐지기를 구현하고자 한다. 이 공격 탐지기는 공격이 진행될 때 공격자에 의해 캐시 미스가 많이 발생한다는 원리를 바탕으로 CPU 마이크로 아키텍처의 정보를 실시간으로 얻기 위해 인텔 PCM(Performance Counter Monitor) 기능을 사용하였다.

특히, Flush+Reload 공격은 여러 마이크로 아키텍처 공격에 적용할 수 있음을 고려하여 Spectre 공격과 AES 비밀 키 추출 공격이 이루어지는 환경을 가정하여 검출 실험을 진행하였다. 또한, 캐시 부채널 공격 여부를 판별하는 방법으로 SVM(Support Vector Machine). RF(Random Forest), MLP(Multi-Level Perceptron)와 같은 머신 러닝 기법들을 활용하였다. 제안하는 탐지 기법들은 Spectre 공격 시에는 거의 100%, AES 암호 및 키 추출 공격 시에는 95% 이상의 정확도로 공격 탐지에 성공하였다.

II. 배경 지식

2.1 Flush+Reload 공격

Flush+Reload 공격은 캐시 부채널 공격의 일종으로서 공격자와 공격 대상이 메모리 페이지를 공유하고 있다는 조건에서 공격자가 공격 대상의 비밀 정보를 획득하는 공격이다. 이 공격은 x86 명령어 중 clflush를 이용하는 공격 기법인데 clflush가 논리적 주소를 기반으로 해당 영역만 캐시에서 flush 할 수 있는 특성을 이용한다. 캐시 메모리는 메인 메모리에서 가장 최근에 사용되는 데이터를 가지고 있어 접근 속도가 빠르며, CPU 내부와 인접한 위치에 탑재된 메모리이다. 캐시 메모리는 여러 층으로 나누어져 있는데 Flush+Reload 공격은 LLC 메모리를 대상으로 공격한다.

Flush+Reload 공격 시나리오는 다음과 같이 3단계로 구성할 수 있다. 먼저 공격자와 공격 대상이 공유하고 있는 메모리 라인을 clflush를 이용하여 제거한다. 두 번째 단계에서 공격자는 공격 대상이 가지고 있는 비밀 정보를 내재한 응용 프로그램이 실행되는 동안까지 대기하며 기다린다. 공격 대상이 응용 프로그램을 실행하였을 때 사용하는 비밀 정보의

값에 따라 메모리 라인의 접근 여부가 달라진다. 세 번째 단계에서 공격자는 메모리 라인에 다시 접근하여 시간을 측정한다. 이러한 재적재 과정에서 공격 대상이 해당 메모리 라인에 접근하였다면 이 정보가 캐시 메모리에 적재되고, 따라서 공격자의 접근 시간이 비교적 짧게 측정된다. 만약 공격 대상이 메모리 라인에 접근하지 않았다면 메인 메모리에서 데이터를 가져오는 것이므로 접근 시간이 길어지게 된다.

이 시차를 이용하여 공격자는 공격 대상이 수행하는 암호용 개인 키, 응용 프로그램의 비밀 정보 등을 추출할 수 있다. 이러한 Flush+Reload 공격은 주로 RSA, AES 등 암호 알고리즘을 대상으로 하여 개인 키를 추출하는 데 많이 활용되고 있으며 최근 제안된 Spectre나 Meltdown 공격에도 활용되고 있다. Flush+Reload 공격이 개발되고 나서 이를 탐지하는 연구도 진행되고 있는데 탐지기들 대부분은 LLC 캐시 미스와 같은 CPU의 이벤트가 비정상적으로 증가하는 것과 같은 공격 징후를 탐색하여 이를 활용하고 있다.

Chiappetta 등은 머신 러닝 기술과 Linux perf 툴을 이용하여 캐시 미스 카운터 값을 분석하여 Flush+Reload 공격을 탐지하는 방법을 제안하였다[10]. 또한, Mushtag 등은 12개의 하드웨어 성능 모니터링 값에 기반하여 12개 머신 러닝 모델에 적용한 검출기를 제안하기도 하였다[11]. Depoix 등은 LLC 캐시 미스와 적중 등을 이용한 딥 러닝 기법을 사용하여 99% 정도의 정확도로 공격을 탐지하였다[12]. 최근에는 PCM을 이용하여 실시간으로 캐시 부채널 공격을 탐지하는 모델이 Cho 등에 의해 제안되기도 하였다[13].

2.2 Spectre 공격

Spectre 공격은 현대 마이크로 프로세서의 성능 최적화를 위하여 적용된 분기 예측(branch prediction) 기술로 인해 발생한 하드웨어 보안 취약점을 이용한다. 분기 예측이란 프로그램의 논리 흐름상 실행 여부가 불확실한 상태에서 예측된 정보에 따라 먼저 명령어를 실행하는 CPU 성능 개선 기법을 말한다. 하지만 예측 실행으로 인해 CPU 성능은 향상되는 반면, 예측 실패임에도 불구하고 이미 명령어의 실행 정보가 캐시에 남게 되는데 이를 Flush+Reload 공격에 적용하여 비밀 정보를 추출하는 데 활용하고 있다.

Spectre 공격과 관련된 두 개의 CVE (Common Vulnerabilities and Exposures)가 존재하는데, 이 중 CVE-2017-5753(Spectre-V1)은 예측 실행 기법에 있는 취약점을 이용하여 배열의 범위를 벗어난 값을 읽어내는 기법이다. Fig. 1에서 3번 줄은 앞에서 설명한 분기 예측 때문에 2번 줄의 조건이 판단되기 전에 실행될 수 있다. 따라서 공격자가 x의 값을 array1 배열 범위를 넘어가는 값으로 조작하면 그 값이 읽히는 현상이 발생할 수 있다. 예측이 잘못된 경우 값이 폐기되기 때문에 프로그램 상에서 값을 확인할 수 없지만, Flush+Reload 공격과 같은 캐시 부채널 공격을 통해 캐시 접근 정보를 공격자가 읽을 수 있다.

CVE-2017-5715(Spectre-V2)는 분기 위치(branch target) 예측 기법을 이용한다. 분기 위치 예측에서는 분기문이 있을 때 다음으로 이동할 분기 위치를 계산하기 전에 예측된 위치를 사용하는데, 공격자가 분기 위치의 예측 과정을 조작하여 자신이 원하는 위치로 프로그램 수행을 분기하도록 만드는 것이다.

```
1. void victim_function(size_t x){
2.     if(x < array1_size) {
3.         y = array2[array1[x]*4096]
4.     }
5. }
```

Fig. 1. Conditional branch example of CVE-2017-5753

2.3 PCM

리눅스에서는 perf라는 라인 명령어를 이용하여 하드웨어 성능을 모니터링하지만 인텔에서 제공하는 PCM 도구를 활용하면 윈도우나 리눅스 환경에서도 모니터링 기능을 활용할 수 있다. PCM은 프로세서의 성능 카운터 값을 모니터링하는 도구로서 싸이클당 명령어 수나 캐시 미스를 비롯한 이벤트가 발생하였을 때 프로세서 동작과 관련된 카운터를 실시간으로 제공한다.

PCM은 PAPI(Performance Application Programming Interface)와 유사하지만, 프로세서의 코어에 대한 카운터 값만 지원하는데, 컴파일 과정을 거쳐 이진 파일처럼 실행할 수도 있으며 CPU의 여러 카운터 값을 CSV 파일로 출력할 수

있는 기능도 있다. 본 논문에서는 인텔 프로세서에서 캐시 부채널 공격이 이루어지고 이를 탐지할 수 있는 모델을 개발하는 것이 목표이므로 인텔 PCM을 공격 징후 모니터링 실험에 사용하였다.

2.4 SVM(Support Vector Machine)

SVM은 머신 러닝 모델 중 하나로 데이터 분류 및 패턴 인식에 많이 사용되는 지도 학습 방법이다. SVM 알고리즘은 두 개의 분류 영역 중 어느 한 곳에 속한 데이터 집합이 있을 때, 그 새로운 데이터가 어느 범주에 속할지를 판단하는 비확률적 이진 선형 분류 모델을 만들 수 있다. 또한, SVM은 선형 분류는 물론 비선형적인 데이터 분류에서도 사용될 수 있다. 다만, 비선형 데이터 분류 작업을 위해서는 주어진 데이터를 고차원의 특정 공간으로 사상(mapping)하는 과정이 필요한데, 이 사상 작업을 효율적으로 하기 위해 커널 트릭(kernel trick)이라는 것을 사용하기도 한다[14, 15].

본 논문에서의 공격 탐지기는 비선형 분류 기능을 사용하며 RBF(Radial Basis Function) 커널을 이용하여 SVM 학습 모델을 구성하였다. SVM에서는 마진(margin)의 너비를 결정하는 비용(cost) 파라미터와 커널의 표준 편차를 조절하는 감마(gamma)를 조정함으로써 정확도를 높일 수 있다.

2.5 RF(Random Forest)

RF는 앙상블(ensemble) 학습 방법의 일종으로, 다수의 의사 결정 트리(decision tree)로부터 데이터 분류 또는 회귀에 사용되는 지도 학습 모델이다. RF의 가장 큰 특징은 랜덤성에 의해 트리들이 서로 조금씩 다른 속성(feature)을 갖는다는 점이다. 이 속성은 각 트리의 예측(prediction)들이 비상관화(decorrelation) 되게 하며, 일반화 성능을 향상시킨다. 랜덤화는 각 트리의 훈련 과정에서 진행되며, 배깅(bagging)과 랜덤 노드 최적화(randomized node optimization) 기술이 주로 사용된다.

배깅은 트리를 만들 때 학습용 데이터의 부분 집합을 활용하여 형성하는 것으로서 한 가지 중요한 것은 바로 중복을 허용하여 선택함으로써 각 트리를 형성한다는 점이다. 또한, RF는 트리를 형성할 때 데이터 세트에만 변화를 줄 뿐만 아니라 속성 값을 선택함에 있어서도 부분 집합을 활용한다. 일반적으로 M개의 속성 값이 존재할 때, 임의로 선택하는 속성 값의 수는 $f=\sqrt{M}$ 을 사용한다. 이러한 배깅 작업을 통해 분산을 줄이고 과적합(overfitting)을 피하게 된다. 랜덤 노드 최적화는 학습 단계에서 학습 목적 함수를 최대로 만드는 노드 분할 함수의 매개 변수의 최적값을 구하게 된다. RF 모델에서 이 두 가지 방법을 서로 동시에 사용하면 랜덤화 속성을 더욱 증가시킬 수 있다[16].

RF에서 사용하는 주요 변수로는 의사 결정 트리의 크기 T, 루트 노드에서 종단 노드까지의 트리의 최대 깊이 D,그리고 학습용 부분 집합을 만들 때 추출하는 임의의 속성 수 f 등이 있다. 본 논문에서의 RF 기반 공격 탐지기에서는 10개의 의사 결정 트리를 사용하였으며, 트리 깊이를 나타내는 파라미터는 None으로 설정하였다. 이는 완벽하게 레이블이 결정될 때까지 분할하거나 최소 분할 데이터까지 자동으로 데이터를 분할하여 트리 깊이를 결정한다. 그리고 임의의 속성 수는 Auto로 설정하여 학습 모델을 구성하였다.

2.6 MLP(Multi-Level Perceptron)

MLP는 인공 신경망(Artificial Neural Network, ANN) 종류 중 하나로, 퍼셉트론으로 구성된 네트워크 계층을 여러 개 순차적으로 붙여놓은 형태이다[17]. 보통 MLP를 정방향 인공 신경망으로 부르기도 하는데, Fig. 2에서 보는 바와 같이 입력층, 은닉층, 출력층으로 구분되며, 각 층에서는 일정한 개수의 퍼셉트론 노드를 갖는다.

입력층은 예측값을 도출하기 위한 입력 변수의 값들을 입력하는 역할을 한다. 은닉층은 입력 노드로부터 입력 값을 받아 가중치 값을 이용하여 각 계층 출력 값을 계산하고, 활성화 함수에 적용하여 최종적으로 다음 계층에 전달한다. 각 층의 노드들은 모두 가중치를 가지는 네트워크로 연결되어 있는데 이 가중치 값은 초기에는 랜덤하게 설정되었다가 예측값에 근접한 정도로 조정된다.

본 논문에서의 입력 노드 개수는 학습하고자 하는 PCM 카운터 값의 개수이며 은닉층을 두 개 사용하여 학습을 진행하였다. 공격 검출기의 출력층은 공격 유무를 판단하는 2개의 노드를 갖게 된다. 본 논문에서의 MLP는 속성 수에 해당하는 입력 노드 그리고 2개의 출력 노드로 구성하였다. 또한, 손실 함수로는 cross-entropy 함수를 이용하였고 최적화를

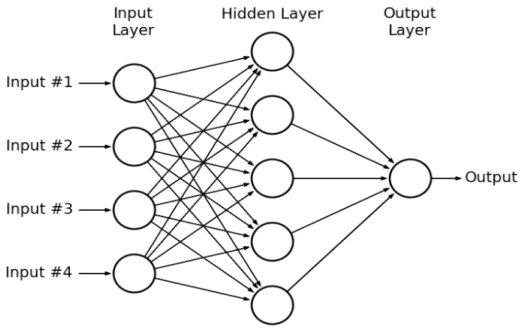


Fig. 2. Multi-Layer Perceptron network

위해 Adam 함수를 사용하였다.

III. 캐시 부채널 공격 실험 환경 및 탐지 모델

3.1 실험 환경

PCM은 캐시 적중률이나 명령어 실행 횟수 등 프로세서 동작 상태를 모니터링하기 위한 여러 성능 카운터 값을 제공한다. Flush+Reload 공격에 기반한 캐시 부채널 공격이 진행될 때 PCM이 제공하는 다양한 카운터 값이 어떻게 변화하는지 확인을 위한 실험 환경은 다음과 같이 구성하였다.

실험에 사용된 컴퓨터는 Intel Core(TM) i5-8400 2.80GHz 프로세서를 탑재하고 8GB의 DDR4 메모리를 사용하고 있으며 Ubuntu 16.04 LTS에서 공격 탐지 실험을 진행하였다. 본 논문에서는 공격 탐지율을 효과적으로 측정하기 위해 편의상 공격자와 공격 대상이 하나의 컴퓨터 내에 존재한다고 가정하였으며 공격 탐지기 역시 같은 컴퓨터 내에서 운영하는 것으로 가정하였다.

3.2 머신 러닝 기반 탐지기 설계

Flush+Reload 기반 부채널 공격이 진행되었을 때 머신 러닝을 이용한 공격 탐지에 사용될 속성을 정하기 위해 PCM 카운터 값을 선정하였다. 머신 러닝 기반 공격 탐지기의 학습을 위해 선정한 11개의 하드웨어 성능 이벤트는 Table 1과 같다. 여기서 11개 성능 이벤트를 선정한 이유는 PCM을 이용하여 CPU를 모니터링할 때 시스템의 변화를 가장 민감하게 감지하는 대표적인 이벤트이기 때문이다.

일반적으로 Flush+Reload 공격이 수행되는 데

Table 1. Specific hardware performance events

Name	Event description
EXEC	Instructions per nominal CPU cycle
IPC	Instructions per CPU cycle
FREQ	Relation to nominal CPU frequency
AFREQ	Relation to nominal CPU frequency while in active state
L3MISS	L3(read) cache misses
L2MISS	L2(read) cache misses
L3HIT	L3(read) cache hit ratio
L3MPI	Number of L3(read) cache misses per instruction
L2MPI	Number of L2(read) cache misses per instruction
READ	Bytes read from main memory controller
WRITE	Bytes written to main memory controller

필요한 시간은 수 십 msec 정도밖에 걸리지 않는다. 따라서 제안하는 공격 탐지기에서는 PCM을 통해 관측된 카운터 값을 1초 단위로 측정하였다. 또한, 사용자가 실제 사용하는 환경과 비슷한 환경하에서 탐지기를 만들기 위해 다른 응용 프로그램을 실행하지 않을 때, 동영상을 보고 있을 때, mp3 파일을 실행하고 있을 때와 같은 3가지 환경으로 구분하여 실험을 진행하였다.

지도 학습 기반의 탐지기 학습 모델은 SVM, RF 그리고 MLP 3가지를 선정하였고 데이터 전처리는 따로 하지 않고 수집한 그대로 사용하였다. 공격 대상의 각 시스템 운영 환경에서 실험용 데이터를 학습용 데이터 세트(training data set)과 검증용 데이터 세트(validation data set)으로 구분하여 50분씩 수집하였다. 그리고 정확도(accuracy) 측정을 위한 테스트 데이터 세트(test data set)도 별도로 수집하였다.

본 논문에서는 탐지기의 성능을 실제로 측정하기 위해 Flush+Reload 공격이 활용된 두 가지 공격을 탐지할 수 있는지 확인하였다. 먼저 Spectre PoC(Proof of Concept) 코드를 활용하여 Spectre 공격 탐지 여부를 확인하였다. 또한, T-Table을 사용하는 AES를 수행할 경우 Flush+Reload를 사용하여 비밀 키를 탈취하는 코드를 사용하여 탐지기 성능을 측정하였다.

IV. 머신 러닝 기반 캐시 부채널 공격 탐지

4.1 PCM 카운터 변화

먼저 Flush+Reload 공격을 활용한 Spectre 공격이 진행되었을 때 PCM을 사용하여 프로세서를 모니터링하면서 크게 변화하는 카운터 값을 관찰하였다. 다음 Fig. 3은 공격자에 의해 Spectre 공격이 성공적으로 진행된 결과를 나타낸 것이다. 그림에서 보는 바와 같이 공격 대상의 비밀 정보 "The Magic Words"가 추출됨을 볼 수 있다.

Fig. 4는 별도의 응용 프로그램을 실행하지 않은 상태에서 Spectre 공격이 실행되는 60초간 수집한 데이터 11가지 카운터 값을 그래프로 표현한 것이다. Time의 단위는 초이며 30초를 기준으로 0~30초 구간은 Spectre 공격이 실행되지 않고 이후 Spectre 공격이 실행된 것이다.

다음 Fig. 5는 Spectre 공격이 진행된 이후 급격한 변화를 보이는 4개의 카운터 값을 상세히 나타낸 것으로 L3MISS, L2MISS는 급격하게 증가하였고 AFREQ가 증가하여 고정화되는 것을 관찰할 수 있었다. 반면 IPC 값은 수치가 감소하는 것을 볼 수 있다. L3MISS, L2MISS는 공격의 특성상 캐

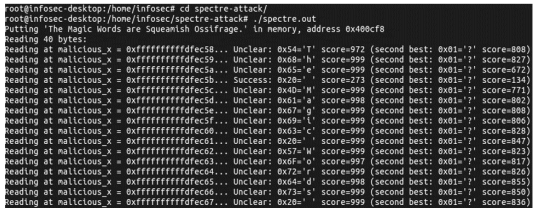


Fig. 3. Information leakage by spectre attack

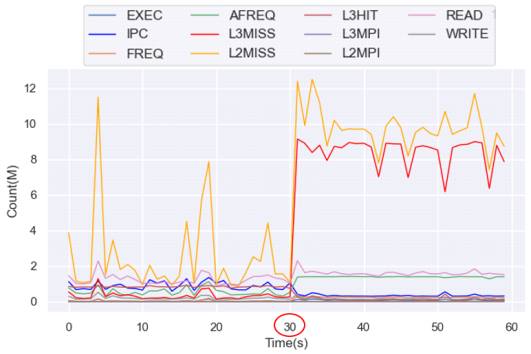


Fig. 4. Performance count values when trying Spectre attack

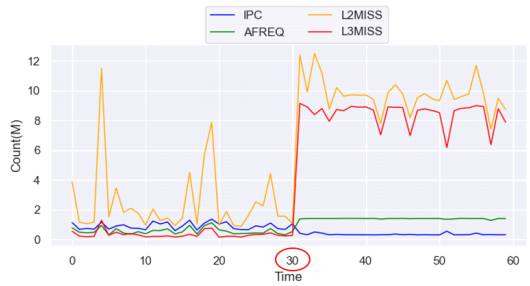


Fig. 5. IPC, AFREQ, L2MISS, L3MISS count values when trying Spectre attack

시 MISS가 많이 발생하기 때문에 증가하는 것이며, IPC은 CPU 싸이클 당 실행되는 명령어가 공격에 관련된 명령어만 실행되므로 그림과 같이 공격 순간부터 감소하는 특징을 보임을 알 수 있다. 이러한 현상은 공격자가 수행하는 공격용 프로그램이 공유 메모리에 대한 접근을 반복적으로 시도하여 시간 측정을 하는 과정이 시작되었다는 것으로 판단할 수 있다.

두 번째로 Flush+Reload를 활용하여 AES의 비밀 키를 추출하는 코드를 실행하였을 때의 공격 탐지 요인을 분석해 보았다. AES 비밀 키 공격 시 하드웨어 성능 카운터 값은 Spectre 공격을 실행하였을 때와 다른 점이 일부 확인되었다. 다음 Fig. 6은 128비트 비밀 키를 사용하는 AES를 사용하는 공격 대상 컴퓨터 비밀 키가 노출되는 것을 나타낸 것이다.

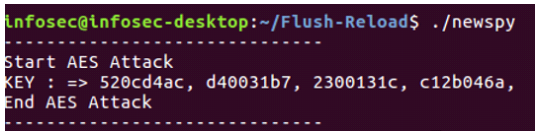


Fig. 6. Secret key extraction attack on AES

Fig. 7. IPC, AFREQ, L2MISS, L3MISS count values when trying key attack on AES

다음 Fig. 7은 다른 응용 프로그램을 실행하지 않은 상태에서 AES 비밀 키 추출 공격을 할 때 위의 4가지 카운터 값을 그래프로 그린 것이다. Spectre 공격과 달리 30초 이후 공격이 실행됨에도 불구하고 L3MISS, L2MISS 값이 급격하게 증가하고 감소하는 형태를 보이지 않았다. 그리고 IPC의 값과 AFREQ의 값이 공격을 실행하지 않을 때와 비교하면 증가하는 것을 확인할 수 있었다.

4.2 공격 탐지 실험 결과

머신 러닝 모델을 이용하여 공격을 어느 정도 탐지할 수 있는지를 실제 공격 대상 사용자들이 사용하는 환경과 최대한 비슷한 환경을 3가지 경우로 나누어 진행하였다. 또한, 모델을 학습하기 위해 50분간 데이터들을 수집하였고 테스트 데이터 역시 50분간 수집하여 실험을 진행하였다.

다음 Table 2는 두 가지 부채널 공격에 대한 머신 러닝 기반 탐지기 성능을 정확도로 나타낸 것이다. 여기에 사용된 속성은 하드웨어 성능 카운터 11개를 모두 머신 러닝의 입력으로 사용하였다. 추가로 공격 시 변화가 심한 4개의 속성만 입력으로 사용한 분석도 시행하였으나 11개 모두를 사용한 것과 큰 차이는 없었다.

Table 2에서 공격 탐지 환경에 따라 응용 프로그램을 실행하지 않았을 때, 비디오 파일을 실행했을 때, mp3 파일을 실행했을 때를 각각 Normal, Video, MP3로 표현하였다. 표에서 Spectre는 Spectre PoC 코드를 실행하였을 때를 나타내며, AES는 AES의 비밀 키 추출 공격 코드를 실행하였을 때를 의미한다.

Table 2. Accuracy for Spectre attack and key attack on AES

Attack	Model	Normal	Video	MP3
Spectre	SVM	0.9989	0.9956	1.0000
	RF	1.0000	0.9922	1.0000
	MLP	0.9990	0.9943	0.9997
AES	SVM	0.9411	0.8200	0.9456
	RF	0.9578	0.9033	0.9589
	MLP	0.8927	0.9743	0.9527

Table 2에서 보는 바와 같이, Spectre 공격이 진행 중일 때 수집한 데이터를 사용하여 모델을 학습시킨 후 테스트 데이터를 사용하여 정확도를 측정해 보았을 때 거의 100% 공격 탐지가 가능했다. 그 이유는 Fig. 4를 보면 알 수 있듯이 공격이 진행될 때 급격하게 변하는 L3MISS 및 L2MISS 카운터 값이 영향이 큰 것으로 판단된다. 머신 러닝별로 분석한 결과도 사용한 3가지 모델 모두 높은 정확도를 가지는 것을 확인할 수 있다.

AES에 대한 비밀 키를 복구하는 코드를 실행하였을 때는 Spectre 공격에 비해 비교적 낮은 정확도로 공격을 탐지하였다. Fig. 7과 Fig. 5를 비교하면 알 수 있듯이 Spectre 공격 때와 달리 카운터 값이 거의 변화하지 않는 것을 확인할 수 있었다. AES에 대한 공격은 Flush+Reload 공격이 실행됨에도 불구하고 캐시 적중률이 높기 때문에 L3MISS와 L2MISS의 값이 급격하게 변화하지 않는 것으로 분석되었다.

그럼에도 불구하고 일부 SVM과 MLP 모델에서

Table 3. Comparison of detector on cache-based side channel attacks

Detector	Tools	Performance counters	Target attacks	Machine Learning model	Accuracy(A) /F-score(F)
Chiappetta et al.[10]	Perf	L3_MISS	Flush+Reload Prime+Probe	ANN (AES, ECDSA)	≈0.93~1.0(F)
Mushtag et al.[11]	PAPI	L1_MISS, L3_MISS	Flush+Reload Prime+Probe	12 ML Models	≈0.95~0.98(A)
Depoix et al.[12]	PAPI	TOT_INS,L3_MISS, L3_ACCESS,	Flush+Reload	ANN (Spectre)	≈0.99(A) ≈0.97(F)
Cho et al.[13]	PCM	L1~L3MISS, IPC, RETIRED_BRANCH	Flush+Reload Prime+Probe	Softmax Clsssification	≈0.92~0.98(A)
Our method	PCM	L2_MISS, L3_MISS IPC, AFREQ	Flush+Reload	SVM, RF, MLP (Speatre, AES)	≈0.95~1.0(A)

높은 정확도를 보였는데 이는 공격이 실행되고 IPC와 AFREQ의 값이 증가하는 것이 제대로 학습된 결과로 볼 수 있다. AES 키 추출 공격 시의 정확도를 공격 모델별로 볼 때, 공격 환경에 따라 차이를 보이고 있으나 최고 0.95(RF)~0.97(MLP) 정도의 정확도로 공격을 탐지할 수 있었다.

다음 Table 3은 기존에 제안된 캐시 부채널 공격 탐지기와 논문에서 구현한 탐지기를 사용 도구, 성능 카운터, 대상이 되는 기본 공격, 머신 러닝 모델 그리고 탐지 성능면에서 비교 분석한 것이다. 제안된 각 탐지기마다 조금씩의 차이는 있지만, 기본적으로 Flush+Reload 공격과 Prime+Probe 공격에 대한 CPU의 성능 카운터 변화를 관찰 분석하였다. 기존 Cho 등이 제안한 기법[13]과 본 논문에서의 기법을 비교해 볼 때, 사전 분석 과정을 거쳐 공격 특성을 반영한 이벤트 카운터를 설정하였다. 또한, 머신 러닝 모델로 Softmax 분류 기법만 사용하던 것을 SVM, RF 및 MLP와 같은 딥러닝 기법을 적용하여 다양화하였다. 실험 결과, 머신 러닝 혹은 딥 러닝 모델에 기반한 지도 학습 방법에 대한 연구를 통해 정확도가 크게 증가함을 볼 수 있었다. 본 연구에서는 두 가지 캐시 부채널 공격에 대해서 최적의 머신 러닝 기법을 적용하면 약 95%~100% 정도의 정확도로 공격을 탐지할 수 있었다.

V. 결 론

본 논문에서는 Flush+Reload 공격을 활용한 캐시 부채널 공격을 시도할 경우 이를 탐지할 수 있는 머신 러닝 기반 탐지 모델을 설계하였다. 탐지 대상 공격으로는 Spectre 공격과 AES 키 추출 공격을 실행하였으며 사용자 환경을 고려하여 다양한 형태로 탐지기를 설계하였다.

캐시 부채널 공격을 탐지하기 위해 인텔 PCM 도구를 이용하여 공격에 크게 반응하는 하드웨어 성능 카운터 값을 프로파일링하고 이를 이용하여 머신 러닝 모델을 학습하도록 하였다. 설계한 SVM, RF 그리고 MLP 모델 모두 Spectre 공격을 거의 100% 탐지할 수 있었다. AES 키 추출 공격을 진행하였을 때에는 Spectre와 달리 캐시 적중률이 높고 각 카운터 값의 변화가 적어 공격 탐지율이 다소 낮아지기는 하였지만 머신 러닝 기반 탐지 기법이 캐시 부채널 공격에 충분한 방어 대책이 될 수 있음을 확인하였다.

References

[1] P. Kocher, J. Jaffe, and B. Jun, "Differential power analysis," CRYPTO'99, LNCS 1666, pp. 388-397, 1999.

[2] D. Bernstein, "Cache-Timing Attacks on AES," Available at http://cr.yp.to/antiforgery/cachetiming-20050414.pdf, Apr. 2005.

[3] P. Kocher, J. Horn, A. Fogh, D. Genkin, D. Gruss, W. Haas, M. Hamburg, M. Lipp, S. Mangard, T. Prescher, M. Schwarz and Y. Yarom, "Spectre Attacks: Exploiting Speculative Execution," IEEE Symposium on Security and Privacy, pp. 1-19, May. 2019.

[4] M. Lipp, M. Schwarz, D. Gruss, T. Prescher, W. Hass, S. Mangard, P. Kocher, D. Genkin, Y. Yarom and M. Hamburg, "Meltdown: Reading Kernel Memory from User Space," Proceedings of the 27th USENIX Security Symposium, pp. 973-990, Aug. 2018.

[5] Y. Yarom and K. Falkner, "FLUSH+RELOAD: a high resolution, low noise, L3 cache side-channel attack," Proceedings of the 23rd USENIX Security Symposium, pp. 719-732, Aug. 2014.

[6] D. Gruss, C. Maurice, K. Wagner, and S. Mangard, "Flush+Flush: A Fast and Stealthy Cache Attack," DIMVA'16, LNCS 9721, pp. 279-299, 2016

[7] D. Osvik, A. Shamir and E. Tromer, "Cache Attacks and Countermeasure: The Case of AES," CT-RSA'06, LNCS 3860, pp. 1-20, Feb. 2006.

[8] G. Irazoqui, M. Inci, T. Eisenbarth and B. Sunar, "Wait a minute! A fast, Cross-VM attack on AES," RAID'14, LNCS 8688, pp. 299-319, Sep. 2014.

[9] Y. Yarom and N. Benger, "Recovering OpenSSL ECDSA Nonces Using the FLUSH+RELOAD Cache Side-channel Attack," IACR Cryptology ePrint Archive, Available at https://eprint.iacr.org/2014/140, 2014.

[10] M. Chiappetta, E. Savas, and C. Yilmaz, "Real time detection of cache-based side channel attacks using hardware performance counters," Applied Soft Computing, Vol. 49, pp. 1162 - 1174, 2016.

[11] M. Mushtaq, A. Akram, M. Bhatti, R. Rais, V. Lapotre, G. G. Gogniat, "Run-time Detection of Prime+Probe Side-Channel Attack on AES Encryption Algorithm," In Proceedings of the Global Information Infrastructure and Networking Symposium (GIIS), Oct. 2018

[12] J. Depoix and P. Altmeyer, "Detecting spectre attacks by identifying cache side-channel attacks using machine learning," Proceedings of Wiesbaden Workshop on Advanced Microkernel Operating Systems(WAMOS'18) pp. 75-85, Aug. 2018.

[13] J. Cho, T. Kim, S. Kim, M. Im, T. Kim, and Y. Shin, "Real-Time Detection for Cache Side Channel Attack using Performance Counter Monitor," Applied Science, Vol. 10, Issue 3, 2019.

[14] C. Cortes, and V. Vapnik, "Support-vector networks," Machine Learning, Vol. 20, Issue 3, pp. 273-297, 1995.

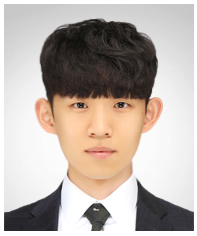
[15] T. Hofmann, B. Scholkopf, and A. J. Smola, "Kernel Methods in Machine Learning," The Annals of Statistics, Vol. 36, No. 3, pp. 1171-1220, 2008.

[16] L. Breiman, "Random Forests," Machine Learning, Vol. 45, pp. 5-32, 2001

[17] R. Collobert and S. Benjio, "Links between perceptrons, MLPs and SVMs," Proceedings of the twenty-first international conference on Machine learning, ICML'04, p. 23, 2004.

〈저자소개〉

황 종 배 (Jongbae Hwang) 학생회원
2017년 3월~현재: 호서대학교 컴퓨터정보공학부 학부과정
2020년 3월~현재: 호서대학교 학·석사연계과정
〈관심분야〉 암호학, 부채널 공격, 인공지능 보안



배 대 현 (Daehyeon Bae) 학생회원
2017년 3월~현재: 호서대학교 컴퓨터정보공학부 학부과정
2020년 3월~현재: 호서대학교 학·석사연계과정
〈관심분야〉 부채널 공격, 암호학, 머신러닝

하 재 철 (Jaecheol Ha) 종신회원
1989년 2월: 경북대학교 전자공학과 학사
1993년 8월: 경북대학교 전자공학과 석사
1998년 2월: 경북대학교 전자공학과 박사
1998년 3월~2007년 2월: 나사렛대학교 정보통신학과 교수
2007년 3월~현재: 호서대학교 컴퓨터정보공학부 교수
2013년 1월~현재: 한국정보보호학회 상임부회장
2009년 1월~현재: 한국산학기술학회 이사
〈관심분야〉 정보보호, 네트워크 보안, 부채널 공격