실시간 모니터링을 이용한 캐시 부채널 공격 탐지 프레임워크

임미옥, 김수진, 신영주
광운대학교 컴퓨터정보공학부 정보 및 사이버보안 연구실
e-mail: mo981014@gmail.com, kipper152@naver.com, yjshin@kw.ac.kr

Framework on Cache Side-channel Attack Detection Using Real-time Monitoring

Miok Im, Soojin Kim, Youngjoo Shin
Information and Cyber Security Lab, School of Computer and Information Engineering
Kwangwoon University

요 약

캐시 부채널 공격은 캐시 기반의 공격 기법으로 개인정보 유출에 대한 위험성이 큰 보안 취약점이다. 해당 취약점을 막기 위해 실시간 공격 탐지 기법에 관한 연구들이 진행되고 있지만 사용자에게 이벤트값과 탐지 결과를 빠르고 편리하게 보여줄 필요성이 있다. 본 논문은 효율적인 캐시 부채널 공격 탐지를 위해 Intel PCM 과 기존의 탐지프로그램을 개선하여 탐지에 필요한 데이터들을 실시간으로 모니터링 및 경고를 보내주는 프레임워크를 제작했다. 해당 프레임워크는 캐시 부채널 공격을 실시간 탐지 및 관련 데이터들을 대시보드로 보여준다.

1. 서론

프로세서는 캐시를 사용하여 CPU 가 메모리에 저장된 데이터를 읽어올 때 빠르게 접근할 수 있다. 캐시는 프로세서의 성능 향상에 기여하며 오늘날 대부분의 프로세서에서 사용되고 있다. 하지만 캐시 취약점은 캐시 부채널 공격에 이용되어 개인정보를 유출할 수 있다는 문제점이 있다. 캐시 부채널 공격은 캐시 기반의 공격 기법으로 공격자와 희생자가 공유하는 LLC(Last Level Cache)를 사용한다. 해당 공격을 막기 위해 탐지 기법에 관한 많은 연구가 진행됐지만 사용자에게 이벤트 값(Cache Miss, IPC, Branch 등)과 어떠한 공격이 들어왔는지 시각적으로 보여줄 필요성이 있다.

우리는 Intel PCM 과 기존의 탐지 프로그램[7]을 활용하여 이벤트값과 탐지 결과를 실시간 모니터링함으로써 캐시 부채널 공격 여부를 쉽게 확인할 수 있도록 프레임워크를 제작했다. 해당 프레임워크는 데이터 수집도구인 Telegraf 를 통하여 이벤트 값과 탐지 프로그램 결과값을 수집한 후 시계열 데이터베이스 Influxdb 에 저장함으로써 최종적으로 Grafana 대시보드에 그래프로 보여주도록 만들었다. 기존 탐지 프로그램에서 모니터링 및 경고 수단을 추가하여 사용자가 편리하고 빠르게 공격 여부를 확인할 수 있었다.

본 논문의 구성은 다음과 같다. 2 장에서는 Grafana, PCM, 캐시 부채널 공격, Softmax Classification 에 대한 배경지식을 설명한다. 3 장에서는 프레임워크 제작 방법에 대해서 데이터 수집 및 저장, 대시보드 설정 및 알람에 대해 말한다. 4 장에서는 3 장의 실험과 결과를 설명한다. 5 장에서는 향후 계획, 마지막 6 장에서는 결론에 대해 기술한다.

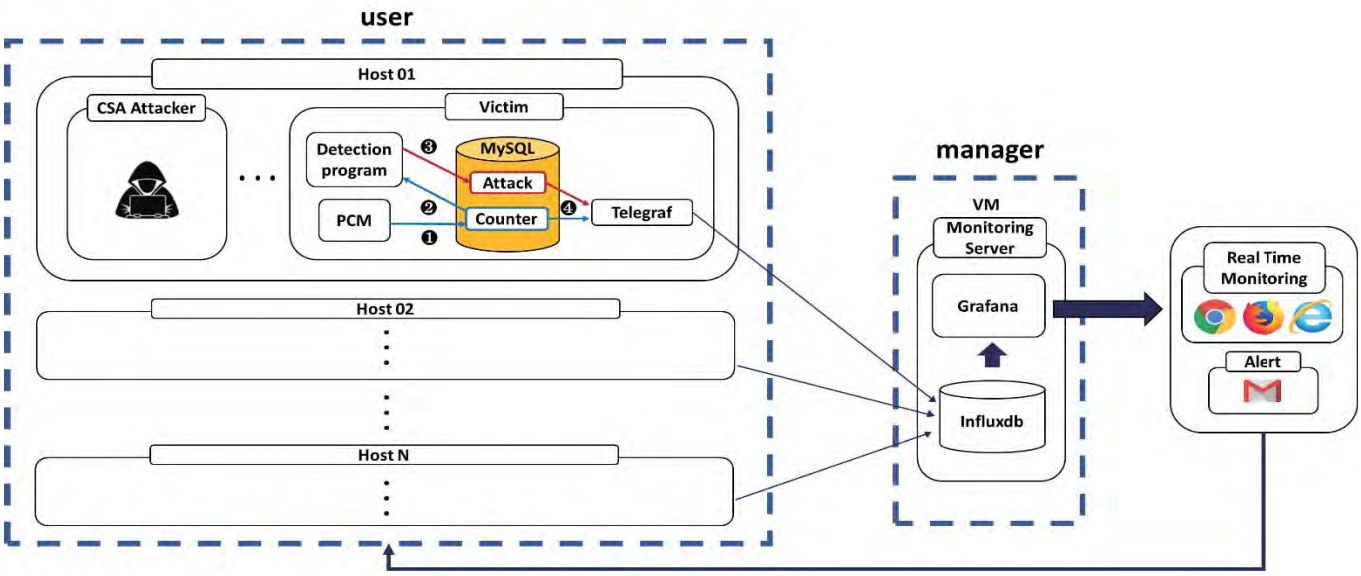
2. 배경지식

2.1 Grafana

Grafana 는 데이터소스(e.g., Cache Miss)를 모니터링 및 관찰하기 위한 오픈 소스 플랫폼이며 시각화를 위해 Graphite, Prometheus, Elasticsearch, OpenTSDB 및 Influxdb 등을 지원한다[1]. 특히, 본 논문에서는 실시간 시각화를 위해 가장 널리 사용되는 시계열 데이터베이스 Influxdb 와 데이터 수집 방법으로 입력 플러그인(e.g., MySQL)과 출력 플러그인(e.g., Influxdb)을 간단한 설정으로 사용 가능한 Telegraf 를 활용하였다[2]. 즉, Telegraf 에서 데이터 수집한 것을 Influxdb 에 저장하여 Grafana 대시보드에 나타낼 수 있다. 또한, Grafana 는 시각화뿐만 아니라 특정 지표에 대한 경고 규칙을 정의하고, 지속적으로 Slack, SMS 와 Email 같은 시스템에 알림을 보낼 수 있다.

2.2 Performance Counter Monitor (PCM)

Performance Counter Monitor(PCM)는 인텔 프로세서 내부의 특수 레지스터를 이용하여 이벤트 값(e.g., Cache Miss)을 실시간으로 관찰할 수 있는 도구이다[3].



(그림 1) 모니터링 시스템 구조

프로세스의 이벤트 변화율은 캐시 부채널 공격을 탐지하는 데 사용된다. Intel PCM 은 컴파일되지 않은 소스코드를 제공하기 때문에 내부 동작을 사용자가 변경할 수 있으며 간단하게 컴파일하여 실행파일로 이용할 수 있다.

2.3 캐시 부채널 공격(Cache Side Channel Attack)

2.3.1 FLUSH+RELOAD 공격

FLUSH+RELOAD[4] 공격은 공격자와 희생자가 공유하는 L3 캐시 라인을 대상으로 하는 공격이다. 공격은 크게 3 단계로 이루어져 있다. 첫 번째로, 공격자는 공격자와 희생자가 공유하는 캐시 라인을 clflush 명령어를 사용하여 L1, L2, L3 캐시에서 모두 비워준다. 두 번째로, 공격자는 희생자가 해당 캐시 라인에 접근할 때까지 기다린다. 마지막으로, 공격자는 다시 해당 캐시 라인에 접근하여 데이터를 로드 한다. 프로세서는 최근에 사용한 데이터를 캐시에 저장한다. 데이터가 캐시 메모리에 존재한다면 프로세서는 메인 메모리에 접근하지 않고 캐시 메모리에서 데이터를 바로 가져올 수 있다. 따라서 데이터를 가져오는 데 있어서 메인 메모리보다 더 짧은 시간 내에 가져온다. 공격자가 데이터를 로드 하는 시간이 느리다면 희생자가 해당 캐시 라인에 접근한 것이고, 반대로 시간이 빠르다면 희생자가 접근하지 않은 것이다. FLUSH+RELOAD 공격은 이러한 시간 차이를 통하여 희생자의 데이터를 알아낼 수 있다.

2.3.2 FLUSH+FLUSH 공격

FLUSH+FLUSH[5] 공격은 FLUSH+RELOAD 공격과 방식이 유사하다. 하지만 세 번째 단계에서 로드를 하는 대신 다시 clflush 명령어를 실행하여 캐시 라인을 비워준다. 캐시 라인에 데이터가 존재한다면 데이터가 존재하지 않을 때보다 캐시를 비우는 데 오랜 시간이 걸린다. 따라서 세 번째 단계에서 시간이 오래 걸린다면 희생자가 해당 캐시 라인에 접근한 것이고, 시간이 오래 걸리지 않는다면 희생자가 접근하지 않은 것이다. FLUSH+FLUSH 공격도 이러한 시간차를 통하여 희생자의 데이터를 알아낼 수 있다.

2.3.3 PRIME+PROBE 공격

PRIME+PROBE[6] 공격은 앞서 설명한 2 개의 공격과 달리 공격자와 희생자가 공유하는 L3 Cache Set 을 대상으로 공격한다. PRIME+PROBE 공격은 크게 3 단계로 이루어져 있다. 첫 번째로, 공격자는 자신의 데이터로 공유 Cache Set 들을 채운다. 두 번째로, 공격자는 희생자가 실행하는 동안 기다린다. 마지막으로, 공격자는 다시 자신의 데이터를 실행하여 로드 하는 시간을 측정한다. 이때 희생자가 Cache Set 에 접근했다면 Cache Set 은 희생자의 데이터로 채워지면서 공격자의 데이터는 evict 된다. 따라서 공격자가 다시 로드하였을 때 시간이 오래 걸린다. 반면에 희생자가 접근하지 않았다면 시간이 오래 걸리지 않는다. PRIME+PROBE 공격은 이러한 시간 차이를 통하여 희생자의 데이터를 알아낼 수 있다.

2.4 Softmax Classification

여러 공격 중에서 어떠한 공격이 진행되었는지 판단하기 위해 다중 클래스 분류인 Softmax Classification 을 사용하였다. Softmax Classification 의 가설 함수 H(x)는 입력 데이터 x 에 대하여 가중치(W)를 곱하고 편향(b)을 더한 값이다. H(x)가 Softmax 함수 S_i의 입력값이 되어 나온 값이 예측값이 된다. Softmax 함수란 분류해야 하는 클래스의 총 개수를 k 라고 하면, k 차원의 벡터를 입력받아 각 클래스에 대한 0~1 사이의 확률값을 구한다. 이와 같이 확률적인 결과 값을 가지고 높은 확률을 가지는 클래스가 예측값이 된다. 가설 함수를 통하여 구한 예측값과 실제 값을 기반으로 비용 함수 Cost(W)를 구한다. Softmax Classification 의 비용 함수는 각 클래스에 대한 예측값과 실제값의 차이를 모두 더한다. 따라서 비용 함수는 예측값이

(그림 2) Grafana 를 통한 각 호스트 별 캐시 부채널 공격 전과 후의 이벤트 값들과 탐지 결과값 변화

실제값과 유사할수록 0 에 가까워지고, 다를수록 값이 커지게 된다. 비용 함수가 최소가 되도록 W, b 의 값을 찾음으로써 가장 적절한 예측을 할 수 있는 가설 함수를 구할 수 있다.

$$H(x) = Wx + b$$

$$S_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}} \quad for \;\; i = 1,2,\dots,k$$

3. 프레임워크

효율적으로 캐시 부채널 공격 탐지를 위해 Intel PCM 과 기존의 탐지 프로그램[7]을 개선하고 Telegraf, Influxdb, Grafana 를 사용하여 실시간 모니터링이 가능하도록 프레임워크를 제작하였다.

(그림 1)과 같이 관리자 서버는 가상 머신이며 Influxdb 와 Grafana 로 이루어져 있고, 사용자 서버는 MySQL, Telegraf, Intel PCM 그리고 탐지 프로그램으로 구성되어 있다. 프레임워크는 사용자와 관리자 서버로 구분되며, 관리자 서버를 통해 호스트간 접근이 불가하여 보안상 안전하다.

3.1 데이터 수집 및 저장 방법

탐지 프로그램은 PCM 을 동작시키고, (그림 1)과 같이 MySQL 로부터 이벤트값들을 읽어와 어떠한 공격을 받았는지에 대한 결과를 MySQL 에 저장한다. 구체적으로 탐지 프로그램이 PCM 을 실행시키면 MySQL 의 'Counter' 테이블에 이벤트값들이 저장된다. 탐지 프로그램은 탐지에 필요한 데이터들을 읽어 오기 위해 반복문을 통하여 'Counter' 테이블에서 실시간으로 값을 가져온다. 해당 데이터들은 Softmax Classification 기법으로 훈련된 머신러닝 모델의 입력값으로 넣어 주어 예측 결과를 출력한다. 모델은 이벤트값들을 토대로 어떠한 공격도 하지 않은 상태를 0, FLUSH+RELOAD 공격은 1, FLUSH+FLUSH 공격은 2, PRIME+PROBE 공격이 진행된 경우는 3 으로 출력하며 이를 MySQL 의 'Attack' 테이블에 저장한다. 최종적으로 MySQL 은 저장된 값들(Attack, Counter)을 Telegraf 를 통하여 대시보드에 나타낼 데이터들을 수집한다.

Telegraf 는 input plugin 으로 MySQL 을 사용하여 Intel PCM 의 이벤트 값과 탐지 프로그램의 결과값을 수집하였다. 이때, Telegraf 의 설정파일을 통해 데이터 수집 주기 1 초, 데이터 전송주기 1 초로 정해주었으며 수집한 데이터를 저장하기 위해 output plugin 은 시계열 데이터베이스 Influxdb 로 설정해주었다.

각 사용자 서버는 (그림 1)과 같이 Telegraf 또는 데이터 수집에 필요한 프로그램 등을 백그라운드로 실행하여 모니터링 서버의 Influxdb 에 데이터를 보낸다. Influxdb 는 데이터를 HTTP 로 받아들이기 때문에 Telegraf 설정 파일 중 output plugin 의 URL 부분을 Influxdb 주소로 변경해주어야 한다. 관리자 서버는 가상 머신이기 때문에 브릿지를 통해 네트워크 대역을 재설정해준 후 모든 사용자가 데이터를 전송할 수 있도록 포트 포워딩이 필요하다.

3.2 대시보드 설정 및 알람

Grafana 는 데이터소스로 Influxdb 를 선택함으로써

이벤트값과 탐지프로그램 [7] 결과값을 대시보드에 나타낼 수 있다. (그림 2)와 같이 여러 쿼리를 만들어 주어 다양한 값들을 대시보드에 그래프로 나타내 주었다. 하나의 호스트당 4 개의 대시보드를 가지도록 구성하였으며 Grafana 의 alert 기능을 사용하여 캐시 부채널 공격이 들어왔을 때 사용자 메일로 관리자 서버에서 경고 메시지를 보내도록 설정했다. 또한, 관리자 서버의 Grafana 대시보드를 모든 사용자가 웹 브라우저로 확인할 수 있도록 브릿지 설정과 포트 포워딩해 주었다.

4. 실험

4.1 캐시 부채널 공격에 따른 PCM 값의 변화

Intel PCM 값들의 변화를 통하여 캐시 부채널 공격들을 탐지할 수 있었다. 캐시 부채널 공격들은 수행하는 명령어에 비해 많은 cycle 이 소요되기 때문에 IPC(Instruction Per Cycle) 값이 감소하지만 공격 코드가 수행하는 반복문으로 인해 Branch 값은 증가한다. 하지만 캐시 부채널 공격들은 Cache Miss 로 구분될 수 있다. 첫 번째, FLUSH+RELOAD 공격은 공격자가 희생자와 공유하는 L3 캐시 라인을 비우고 reload 를 반복한다. 따라서 공격이 진행되는 동안 모든 Cache Miss 값들이 급격하게 증가한다. 두 번째, PRIME+PROBE 공격은 PROBE 시 대부분 자신의 코드와 데이터들로 이루어진 Cache Set 들을 reload 하기 때문에 L3 Cache Miss 값 변화는 크게 없다. 하지만 L1, L2 Cache 는 구조상 L3 에서 reload 하고자 하는 Cache Set 들을 한 번에 가져올 수 없으므로 L1, L2 Cache Miss 값은 급격하게 증가한다. 마지막 FLUSH+FLUSH 공격은 공격자가 희생자와 공유하는 L3 캐시 라인을 reload 하지 않고 비워주기만 하므로 Cache Miss 값에는 큰 변화가 없다. 따라서 해당 이벤트값들로 공격을 탐지하고 Cache Miss 를 통해 분류할 수 있다.

4.2 실험 결과 및 관찰

Intel® Core™ i5-7400, Intel® Xeon® E5-2620, Intel® Core™ i9-9900KF 총 3 개의 프로세서에서 각각 Intel PCM 값들을 읽어와 탐지 프로그램의 입력 값으로 넣어주고, 탐지 프로그램이 출력한 값을 가지고 어떠한 공격이 들어왔는지를 탐지하였다. (그림 2)는 캐시 부채널 공격 전과 후의 PCM 값들과 탐지 결과를 각각의 호스트별로 Grafana 에서 나타낸 것으로 공격이 10 초간 지속되는 경우 빨간색 라인이 생기며, 이를 기준으로 공격 전과 후를 구분할 수 있다. E5-2620 프로세서는 FLUSH+RELOAD 공격을, i9-9900KF 프로세서는 FLUSH+FLUSH 공격을, i5-7400 프로세서는 PRIME+PROBE 공격을 진행하였다. (그림 2)에서 E5-2620 프로세서를 보면 FLUSH+RELOAD 공격이 진행되는 동안 모든 Cache Miss 값들이 급격하게 증가하는 것을 볼 수 있다. 또한 i5-7400 프로세서를 보면 PRIME+PROBE 공격이 진행되는 동안 L3 캐시와 달리 L1, L2 의 Cache Miss 값이 증가하는 것을 볼 수 있다. 하지만 i9-9900KF 프로세서는 FLUSH+FLUSH 공격이 진행되는 동안 Cache Miss 값은 변화가 없는 것을 볼 수 있다. 또한 모든 캐시 부채널 공격들이 진행되는 동안 Branch 값은 증가, IPC 값은 감소하는 것을 볼 수 있으며 Attack 값의 경우 FLUSH+RELOAD 공격이 진행된 경우 1, FLUSH+FLUSH 공격이 진행된 경우 2, PRIME+PROBE 공격이 진행된 경우 3 으로 변경되는 것을 확인할 수 있다. 이를 기반으로 Counter 값들의 변화에 따라서 어떠한 공격을 받았는지를 알아낼 수 있었다. (그림 2)에서와 같이 호스트별로 Intel PCM 값들과 탐지한 공격을 Grafana 에서 실시간으로 나타내 주고 공격이 탐지될 경우 사용자에게 경고 메일을 보내준다.

5. 향후계획

현재 탐지 프로그램은 L1, L2, L3 Cache Miss 값들을 기반으로 캐시 부채널 공격들을 탐지하기 때문에 실제로 공격을 하지 않아도 Cache Miss 값이 증가하면 공격으로 탐지하는 경우가 있다. 따라서 이러한 오탐률을 줄일 수 있도록 현재 탐지 프로그램을 개선하려 한다.

6. 결론

본 논문에서는 각 호스트별로 Intel PCM 값들과 FLUSH+RELOAD, FLUSH+FLUSH, PRIME+PROBE 중에서 어떠한 캐시 부채널 공격이 실행되었는지에 대한 데이터들을 실시간으로 Grafana 를 통하여 나타내어 주고 공격을 받은 호스트에게 메일을 통하여 알려주는 방법에 대해서 설명하였다. 또한 Intel PCM 값들을 기반으로 어떠한 캐시 부채널 공격이 실행되었는지를 머신러닝을 통하여 탐지하는 기술에 대하여 설명하였다.

Acknowledgement

이 논문은 2019 년도 정부 (과학기술정보통신부)의 재원으로 정보통신기술진흥센터의 지원을 받아 수행된 연구임 (2019-0-00533, 컴퓨터 프로세서의 구조적 보안 취약점 검증 및 공격 탐지 대응)

참고문헌

[1] https://grafana.com/oss/grafana/ - Grafana Labs

[2] https://github.com/influxdata/telegraf/ - telegraf

[3] Intel® Performance Counter Monitor - A Better Way to Measure CPU Utilization

[4] Yarom Yuval, and Katrina E. Falkner. “Flush+Reload : a High Resolution, Low Noise, L3 Cache Side-Channel Attack”. USENIX Security, 2014.

[5] Daniel Gruss, Clémentine Maurice, Klaus Wagner, Stefan Mangard. “Flush+Flush : A Fast and Stealthy Cache Attack”. DIMVA, 2016.

[6] Liu, F.; Yarom, Y.; Ge, Q.; Lee, R.B. “Last-Level Cache Side-Channel Attacks are Practical”. IEEE Symposium on Security and Privacy, 2015.

[7] Jonghyen Cho, Taehun Kim, Soojin Kim, Miok Im, Taehyun Kim, and Youngjoo Shin “Real-Time Detection for Cache Side Channel Attack using Performance Counter Monitor”. Applied Sciences-Basel, 2020.