Secure Computing 을 위한 동형암호 가속기 연구에 대한 조망

정헌희 [1], 남기빈 [2], 이동주 [3], 백윤흥 [4]
[1] 서울대학교 전기정보공학부, 반도체 공동연구소
[2] 서울대학교 전기정보공학부, 반도체 공동연구소
[3] 서울대학교 전기정보공학부, 반도체 공동연구소
[4] 서울대학교 전기정보공학부, 반도체 공동연구소

honeyjung@snu.ac.kr, kvnam@sor.snu.ac.kr, djlee@sor.snu.ac.kr, ypaek@snu.ac.kr

A Survey on Homomorphic Encryption Acceleration Technology for Secure Computing

Heonhui Jung[1], Kevin Nam[2], Yun-Heung Paek [3]
[1]Dept. of Electrical and Computer Engineering and
Inter-University Semiconductor Research Center (ISRC), Seoul National University
[2]Dept. of Electrical and Computer Engineering and
Inter-University Semiconductor Research Center (ISRC), Seoul National University
[3]Dept. of Electrical and Computer Engineering and
Inter-University Semiconductor Research Center (ISRC), Seoul National University
[4]Dept. of Electrical and Computer Engineering and
Inter-University Semiconductor Research Center (ISRC), Seoul National University

요 약

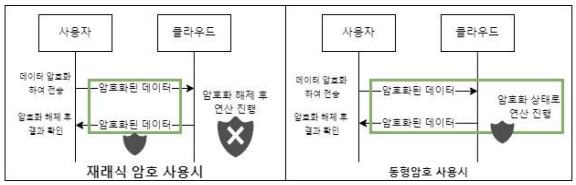
증가하는 클라우드 서비스에 대한 보안 위협에 대응하기 위해 데이터를 암호화한 상태로 연산을 수행하는 동형암호에 대한 연구가 활발히 진행되고 있다. 동형암호는 암호화된 상태로 처리를 진행하기에 데이터 유출 위협으로부터 자유로우나, 2^{10} 차 이상의 다항식에 대해 연산을 수행해야 되기에 많은 양의 연산 자원과 메모리 자원을 필요로 한다. 본 논문에서는 이러한 동형암호 연산의 특성과 이를 실용적으로 사용하기 위한 가속기 개발 연구들에 대해 서술하였다.

1. 서론

통신 및 반도체 기술의 발전으로 인해 이전엔 불가능했던 여러 AI, 클라우드 서비스들이 생겨나게 되었다. 클라우드 서비스의 경우, 기본적으로 사용자가 연산을 클라우드로 offloading 하는 식으로 발달하게 되었으며, 특히 MLaaS 서비스의 경우, 서비스 제공자 측에서는 AI 모델을 훈련시키기 위해 많은 시간과 비용이 필요하기에 모델을 외부로 반출하는 것을 매우 꺼려한다.[1] 따라서 현재 상용화 된 많은 클라우드 및 MLaaS 서비스들은 개인의 데이터들을 수집하여 클라우드로 전송하여 연산을 진행하고 그 결과를 다시 개인에게 보내주는 식으로 활용되고 있다.

하지만 클라우드 컴퓨팅 서비스가 증가함에 따라 크고 작은 정보 유출의 피해가 커져가고 있으며, IBM 에 따르면, 2024 년, 전 세계적으로 데이터 유출로 인해 미화 488 만 달러의 비용이 발생하였으며, 이전 년도에 비해 10% 증가하였다.[2] 정보 유출에 대한 피해를 막기 위해 재래식 암호화를 사용하여 클라우드와 사용자 간 정보를 주고받는 단계를 암호화하여 최대한 외부로의 정보 유출을 막고 있으나[3], 여전히 클라우드 자체가 공격을 당하거나 내부자로 인한 유출이 일어날 경우, 대규모의 개인정보 유출을 피할 수 없다.

이런 문제를 해결하기 위해 완전 동형암호(Fully-Homomorphic Encryption)[4][5]의 개발과 함께 암호화된 상태에서 연산을 진행하여 정보 유출로부터 안전한 시스템을 만들고자 하는 연구가 진행되었다.[6]



(그림 1) 동형암호와 재래식 암호의 비교

이런 시스템의 경우, 클라우드 내에서도 암호화 상태가 유지되며 정보유출 사고가 일어났을 때도 정보가 안전히 보호될 수 있다는 장점이 있다. 하지만 동형암호를 사용한 상태로 연산을 진행할 경우, 동형암호 암호문 간의 큰 규모의 연산이 필요하며, 이는 같은 평문 연산에 비해 큰 규모의 연산 및 메모리 자원을 필요로 하게 된다. 이에 따라 FPGA 나 ASIC 를 활용한 전용 가속 플랫폼에 대한 연구가 활발히 이루어지고 있으며, 본 논문에서는 이러한 전용 하드웨어를 활용한 동형암호 가속기 설계 연구에 대해 알아보고자 한다.

2. 동형암호 연산의 특징

기본적으로 동형암호의 암호문은 (그림 2)와 같이 구성된다.

(그림 2) 동형암호 암호문의 구성

각 동형암호 scheme 별로 차이는 있으나 기본적으로 N 개의 항으로 이루어진 다항식 L 개가 모여 하나의 암호문을 구성하게 된다.

N 은 암호문의 보안성에 크게 관여하는 parameter 이며, 동형암호에서 일반적으로 사용되는 128-bit security 를 보장하기 위해서는 2^{14} 이상을 사용하여야 한다[7]. L 은 하나의 암호문에 들어있는 전체 다항식의 개수에 관여하며, 큰 값을 사용할수록 후술할 *bootstrapping* 을 적게 할 수 있어 데이터에 대해서 연산을 많이 해야 할 경우 유리하다.

연산해야 되는 각 암호문이 다항식으로 이루어져 있기 때문에 벡터 연산이 주가 되며, 앞서 본 2^{14} 개 이상의 항 수를 가지고 있기에 큰 연산 자원을 필요로 한다. 또한 후술할 *keyswitching* 및 *bootstrapping* 과정에서 다항식 간 곱셈이 자주 발생하게 되며, 큰 차수의 다항식 간의 컨벌루션 연산을 피하기 위해 NTT(Number Theoric Transform)나 FFT(Fast Fourier Transform)가 사용되며, 이는 상당한 연산 부하를 일으키게 된다.[8]

동형암호 암호문에서 연산을 진행하면 할수록 암호문에 오차가 계속해서 축적되게 된다. 이를 계속해서 방치하고 연산을 진행할 경우, 복호화 시에 결과를 확인할 수 없게 된다. 따라서 오차가 많이 축적되었을 경우, *bootstrapping* 이란 과정을 통해서 축적된 오차를 떨어뜨릴 필요가 있다. *bootstrapping* 알고리즘은 대부분 필요로 하는 연산량이 매우 크며, *keyswitching* 및 *rotation* 과정을 추가로 필요로 한다. 이 과정에서, 암호문뿐만 아니라 각 과정에 필요한 특별한 다항식들의 집합인 공개키, 즉 evaluation key 가 필요하며, 공개키의 다항식과도 NTT 를 활용한 다항식 곱셈을 필요로 하기에 큰 연산 부하와 evaluation key 를 불러오기 위한 메모리 부하를 일으킨다.

3. 동형암호 가속기 연구 동향

2 절에서 확인한 동형암호 연산의 특징들을 통해 벡터 및 다항식 연산과 NTT/FFT 연산이 주가 된다는 것을 알 수 있으며, 현재 진행되어 있는 가속기 연구들도 이러한 특징을 이용하여 아래 표와 같이 가속을 시도하고 있다.

		[9]F1	[10]Craterlake	[11]BTS
Parameter	N	2^14	2^16	2^17
	L	24	60	44
	q size(bits)	32	28	64
HW	Area(mm^2)	151.4	472.3	373.6
	Technology	14nm	14nm	14nm
	Frequency	1GHz	1GHz	1.2GHz
	SRAM	64MB	256MB	512MB
	# of vector lane	128	2048	2048
Performance	Logistic regression	1024ms	119.52ms	39.9~43.5ms

(표 1) 동형암호 가속기 성능간 비교

3-1 F1[9]

F1 은 최초의 동형암호 scheme 을 가속할 수 있는 ASIC 가속기이다. 동형암호 연산이 메모리 부하를 크게 일으킴을 확인하고, DDR4 와 같은 일반적인 DRAM 이 아닌 HBM[12]을 사용하였다. 또한 데이터에서의 메모리 이동을 최소화하기 위해 큰 SRAM scratch pad 를 사용하고, 그 외, NTT 의 가속 및 기타 polynomial 연산을 지원하는 구성요소들을 설계하여 암호문간 곱셈 연산 시 CPU 대비 48,640 배 및 *bootstrapping* 시 1,195 배 빠르게 가속하는데 성공하였다.

3-2 Craterlake[10]

Craterlake 의 경우, 앞서 본 동형암호 parameter 중 L 크기를 늘림을 통해서 deep neural network 과 같이 연산이 많이 필요한 작업에 대해서 효율을 높이는 큰 parameter 를 사용하고자 하였다. 하지만 L 이 늘어나

게 될수록, 연산해야 할 다항식이 더 많아지고, 암호문의 크기가 커지게 되므로, 이에 대한 sweet spot 를 찾아 해당 구간에서 이전의 가속기들이 효율적이지 못함을 보였다. 또한 큰 parameter 를 사용할 때 효율을 높이기 위해 새로운 keyswitching 알고리즘의 도입하였으며, NTT 를 진행할 경우의 복잡한 메모리 이동 연산에서 데이터 이동 경로를 고정하는 방법을 제시하여 최적화하였다. 그 결과 CPU 대비 4,611 배 빠른 연산이 기능해졌다. 하지만 최적화 과정에서 bit width 가 28bit 로 크게 줄어드는 한계를 보였다.

3-3 BTS[11]

F1[9]에서 *bootstrapping* 시의 throughput 이 크게 떨어지는 것을 개선하고, multi-slot *bootstrapping* 을 지원하도록 하여 실용적인 수준에서 *bootstrapping* 이 가능한 가속기를 목표하였다. 또한 $N=2^{17}$ 이상의 큰 parameter 를 사용하기에, 이러한 경우 모든 데이터들을 on-chip memory 에 올리기는 어렵기에 HBM 의 bandwidth 를 고려하여 사용할 NTT 모듈의 최소 개수를 계산하였으며, 용량이 큰 evaluation key 의 경우, load 되는 시간 동안 앞의 연산이 끝나 있도록 하는 등 균형 맞는 설계를 추구하였다. 또한 NTT 시의 데이터 이동을 위한 연산기 간 NoC(Network on Chip)를 함께 구현하였다. 그 결과, CPU 에 대비하여 ResNet-20 을 동형암호 연산으로 진행할 경우, 5,000 배 이상 가속이 가능하였으며, 이는 bootstrapping 이 잦을수록 더 큰 가속효과를 누릴 수 있었다.

3-4 CryptoPIM[13], MemFHE[14]

CryptoPIM[13]의 경우, 기존의 전통적인 가속기의 형태에서 벗어나 Processing-in-Memory(PIM) 기술을 사용하여 전반적인 격자 기반 암호에 대한 연산을 가속하고자 하였다. 완전한 동형암호 알고리즘을 가속하진 않았지만, 동형암호 연산의 대부분을 차지하는 NTT 연산을 가속할 수 있었다. Memristor 로 구성된 ReRAM 을 사용하며 memristor 의 성질을 활용하여 NTT 를 진행하는데 필요한 modulo 연산 및 곱셈 덧셈 등을 수행한다. 이후 MemFHE[14]에서 완전 동형암호를 직접 가속할 수 있었다.

PIM 기술을 사용한 가속기들의 경우, 칩의 물리적인 I/O 포트 제한을 받지 않아 병렬성을 온전히 살릴 수 있으며, off-chip 통신으로 인한 에너지를 크게 아낄 수 있다는데 장점이 있어 CryptoPIM 의 경우 FPGA 구현체에 비해 31 배 효율적인 에너지 소비량을 기록할 수 있었다.

4. 결론

본 논문에서는 동형암호 연산에서 어떤 연산이 필요한지와 이에 따른 메모리 부하와 연산 부하를 앞선 연구들에서 어떻게 해결하고자 하였는지에 대해 살펴보았다. F1[9]를 시작으로 HBM(High Bandwidth Memory)[12]를 활용하여 큰 크기의 암호문과 evaluation key 를 빠르게 불려오려는 구조와, 최대한 on-chip 에서 reuse 를 최대화 하기 위해 암호문 들을 저장해 두는 큰 SRAM 용량이 공통적으로 드러난다. 큰 parameter 대신 작은 parameter 를 사용하여 메모리 대역폭을 아끼거나 evaluation key 의 크기에 관한 부분을 추가적으로 차후에 연구할 계획이며, 성능 향상을 꾀할 수 있을 것으로 기대된다.

사사

이 논문은 2024 년도 BK21 FOUR 정보기술 미래인재 교육연구단에 의하여 지원되었으며, 2024 년도 정부(과학기술정보통신부)의 재원으로 한국연구재단의 지원을 받아 수행된 연구임 (RS-2023-00277326).

참고문헌

[1] J. -W. Lee et al., "Privacy-Preserving Machine Learning With Fully Homomorphic Encryption for Deep Neural Network," in IEEE Access, vol. 10, pp. 30039-30054, 2022

[2] 2024 년 데이터 유출 비용 보고서", IBM, 2024 년 9 월 18 일 접속, https://www.ibm.com/kr-ko/reports/data-breach

[3] Eric Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.3" in Internet Engineering Task Force (IETF) RFC, RFC8446, https://www.rfc-editor.org/info/rfc8446, 2018

[4] Ilaria Chillotti, Nicolas Gama, Mariya Georgieva, and Malika Iz-abach'ene. Tfhe: Fast fully homomorphic encryption over the torus. (2018). https://ia.cr/2018/421

[5] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. Cryptology ePrint Archive, Report 2016/421, (2016). https://ia.cr/2016/421.

[6] Kim, A., Song, Y., Kim, M. *et al.* Logistic regression model training based on the approximate homomorphic encryption. *BMC Med Genomics* **11** (Suppl 4), 83 (2018). https://doi.org/10.1186/s12920-018-0401-7

[7] Yongwoo Lee, Joonwoo Lee, Young-Sik Kim, HyungChul Kang, and Jong-Seon No. 2020. High-Precision and Low-Complexity Approximate Homomorphic Encryption by Error Variance Minimization. IACR Cryptology ePrint Archive 1549 (2020)

[8] Over 100x Faster Bootstrapping in Fully Homomorphic Encryption through Memory-centric Optimization with GPUs. (2021). *IACR Transactions on Cryptographic Hardware and Embedded Systems*, *2021*(4), 114-148.

[9] Nikola Samardzic, Axel Feldmann, Aleksandar Krastev, Srinivas Devadas, Ronald Dreslinski, Christopher Peikert, and Daniel Sanchez. 2021. F1: A Fast and Programmable Accelerator for Fully Homomorphic Encryption. In MICRO-54: 54th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO '21).

Association for Computing Machinery, New York, NY, USA, 238–252. https://doi.org/10.1145/3466752.3480070

[10] Nikola Samardzic, Axel Feldmann, Aleksandar Krastev, Nathan Manohar, Nicholas Genise, Srinivas Devadas, Karim Eldefrawy, Chris Peikert, and Daniel Sanchez. 2022. CraterLake: a hardware accelerator for efficient unbounded computation on encrypted data. In Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA '22). Association for Computing Machinery, New York, NY, USA, 173–187. https://doi.org/10.1145/3470496.3527393

[11] Sangpyo Kim, Jongmin Kim, Michael Jaemin Kim, Wonkyung Jung, John Kim, Minsoo Rhu, and Jung Ho Ahn. 2022. BTS: an accelerator for bootstrappable fully homomorphic encryption. In Proceedings of the 49th Annual International Symposium on Computer Architecture (ISCA '22). Association for Computing Machinery, New York, NY, USA, 711–725. https://doi.org/10.1145/3470496.3527415

[12] JEDEC, High Bandwidth Memory (HBM) DRAM, JEDEC Standard, JESD235D, (2020)

[13] H. Nejatollahi, S. Gupta, M. Imani, T. S. Rosing, R. Cammarota and N. Dutt, "CryptoPIM: In-memory Acceleration for Lattice-based Cryptographic Hardware," *2020 57th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, CA, USA, 2020, pp. 1-6, doi: 10.1109/DAC18072.2020.9218730.

[14] Saransh Gupta, Rosario Cammarota, and Tajana Šimunić. 2024. MemFHE: End-to-end Computing with Fully Homomorphic Encryption in Memory. ACM Trans. Embed. Comput. Syst. 23, 2, Article 28 (March 2024), 23 pages. https://doi.org/10.1145/3569955