Université de Neuchâtel
Institut d'informatique

In-memory acceleration for HE with UPMEM PIM

Mpoki Mwaisela, Peterson Yuhala, James Menetrey, Pascal Felber, Valerio Schiavoni

ABUMPIMP

August 26, 2024

Swiss National
Science Foundation

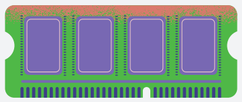
Agenda

01

HOMOMORPHIC ENCRYPTION

02

UPMEM PIM



03

HE KERNELS

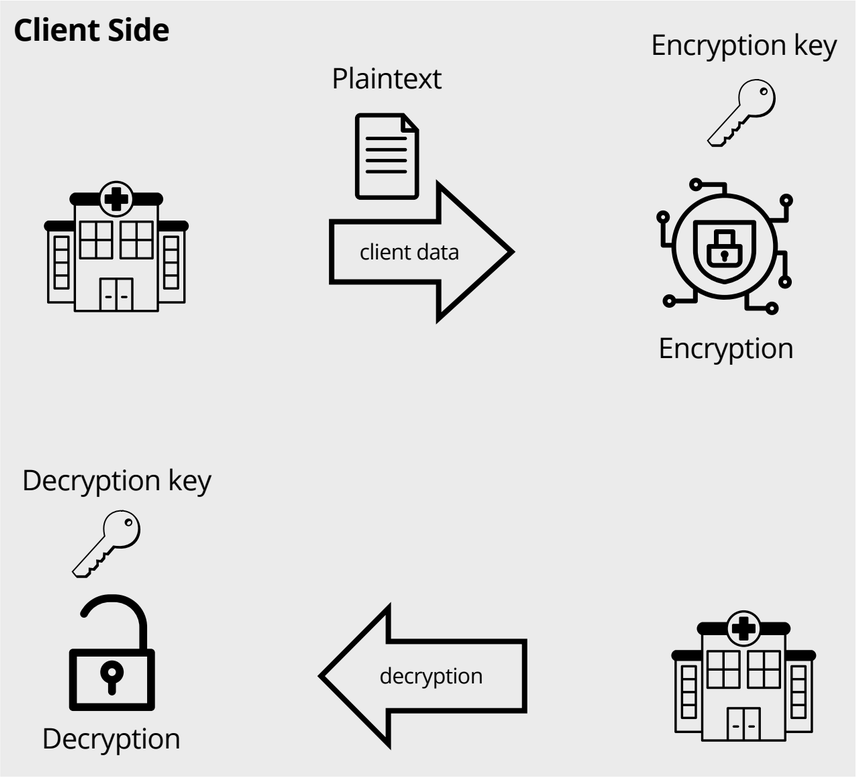
04

EVALUATION

unine
Université de Neuchâtel
Institut d'informatique

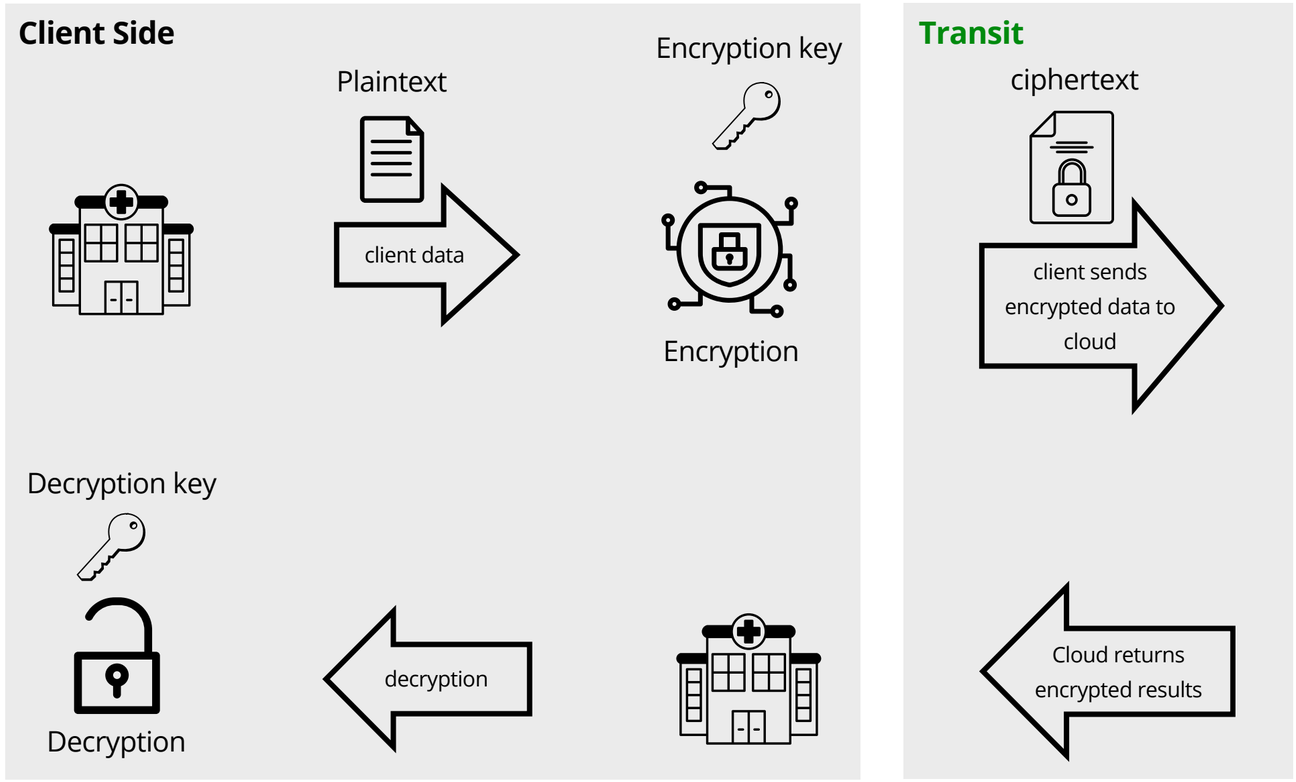
Homomorphic Encryption (HE)

- Enables computations on encrypted data without prior decryption
- Data is encrypted during **transit** and **processing**



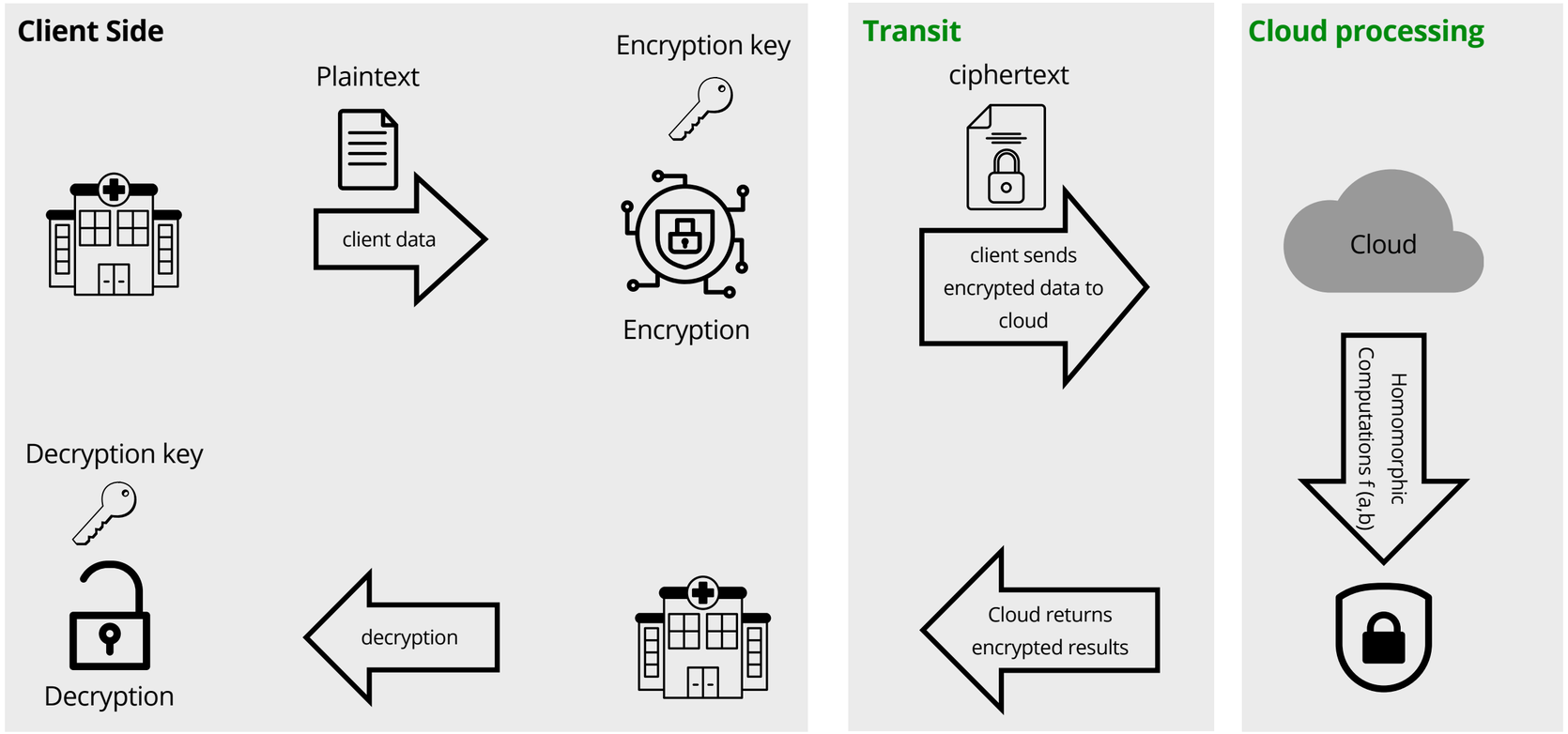
Homomorphic Encryption (HE)

- Enables computations on encrypted data without prior decryption
- Data is encrypted during **transit** and **processing**



Homomorphic Encryption (HE)

- Enables computations on encrypted data without prior decryption
- Data is encrypted during **transit** and **processing**



Potential Usecases

- Privacy preserving machine learning
- Secure data analysis and processing
- Secure systems and transactions

Potential Usecases

- Privacy preserving machine learning
- Secure data analysis and processing
- Secure systems and transactions

HE has a significantly large overhead for these application

Potential Usecases

- Privacy preserving machine learning
- Secure data analysis and processing
- Secure systems and transactions

HE has a significantly large overhead for these application

can UPMEM PIM help minimize the HE bottlenecks ?

Overhead of HE

- HE supports computation on encrypted data (Ciphertext)

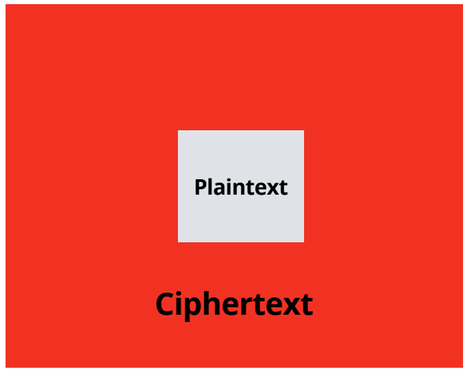
- Ciphertexts can be significantly larger than plaintexts
  - **At least a 50x** increase in data size *[F1 Accelerator, Samardzic et al., MICRO '21]*
  - Leads to **challenging data movements** to perform complex operations

Plaintext

Overhead of HE

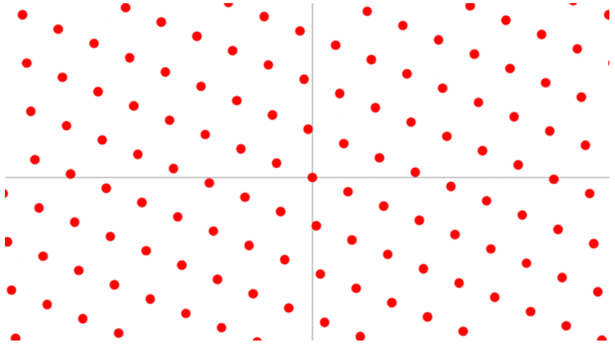
- HE supports computation on encrypted data (Ciphertext)

- Ciphertexts can be significantly larger than plaintexts
  - **At least a 50x** increase in data size *[F1 Accelerator, Samardzic et al., MICRO '21]*
  - Leads to **challenging data movements** to perform complex operations



HE Background

- **Foundations in lattice cryptography**
  - Resistant to quantum computing attacks!
  - Utilizes the complexity of (ring) learning with errors



- **Cryptographic Constructs**
  - **schemes:** BGV, BFV, CKSS, etc
  - **primitives:** keygen, encrypt, decrypt, evaluation, i.e EvalAdd, EvalMult, etc

- **Data representation in HE**
  - Polynomials are expressed in a quotient ring format: $R_q = Z_q/(x^N + 1)$
  - Typical polynomial forms: $a_0 + a_1x + a_2x^2 + \ldots\ldots + a_{N-1}x^{N-1}$

HE Polynomial Arithmetic

- **Operations:** addition and multiplication are carried out on each coefficient-wise
- **Addition:** performed using modular adder unit: $(a+b) \mod q = \begin{cases} a+b & \text{if } a+b < q \\ a+b-q & \text{if } a+b \geq q \end{cases}$
- **Multiplication:** uses barrett's and montgomery reduction algorithm for efficient computation.
- **HE bottleneck:**
  - Operations on high bit-width coefficients (up to 1000 bits) solved by RNS
  - Significant computational overhead for high-degree polynomials, common in HE schemes

Accelerating FHE Schemes

- Focuses on algorithm optimization
  - Transforms (NTT), Bootstrapping, Encoding
- Utilizes hardware for acceleration
  - GPUs, FPGAs, AVX, ASIC

F1: A Fast and Programmable Accelerator for Fully Homomorphic Encryption (Extended Version)

Axel Feldmann[1*], Nikola Samardzic[1*], Aleksandar Krastev[1], Srini Devadas[1], Ron Dreslinski[2], Karim Eldefrawy[3], Nicholas Genise[3], Chris Peikert[2], Daniel Sanchez[1]

cuHE: A Homomorphic Encryption Accelerator Library

Wei Dai and Berk Sunar

Worcester Polytechnic Institute,
Worcester, Massachusetts, USA

Intel HEXL: Accelerating Homomorphic Encryption with Intel AVX512-IFMA52

Fabian Boemer
fabian.boemer@intel.com
Intel Corporation
Santa Clara, CA, USA

Sejun Kim
sejun.kim@intel.com
Intel Corporation
Hillsboro, Oregon, USA

Gelila Seifu
gelila.seifu@intel.com
Intel Corporation
Santa Clara, CA, USA

Fillipe D. M. de Souza
fillipe.souza@intel.com
Intel Corporation

Vinodh Gopal
vinodh.gopal@intel.com
Intel Corporation

Fast Arithmetic Hardware Library For RLWE-Based Homomorphic Encryption

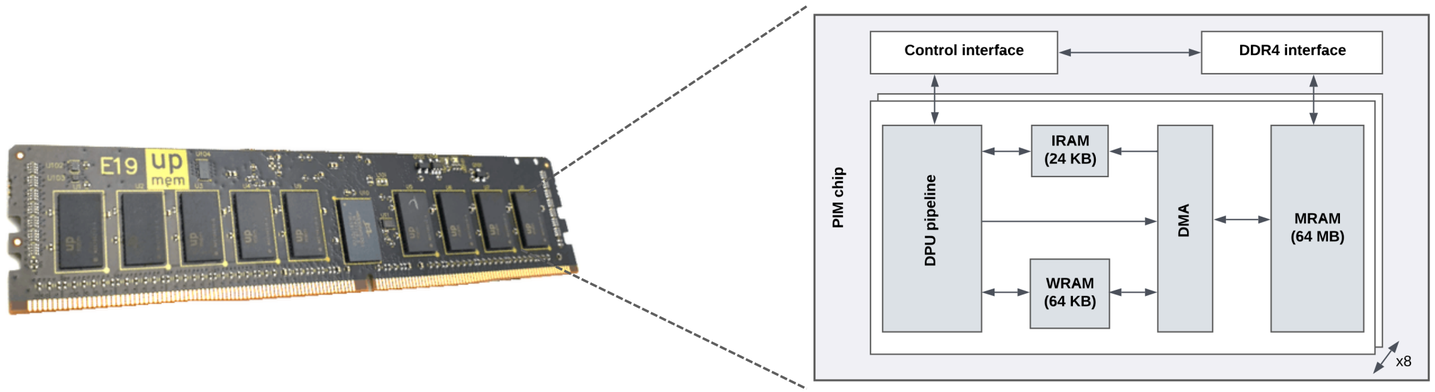
Rashmi Agrawal[1], Lake Bu[2], Alan Ehret[1] and Michel A. Kinsy[1]

[1] Adaptive and Secure Computing Systems (ASCS) Laboratory, Boston University
rashmi23,ehretaj,mkinsy@bu.edu
[2] The Charles Stark Draper Laboratory Inc., USA, lbu@draper.com

Abstract. With billions of devices connected over the internet, the rise of sensor-based electronic devices has led to the use of cloud computing as a commodity technology service. These sensor-based devices are often small and limited by power, storage, or compute capabilities; hence, they achieve these capabilities via cloud ser-

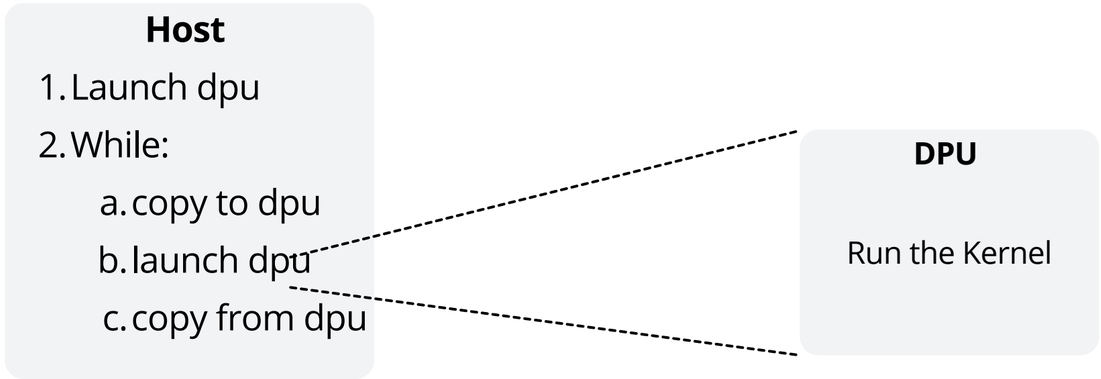
Processing In-Memory (PIM)

- **Concept:** bringing computation closer to data to reduce data movement bottleneck
- **UPMEM PIM** : features thousands of innovative Data Processing Units (DPUs) embedded within DRAM memory chips
- **Organization:** each DRAM memory chip comprises of 8 DPUs with each DPU having explicit access to 64MB of MRAM
- **DPUS:** General purpose processor, up to 24 tasklets with 11 pipeline stages



Programming DPU

- **Control mechanism:** DPUs are managed by high-level applications running on the main CPU (Host)
- **Task orchestration:** host ensures task coordination and execution across the DPUs.
- **Programmability:**
  - SDK Available: host applications can be developed in C/C++, Java, and Python
  - DPU programming: currently only supports C



UPMEM PIM as an Accelerator for HE

- **UPMEM PIM features:** provides extensive parallelism up to 2560 DPUs with up to 256 GB of PIM RAM

- **Main idea:** use polynomial arithmetic algorithms as kernels for the DPUs

Evaluation

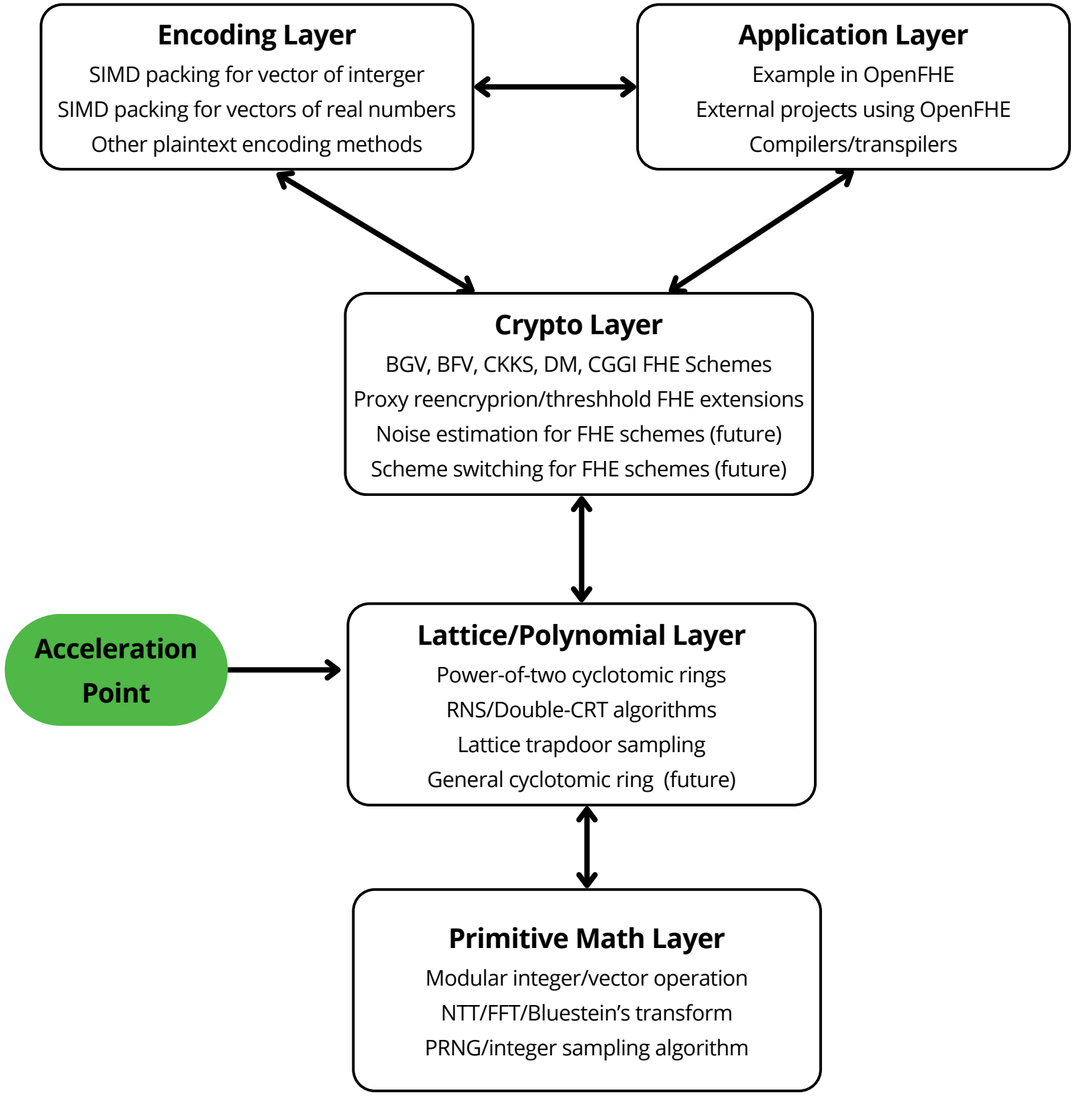
- **Host:** two-socket server 10-core Intel Xeon Silver 4210R (2.40GHz) per socket, hyper-threading , Cache Hierarchy - 32KB L1d, 32KB L1i, 1MB L2, 13.75MB L3

- **PIM:** DPU 32 ranks (2048 DPUs, 350MHz each), 128GB MRAM, 1GB/s per DPU, total 2.048TB/s bandwidth

- **Evaluation methodology**
  - **Metrics**: CPU execution time (cpu-time), DPU execution time (dpu-time), host-to-DPU copy time (host-dpu), and DPU-to-host copy time (dpu-host).
  - **Setup**: CPU experiments use 16 threads, and DPU experiments use 16 tasklets.

Case Study: OpenFHE

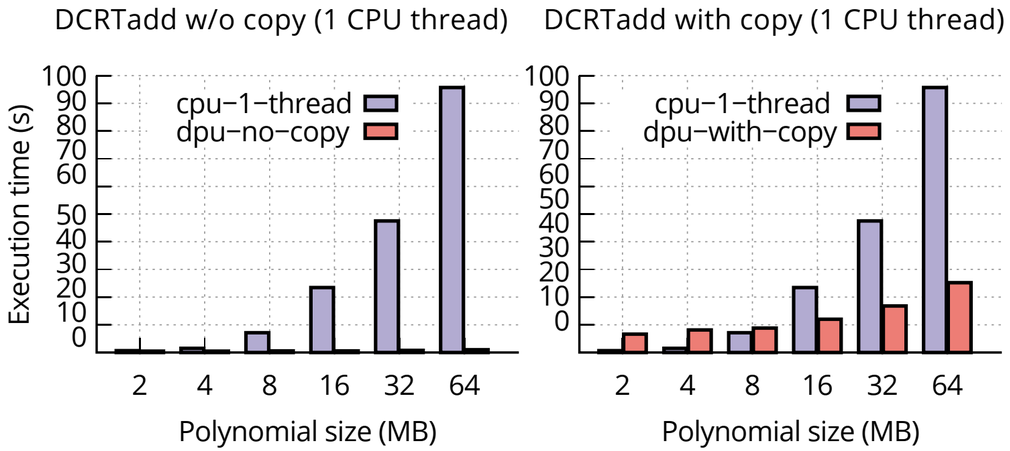
- **What :** library for FHE implementations, incorporating design features from PALISADE, HElib, HEAAN, and FHEW
- **Cryptographic capabilities:** supports efficient implementations of all common FHE schemes i.e BGV, BFV, CKKS, DM/FHEW and CGGI/TFHE
- **Ciphertext representation :** pair of polynomials in DCRT format which is combination of NTT and RNS representations
- **Key feature:** includes a hardware abstraction layer (HAL) supports for multiple hardware acceleration back-ends such as AVX, GPU, FPGA etc

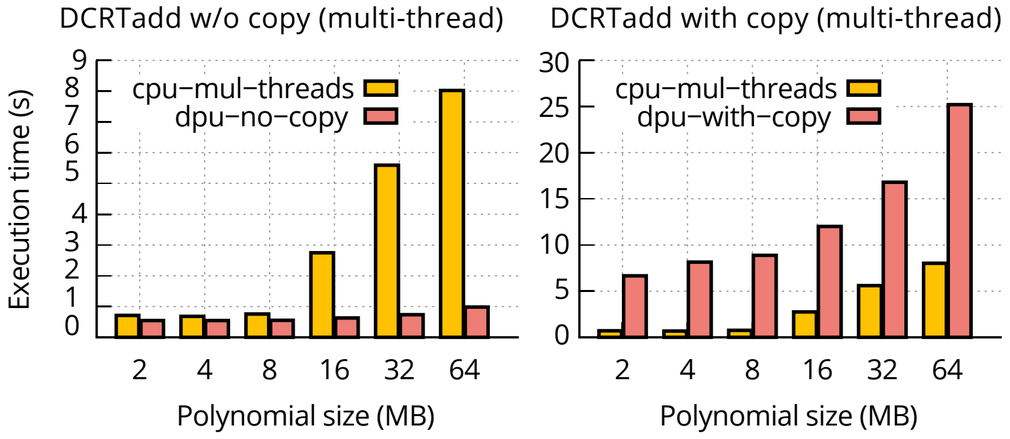


OpenFHE design paper

Case Study: OpenFHE

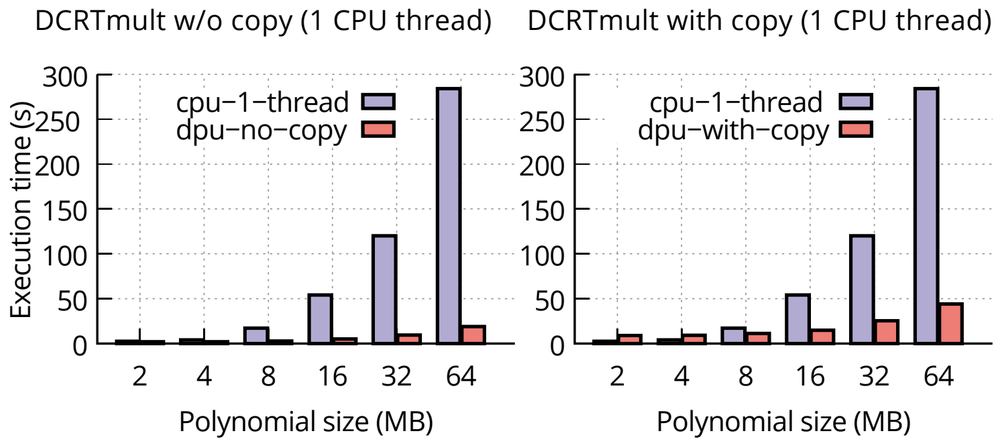
DCRT Polynomial Addition

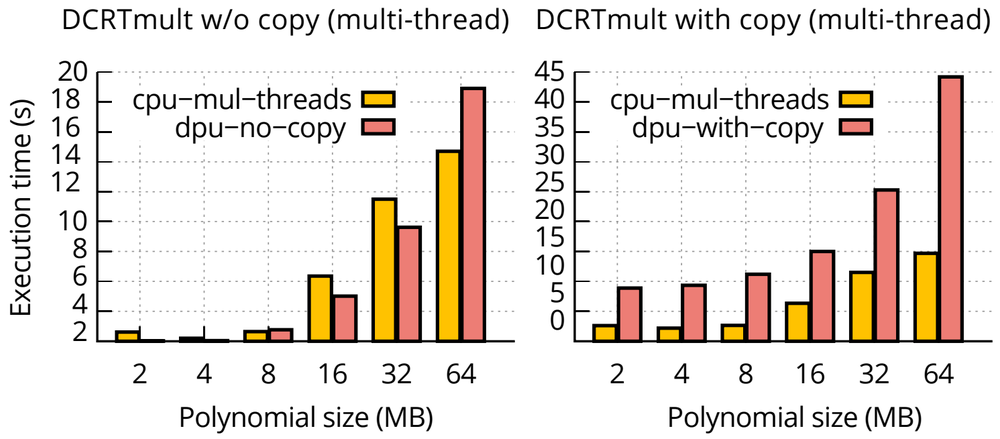




Case Study: OpenFHE

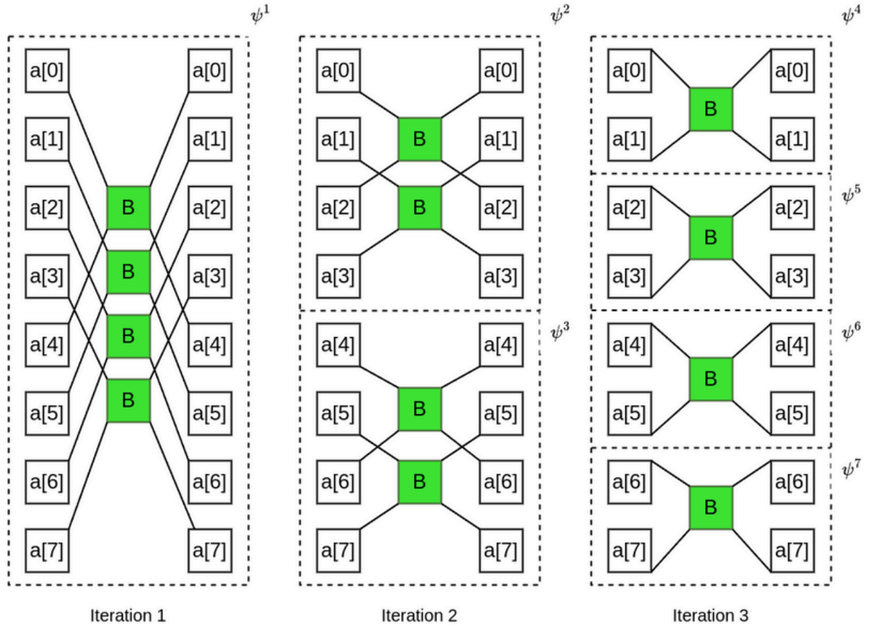
DCRT Polynomial Multiplication

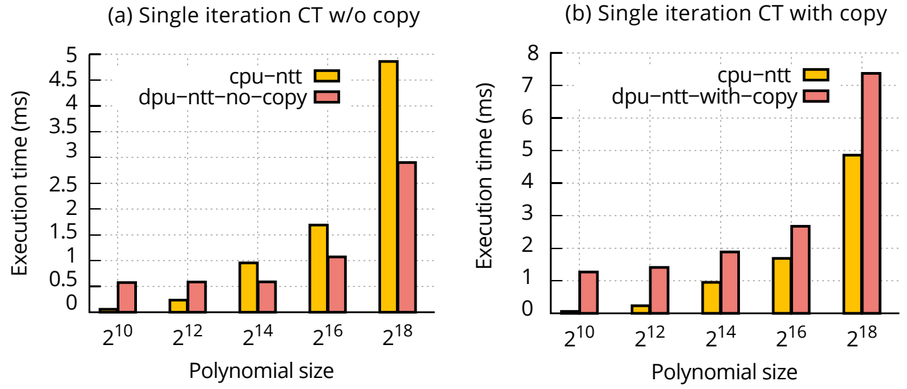




Number Theoretic Transform

- Transforms a polynomila from NTT for efficient Multiplications
- **Core operations :** cooley -tukey butterfly for NTT and gentleman -sande harvey butterfly for iNTT
- **Butterfly operation “B”:** generally involves combination of modular addition and multiplication algo. **Offload butterflies to DPUs**
- **Data dependency:** between iteration need to update tables in the DPUs





CONCLUSION

- HE allows computations to be performed directly on encrypted data, but slow
- HE applications can benefit from the UPMEM PIM massive parallization
- Need to efficiently manage the data copies to/from DPUs and host


A
Q