정 보 보 호 학 회 지
제 33 권 제5호, 2023. 10

메모리내 연산 기술의 클라우드 신뢰실행 관련 연구 전망

신 수 환*, 이 호 준**

요 약

오늘 날의 클라우드 워크로드는 인공지능 및 빅 데이터 활용의 비약적인 발전으로 인하여 메모리 대역폭이 프로세서의 연산 속도를 따라가지 못해 병목 현상을 겪고 있다. 이러한 이른바 메모리 벽 문제 (Memory Wall Problem)를 해결하기 위해 컴퓨터 아키텍처 및 운영체제는 변화해 나가고 있다. 그 중 최근 가장 주목 받는 기술 중 하나인 메모리내 연산 기술 (Processing-In-Memory)는 프로세서를 메모리 디바이스 내에 탑재함으로써, 데이터를 메인 프로세서에 이동시켜 처리할 필요 없이 데이터 내부에서 처리한다. 이로 인해 대용량 데이터의 처리속도 향상과 동시에 메인 메모리 버스의 부하를 줄여 클라우드 시스템의 전반적인 성능 향상 또한 꾀할 수 있다.

한편, 클라우드 아키텍처는 또 다른 요구에 의하여 변화를 거치고 있으며, 이는 바로 보안이다. 오늘 날의 컴퓨터 아키텍처 및 GPU등의 가속기들은 신뢰실행 기술 (Trusted Execution)의 지원을 통하여 클라우드에서의 민감한 연산을 격리 및 보호하고자 한다. Intel의 SGX와 NVIDIA GPU의 confidential computing기능 지원이 이러한 흐름을 대표한다.

최근 PIM을 활용한 보안기술의 새로운 방향들을 제시하는 연구들이 소개되고 있는 가운데, 본 논문은 클라우드 신뢰실행 (Trusted Execution)에서의 PIM을 적용한 최신 연구들의 방향을 소개하고 또한 향후 연구 전망을 제공하고자 한다. PIM기술의 동향과 PIM을 보안에 특화시킨 연구, 그리고 앞으로 해결되어야할 과제들을 논함으로써, 새로이 주목받는 PIM 기반 보안 기술들을 정리하고 향후 전망을 제공한다.

Ⅰ. 서 론

오늘 날의 클라우드는 인공지능 등의 발전으로 갈수록 초대용량의 데이터를 처리하는 연산의 효율적인 처리를 요구 받고 있다. 최근 CPU, GPU 와 NPU (Neual Processing Unit)등이 급격하게 발전하면서 연산 속도가 빨라지고 있다. 그러나, 기존 Von Neumann 아키텍처는 메모리와 프로세스가 분리되어 있어, 메모리의 속도가 프로세스의 연산 처리 속도를 따라 가지 못해 메모리 벽 문제 (Memory-wall problem)이 발생하고 있다.

메모리 벽 문제는 메인 프로세서 혹은 GPU, NPU 등의 가속기 속도의 비약적인 발전으로 메모리 디바이스의 대역폭이 연산 속도를 따라가지 못해 전체적인 데이터 처리 속도에 병목지점 (wall)으로 작용하게 되는 현상을 의미한다. 데이터가 처리되는 속도보다 메모리에서 메인 프로세서 및 가속기에 데이터 불러오거나 역으로 처리된 결과를 저장하는 속도가 느린 것이다. 이러한 문제는 데이터의 크기가 커질 수록 악화된다.

메모리 벽 문제를 해결하기 위해 메모리 내 연산 (Processing-In-Memory, PIM)기술이 개발되었다. PIM기술은, 메모리 벽 문제를 근본적으로 해결하기 힘든 기존 폰 노이만 컴퓨터의 구조의 대안으로써, 물리적으로 메모리와 프로세서를 분리하지 않고 통합되어 있도록 설계한 것이다. 이를 바탕으로 메모리가 데이터를 저장하고, 프로세서가 곧바로 데이터에 접근할 수 있도록 하여 데이터 처리 속도를 높이는 것을 목표로 한다.

이러한 PIM은 여러가지 특징을 가지고 있는데, 몇가지 기술을 통해 클라우드 신뢰환경에서도 빠르게 실행될 수 있는 환경을 구축할 수 있을 것으로 예상된다. 본 논문에서는 PIM의 하드웨어적인 구성을 살펴보고, 클라우드 신뢰실행의 보안 측면에서 진행된 연구들을 소개하고, 이후 진행 될 수 있는 연구 방향에 대해 논하고자 한다.

* 성균관대학교 소프트웨어학과 컨버전스 연구소 (대학원생, hong@paper.korean.ac.kr)
** 성균관대학교 (부교수, kim@hankook.re.kr)

Ⅱ. 배 경

2.1. PIM

Procesing-In-Memory(PIM)은 메모리 내부에서 기존의 Von Neumann 아키텍처와 다르게 프로세서와 메모리가 물리적으로 분리되지 않고 통합되어 있도록 설계한 것이다. 기존 아키텍처의 경우, 메모리는 정보의 저장만을 담당했으나 PIM에서는 각 메모리가 데이터를 저장하고 프로세서가 데이터에 접근할 수 있도록 한다.

PIM에서의 메모리는 데이터를 저장하고 동시에 연산을 수행할 수 있는 능력을 갖추고 있어 데이터 이동, 복사 등의 지연이 크게 감소하며, 성능 향상을 바탕으로 궁극적으로 메모리 병목 현상 등을 해결하는데 도움이 될 것으로 기대되고 있다. 구체적으로 PIM은 다음과 같은 장점들을 지니고 있다.

먼저, PIM을 활용하게 되면 데이터 동선이 단순화되고, 메모리 대역폭을 효율적으로 사용할 수 있는 효과를 가져온다. PIM은 데이터를 메모리와 프로세서 간에 주고받는 데 드는 시간과 에너지를 감소하는데 주 역할을 할 수 있다. 또한 데이터와 계산이 메모리 내에서 동시에 이루어지므로 데이터 처리 속도가 빨라진다. 이는 실시간 데이터 처리에 빠르게 사용될 수 있다.[1] 또한, PIM은 메모리 병목 현상을 줄이고 데이터 접근 속도를 향상시킬 수 있다.[2]

다음으로, PIM을 활용하게 되면 자원을 효율적으로 사용할 수 있게 된다. PIM을 활용하게 되면 이전에 사용되었던 기기들에 비해 저전력으로 사용할 수 있게 되며, 고밀도로 집적된 회로로 구성될 수 있어 작은 공간에 많은 처리 유닛을 넣을 수 있다.[3][4]

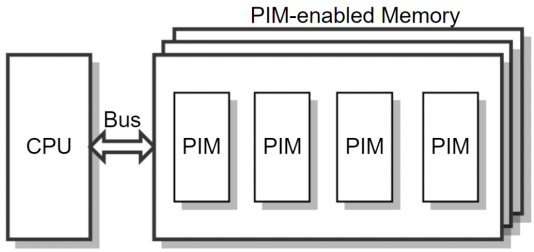
마지막으로, PIM은 메모리 내에서 데이터를 즉시 처리하기 때문에 데이터 노출 위험이 감소한다. 본 논문에서는 이러한 PIM의 장점들을 바탕으로 어떻게 보안 분야에서 PIM이 활용될 수 있을지 논할 것이다.

〔그림 1〕 UPMEM PIM 보드

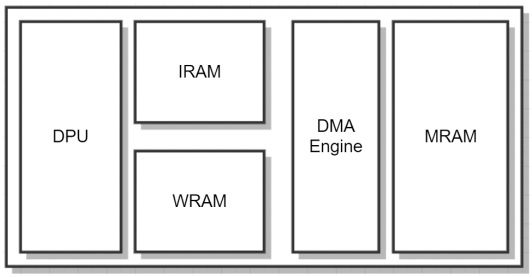
2.2. UPMEM

PIM은 국내외 다양한 기업들에서 현재 개발 및 상용화를 연구중이다. 이 중에서 본 논문에서는 UPMEM사에서 개발한 PIM(이하 UPMEM)을 다룬다.[5] UPMEM의 경우 PIM을 상용화 하여 국내외 여러 연구가 UPMEM을 바탕으로 진행되어지고 있다.[6]-[8]

현재 상용화된 UPMEM은 랭크 단위로 구분되어 있으며, 총 32개의 랭크와 4096개의 DPU (Data Processor Unit)로 구성되어 있다. 그림 2는 UPMEM 내부의 각 랭크별로 할당되어있는 PIM을 보여주며, 그림 3은 각 PIM이 어떻게 구성되어 있는지를 보여준다. 각 PIM은 DPU, IRAM, WRAM, DMA 엔진으로 구성되어 있으며, DMA 엔진은 각 부위간의 통신을 중개하는 역할을 한다. WRAM은 PIM을 활용할 때 스크래치 패드로 활용하게 된다.



〔그림 2〕 UPMEM 내 PIM 메모리 위치



〔그림 3〕 UPMEM 내 각 PIM의 구성

Ⅲ. PIM 기술의 클라우드 신뢰실행 관련 연구

본 부문에서는 PIM의 성능을 측정한 연구에 대해 살펴보고, 해당 연구에서 밝혀진 PIM의 특성에 맞추어 이루어진 PIM이 클라우드 컴퓨팅에 사용될 수 있는 방법에 대한 연구에 대해 소개한다.

〔표 1〕 클라우드 신뢰 실행 환경 관련 연구

연구 제목	목적	하드웨어	연구결과
SE-PIM〔6〕	큰 데이터를 활용할 수 있는 신뢰 실행 환경 구축	UPMEM	메모리 접근 패턴 유추 공격 방어
UPMEM 성능 측정〔9〕	PIM의 성능 측정	UPMEM	간단한 연산에서 CPU보다 빠르나, 복잡한 연산에서 느린 차이를 보임
Secure DIMM〔16〕	Path ORAM의 가속화	자체 하드웨어	기존 ORAM에 비해 1.9배 빠른 속도
O-PIM〔7〕	PIM을 활용한 ORAM의 가속화	UPMEM	data-intensive 한 workload에서 CPU 대비 20% 향상을 보임

3.1. PIM의 성능 측정

UPMEM의 성능을 측정한 연구[9]에 따르면, 데이터 복사 등의 간단한 연산을 수행하였을 때는 기존 CPU만으로 이루어진 환경보다 파일의 크기에 따라 최대 12.5배 빠른 속도를 보여주었다. 그러나, 암호화 등의 복잡한 알고리즘이 들어간 작업을 실행할 때는 CPU보다 최대 4배 느린 속도를 보여주었다. 이를 바탕으로 생각해 볼 때, PIM을 클라우드에 신뢰환경 목적으로 활용 할 때는 간단한 연산을 바탕으로 설계해야 함을 알 수 있다.

3.2. 신뢰 실행 환경 관련 연구

3.2.1. 신뢰 실행 환경

신뢰 실행 환경(Trusted Execution Environment)은 민감한 데이터와 코드를 신뢰하지 못하는 환경으로부터 격리하여 안전한 실행을 확보하는 기술이다. 이러한 클라우드에서 사용 가능한 대표적인 신뢰 실행 환경의 예시로는 Intel SGX가 존재한다. SGX는 EPC(Enclave Page Cache)라는 보호된 메모리 페이지를 제공하고, 해당 영역을 메모리에서 자동으로 암호화함으로써 프로그램 및 처리된 데이터를 SGX 내부에서 보호하게 된다. [10]

이러한 신뢰 실행 환경의 가장 큰 장벽은 메모리 제한이다. SGX의 경우 EPC의 크기는 128MB 밖에 되지 않아, 큰 데이터의 경우 작은 데이터로 잘라 처리해야 한다. 따라서, 큰 데이터를 처리할 때는 일반 CPU에서 처리할 때와 SGX에서 처리할 때 1000배가 넘는 차이가 나는 경우도 있다고 한다. [11]

3.2.2. PIM을 활용한 신뢰 실행 환경 연구

SE-PIM[6] 연구의 경우 PIM을 활용하여 신뢰 실행 환경에서 큰 데이터를 활용하고자 하였다. SE-PIM은 사용된 데이터의 무결성과 기밀성을 보호하고자 하고, 사용한 데이터에 대한 접근 패턴을 파악하지 못하게 하는 것이 목적이다. 이러한 방어 목적 하에, 공격자는 물리적 할 수 있고, 악성코드 등으로 OS가 손상될 수 있는 가능성, 그리고 메모리의 버스 공격으로 인해 정보를 유출하는 공격이 가능하다고 가정하였다.

SE-PIM은 위에 언급된 공격 방법에 의한 방어를 달성하기 위해 일반적인 PIM 모델에 여러 기능을 추가하였다. 먼저, DRAM Lockdown 유닛을 활용하여 DRAM을 외부에서 무단 접근하지 못하도록 하고, AES/DMA 엔진을 바탕으로 암호화된 데이터 전송이 가능하도록 하였다.

또한 보호되고 있는 각 PIM에 접근하기 위해서는 원격 인증(Remote Attestation) 기술을 이용하여 호스트 CPU Enclaves 만 보호되고 있는 PIM에 접근 가능하도록 하였다.

실험 결과, CPU만으로 실행된 상황을 공격했을 때

는 메모리 내 특정 위치에 접근했다는 사실을 파악할 수 있었지만, PIM으로 실행된 상황을 동일한 방법으로 공격하였을 때는 단일 채널 메모리에 매핑되어 있어 접근한 위치를 특정할 수 없었다. 그러나 기존 PIM에 비해 접근 시간이 약간 늘어났으며, 마이크로 벤치마크로 실험한 결과 일반적인 접근에 비해서는 살짝 느린 데이터 처리 속도를 보여주었다.

3.3. ORAM 관련 연구

3.3.1. ORAM

Oblivious RAM (ORAM) 알고리즘[12]은 사용자가 믿을 수 없는 클라우드를 이용할 때, 해당 클라우드에서 발생할 수 있는 부채널 공격을 방어하는 알고리즘이다. 이러한 알고리즘은 정보 접근 패턴을 파악할 수 없도록 해, 부채널 공격이 성공하지 못하도록 하는 역할을 한다.

PATH ORAM[13]은 현재까지 알려진 가장 단순한 알고리즘이다. PATH ORAM은 Statsh와 정보를 저장하는 이진 트리로 구성되며, 정보를 임시로 저장하는 역할을 하게 되는 Statsh는 사용자에게, 그리고 정보를 저장하게 되는 이진 트리는 서버에 존재하게 된다. 사용자가 정보를 PATH ORAM에 저장하게 되면, 이는 무작위하게 이진 트리 상에 저장되게 된다. 각 정보가 어디에 저장되었는지는 포지션 맵에 저장되게 되는데, 이는 신뢰할 수 있는 공간에 따로 저장되게 된다.

그림 4는 PATH ORAM의 Access 알고리즘을 보여준다. 서버가 사용자에게 정보 접근 요청을 받게 되면 해당 알고리즘을 거치게 된다. 접근 요청이 들어온 블록의 ID를 바탕으로 포지션 맵에서 해당 정보의 주소를 찾아 저장하게 되고, 이후 해당 블록의 주소를 무작위하게 재선정하고 해당 주소에 블록을 옮겨 저장하게 된다. 즉, 한번 접근한 데이터의 경우 다시 같은 주소에 저장되지 않게 되는 것이다. 만약 사용자가 쓰기 요청을 하였을 경우, Statsh에 있는 데이터를 교체하고, 해당 Statsh를 바탕으로 PATH ORAM의 이진 트리를 업데이트하게 된다.

Algorithm 1 PathORAM [23]'s Access(op, a, data*) :

1: $x \leftarrow position[a]$
2: $position[a] \leftarrow UniformRandom(0 \dots 2^L - 1)$
3: **for** $l \in \{0, 1, \dots, L\}$ **do**
4: $\quad S \leftarrow S \cup ReadBucket(P(x, l))$
5: **end for**
6: $data \leftarrow$ Read block a from S
7: **if** $op = write$ **then**
8: $\quad S \leftarrow (S - \{(a, data)\}) \cup \{(a, data^*)\}$
9: **end if**
10: **for** $l \in \{L, L-1, \dots, 0\}$ **do**
11: $\quad S' \leftarrow \{(a', data') \in S : P(x, l) = P(position[a'], l)\}$
12: $\quad S' \leftarrow$ Select $min(|S'|, Z)$ blocks from S'
13: $\quad S \leftarrow S - S'$
14: $\quad WriteBucket(P(x, l), S')$
15: **end for**
16:
17: **return** $data$

〔그림 4〕 PATH ORAM Access Algorithm

3.3.2. PIM을 활용한 ORAM 연구

Secure DIMM[14]은 PATH ORAM을 바탕으로 새로운 하드웨어를 만들어 ORAM을 가속화 하려고 시도한 모델이다. 이 연구에서는 CPU는 안전하다는 가정 하에 메모리에 물리적 부 채널 공격이 가능하다는 가정하에, 메모리 접근 패턴을 파악하지 못하게 하기 위한 목적으로 연구되었다. 해당 연구에서 CPU와 각 SDIMM간의 버스는 안전하다고 가정하였다.

이를 해결하기 위해 PIM과 유사한 새로운 하드웨어를 만들었으며, 이를 SDIMM이라고 명명하였다. 하드웨어 내부에 들어가는 SDIMM의 개수만큼 ORAM을 쪼개어 각 SDIMM에 할당하였고, CPU가 안전하다는 가정하에 포지션 맵을 CPU에 위치시켰다. 또한, 각 SDIMM에 보안 버퍼 (Secure Buffer)을 위치시켜, 해당 버퍼에서 암호화와 복호화를 할 수 있도록 설계하였다.

해당 연구의 실험 결과, Freecursive ORAM과 비교하였을 때, 1.9배 빠른 속도를 보였고, 2.55배 전력을 아낄 수 있던 것으로 파악되었다.

O-PIM[7]은 상용화된 PIM, UPMEM을 바탕으로 ORAM을 가속화 하려고 하였다. O-PIM은 현재까지 개발된 가장 간단한 ORAM 알고리즘인 PATH ORAM을 사용하였다. O-PIM에서는 공격자가 물리적 하드웨어 부채널 공격 및 소프트웨어적인 공격을 할 수 있다고 가정하였다. 해당 공격을 통해 공격자는 데이터 접근 패턴을 파악하는 것을 목적으로 하고, O-PIM은 이를 방어하는데 그 목적을 둔다. 또한 CPU와 메모리 사이의 버스에 타이밍 공격을 하여 접근하는 메모리를 파악하거나, 버스 탈취 공격 등을 하는 경우를 가정하였다.

이를 위해 O-PIM은 UPMEM 내부의 WRAM과 IRAM을 신뢰실행환경 하에 있어 믿을 수 있다고 가정하였다. 따라서, 유출되어서는 안되는 포지션 맵을 공격자가 정보를 얻어낼 수 없는 WRAM에 위치하였다.

O-PIM은 유저가 데이터를 입력하게 되면 먼저 데이터를 분리된 ORAM에 들어갈 수 있도록 분리하고, 해당 분리된 데이터를 무작위하게 주소를 배정하여 ORAM에 업데이트 하게 된다.

해당 연구의 실험 결과, 일반적인 환경 하에서는 O-PIM이 PATH ORAM보다 16배정도 느린 결과가 나왔다. 그러나, 바쁜 클라우드 환경을 가정한 경우, CPU보다 DPU의 처리속도가 빠르기 때문에 CPU는 5.8배 가량 느려진 방면 DPU는 1.22배 느려졌다. 이는 실제 클라우드 환경에서 ORAM을 적용해야 할 때, 개선된 O-PIM이 활용될 수 있는 방안을 보여주었다고 할 수 있다.

Ⅳ. 신뢰실행 환경과 PIM의 미래 연구 방향에 관한 제언

본 논문에서는 PIM을 활용한 클라우드 신뢰환경 관련 연구들에 대해서 살펴보았다. PIM은 데이터 이동 등의 간단한 연산을 할 때 기존 CPU보다 빠른 속도를 보여 여러가지 보안 대책에서 사용할 수 있을 것이라고 예상된다. 클라우드 신뢰 실행 환경을 구축할 때, 큰 데이터를 다룰 때 SE-PIM 같은 연구를 활용하여 메모리를 늘릴 수 있을 것이며, 혹은 데이터의 읽고 쓰기가 많을 때, ORAM 관련 연구와 PIM을 활용하여 클라우드를 구축할 수 있을 것이다.

하지만 현재까지 상용화된 ORAM의 한계점도 존재한다. 현재 상용화된 ORAM의 경우 DPU에서의 연산 속도 자체에는 문제가 없으나 DPU에서 CPU로 복사를 할 때 느린 문제가 있다. 또한, 소프트웨어적인 부분의 방어와 여러가지 물리적 공격의 방어를 위해 암호화 및 복호화가 요구되는 부분이 많으나 현재 PIM은 복잡한 연산이 들어가게 되면 매우 느려지는 단점이 있다. PIM을 위한 암호화 및 복호화 방식을 개발하는 것도 좋은 미래 연구 과제가 될 것이다.

Ⅴ. 결 론

클라우드 환경에서 PIM의 활용은 신뢰환경 구축에 여러가지로 사용될 수 있다. 본 논문에서는 현재까지 이루어진 연구를 신뢰실행환경 관련 연구와 ORAM 관련 연구로 나누어 살펴보았다.

현재까지 상용화된 PIM은 보안의 관점에서 봤을 때 장점과 단점을 동시에 가지고 있다. PIM이 가지고 있는 몇 가지 단점을 보완할 수 있는 연구가 진행된다면, PIM을 보안 분야에서 실용적으로 사용할 수 있게 될 것이다.

참 고 문 헌

[1] H. Zhang, G. Chen, B. C. Ooi, K. -L. Tan and M. Zhang, "In-Memory Big Data Management and Processing: A Survey," in IEEE Transactions on Knowledge and Data Engineering, vol. 27, no. 7, pp. 1920-1948, July 2015.

[2] Kazi Asifuzzaman, Narasinga Rao Miniskar, Aaron R. Young, Frank Liu, Jeffrey S. Vetter, A survey on processing-in-memory techniques: Advances and challenges, Memories - Materials, Devices, Circuits and Systems, Volume 4, Jul. 2023.

[3] Mohsen Imani, Saransh Gupta, and Tajana Rosing. Ultra-Efficient Processing In-Memory for Data Intensive Applications. In Proceedings of the 54th Annual Design Automation Conference 2017 (DAC '17). Association for Computing Machinery, New York, NY, USA, Article 6, pp. 1–6, Jun. 2017.

[4] Mittal, S. A Survey of ReRAM-Based Architectures for Processing-In-Memory and Neural Networks. Mach. Learn. Knowl. Extr. 1, pp. 75-114, April 2018.

[5] UPMEM, "UPMEM PIM solution", https://www.upmem.com/technology/

[6] K. D. Duy and H. Lee, "SE-PIM: In-Memory Acceleration of Data-Intensive Confidential Computing," in IEEE Transactions on Cloud Computing, vol. 11, no. 3, pp. 2473-2490, 1 Jul. 2023.

[7] 신수환, 이호준. "PIM을 활용한 ORAM 가속화 연구", 정보보호학회논문지, 33(2), pp. 235-242,

Apr. 2023.

[8] J. Gomez-Luna, I. E. Hajj, I. Fernandez, C. Giannoula, G. F. Oliveira and O. Mutlu, "Benchmarking a New Paradigm: Experimental Analysis and Characterization of a Real Processing-in-Memory System," in IEEE Access, vol. 10, pp. 52565-52608, May 2022.

[9] J. Nider et al., 'A case Study of Processing-in-Memory in off-the-Shelf Systems', in 2021 USENIX Annual Technical Conference (USENIX ATC 21), pp. 117-130. Jul. 2021.

[10] Somnath Chakrabarti, Thomas Knauth, Dmitrii Kuvaiskii, Michael Steiner, Mona Vij, Chapter 8 - Trusted execution environment with Intel SGX, Responsible Genomic Data Sharing, Academic Press, pp. 161 - 190. 2020.

[11] S. Sasy, S. Gorbunov, and C. W.Fletcher, “ZeroTrace : Obliviousmemory primitives from Intel SGX, ”Proceedings 2018 Network and Distributed System SecuritySymposium, Jan. 2018.

[12] O. Goldreich, “Towards a theory ofsoftware protection and simulation by Oblivious Rams,” Proceedings of the nineteenth annual ACM conference on Theory of computing, pp. 182-194,Jan. 1987

[13] E. Stefanov, M. van Dijk, E. Shi, C.Fletcher, L. Ren, X. Yu, and S.Devadas, “Path Oram,” Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, pp.1-26, Apr. 2013.

[14] A. Shafiee, R. Balasubramonian, M.Tiwari, and F. Li, “Secure DIMM:Moving ORAM Primitives Closer to Memory,” in 2018 IEEE International Symposium on High Performance Computer Architecture (HPCA), pp.428 – 440, Feb. 2018.

〈저 자 소 개〉

신 수 환 (Suhwan Shin)

학생회원

2022년 2월 : 성균관대학교 소프트웨어학과 졸업

2022년 3월~현재 : 성균관대학교 소프트웨어학과 석사과정

<관심분야> 시스템보안, 하드웨어 보안, 클라우드 AI 보안

이 호 준 (Hojoon Lee)

정회원

2010년 12월 : The University of Texas at Austin 전자컴퓨터공학 학사

2013년 8월 : KAIST 정보보호 석사과정

2018년 2월 : KAIST 정보보호 박사과정

2019년 9월 ~ 현재 : 성균관대학교 소프트웨어학과 조교수

<관심분야> 정보보호, 프로그램 분석, 소프트웨어보안, 시스템보안, TEE, 클라우드 AI보안