ZAMA

MARSEILLE • JANUARY 27, 2026

THE REVOLUTION OF HOMOMORPHIC ENCRYPTION

PROCESSING DATA WITHOUT REVEALING IT

MARC JOYE

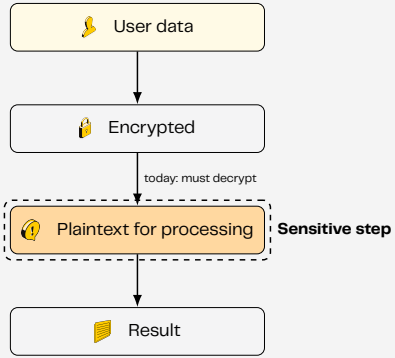
Why We Need a Revolution

We want the cloud… but not the risk

- Data is valuable ⇝ attackers, leaks, misuse
- Regulations & trust demand privacy
- Yet: we still want powerful services (e.g., analytics, AI, …)

Key idea

Traditional encryption protects data at rest and in transit, *but not while computing*



FHE in One Sentence

"Compute on encrypted data as if it were plaintext"

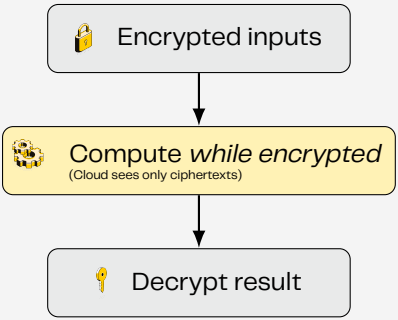
The magic property

If you encrypt x and y, you can compute $f(x, y)$ **directly on ciphertexts**

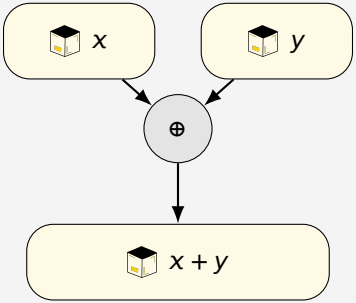
$$\mathrm{Dec}\big(f(\mathrm{Enc}(x), \mathrm{Enc}(y))\big) = f(x, y)$$

Key idea

The server never sees x, y, or intermediate values; only the user gets the result in the clear



Intuition: The Lockbox Analogy



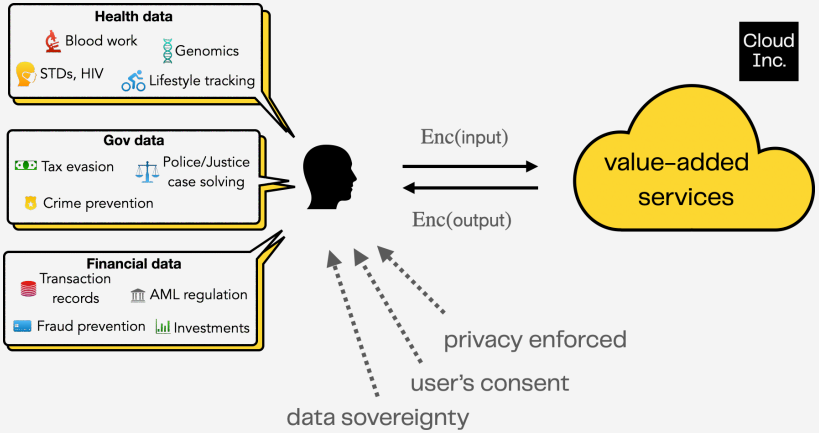
Still sealed
(only owner can open)

- Imagine numbers inside sealed lockboxes
- You can *combine* lockboxes with special tools
- You get a new lockbox containing the answer
- Only the owner has the key to open it

Key idea

FHE is that special toolset implemented with mathematics

How FHE Will Change the World



A Concrete Example

Private Health Flag (Encrypted Lab Results)

Scenario

A hospital computes an early-warning *score* from sensitive lab values (e.g., inflammation markers), then derives a simple *risk flag* for alerting

Example (simple rule-based score):

$$\text{score} = \mathbb{1}[\text{CRP} > t_1] + \mathbb{1}[\text{WBC} > t_2] + \mathbb{1}[\text{Temp} > t_3]$$

With a model hosted in the cloud

1. Patient/lab encrypts {CRP, WBC, Temp}
2. Cloud computes the encrypted score (thresholds stay server-side)
3. Cloud derives encrypted **risk flag** (e.g. score $\geq t$)
4. Hospital decrypts only the risk flag

Key idea

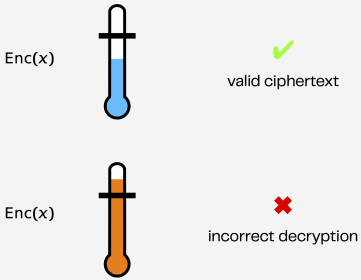
The cloud learns nothing about the patient's lab values (or intermediate results)

Taming the Complexity Behind Encrypted Computation

- FHE ciphertexts carry **noise**
- Operations increase noise
- If noise gets too big ⇝ decryption fails
- Breakthrough: **bootstrapping**
  - refresh ciphertext
  - can be made programmable

Key idea

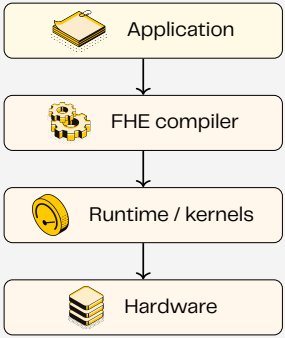
Modern schemes engineer noise growth and refresh to unlock deep computations



From Theory to Engineering

What improved in the last decade?

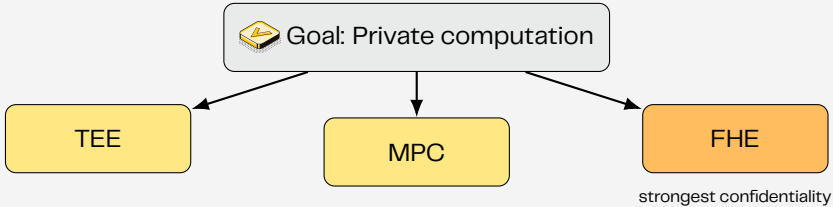
- Better algorithms for bootstrapping
- Hardware acceleration (CPU/GPU/FPGA)
- Compiler toolchains to map programs to FHE operations
- Practical parameter selection & security standards



Key idea

The “revolution” is not one trick; it’s a full stack: math + compilers + systems

FHE vs Other Privacy Technologies



- **TEEs** (trusted enclaves): fast, but rely on hardware trust
- **MPC**: joint computation, often interactive/communication-heavy
- **FHE**: cryptographic confidentiality during computation
- **Differential Privacy**: protects aggregates, not raw computation secrecy

In practice, hybrids are common: FHE + MPC, or FHE + TEEs, depending on the threat model

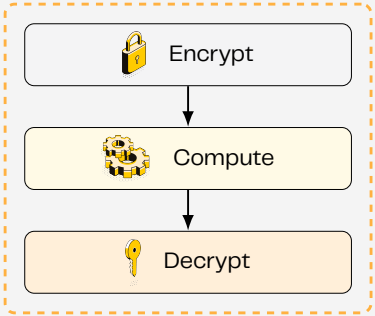
Takeaways: Why This Is a Revolution

What to remember

1. FHE lets you **compute on encrypted data**
2. It enables new products where **privacy is the default**
3. The “revolution” is the full stack becoming usable

Key idea

Future: confidential apps, safer data sharing, and stronger user control



Privacy-by-design for computation

Questions?

Contact and Links

marc@zama.ai

zama.ai

github.com/zama-ai

zama.ai/community