# Transciphering Framework for Approximate Homomorphic Encryption

Jihoon Cho[1], Jincheol Ha[2], **Seongkwang Kim[2]**, Byeonghak Lee[2],

Joohee Lee[1], Jooyoung Lee[2], Dukjae Moon[1], Hyojin Yoon[1]

[1]Samsung SDS, Seoul, Korea

[2]KAIST, Daejeon, Korea

Asiacrypt 2021

SAMSUNG SDS KAIST GSIS Graduate School Of Information Security CryptLab

# Overview

- We present the first **transciphering framework for real numbers**
  - Transciphering framework is a conversion of ciphertexts from a symmetric cipher to an HE-ciphertext
- We propose HE-friendly cipher HERA targeting **practically shorter homomorphic evaluation time**
  - It has been widely believed that mult. depth and complexity are most relevant
  - It turned out that the linear layer significantly affect the evaluation time
  - We devised randomized key schedule rather than randomized linear layer
- We achieve **23x smaller ciphertext expansion** and **9085x smaller latency** than the CKKS-only environment.

# Homomorphic Encryption

- Homomorphic encryption (HE) is an encryption scheme that enables addition and multiplication over encrypted data without decryption key*
  - $\mathrm{a} * \mathrm{b} = \mathrm{Dec}(\mathrm{Enc}(a) * \mathrm{Enc}(b))$
  - E.g., FV ($\mathbb{Z}_t$, +, ×), CKKS($\mathbb{C}$, +, ×)
- HE can protect data even when they are being used
  - E.g., ML inference, statistics of sensitive data on a cloud server

* We do not take into account partially homomorphic encryption (PHE) in this talk.

# RLWE-based HE Schemes

**Message** : data vector which we want to manipulate



*Encode*: function from vector to poly (without key)

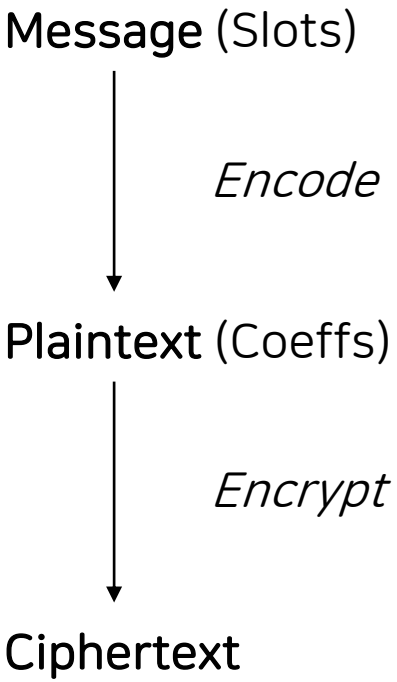
**Plaintext** : polynomial to be encrypted



*Encrypt*: function from poly to poly^2 (with key)

**Ciphertext** : a (pseudorandom) pair of polynomials

# RLWE-based HE Schemes



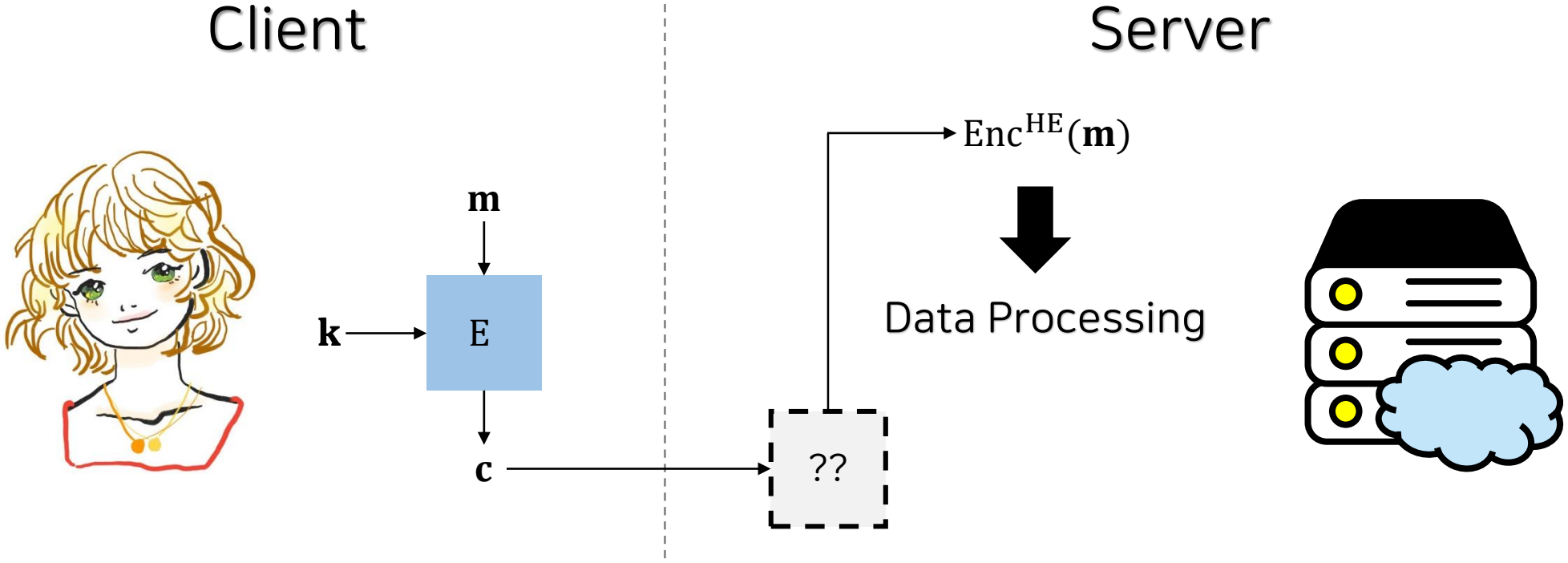

| | FV (BGV) | CKKS |
|---|---|---|
| Message Space | $\mathbb{Z}_t^N$ Exact computation | $\mathbb{C}^{N/2}$ Approximate computation |
| Encoding Function | NTT-like function | DFT-like function |
| Plaintext Space | $\mathcal{R}_t = \mathbb{Z}_t[X]/(\Phi_{2N}(X))$ | $\mathcal{R} = \mathbb{Z}[X]/(\Phi_{2N}(X))$ |
| Encryption Function | RLWE | RLWE |
| Ciphertext Space | $\mathcal{R}_q \times \mathcal{R}_q$ $(q \gg t)$ | $\mathcal{R}_q \times \mathcal{R}_q$ |

# Demerits of HE

- Slow encryption speed
  - Slower than usual public key encryption
  - Inadequate to bulk encryption

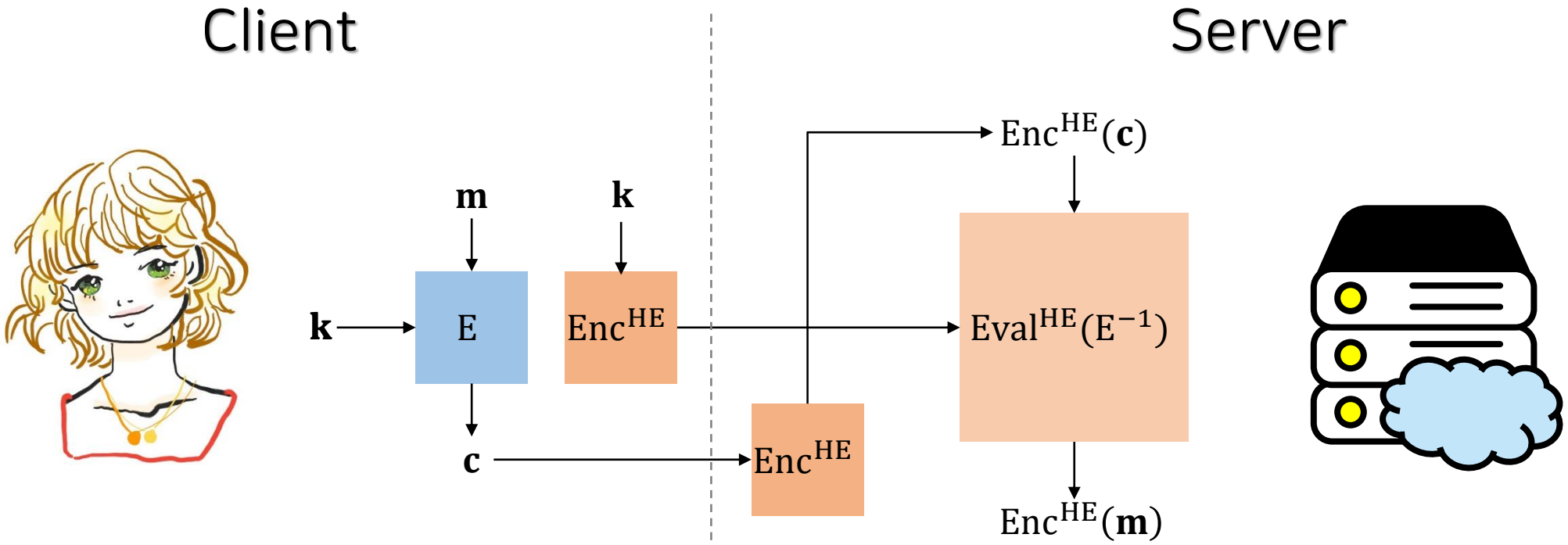
- Large ciphertext expansion
  - 10x – 1,000,000x according to the choice of parameters
  - Disadvantage for encryption of small messages
  - Large memory & network bandwidth overhead

# Transciphering Framework




| Scheme | Enc. Speed | Ctxt. Exp. Ratio | Parameters |
|---|---|---|---|
| AES-GCM | 4.59 GB/s | 1 | 128-bit secure |
| CKKS | 1.22 MB/s | 35.3 | $(N = 2^{14}, \log q = 320)$ |

# Transciphering Framework




Fast encryption speed
Smaller ciphertext expansion

* K. Lauter et al., “Can Homomorphic Encryption Be Practical?”, ACM CCSW 2011

# HE-friendly Ciphers

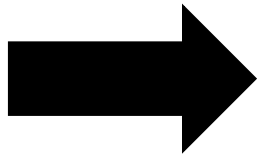
- HE-friendly cipher is a cipher which is efficiently evaluated using homomorphic encryption
- New design strategy is required
  - So far, AND gates and XOR gates need roughly same resources in most hardware
  - However, cost of addition (XOR in binary) is almost negligible compared to multiplication (AND in binary) in HE setting
  - Low multiplicative depth/complexity
- Since then, several HE-friendly ciphers were proposed
  - E. g., LowMC (EC ‘15), Kreyvium (FSE ‘16), FLIP (EC ‘16), Rasta (Crypto ‘18), and Masta (IEEE Access ‘20)

# Problems

- There is no transciphering framework for real numbers
  - In real-world applications, the computation of real numbers is needed
  - (Deterministic) cipher over real number is hard to design
- Previous works did not consider practicality enough
  - Cost model was too idealized to reflect the real cost of HE
  - Client-side encryption is too slow as a symmetric cipher (~ 100 M Cycle)

Solving equation

$$\begin{cases} \mathrm{E}_{\mathbf{k}}(\mathbf{m}_1) = \mathbf{c}_1 \\ \mathrm{E}_{\mathbf{k}}(\mathbf{m}_2) = \mathbf{c}_2 \\ \vdots \\ \mathrm{E}_{\mathbf{k}}(\mathbf{m}_\ell) = \mathbf{c}_\ell \end{cases}$$

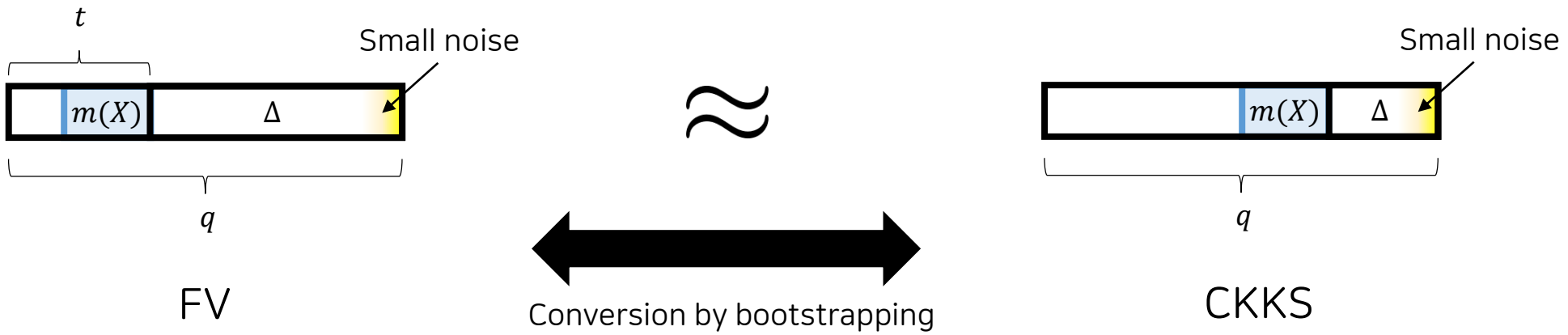


Minimizing function

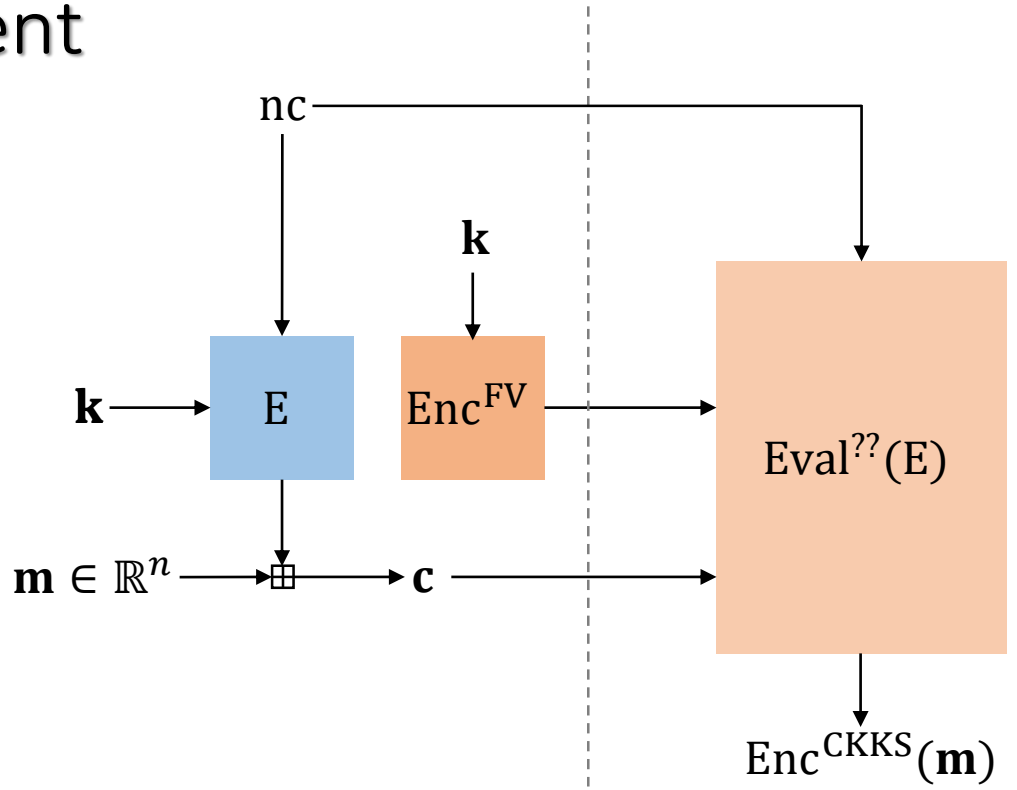
$$F_\ell(\mathbf{k}) = \sum_{i=1}^{\ell} \|\mathrm{E}_{\mathbf{k}}(\mathbf{m}_i) - \mathbf{c}_i\|^2$$

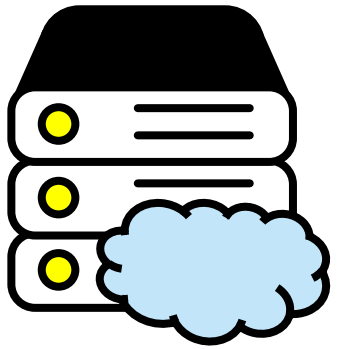
# Observation

- **Observation 1**: CKKS and FV have similar encryption algorithms
  - $\mathrm{Enc}^{\mathrm{CKKS}}(m) = v \cdot pk + (\lfloor \Delta \cdot m \rceil + e_0, e_1) \bmod q,\ \Delta = \mathrm{scale}$
  - $\mathrm{Enc}^{\mathrm{FV}}(m) = v \cdot pk + (\Delta \cdot m + e_0, e_1) \bmod q,\ \ \Delta = \lfloor q/t \rfloor$
- **Observation 2**: both schemes have similar (encoded) plaintext space
  - CKKS: $\mathbb{Z}[X]$ with bounded coefficients
  - FV: a fixed representative of $\mathbb{Z}_t[X]$



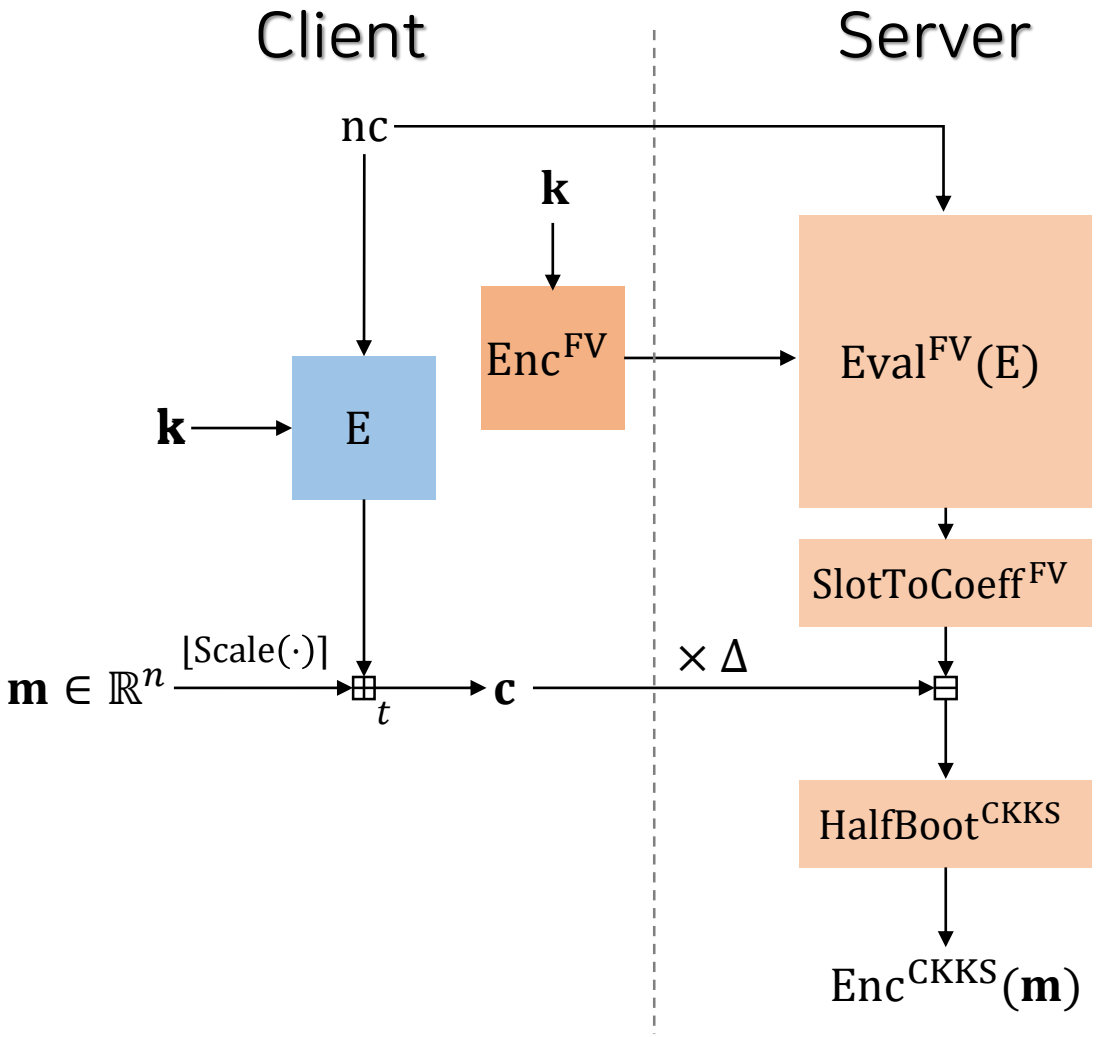
# We Want …



# RtF Transciphering Framework

- Real-to-Finite-field (RtF)
- Client side over $\mathbb{Z}_t$
  - Conversion of real message to integers modulo $t$ by scaling and rounding off
  - Nonce-based stream cipher over $\mathbb{Z}_t$
  - Sending FV-ciphertext of key $\mathbf{k} \in \mathbb{Z}_t^n$



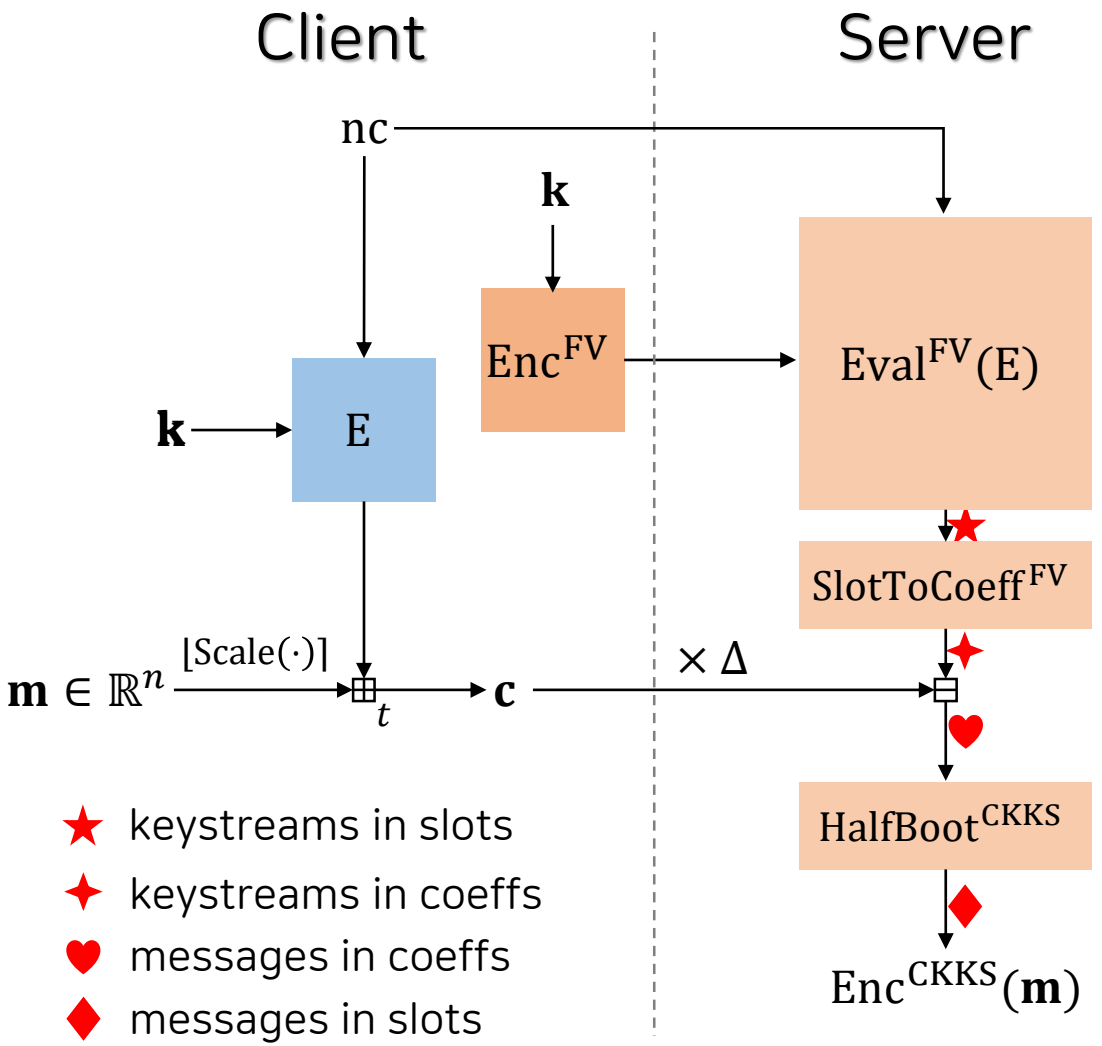
# RtF Transciphering Framework

- Offline server side
  - FV-evaluation of cipher $\mathrm{E}$
  - Move keystreams: Slots → Coefficients
- Online server side
  - Concat & scale up to FV-ciphertext space

    $\boldsymbol{c} \mapsto C \mapsto (\Delta \cdot C, 0), C = \mathrm{poly}$
  - Homomorphically subtract keystream
  - CKKS-Bootstrapping (HalfBoot)

    $\mathrm{ModRaise} \to \mathrm{SubSum} \to \mathrm{CoeffToSlot}^{\mathrm{CKKS}}$

    $\to \mathrm{EvalMod} \to \mathrm{SlotToCoeff}^{\mathrm{CKKS}}$



# Stream Cipher HERA

- HE-friendly ciphers with a RAndomized key schedule (HERA)
- Stream cipher outputting keystream in $\mathbb{Z}_t^{16}$
- Substitution-permutation network (SPN) with randomized key schedule
- Start and end with linear layers
  - $\mathrm{HERA}^r = \mathrm{Fin}_r \circ \mathrm{Round}_{r-1} \circ \cdots \circ \mathrm{Round}_1 \circ \mathrm{ARK}_0$
  - $\mathrm{Round}_i = \mathrm{ARK}_i \circ \mathrm{Cube} \circ \mathrm{Lin}$
  - $\mathrm{Fin}_r = \mathrm{ARK}_r \circ \mathrm{Lin} \circ \mathrm{Cube} \circ \mathrm{Lin}$
- Fixed constant input

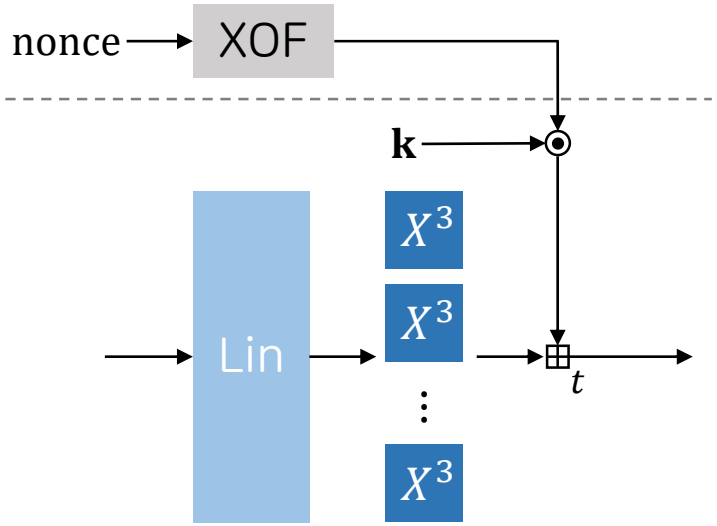
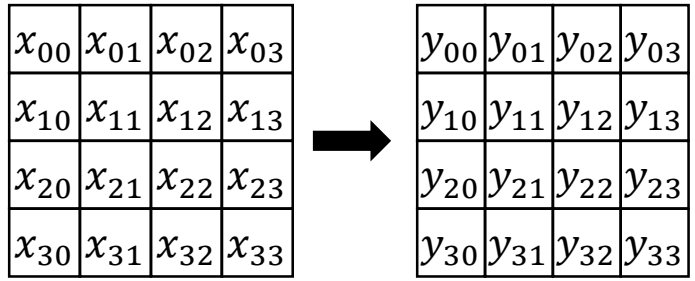



Fig. Round function of HERA

# Design Aspects of HERA

- Random affine layers → Fixed affine layers
  - It has been popular to use ‘simpler S-box and random affine layer’
  - Much faster encryption speed on the both sides
  - AES-like MDS matrix is used for high diffusion

| Function | Time (ms) | Consumed Budget (bits) |
|---|---|---|
| $\mathrm{MixRows} \circ \mathrm{MixColumns}$ | 23.55 | 4 |
| Fixed random matrix | 461.7 | 27 |
| Freshly-generated matrix | 4006 | 35 |
| Cube | 3479 | 86 |




$$\begin{bmatrix} y_{0c} \\ y_{1c} \\ y_{2c} \\ y_{3c} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_{0c} \\ x_{1c} \\ x_{2c} \\ x_{3c} \end{bmatrix}$$

**Fig.** MixColumns

$$\begin{bmatrix} y_{c0} \\ y_{c1} \\ y_{c2} \\ y_{c3} \end{bmatrix} = \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \cdot \begin{bmatrix} x_{c0} \\ x_{c1} \\ x_{c2} \\ x_{c3} \end{bmatrix}$$

**Fig.** MixRows

# Design Aspects of HERA

- Simple but randomized key schedule
  - Component-wise multiplication by random value to master key
  - $\mathrm{ARK}[\mathbf{k}, \mathrm{nc}, i](\mathbf{x}) = \mathbf{x} + \mathbf{k} \odot \mathbf{rc}_i$
  - Efficient in homomorphic evaluation

- Component-wise cube map
  - $(x_0, x_1, \ldots, x_{15}) \mapsto (x_0^3, x_1^3, \ldots, x_{15}^3)$
  - Invertible low-degree nonlinear map
  - High-degree inverse → prevent algebraic MitM

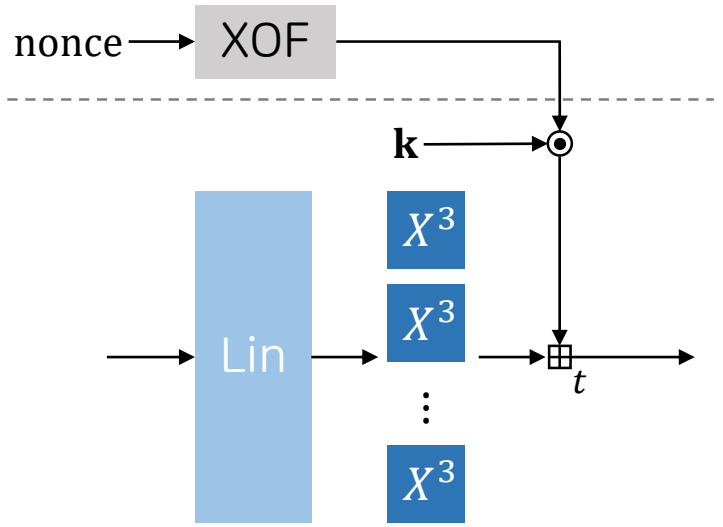



**Fig.** Round function of HERA

# Security Analysis of HERA

- Statistical attack
  - Differential-based cryptanalysis → Impossible (CPA model X)
  - Integral attack (including Out-of-oddity attack*) → Impossible (CPA model X)
  - We compute the expected linear/differential probability
    - Linear probability of Cube $\leq 4/t$
    - Differential probability of Cube $\leq 2/t$
    - Branch number of linear layer = 8
    - HERA is secure if $\left(\frac{t}{2}\right)^{8\cdot\left\lfloor\frac{r}{2}\right\rfloor} > 2^{\lambda}$

* T. Beyne et al., "Out of oddity – New Cryptanalytic Techniques against Symmetric Primitives Optimized for Integrity Proof System", Crypto 2020

# Security Analysis of HERA

- Algebraic attack
  - Trivial linearization & Interpolation attack
    - Count the number of monomials
    - We believe that monomials appear densely since $\mathrm{MixRows} \circ \mathrm{MixColumns}$ is dense
    - Hard to make meet-in-the-middle attack because of high-degree inverse
  - GCD attack & Gröbner basis attack
    - Compute the degree of regularity
    - Consider the hybrid approach (guess & determine)

# Security Analysis of HERA

| Attack \ Security | 80 | 128 | 192 | 256 |
|---|---|---|---|---|
| LC/DC | 2 | 4 | 4 | 6 |
| Trivial linearization | 4 | 5 | 6 | 7 |
| Interpolation attack | 4 | 5 | 6 | 7 |
| GCD attack | 1 | 1 | 1 | 7 |
| Gröbner basis attack | 4 | 5 | 6 | 7 |

**Tab.** Recommended number of rounds with respect to each attack

# Performance Comparison

| Scheme | $\log N$ | Log of #slots | Ct size (KB) | Ct. Exp. Ratio | Client | | Server | | Prec. (bits) | Level |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | Lat. ($\mu s$) | Thrp. (MB/s) | Lat. (s) | Thrp. (KB/s) | | |
| RtF-HERA | 16 | 16 | 0.055 | 1.54 | 1.599 | 21.73 | 147.68 | 4.738 | 17.2 | 11 |
| RtF-HERA | 16 | 9 | 0.055 | 1.53 | 1.599 | 21.78 | 117.71 | 0.051 | 17.2 | 11 |
| LWE * | 16 | 9 | 0.007 | 4.84 | 21.60 | 0.051 | 89.61 | 0.006 | 9.2 | 6 |
| CKKS only | 14 | 14 | 640 | 35.31 | 14527 | 1.218 | none | | 17.1 | 11 |

* W. Lu et al., “Pegasus: Bridging Polynomial and Non-polynomial Evaluations in Homomorphic Encryption”, IEEE S&P 2021

Thank you