FHECAP: An Encrypted Control System with Piecewise Continuous Actuation

Song Bian, *Member, IEEE,* Yunhao Fu, Dong Zhao, *Senior Member, IEEE,* Haowen Pan, Yuexiang Jin, Jiayue Sun, *Member, IEEE,* Hui Qiao, and Zhenyu Guan, *Member, IEEE*

***Abstract*—We propose an encrypted controller framework for linear time-invariant systems with actuator non-linearity based on fully homomorphic encryption (FHE). While some existing works explore the use of partially homomorphic encryption (PHE) in implementing linear controller systems, the impacts of the non-linear behaviors of the actuators on the systems are often left unconcerned. In particular, when the inputs to the controller become too small or too large, actuators may burn out due to unstable system state oscillations. To solve this dilemma, we design and implement FHECAP, an FHE-based controller framework that can homomorphically apply non-linear functions to the actuators to rectify the system inputs. In FHECAP, we first design a novel data encoding scheme tailored for efficient gain matrix evaluation. Then, we propose a high-precision homomorphic algorithm to apply non-arithmetic piecewise function to realize the actuator normalization. In the experiments, compared with the existing state-of-the-art encrypted controllers, FHECAP achieves $4\times$–$1000\times$ reduction in computational latency. We evaluate the effectiveness of FHECAP in the real-world application of encrypted control for spacecraft rendezvous. The simulation results show that the FHECAP achieves real-time spacecraft rendezvous with negligible accuracy loss.**

***Index Terms*—Fully homomorphic encryption, encrypted control, noise analysis, piecewise nonlinearity.**

I. Introduction

In recent years, various security issues have emerged in cloud-based control systems, such as close encounters of spacecraft [1], [2] and data breaches in industrial control systems [3]–[5]. It can be said that improving the security of the control system is very important [6], [7]. Applying HE to cloud controllers can achieve secure outsourced control.

Homomorphic encryption (HE) is a cryptographic primitive that allows multiple participating parties to jointly complete a computational task over encrypted data without decryption. In Figure 1, we show a general protocol of applying HE on the cloud-based control system, where the main participating parties are the controlled plant and the controller. First, in (1), the sensor of the plant collects the system status and encrypts the data using its private key. The encrypted data are then sent to the cloud controller. Second, in (2), the encrypted cloud controller combines the encrypted system state with the set of control laws to obtain the encrypted system input. Next, the results from (2) are returned to the plant. Finally, the plant decrypts the received ciphertexts to obtain the control commands and execute the commands through the actuator. Within the protocol, we observe that the main performance bottleneck lies in the application of complex control laws over homomorphic ciphertexts. Therefore, it is critical to develop control-specific HE operators to enhance both the expressiveness and the efficiency of the overall protocol.

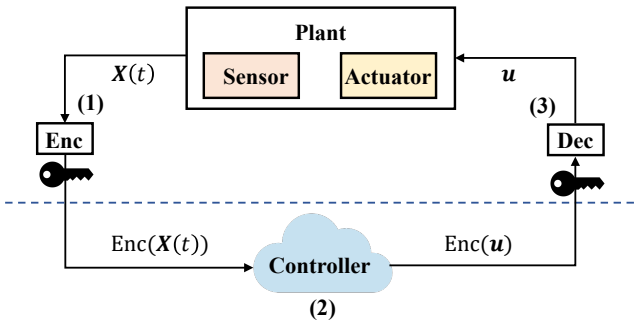


Figure 1. The overview of a round of cloud-based encrypted control protocol via FHE.

Existing works on encrypted control systems can be classified based on the underlying cryptosystems, namely, Paillier-based [8]–[17] and lattice-based [18]–[27]. Since the Paillier cryptosystem can only encrypt one integer at a time, Paillier-based encrypted control systems suffer from high communication costs. To mitigate the high level of communication overheads, lattice-based encrypted control solutions are proposed, where the computations are carried out over leveled homomorphic encryption (LHE) schemes such as BFV [28] and CKKS [29]. The single-instruction-multi-data (SIMD) capabilities [30] of LHE-based encrypted control systems can significantly reduce communication bandwidth by packing multiple messages into one ciphertext.

Despite the significant progress made, we observe that most current encrypted control systems can only support linear control systems. In real-world applications, a large number of control systems incorporate non-linearity in their control behaviors. In particular, an important class of control systems incorporates the non-linear characteristics of actuators into the system descriptions [31]–[38]. In such systems, non-linear

This work is partially supported by the National Key R & D Program of China (2023YFB3106200), the National Natural Science Foundation of China (T2425023, 62202028, 62172025, U2241213). This work is also supported by Huawei Technologies Co., Ltd.

Corresponding author is Zhenyu Guan.

S.Bian, Y. Fu, D. Zhao, H. Pan, Y. Jin and Z. Guan are with School of Cyber Science and Technology, Beihang University, Beijing 100191, China (e-mail: {sbian, fyhssgss, dzhao, panhaowen, yuexjin, guanzhenyu}@buaa.edu.cn).

J. Sun is with the College of Information Science and Engineering, Northeastern University, Shenyang 110819, China (e-mail: jyuesun@163.com)

H. Qiao is with the Department of Automation, Beijing National Research Center for Information Science and Technology, Institute for Brain and Cognitive Sciences, Tsinghua University, Beijing 100084, China (e-mail: qiaohui@mail.tsinghua.edu.cn).

functions such as saturation, dead-zone, and hysteresis are applied after the linear transformations to stabilize the system states and protect the actuating parts. In this work, we refer to this type of setting as linear-control-with-non-linear-actuation (LCNA) systems. We point out that, due to the absence of non-linear operators over HE ciphertexts, most (if not all) existing encrypted control schemes do not support LCNA systems.

In this work, we propose FHECAP, an FHE-based encrypted control system capable of processing linear control states followed by non-linear actuation functions. We observe that most existing encrypted control systems, whether based on PHE or LHE, have limited operator expressiveness and thereby do not support the homomorphic evaluation of non-linear functions. Contrarily, based on the idea of piecewise interpolation, we segment a general non-linear function into sub-pieces of non-linear functions, where each of the function pieces can be evaluated by a separate lower-level FHE primitive. In this way, FHECAP simultaneously achieves higher evaluation accuracy and faster computation speed for the evaluation of control-related non-linear functions over HE ciphertexts. Specifically, to the best of our knowledge, FHECAP is the first FHE-based controller framework that can evaluate linear control laws concatenated by non-linear actuation without additional rounds of communication. The contributions of this work are summarized as follows.

- **A General Encrypted Control Framework**: We propose an encrypted control framework for linear control with non-linear actuation functions. To the best of our knowledge, FHECAP is the first FHE-based encrypted controller for LCNA systems.
- **Cross-Scheme FHE Infrastructure**: We observe that it is essential to adapt multiple HE schemes (including both LHE and FHE schemes) for the expressive and secure control state evaluation. In particular, we devise a general formulation for the homomorphic evaluation of arbitrary non-linear functions through piecewise interpolation of cross-scheme FHE operators.
- **Automated Noise Analysis**: To ensure the usability of FHECAP, we rigorously study the noise characteristics of FHECAP under different homomorphic parameters. Thus, the participating parties can adjust parameters according to actual control requirements to guarantee correct and stable operating states while retaining system performance.
- **Thorough Evaluations**: In experiments, we show that FHECAP can be $4\times$–$1000\times$ faster than existing works when evaluating linear control laws. Furthermore, FHECAP is capable of evaluating a large number of highly complex non-linear functions over FHE ciphertexts, including saturation, dead-zone, and relay. As a case study, we demonstrate the effectiveness of FHECAP in protecting both the system safety and the data security through the example control system of cooperative spacecraft rendezvous.

The rest of this paper is organized as follows. First, in Section II, we introduce preliminaries on basic HE schemes and operators. Second, the abstract formulation of the LCNA systems and the threat models are outlined in Section III. Third, in Section IV, we sketch the overview on the FHECAP framework and explain the detailed algorithmic constructions for each of the components in the encrypted controller. Fourth, in Section V, we derive theoretical bounds on the noise growths of FHECAP and suggest methods to properly set the encryption parameters. Fifth, the performance of FHECAP is illustrated through mini-benchmark experiments and an end-to-end case study in Section VI. Finally, we conclude our work in Section VII.

TABLE I
LIST OF ABBREVIATIONS

Abbreviation	Full Form
BK	Bootstrapping Key
CB	Circuit Bootstrapping
CDF	Cumulative Distribution Function
FHE	Fully Homomorphic Encryption
HE	Homomorphic Encryption
KSK	Key Switching Key
LCNA	Linear Control with Non-linear Actuation
LHE	Leveled Homomorphic Encryption
LUT	Look-up Table
LWE	Learning with Errors
MPC	Multi Party Computation
NTT	Number Theoretic Transform
PA	Polynomial Approximation
PHE	Partially Homomorphic Encryption
RGSW	Ring Gentry-Sahai-Waters
RLWE	Ring Learning with Errors
SK	Secret Key
SIMD	Single-Instruction-Multi-Data

II. PRELIMINARIES

In this section, we introduce the basic notations and concepts associated with the state-of-the-art homomorphic encryption schemes [28], [29], [39]–[41], as well as existing work on encrypted control and FHE-based non-linear function evaluation.

A. Acronyms and Notations

We use lowercase letters with tilde to denote polynomials (e.g., $\widetilde{a}$), boldface lowercase letters for vectors (e.g., $\boldsymbol{a}$), and capital letters for matrices (e.g., A). Similarly, capital letters with tilde are matrices whose entries are polynomials. For a vector $\boldsymbol{a}$, we denote by $\|\boldsymbol{a}\|_i$ its i-th norm. For a polynomial $\widetilde{a}$, $\|\widetilde{a}\|_i$ is the i-th norm of the coefficient vector of $\widetilde{a}$. For a matrix A, we denote by A_i its i-th row vector. $\mathbb{Z}$ refers to the set of integers while $\mathbb{Z}_q$ represents to the set of integers modulo q. Additionally, $\mathbb{Z}_q[\tau]$ denotes the set of polynomials with coefficients in $\mathbb{Z}_q$. For a comprehensive list of notations, terminologies, and abbreviations used in this paper, please refer to Table A1 and Table I.

B. Homomorphic Encryption Schemes

In this work, we adopt three types of ciphertext across different HE schemes: the ring learning with errors (RLWE) ciphertexts from the CKKS [29] scheme, the learning with errors (LWE) ciphertext from the TFHE scheme [40], and the ring Gentry-Sahai-Waters (RGSW) ciphertext from the GSW scheme [39]. The ciphertexts for the respective HE schemes are sketched as follows.

- $\mathsf{LWE}_{\boldsymbol{s}}^{n,q}(m)$: The Learning with Error (LWE) ciphertext. An LWE ciphertext is defined as $(\boldsymbol{a}, b) \in \mathbb{Z}_q^{n+1}$ which encrypted a plaintext message $m \in \mathbb{Z}_p$, satisfying the following equation:
$$b = \langle \boldsymbol{a}, \boldsymbol{s} \rangle + \Delta m + e_{\text{fresh}} \pmod q. \tag{1}$$
where $\boldsymbol{a} \in \mathbb{Z}_q^n$ is chosen uniformly at random, secret key $\boldsymbol{s} \in \mathbb{Z}_2^n$ is a uniform random binary vector, e is a random noise sampled from the χ_σ distribution, and Δ is a scaling factor to separate the message from the noise.
- $\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{m})$: The Ring Learning with Error (RLWE) ciphertext. Similar to the LWE ciphertext, an RLWE ciphertext is also a tuple $(\widetilde{a}, \widetilde{b}) \in R_Q^2$, where
$$\widetilde{b} = \widetilde{a} \cdot \widetilde{s} + \Delta \widetilde{m} + \widetilde{e}_{\text{fresh}}. \tag{2}$$
RLWE ciphertexts are usually used to encrypt a vector of plaintext integers, and there are two methods to encode a plaintext vector $\boldsymbol{a} \in \mathbb{Z}^N$ into a polynomial $\widetilde{m} \in R_q$: the number-theoretic-transform-domain (NTT-domain) encoding and the coefficient-domain encoding.
NTT-Domain Encoding: For NTT-domain encoding, the plaintext vector $\boldsymbol{a}$ is first transformed into the NTT domain $\texttt{NTT}(\boldsymbol{a})$. Then, the elements of $\texttt{NTT}(\boldsymbol{a})$ are embedded as the coefficients of the encoded polynomial $\widetilde{a}$, denoted as $\widetilde{a}_{\texttt{NTT}} = \texttt{NTTEcd}(\boldsymbol{a})$.
Coefficient-Domain Encoding: For coefficient-domain encoding, the elements of $\boldsymbol{a}$ are directly embedded into the coefficients of the encoded polynomial $\widetilde{a}$, depicted as $\widetilde{a}_{\texttt{Coef}} = \texttt{CoefEcd}(\boldsymbol{a})$.
- $\mathsf{RGSW}_{\widetilde{s}}^{n,q,\mathsf{Bg},d}(\widetilde{m})$: The RGSW ciphertext is a tuple of $(\widetilde{A}, \widetilde{B}) \in R_Q^{2\times 2d}$, which can be regarded as a composition of $2d$ RLWE ciphertexts. Here, we have
$$\widetilde{B} = \widetilde{A} + \widetilde{m} \cdot H, H = I_2 \otimes D_{\mathsf{Bg},d}, \tag{3}$$
where $\widetilde{A}$ is $2d$ encrypted zero polynomial RLWE ciphertext matrix, and H denote the gadget matrix where $D_{\mathsf{Bg},d} = \left[\frac{1}{\mathsf{Bg}} \quad \cdots \quad \frac{1}{\mathsf{Bg}^d}\right]^T$ and I_2 is a 2×2 identity matrix. The detailed construction can be referred to in [39].

C. Homomorphic Operators

Here, we explain the three classes of fundamental HE operators used throughout this work: arithmetic operators, logic operators, and conversion operators. Note that when the parameters are not important to the discussion, we abbreviate the above ciphertexts notation as $\mathsf{LWE}(m)$, $\mathsf{RLWE}(\widetilde{m})$, and $\mathsf{RGSW}(\widetilde{m})$.

Arithmetic Operators:

- $\pm$ of LWE/RLWE: Ciphertext addition and subtraction. The underlying computations are simply the coefficient-wise addition and subtraction of the polynomials in the inputs ciphertexts. For instance, for two LWE ciphertexts $\mathsf{LWE}(m_0) = (\boldsymbol{a}_0, b_0)$ and $\mathsf{LWE}(m_1) = (\boldsymbol{a}_1, b_1)$, the result of an addition or subtraction operator is $\mathsf{LWE}(m_0 \pm m_1) = (\boldsymbol{a}_0 \pm \boldsymbol{a}_1, b_0 \pm b_1)$. The same procedure applies to RLWE ciphertext.
- $\cdot$ of RLWE: Plaintext-ciphertext multiplication. This operation takes as input a plaintext polynomial $\widetilde{m}_0$ and a ciphertext $\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{m}_1)$, and the output is $\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{m}_0 \cdot \widetilde{m}_1)$. The modulus of RLWE is reduced from Q' to Q through `Rescaling` [29].
- $\times$ for RLWE and RGSW: External product. This operation requires an RLWE and an RGSW, with the output satisfied as follows:
$$\mathsf{RLWE}(\widetilde{m}_0) \times \mathsf{RGSW}(\widetilde{m}_1) = \mathsf{RLWE}(\widetilde{m}_0 \cdot \widetilde{m_1}). \tag{4}$$
- Rotation of RLWE: Give an $\mathsf{RLWE}(\widetilde{a}_{\texttt{NTT}})$ encoded and encrypted from the plaintext vector $\boldsymbol{a} = \{a_i\}_{0\le i<N}$, this operation homomorphically rotates the first l slots of $\boldsymbol{a}$ to the end, resulting in a new ciphertext $\mathsf{RLWE}(\widetilde{a}'_{\texttt{NTT}})$ where the corresponding plaintext vector $\boldsymbol{a}' = \{a_l, \cdots, a_{N-1}, a_0, \cdots, a_{l-1}\}$, which is denoted as $\texttt{HomRot}(\mathsf{RLWE}(\widetilde{a}_{\texttt{NTT}}, l)$.

Logic Operators:

- `CMUX`: Homomorphic selector. Given inputs $\mathsf{RLWE}(\widetilde{a})$ and $\mathsf{RLWE}(\widetilde{b})$ with a section signal $\mathsf{RGSW}(\widetilde{t})$ where $\widetilde{t} \in \{0, 1\}$, `CMUX` computes $\mathsf{RLWE}(\widetilde{c})$ where $\widetilde{c} = \widetilde{a}$ if $\widetilde{t} = 1$ or $\widetilde{c} = \widetilde{b}$ if $\widetilde{t} = 0$. We denote this operator as $\texttt{CMUX}(\mathsf{RGSW}(\widetilde{t}), \mathsf{RLWE}(\widetilde{a}), \mathsf{RLWE}(\widetilde{b}))$.
- `LUT`: Look-up table function (LUT). Initially, the encoded LUT polynomial $\widetilde{tab}$ is constructed by giving an arbitrary discrete function $f(\cdot)$. After preprocessing, this operation takes $\mathsf{LWE}(m)$ as input each time and obtains the output $\texttt{HomLUT}(\mathsf{LWE}(m), \widetilde{tab}) = \mathsf{LWE}(f(m))$ by homomorphically looking up the function value of m on $\widetilde{tab}$.
- `HomComp`: Homomorphic comparison. Given two LWE ciphertexts $\mathsf{LWE}(a)$ and $\mathsf{LWE}(b)$, this operation outputs a new LWE ciphertext that indicates the relative magnitude of these two ciphertexts. In this work, we use the notation $\texttt{HomComp}(\mathsf{LWE}(a), \mathsf{LWE}(b)) = \mathsf{LWE}(0)$ to denote the case where $a \ge b$ and $\texttt{HomComp}(\mathsf{LWE}(a), \mathsf{LWE}(b)) = \mathsf{LWE}(1)$ for $a < b$.

Conversion Operators:

- LWEToRLWE: Let $\mathsf{LWE}_{\boldsymbol{s}}^{n,Q}(m) = (\boldsymbol{a}, b)$, by rearranging $\boldsymbol{a}$ to the coefficients of the new polynomial $\widetilde{a}$, and placing b as a constant term of a other-wise zero polynomial $\widetilde{b}$, $(\widetilde{a}, \widetilde{b})$ forms the new ciphertext $\mathsf{RLWE}_{\widetilde{s}}^{n,Q}(\widetilde{m}')$ where the constant term of $\widetilde{m}'$ is m.
- RLWEToLWE: Taking as an input the index parameter i $(0 \le i < N)$, RLWEToLWE extracts the LWE ciphertext encrypting the i-th plaintext message $\mathsf{LWE}_{\boldsymbol{s}}^{n,Q}(\widetilde{m}_i)$ from $\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{m})$ [40] which denoted by $\mathsf{RLWEToLWE}(\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{m}), i)$.
- LWEToRGSW: The conversion from $\mathsf{LWE}(m)$ to $\mathsf{RGSW}(\widetilde{m})$, which is referred to as `CB` in [40].

D. Related Works on Encrypted Control

In this work, we classify related literature over existing encrypted control systems into main groups: Paillier-based encrypted control systems, and LHE-based encrypted control systems. In what follows, we give a brief review on the recent advances in both groups of encrypted control systems.

1) Paillier-based Protocols: A number of Paillier-based encrypted control systems are recently proposed to enhance data security while retaining high computation efficiency. The Paillier encryption scheme is a typical representative for partially homomorphic encryption, which supports homomorphic

addition between ciphertexts, as well as scalar multiplication between plaintext and ciphertext. By utilizing such operations, we can implement homomorphic linear transformation over ciphertexts, which corresponds to the instantiation of encrypted linear control systems [9], [10], [12]–[17]. To encode real-valued plaintext data into integers, [10] proposes to equip quantizers to plaintexts, where the exact quantization factors are determined by the range of the control value. Similarly, [12] rationalizes fraction numbers into a new encoding scheme to convert the real-valued inputs to integers. Building upon these works, recent studies have further advanced quantization in Paillier-based systems. [13] takes into account the noises caused by ciphertext addition and multiplication in the design of the quantizer. Furthermore, [17] introduced a novel approach for designing quantization sensitivity that considers system performance, while [16] demonstrated the practical stability of systems under stable control laws based on quantization error analysis. Although the above works explore the application capabilities of Paillier-based encrypted control systems, it is still difficult to apply such protocols in real-world scenarios due to the limitations in operator expressiveness. In particular, partially homomorphic encryption schemes do not have adequate support for non-linear function evaluations.

2) LHE-based control: LHE-based encrypted control systems rely on lattice-based cryptosystems, and support batched encryption of multiple plaintext values into a single ciphertext. Therefore, LHE-based control systems like [19], [20], [23]–[27] can realize encrypted linear control systems much more efficiently than the Paillier-based protocols. For example, [24] utilizes homomorphic rotation to facilitate efficiently achieves matrix-vector multiplication over ciphertexts. Moreover, [19] avoids `HomRot` by performing vector accumulation on the decrypted ciphertexts, which, in turn, increases the computational burden on the plant. [20], [23] employ similarity transformations to convert decimals into integers, enabling more efficient ciphertext multiplications at the cost of increased decryption failure probability. [25] focus on improving computational efficiency through precomputed table-based methods, with subsequent work [26] enhancing security at the cost of increased storage and reduced robustness to disturbances. In short, LHE-based control systems can support more efficient evaluation of encrypted linear systems, where the communication overheads can be independent of the system states. Unfortunately, similar to the case of Paillier-based systems, most existing LHE-based solutions can only support evaluating linear control systems over ciphertexts.

E. Related Works on Non-linear Function Evaluation

In this work, we classify the technical approaches of existing FHE-based non-linear function evaluation works into two major groups: look-up table (LUT) algorithms based on DM/CGGI schemes [40], [41] and polynomial approximation (PA) algorithms based on the CKKS scheme. In the following, we provide a brief review of the recent advancements in these two groups.

1) LUT algorithms based on DM/CGGI schemes: In DM/CGGI schemes, the bootstrapping operation, which refreshes ciphertext noise, can evaluate an LUT polynomial. In TFHE, a new ciphertext representing the targeted function value can be obtained by modifying the encoding of the LUT polynomial. [42], [43] implement the evaluation of certain non-linear functions within the TFHE cryptosystem, such as the sign function. Due to the limited size of the LUT, the ciphertext can only support precise lookup within a small bit range, resulting in low precision. Achieving high-precision LUT evaluation requires increasing the parameters of the cryptographic scheme, which leads to significant performance degradation. To deal with this issue, [44] proposed a tree-based LUT evaluation algorithm, which decomposes a large LUT into multiple smaller LUTs, thereby reducing parameter size and improving precision. However, in control theory, non-linear control functions require continuity and much higher precision, and none of the aforementioned algorithms can meet the practical requirements of encrypted control systems.

2) PA algorithms based on CKKS schemes: Since CKKS schemes support homomorphic polynomial evaluation, the evaluation of non-linear functions is transformed into the task of finding an appropriate approximating polynomial that closely fits the non-linear function. The selection of the polynomial involves a trade-off between low degree (for reduced latency) and low approximation error (for high accuracy). Several methods have been proposed to approximate non-polynomial functions, such as using Chebyshev polynomial bases [45], minimax polynomials [46]. [47], [48] introduce a series of low-degree polynomials tailored for specific functions. However, to reduce the approximation error, this approach requires increasing the degree of the approximating polynomial, which in turn demands greater multiplicative depth and ring dimension, leading to higher communication overhead and computational burden.

III. Problem Statement

In this section, we first formulate the linear control system with non-linear piecewise actuation in Section III-A. We then establish the threat model for the system in Section III-B.

A. System Formulation

An LCNA system $\mathcal{L}$ has the general mathematical model

$$\dot{\boldsymbol{x}}(t) = A \cdot \boldsymbol{x}(t) + B \cdot \boldsymbol{u}(t), \boldsymbol{u}(t) = G_\Gamma(K \cdot \boldsymbol{x}(t)), \quad (5)$$

where $A \in \mathbb{R}^{r\times r}$ and $B \in \mathbb{R}^{r\times l}$ are two constant matrices, $\boldsymbol{x} \in \mathbb{R}^r$ is the system state, and $\boldsymbol{u} \in \mathbb{R}^l$ is the system input. $K \in \mathbb{R}^{l\times r}$ is the linear feedback matrix and $G : \mathbb{R}^l \to \mathbb{R}^l$ represents a family of piecewise non-linear functions specifically designed to fit the non-linear characteristics of the actuators (e.g., saturation functions or dead-zone functions). For the control input vector $\boldsymbol{u} = (u_1, u_2, ..., u_l)^T$, $G_\Gamma(\boldsymbol{u})$ can be expressed as

$$G_\Gamma(\boldsymbol{u}) = (G_{\boldsymbol{\gamma}_1}(u_1), G_{\boldsymbol{\gamma}_2}(u_2), ..., G_{\boldsymbol{\gamma}_l}(u_l))^T, \quad (6)$$

where actuation parameter $\Gamma = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, ..., \boldsymbol{\gamma}_l)^T$ is an $l \times k$ matrix composed parameters for each of the non-linear functions. Explicitly, the threshold of $G_{\boldsymbol{\gamma}_i}$ is $[\zeta_{l,i}, \zeta_{r,i}]$, and the expression is defined as

$$G_{\boldsymbol{\gamma}_i}(x) = G_{i,j}(x), x \in [\gamma_{i,j-1}, \gamma_{i,j}). \quad (7)$$

where $1 \le j \le k, \gamma_{i,0} = \zeta_{l,i}$, $\gamma_{i,k} = \zeta_{r,i}$. Each piece of functions in Equation (7) can take three forms: (i) a

constant function (e.g., $f(x) = \alpha$), (ii) a linear function (e.g., $f(x) = \beta(x-\alpha)$), (iii) or a non-linear function. We emphasize that, for *plaintext* control systems, formulating a general non-linear function into pieces can still be non-linear may not be practically useful. Nevertheless, as later demonstrated in Section VI, segmenting a general non-linear function into small pieces can significantly enhance the function evaluation accuracy over HE ciphertexts, which is critical in ensuring correct system behaviors.

Throughout this paper, we make the following assumptions about the target system $\mathcal{L}$:

Assumption 1: The system states and the inputs of $\mathcal{L}$ are bound in the closed-loop. We denoted u_m to be the maximum of $|u(t)|$.

Assumption 2: For $\mathcal{L}$, given a parameter $\epsilon > 0$, there exists $\eta(\epsilon) > 0$ such that if there is an error $e_{\boldsymbol{u}(t)} < \eta(\epsilon)$ in the system input, then the error in the system state $e_{\boldsymbol{x}(t)} < \epsilon$.

We note that our work focuses on encrypted computations over controllers rather than the design of control systems or controllers themselves. The reasonableness of the above assumptions lies in enabling a quantitative co-design framework between control and cryptography. Specifically, Assumption 1 defines a bounded range for the state and input of the control system, predetermining an upper bound to facilitate the selection of an appropriate quantizer in Section V-A. Assumption 2, which aligns with [23], is introduced to analyze the impact of actuator errors on system state errors, thereby enabling a quantitative analysis of system state errors induced by HE decrypted input errors.

B. Threat Model and Security

As sketched in Figure 1, in a cloud-based control system, we have two participating parties: a plant $\mathcal{P}$ and a cloud controller $\mathcal{C}$. We assume that $\mathcal{P}$ fully outsources its control computation tasks to $\mathcal{C}$. Hence, the threat model for the general cloud-based control protocol in Figure 1 can be formulated as follows.

Threat Model: We assume that both $\mathcal{P}$ and $\mathcal{C}$ are semi-honest adversaries [49], in that both parties strictly follow the prescribed protocol but wish to infer as much as possible the private data of the other party. In more detail, we have two types of settings based on the semi-honest adversary assumptions, namely, the outsourced computation setting and the two-party computation setting. We specify the public and private data for each of the settings in Table II. Roughly speaking, we assume the dimension of system state r, l and actuation function G, to be public. For outsourced controlling, $\mathcal{C}$ does not have any private data, and only needs to perform the outsourced control commands. Thus, in such case, the gain matrix K is public. Meanwhile, and state $\boldsymbol{x}(t)$ is private to the client $\mathcal{P}$. In the case of secure two-party controlling, both parties need to protect their respective private data, where the gain matrix K is private to $\mathcal{C}$ and system state $\boldsymbol{x}(t)$ is private to $\mathcal{P}$. Lastly, if the actuation function has parameters, the data can be either public or private to the two participating parties depending on the exact use-case.

Security: Since the entire controlling process is performed over FHE ciphertexts, the security of FHECAP directly follows from that of the underlying FHE schemes, which can be reduced to the hardness of the LWE and RLWE problems over lattices [50]. Note that, all FHE schemes adopted in this work achieves IND-CPA security [51], which means that the FHE ciphertexts are indistinguishable under chosen-plaintext attacks. Meanwhile, it is well-known that an encryption scheme achieving IND-CPA security is secure under the semi-honest adversary. Hence, we say that the overall protocol of FHECAP is secure against the threat model defined above.

TABLE II
SUMMARY OF SECURITY PROPERTIES OF CONTROL PARAMETERS

Parameter	Outsourced	Two-Party
State Dimension r, l	Public	
Actuation Function G	Public	
Gain Matrix K	Public	Private for $\mathcal{C}$
System State $\boldsymbol{x}(t)$	Private	Private for $\mathcal{P}$
Actuation Parameter Γ	Either	Either

Remark: The concrete security level of FHE is primarily determined by the choice of encryption parameters, i.e., the lattice dimension n the sizes of the ciphertext moduli q and Q. In general, higher security levels require larger parameter sets, which in turn results in lengthened computation latency.

IV. System Specification

Here, we first provide an overview of the workflow of FHECAP in Section IV-A, where we briefly describe each of the steps for the overall protocol. Next, we introduce how cross-scheme HE primitives are used to construct FHECAP in Section IV-B.

A. FHECAP Workflow

Figure 2 shows the overall process of FHECAP in one round of interaction between $\mathcal{P}$ and $\mathcal{C}$, which is based on the client-server model. Here, the client is the controlled plant, and the server is the cloud-based controller. On the client side, $\mathcal{P}$ is equipped with the FHECAP client interface, which contains a `Pre-Install` module, an `Encode` module, an `Encrypt` module, a `Decode` module, and a `Decrypt` module. On the server side, $\mathcal{C}$ runs the main service of FHECAP, which contains multiple calls of heavy HE operators to enforce the control laws over the HE ciphertext. The protocol in Figure 2 is further decomposed into the following concrete steps.

Protocol Setup. Before the start of the control round, $\mathcal{P}$ initiates the `Pre-Install` module to create the secret key (SK) embedding encrypt and decrypt modules. Furthermore, the bootstrapping key (BK) and key switching key (KSK) are derived by utilizing SK. $\mathcal{P}$ then proactively transmits BK, KSK, Γ and public parameters to $\mathcal{C}$. If Γ is public to $\mathcal{P}$, $\mathcal{P}$ will send the plaintext list; otherwise, encrypt them into ciphertexts and send. More specifically, the split points γ_i of the piecewise function are encrypted as LWE ciphertexts; α of the sub-functions of (i) and (ii) in Section III-A are encrypted as RLWE ciphertexts, and β of (ii) sub-functions are encrypted as RGSW ciphertexts. This data will be frequently employed in subsequent protocol interactions, and preemptively transmitting it effectively minimizes the volume of protocol communication.

① **The Encoding Step**. The plant collects the system state $\boldsymbol{x} \in \mathbb{R}^r$ from the sensor and encodes the state through the

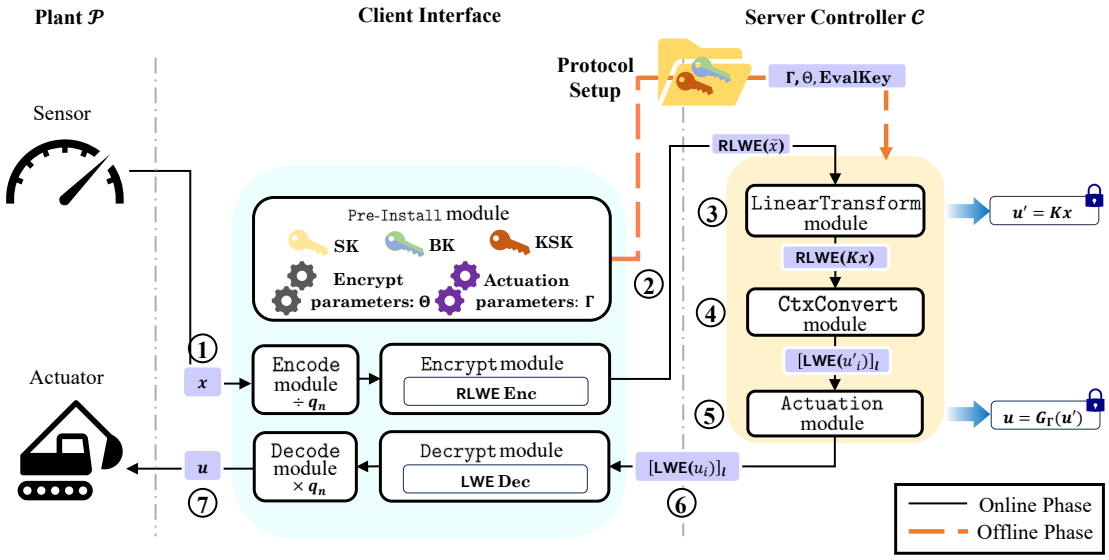


Figure 2. A conceptual illustration of the FHECAP framework.

`Encode` module using the client interface. The main purpose of the encoding step is to normalize each of the r elements in $\boldsymbol{x}$ by the quantizer δ (the choice of δ will be discussed in Section V-A). Then, we put the i-th element of $\boldsymbol{x}$ on the $(\frac{i \cdot N}{l \cdot r} - 1)$-th coefficient of the polynomial $(0 < i \leq r)$, and fill the rest with 0. At the end of this step, we obtain an encoded plaintext polynomial $\widetilde{x}$ of degree N.

② **The Encryption Step**. The client interface uses the `Encrypt` module to combine the secret key SK and the encoded $\widetilde{x}$ from the previous step to obtain an RLWE ciphertext $\mathsf{RLWE}(\widetilde{x})$, and send the resulting ciphertext to the cloud-based controller $\mathcal{C}$.

③ **The Linear Transformation Step**. The controller first applies linear control laws over the input ciphertext $\mathsf{RLWE}(\widetilde{x})$, which translates to the homomorphic evaluation of the inner product between the gain matrix K and the system state $\boldsymbol{x}$. As explained in Section III-B, K is stored in plaintext on $\mathcal{C}$. To encode K, we first flatten the matrix into a vector, i.e., we concatenate all of the rows in K to form a single-row vector $\boldsymbol{k}$. We then encode $\boldsymbol{k}$ into a plaintext polynomial $\widetilde{k}$. Therefore, to compute $K \cdot \boldsymbol{x}$, we actually need to perform a multiplication between the plaintext polynomial $\widetilde{k}$ and the ciphertext polynomial $\mathsf{RLWE}(\widetilde{x})$, which can be efficiently carried out using the $\cdot$ operator over RLWE ciphertexts. Further details on the exact computations can be found in Section IV-B1.

④ **The Ciphertext Conversion Step**: As mentioned in Section II-C, we adopt two distinct types of HE schemes (arithmetic and logic) to treat the linear and non-linear control functions. As a result, a ciphertext format conversion step is required to convert the RLWE after the linear transformation to to a set of l LWE ciphertexts for the subsequent non-linear functions.

⑤ **The Non-Linear Actuation Step**: After applying the linear transformation and converting the results to a set of LWE ciphertexts encrypting the intermediate result $\boldsymbol{u}'$, we compute system input $\boldsymbol{u}$ by applying a non-linear actuation function, which is segmented into a series of piecewise non-linear functions. In a nutshell, we need to carry out the following

Algorithm 1: Homomorphic Linear Transformation

Input : Matrix K
Input : An RLWE ciphertext $\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{x}_{\mathtt{Coef}})$ where $\widetilde{x}_{\mathtt{Coef}} = \mathtt{CoefEcd}(\boldsymbol{x})$
Output: An RLWE ciphertext $\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{\mathsf{k}x}_{\mathtt{Coef}})$

$\mathbf{k} \leftarrow (0)_{lr}$
for $i = 0$ *to* $l - 1$ **do**
  $\boldsymbol{k}_{\mathrm{rev},i} \leftarrow \mathtt{reverse}(\boldsymbol{k}_i)$
$\mathbf{k}_{\mathtt{Coef}} \leftarrow \boldsymbol{k}_{\mathrm{rev},0} \| \cdots \| \boldsymbol{k}_{\mathrm{rev},l-1}$
$\widetilde{\mathsf{k}}_{\mathtt{Coef}} \leftarrow \mathtt{CoefEcd}(\mathbf{k}_{\mathtt{Coef}})$
$\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{\mathsf{k}}_{\mathtt{Coef}} \cdot \widetilde{x}_{\mathtt{Coef}}) \leftarrow \widetilde{\mathsf{k}}_{\mathtt{Coef}} \cdot \mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{x}_{\mathtt{Coef}})$
return $\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{\mathsf{k}}_{\mathtt{Coef}} \cdot \widetilde{x}_{\mathtt{Coef}}))$

computation homomorphically:

$$\mathsf{LWE}(u'_i) \rightarrow \mathsf{LWE}(G_{\gamma_i}(u'_i)) = \mathsf{LWE}(u_i), 1 \leq i \leq l. \quad (8)$$

After the application of Equation (8), we acquire the set of encrypted control values $\{\mathsf{LWE}(u_i)\}_l$, which are essentially the system inputs returned to the client $\mathcal{P}$ with proper data ranges.

⑥ **The Decryption Step**: After receiving the set of system inputs $\{\mathsf{LWE}(u_i)\}_l$ from the cloud controller, the plant calls the `decrypt` module through the client interface, and obtain the final control values $\{u_i\}$ for $1 \leq i \leq l$.

⑦ **The Decoding Step**: The l integers obtained in the previous step are encoded. We perform inverse transformation and multiply by ϵ to obtain a real system input vector with practical significance. The input is fed to the actuation to perform actions.

B. Cryptographic Building Blocks

Here, we make a deeper dive into the cryptographic details for the homomorphic algorithms on the cloud controller.

1) Homomorphic Linear Transformation: To apply the linear transformation of an $l \times r$ matrix K on the encryption of the vector $\boldsymbol{x}$ homomorphically, existing works [24], [52]–[55] adopt the NTT-domain encoding to realize the multiplication-accumulation process for homomorphic matrix-vector multiplication. More specifically, the rows of K are transposed and arranged to form a vector $\mathbf{k}_{\mathrm{NTT}}$ of length

$r \cdot l$, where $K_{i,j}$ corresponds to $\mathbf{k}_{\texttt{NTT},i\cdot r+j}$. Then, $\mathbf{k}_{\texttt{NTT}}$ is encoded as $\widetilde{\mathsf{k}}_{\texttt{NTT}}$. Meanwhile, $\boldsymbol{x}$ is duplicated l times to form a new vector $\mathbf{x}_{\texttt{NTT}}$ of the same length as $\mathbf{k}_{\texttt{NTT}}$, where $\mathbf{x}_{\texttt{NTT}}$ is further encoded and encrypted as $\mathsf{RLWE}(\widetilde{\mathbf{x}}_{\texttt{NTT}})$. Directly multiplying the two together gives the RLWE ciphertext of the Hadamard product of $\mathbf{k}_{\texttt{NTT}}$ and $\mathbf{x}_{\texttt{NTT}}$. Subsequently, the final result $c_1 = \mathsf{RLWE}(\texttt{NTTEcd}(\mathbf{k}_{\texttt{NTT}} \bigodot \mathbf{x}_{\texttt{NTT}}))$ is obtained by summing each slot of the vector through $\log_2 \frac{N}{l}$ times $\texttt{HomRot}$ operations according to the following transformation:

$$c_{\text{out}} \leftarrow c_1 + \texttt{HomRot}(c_1, 2^i), 0 \le i \le \log_2\lceil\frac{N}{l}\rceil. \tag{9}$$

Equation (9) requires in total of one multiplication, $\log_2\lceil\frac{N}{l}\rceil$ additions and $\log_2\lceil\frac{N}{l}\rceil$ rotations. However, since homomorphic rotation is as heavy as homomorphic multiplication, the overall algorithmic complexity of Equation (9) remains high.

More recently, it is demonstrated that coefficient-domain encoding can be more efficient than NTT-domain encoding under the specific context of MPC-based privacy-preserving machine learning applications [56], [57]. To carry out homomorphic linear transformation over RLWE ciphertexts with coefficient encodings, $\boldsymbol{x}$ is directly fed into the plaintext polynomial encoding process without pre-encoding as in the NTT-domain encoding case, where we have $\widetilde{\boldsymbol{x}}_{\texttt{Coef}} = \texttt{CoefEcd}(\boldsymbol{x})$. On the other hand, to prepare K for the linear transformation, we still need to pre-encode the matrix K into a vector $\mathbf{k}_{\texttt{Coef}}$. The encoding process of $\mathbf{k}_{\texttt{Coef}}$ is described on Line 1–5 in Algorithm 1. First, on Line 3, we reverse each element of $\boldsymbol{k}_i$, i.e., the i-th row of K, to get $\boldsymbol{k}_{\text{rev},i}$. Then, on Line 4, we concatenate the resulting vectors $\{\boldsymbol{k}_{\text{rev},i}\}$ to get the encoded vector $\mathbf{k}_{\texttt{Coef}}$. Next, we encode the vector $\mathbf{k}_{\texttt{Coef}}$ into the polynomial $\widetilde{\mathsf{k}}_{\texttt{Coef}} = \texttt{CoefEcd}(\mathbf{k}_{\texttt{Coef}})$. Note that, since the cloud controller knows K in advance, the encoding process can be pre-processed. After data encoding, the multiplication between $\widetilde{\mathsf{k}}_{\texttt{Coef}}$ and $\mathsf{RLWE}(\widetilde{x}_{\texttt{Coef}})$ on Line 6 corresponds to the homomorphic convolution between $\widetilde{\mathsf{k}}_{\texttt{Coef}}$ and $\widetilde{x}_{\texttt{Coef}}$, where i-th coefficient of $\widetilde{\mathsf{k}}_{\texttt{Coef}} \cdot \widetilde{x}_{\texttt{Coef}}$ can be formulated as

$$\begin{aligned}(\widetilde{\mathsf{k}}_{\texttt{Coef}} \cdot \widetilde{x}_{\texttt{Coef}})_i &= \sum_{j=0}^{N-1} \widetilde{x}_{\texttt{Coef},j}\widetilde{\mathsf{k}}_{\texttt{Coef},i-j \bmod N} \\ &= \sum_{j=0}^{r\cdot l-1} \boldsymbol{x}_j \mathbf{k}_{\texttt{Coef},\frac{i\cdot l\cdot r}{N}-j \bmod r\cdot l}\end{aligned} \tag{10}$$

Based on the insights of Equation (10), it is discovered that, when $\frac{i\cdot l\cdot r}{N} \equiv -1 \pmod r$, the right hand side Equation (10) becomes $\sum_{j=0}^{r-1} \boldsymbol{x}_j K_{\lfloor\frac{i\cdot l}{N}\rfloor,j}$, which is essentially the inner product of the $\lfloor\frac{i\cdot l}{N}\rfloor$-th row of K and $\boldsymbol{x}$. Thus we obtain the results of matrix-vector multiplication over ciphertext via selecting appropriate indices of $\mathsf{RLWE}(\widetilde{\mathsf{k}}_{\texttt{Coef}} \cdot \widetilde{x}_{\texttt{Coef}})$ without requiring any rotation operations.

2) Homomorphic Ciphertext Conversion: To generate the pre-actuation system input ciphertext, we convert the resulting ciphertext $\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{\mathsf{k}}_{\texttt{Coef}} \cdot \widetilde{x}_{\texttt{Coef}})$ from the previous step into a set of $\mathsf{LWE}_{\boldsymbol{s}}^{n,q}$ ciphertexts. During this conversion, two key tasks must be addressed: (1) homomorphically extracting the valid inner products at the correct positions and (2) ensuring parameter compatibility between the CKKS and TFHE schemes. Hence, we apply

$$\begin{aligned}\mathsf{LWE}_{\boldsymbol{s}}^{n,q}(u'_{\lfloor\frac{il}{N}\rfloor+1}) &= \frac{q}{Q'}\mathsf{LWE}_{\boldsymbol{s}}^{n,Q'}(u'_{\lfloor\frac{il}{N}\rfloor+1}) \\ &\leftarrow \mathsf{RLWEToLWE}(\mathsf{RLWE}_{\widetilde{s}}^{N,Q'}(\widetilde{\mathsf{k}}_{\texttt{Coef}} \cdot \widetilde{x}_{\texttt{Coef}}), i),\end{aligned} \tag{11}$$

where $i \equiv -1 \pmod{\frac{N}{l}}$. Consequently, we obtain l LWE ciphertexts, each encrypting one element.

3) Homomorphic Non-linear Actuation: Due to the limited data precision and inherent algebraic properties, HE does not perform well for general non-linear functions. Consequently, to the best of our knowledge, none of the existing HE-based encrypted control solutions support non-linear functions in any part of their systems. To solve the usability and compatibility issues, we propose a piecewise approach towards common non-linear actuation functions to achieve both accurate and fast non-linear function evaluation. Specifically, before actually performing any computation, we first segment the non-linear actuation function G into the corresponding sub-functions. Note that, G actually contains l different actuation functions for each of the vector dimensions of the system state (x). Hence, we need to segment each of the i-th function G_i into k pieces of sub-functions $\{G_{i,j}\}$, where $1 \le i \le l$ and $1 \le j \le k$. Notice that, since G is public to both $\mathcal{P}$ and $\mathcal{C}$, both parties can analyze the structures of G to determine the best segmentation strategy. Overall, let α, β be some real numbers, given the set of actuation parameters Γ for each of functions $G_i \in G$, the segmentation of is based on the following three criteria:

- Case 1: If G_i contains a region where the function outputs remain constant, i.e., $G_{i,j}(x) = \alpha$ for $x \in [\gamma_{i,j-1}, \gamma_{i,j})$, we cut the region out as an independent sub-function. Since the output is α regardless of the input, we let $\mathcal{P}$ generate the ciphertext $\mathsf{RLWE}(\widetilde{\alpha})$ during the protocol setup step in Figure 2, where $\widetilde{\alpha}$ is a polynomial with only constant term α. $\mathcal{P}$ then transfer $\mathsf{RLWE}(\widetilde{\alpha})$ in combined with the interval parameters $\gamma_{i,j}$'s as a part of the actuation parameter Γ to the controller $\mathcal{C}$. Whenever $\mathcal{C}$ needs to calculate this sub-function, $\mathcal{C}$ directly returns $\mathsf{RLWE}(\widetilde{\alpha})$.
- Case 2: If a region in G_i is linear, i.e., $G_{i,j}(x) = \beta(x-\alpha)$ for $x \in [\gamma_{i,j-1}, \gamma_{i,j})$. For linear segments, in addition to the interval parameters $\gamma_{i,j}$'s, $\mathcal{P}$ sends $\mathsf{RGSW}(\widetilde{\beta})$ and $\mathsf{RLWE}(\widetilde{\alpha})$ to $\mathcal{S}$ during protocol setup. When evaluating such piece of sub-function, $\mathcal{C}$ calculates $\mathsf{RGSW}(\widetilde{\beta}) \times (\mathsf{RLWE}(\widetilde{x}) - \mathsf{RLWE}(\widetilde{\alpha})) = \mathsf{RLWE}(\beta(x-\alpha))$.
- Case 3: If $G_{i,j}$ is neither constant nor linear function, we consider the segment to be a general non-linear function in the range $[\gamma_{i,j-1}, \gamma_{i,j})$. In such case, $\mathcal{P}$ sends an encoded LUT polynomial $\widetilde{tab}_{i,j}$ along with the interval parameters $\gamma_{i,j}$'s to the cloud controller in the protocol setup step. This evaluation of such sub-function $\mathsf{RLWE}(G_{i,j}(x)) = \mathsf{RLWEToLWE}(\texttt{HomLUT}(\mathsf{LWE}(x), \widetilde{tab}_{i,j}))$. As discussed in Section III-A, due to the inherent accuracy loss in the $\texttt{HomLUT}$ operator, the region $[\gamma_{i,j-1}, \gamma_{i,j})$ cannot be too wide, or the evaluation results become indecipherable.

Building upon the above classification, we illustrate the proposed method of the $\texttt{CMUX}$-tree-based piecewise non-linear actuation function evaluation in Figure 3, where the concrete

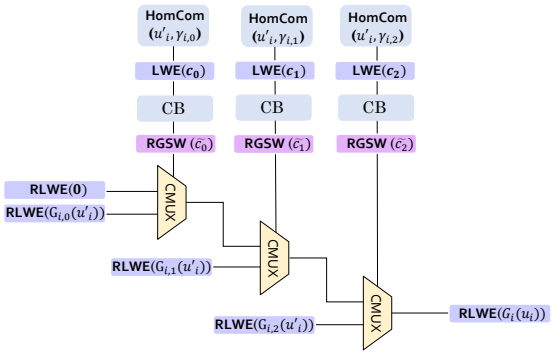


Figure 3. The method of evaluating the piecewise non-linear actuation function.

Algorithm 2: Piecewise Function Evaluation

Input : An LWE ciphertext $c = \mathsf{LWE}(u')$
Input : Actuation parameters $\boldsymbol{\gamma}_i$ for function $G_{\boldsymbol{\gamma}_i}$
Output: An LWE ciphertext $lct = \mathsf{LWE}(u)$ where $u = G_{\boldsymbol{\gamma}_i}(u')$

1 $rct_0 \leftarrow \mathsf{RLWE}(\widetilde{0})$
2 **for** $j = 1$ *to* k **do**
3 **if** $G_{i,j}$ belongs to Case 1 **then**
4 $cf_j \leftarrow \mathsf{RLWE}(\widetilde{\alpha})$ from $\boldsymbol{\gamma}_i$
5 **else if** $G_{i,j}$ belongs to Case 2 **then**
6 $cf_j \leftarrow \mathsf{RGSW}(\widetilde{\beta}) \times (c - \mathsf{RLWE}(\widetilde{\alpha}))$ from $\boldsymbol{\gamma}_i$
7 **else**
8 $cf_j \leftarrow \mathsf{LWEToRLWE}(\mathtt{HomLUT}(c, \widetilde{tab}_{i,j}))$ from $\boldsymbol{\gamma}_i$
9 $cs_j = \mathtt{CB}(\mathsf{LWE}(x < \gamma_{i,j})) \leftarrow \mathtt{HomComp}(c, \gamma_{i,j})$
10 $rct_j \leftarrow \mathtt{CMUX}(cs_j, rct_{j-1}, cf_j)$
11 $lct \leftarrow \mathsf{RLWEToLWE}(rct_k, 0)$
12 **return** lct

procedures are detailed in Algorithm 2. First, the outer loop on Line 2 of Algorithm 2 traverses each of the function segments in G. For the j-th function segment, we compute the homomorphic evaluation result cf_j based on the function type of $G_{i,j}$ on Line 3–8. Next, we determine if the input $c = \mathsf{LWE}(u'_i)$ is in the range of $[\gamma_{i,j-1}, \gamma_{i,j})$ by calculating the boundary condition $cs_j = \mathsf{LWE}(u'_i < \gamma_{i,j})$ through the application of the homomorphic comparison operator $\mathtt{HomComp}$. If the condition is not satisfied (i.e., $cs_j = \mathsf{RGSW}(\widetilde{0})$), subsequent computations are basically ignored by the $\mathtt{CMUX}$ operator on Line 10, which outputs the unmodified rct_j ciphertext generated in the previous iteration (or the initial ciphertext specified on Line 1 in Algorithm 2). In contrast, when the condition holds true, the $\mathtt{CMUX}$ on Line 10 homomorphically selects cf_j to be the evaluation result of the function on the interval $[\gamma_{i,j-1}, \gamma_{i,j})$. Finally, on Line 11, we apply $\mathsf{RLWEToLWE}$ to rct to homomorphically extract the function evaluation result $lct = \mathsf{LWE}(u)$, where u is the final system input to the plant.

Here, we take the saturation function as an example to demonstrate the process of non-linear function segmentation. Let G_i be the saturation function given by:

$$\mathrm{sat}_{\boldsymbol{\gamma}_i}(x) = \begin{cases} \gamma_{i,1}, & \gamma_{i,0} \le x < \gamma_{i,1} \\ x, & \gamma_{i,1} \le x < \gamma_{i,2} \\ \gamma_{i,2}, & \gamma_{i,2} \le x < \gamma_{i,3} \end{cases} \tag{12}$$

From Equation (12), we can see that a saturation function (which is apparently non-linear) can actually be implemented using three pieces of constant and linear functions. Hence, using Algorithm 2, the evaluation of Equation (12) falls into Case 1 and 2 on Line 4 and 6, respectively, where we produce $cf_0 = \mathsf{RLWE}(\widetilde{\gamma}_{i,1})$, $cf_1 = \mathsf{RLWE}(\widetilde{x})$, and $cf_2 = \mathsf{RLWE}(\widetilde{\gamma}_{i,2})$. Lastly, rct will be set to one of the cf_i's depending on the exact value of u' through the comparisons $u' < -\gamma_{i,1}$, $u' < \gamma_{i,2}$ and $u' \ge \gamma_{i,2}$. As a result, we can evaluate complex non-linear functions in a piecewise manner without using high-degree approximation functions [58] or rely on the low-precision $\mathtt{HomLUT}$ operator [40].

Complexity Analysis: By segmenting the piecewise non-linear function into k sub-functions, the evaluation process requires require k times $\mathtt{HomComp}$, k times $\mathtt{CB}$, k times $\mathtt{CMUX}$, and k times sub-function evaluations. Note that the heaviest operation among them is $\mathtt{CB}$, Algorithm 2 achieves a complexity reduction of nearly $O(k)$ $\mathtt{CB}$ operations. In terms of accuracy, the precision of this algorithm depends on the accuracy of each sub-function evaluation. Given that most non-linear actuation functions can be segmented into linear functions, Algorithm 2 is able to maintain accuracy during evaluation, with the only error arising from rounding during encoding, as will be further discussed in Section V.

V. Noise Analysis for FHECAP

To closely inspect the noise characteristics of FHECAP, we first introduce the proposed plaintext space quantization scheme in Section V-A, and then describe the concrete noise characterization steps in Section V-B.

A. Plaintext Space Quantization

Due to the inherent structure of the FHECAP protocol, it is much more important to properly handle the plaintext space of the LWE ciphertext than the RLWE ciphertext, since the final decryption acts over a set of small-parameter ciphertexts $\{\mathsf{LWE}^q_{\boldsymbol{s}}(u_i)\}_l$. Hence, we mainly focus on developing a concrete quantization scheme for the LWE ciphertexts. Let the plaintext space of the LWE ciphertext be ℓ_p-bit, i.e., an LWE ciphertext can only encrypt integers of at most ℓ_p bits, given a real-valued control input $u \in \mathbb{R}$, we embed u into a ℓ_p-bit signed integer as follows:

$$\mathtt{Quant} : \mathbb{R} \to \mathbb{Q}(\ell_p, \ell_d) \to \mathbb{Z}_{2^{\ell_p - 1}}, \tag{13}$$

where

$$\mathbb{Q}(\ell_p, \ell_d) = \{y | y = -\mathfrak{b}_{\ell_p} 2^{\ell_p - \ell_d - 1} + \sum_{i=0}^{\ell_p - 1} 2^{i - \ell_d} \mathfrak{b}_i, \\ \mathfrak{b}_i \in \{0, 1\}, 0 \le i < \ell_p\}. \tag{14}$$

Essentially, Equation (14) converts a real number into an integer of ℓ_p bits in length via the rational number encoding, where the most significant bit $\mathfrak{b}_{\ell_p - 1}$ represents the sign bit, the middle $\ell_p - \ell_d - 1$ bits are used for the integer part, and the last ℓ_d bits are for the fractions. Hence, the range of the system input based on the proposed mapping method of Equation (13) is the interval $[-2^{\ell_p - \ell_d - 1}, 2^{\ell_p - \ell_d - 1}]$.

Let $\upsilon = \mathtt{Quant}(u) = \delta\lambda$ where u is the system input, $\upsilon \in \mathbb{Z}_{2^{\ell_p - 1}}$ is the encoded integer, $\lambda \in \mathbb{Q}(\ell_p, \ell_d)$ is some rational number, and $\delta = 2^{\ell_d} \in \mathbb{Z}$ is the quantizer. (i.e., a special scaling factor for the system input) To properly quantize the input u, we need to derive a set of inequalities to constrain δ from the perspectives of both the valuation range and the noise characteristics. For the valuation range, notice that

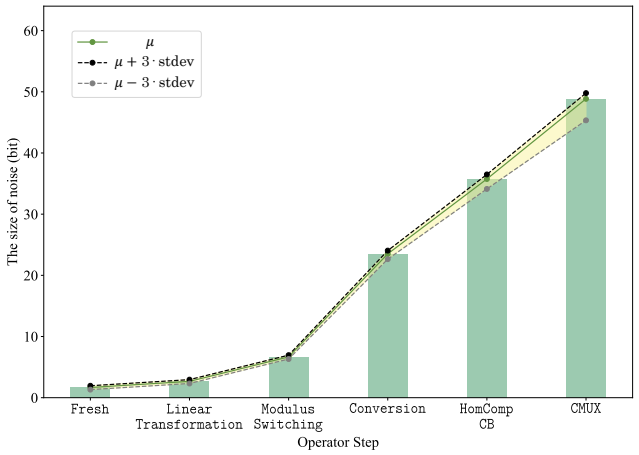


Figure 4. Empirical noise growth in one particular instance of the FHECAP protocol round under the given encryption parameters in Table III, as observed through Monte Carlo simulations.

$$\ell_p - \ell_d - 1 \geq \log_2 u_m, \tag{15}$$

i.e., the range of the encoded integer has to be larger than the maximum possible value u can take. Meanwhile, for noise characteristics, we point out that the truncation error generated during the encoding process in Equation (13) is the main source of noises in our encrypted control system, for that the truncation error can be directly observed after decrypting the control values. To formalize the error generated during data encoding, $|u|$ makes all the valid digits of the ℓ_d-bit after the decimal point of u erased, which is at most $2^{-\ell_d}$. Combining with Assumption 2 , the system input error can be written as $\eta(\epsilon) = |u - \lambda| < 2^{-\ell_d}$. Subsequently, we have that $\delta < \frac{1}{\eta(\epsilon)}$. Through the above equations, we can decide the most suitable δ under a given plaintext space size, u_m and $\eta(\epsilon)$ for the target control system.

B. Noise Analysis

Empirical Noise Analysis: Before devising the formal expressions, we first give an empirical study on the ciphertext noises characteristics of FHECAP. We analyze the sizes of noises for the ciphertext results after the application from step ③ to step ⑤ in Figure 2. Within the steps, there exist six intermediate ciphertexts, namely, the fresh RLWE input, the RLWE ciphertext after linear transformation, the LWE ciphertext after `ModulusSwitching`, the LWE after conversion, the LWE ciphertext after `HomComp` / RGSW after `CB`, and the LWE ciphertext after the final `CMUX`. To intuitively show the growth of noise, we perform 10,000 Monte Carlo tests on the six intermediate ciphertexts noise. Figure 4 shows the mean (μ) and three standard deviation range ($[\mu - 3\cdot\texttt{stdev}, \mu + 3\cdot\texttt{stdev}]$) of the L2 norm of the noises in such six kinds of ciphertexts. As we can see from Figure 4, after the homomorphic scheme switches from CKKS to TFHE, the noise growth of the ciphertext increases significantly.

Formal Noise Analysis: We use the tail-probability-based noise analysis tools developed in [56], [59], [60] to rigorously study the noise characteristics of HE ciphertexts in FHECAP. Our goal is to derive a theoretical bound that closely matches the real level of noise in the target ciphertexts. Let $\mathsf{LWE}(m) = (\boldsymbol{a}, b)$, $\mathsf{RLWE}(\widetilde{m}) = (\widetilde{a}, \widetilde{b})$, $\mathsf{RGSW}(\widetilde{m}) = (\mathsf{RGSW}_0 \cdots \mathsf{RGSW}_{2l-1})^T$, we use $e_{\mathsf{LWE}} = b - \langle \boldsymbol{a}, \boldsymbol{s} \rangle - \Delta \cdot m$, $\widetilde{e}_{\mathsf{RLWE}} = \widetilde{b} - \widetilde{a} \cdot \widetilde{s} - \Delta \cdot \widetilde{m}$, and $\widetilde{\boldsymbol{e}}_{\mathsf{RGSW}} = (\widetilde{e}_{\mathsf{RGSW}_0} \cdots \widetilde{e}_{\mathsf{RGSW}_{2d-1}})^T$ to denote the noises in the respective ciphertexts. When the cloud controller $\mathcal{C}$ initially receives the ciphertexts from the plant $\mathcal{P}$, we assume all ciphertexts produced by $\mathcal{P}$ are freshly encrypted, i.e., the ciphertexts only contain the initial encryption noise $e_{\mathsf{LWE,fresh}}$, $\widetilde{e}_{\mathsf{RLWE,fresh}}$ or $\widetilde{\boldsymbol{e}}_{\mathsf{RGSW,fresh}}$ depending on the ciphertext types.

Step ③ Linear Transformation: Evaluating the linear control law over HE ciphertexts involves a single application of ciphertext multiplication between the encrypted system state $\mathsf{RLWE}(\widetilde{x})$ and the transformation matrix K, followed by a `Rescaling` operator. Because K is encoded to $\texttt{CoefEcd}(\boldsymbol{\kappa})$ and then multiplied, the noise after multiplication is $\|\texttt{CoefEcd}(\boldsymbol{\kappa})\|_1 \widetilde{e}_{\text{fresh}}$. After the modulus switched in `Rescaling`, the noise grows to $\frac{Q}{Q'}\|\texttt{CoefEcd}(\boldsymbol{\kappa})\|_1 \widetilde{e}_{\text{fresh}} + \widetilde{e}_{\texttt{round}}$ where $e_{\texttt{round}}$ is the noise caused by rounding. Due to the parameters chosen for encoding , $\frac{Q}{Q'}\|\texttt{CoefEcd}(\boldsymbol{\kappa})\|_1 \approx \|K\|_1$, while the magnitude of $\widetilde{e}_{\text{fresh}}$ is independent of Q, Q' and can be ignored. (for details, see [29]) To sum up, the resulting noise grows from $\widetilde{e}_{\text{fresh}}$ to $\widetilde{e}_{\texttt{mult}} = \|K\|_1 \cdot \widetilde{e}_{\text{fresh}}$.

Step ④ Ciphertext Conversion: In this step, the modulus of the ciphertext switches from Q' to q, and the lattice dimension switches from N to n. The starting point in this step is $e_{\texttt{mult}}$. Similar `Rescaling`, `ModulusSwitching` switch the modulus from $c_0 = \mathsf{LWE}_{\boldsymbol{s}}^{n,Q'}(m) = (\boldsymbol{a}, b)$ to $c_1 = \mathsf{LWE}_{\boldsymbol{s}}^{n,q}(m) = (\lfloor \frac{q}{Q'} \rceil \boldsymbol{a}, \lfloor \frac{q}{Q'} \rceil b)$, inducing a multiplicative noise amplification factor of $\frac{q}{Q'}$ and an additive rounding noise $e_{\texttt{round},2}$. Therefore, we have that $e_{\texttt{MS}} = \frac{q}{Q'} e_{\texttt{mult}} + e_{\texttt{round},2}$ where q and Q' are the moduli of c_0 and c_1, respectively. Subsequently, the end-to-end noise growth for the RLWEToLWE can be formalized as

$$e_{\mathsf{LWE}_{\boldsymbol{s}}^{n,q}(m)} = e_{\texttt{MS}} - \sum_{i=0}^{N-1} (\widehat{a}_i - a_i) - \sum_{i=0}^{N-1} \sum_{j=0}^{d'-1} \widehat{a}_{i,j} e_{\mathsf{KSK}_{i,j}}, \tag{16}$$

where $\widehat{a}_{i,j} \in [-\frac{\mathsf{Bg}'}{2}, \frac{\mathsf{Bg}'}{2}) \cap \mathbb{Z}$. More specifically, $\{\widehat{a}_{i,j}\}$ $(0 \leq j < d')$ is the unique decomposition set of a_i based on Bg', satisfying $a_i \approx \sum_{j=0}^{d'-1} \widehat{a}_{i,j} \cdot \frac{q}{\mathsf{Bg}^p} = \widehat{a}_i$, where we have that $|\widehat{a}_i - a_i| \leq \frac{q}{2 \cdot \mathsf{Bg}'^{d'}}$. Here, we say that $\widehat{a}_i$ is the approximate reconstruction of a_i.

Step ⑤ Piecewise Actuation: As mentioned in Section V-B, applying a set of piecewise non-linear functions over ciphertexts require the application of three main homomorphic operators, namely, (i) `HomComp`, (ii) `CB`, and (iii) `CMUX`. Here, the main source of noise here comes from the final `CMUX` operator, whose noise characteristics are analyzed as follows. First, we point out that the selection signal into the `CMUX` operator here is not a fresh RGSW ciphertext, but one generated by the `CB` operator. Hence, we have that

$$\begin{aligned} \|\widetilde{e}_{\texttt{CMUX}}\|_\infty = & \|\widetilde{e}_{ct_3 \times (ct_1 - ct_2) + ct_2}\|_\infty \leq \\ & \|\widetilde{e}_{ct_3 \times (ct_1 - ct_2)}\|_\infty + \max(\|\widetilde{e}_{ct_1}\|_\infty, \|\widetilde{e}_{ct_2}\|_\infty). \end{aligned} \tag{17}$$

where ct_1 and ct_2 correspond to the two choices ciphertexts rct_{j-1} and cf_j belongs to three criteria from Section IV-B3. on Line 10 in Algorithm 2, respectively, and ct_3 is the cs_j selection signal. After the k iterations depicted on Line 11 in Algorithm 2, the final noise contained in lct is roughly $\sqrt{k} \cdot \|\widetilde{e}_{\texttt{CMUX}}\|_\infty$ (RLWEToLWE operation here does not generate any noise [40]). Due to space limitations, the exact sizes of

the noises in ct_1, ct_2, and ct_3 are further formalized in the supplemental document.

Combining all the noise analysis explained above, we formalize the overall noise bound for FHECAP noise in Theorem V.1.

Theorem V.1. ***End-to-end protocol noise bound*** *Let Ψ_a be the distribution of discrete variables $\mathscr{Y} = \mathscr{X}^2$ where $\mathscr{X} \sim U[-\frac{a}{2}, \frac{a}{2}) \cap \mathbb{Z}$. Let q, Q', Q be the ciphertext moduli, n, N the lattice dimensions, $\mathsf{Bg}, d, \mathsf{Bg}', d'$ the decomposition parameters, k the number of segments in a piecewise function. Let $\|K\|_1$ represent the L1-norm of the gain matrix, and $\sigma_{\text{fresh}}, \sigma_{\mathsf{BK}}, \sigma_{\mathsf{KSK}}$ be the variance of the initial ciphertext noise. Given the failure probability ξ, we have the following bound on the tail probability for the noise in $\mathsf{LWE}(u_i)$ $(0 \le i < l)$*

$$\Pr[e_{\mathsf{LWE}(u_i)} > \mathfrak{e}_{\mathtt{Sub}} + \mathtt{CCB}_{\Phi_{\mathtt{CMUX}},k}(\xi)] \le \xi. \tag{18}$$

where

$$\mathfrak{e}_{\mathtt{Sub}} = \lceil \frac{q}{Q'} \rceil \|K\|_1 z_{1-\xi} \cdot \sigma_{\text{fresh}} + \mathfrak{e}_{\mathtt{KS},0} + \mathfrak{e}_{\mathtt{KS},1} + \mathtt{CCB}_{\Phi_{\mathtt{EP}},\underline{n}}(\xi),$$

$$\mathfrak{e}_{\mathtt{KS},0} = \mathtt{SDU}^{-1}_{-\frac{q}{2\mathsf{Bg}'^{d'}}, \frac{q}{2\mathsf{Bg}'^{d'}}, N}(\xi),$$

$$\mathfrak{e}_{\mathtt{KS},1} = \sigma_{\mathsf{KSK}} \sqrt{-2 \ln \xi \cdot \mathtt{CCB}_{\Psi_{\mathsf{Bg}'}, N \cdot d'}(\xi)}. \tag{19}$$

Here, $\Phi_{\mathtt{CMUX}}$ can be expressed as $\sum_{j=0}^{\frac{3}{2}nd-d-1} \mathscr{X}_j \cdot \mathscr{Y}_j + \sum_{j=0}^{\frac{1}{2}nd+d-1} \mathscr{X}_j \cdot \mathscr{Z}_j + \mathscr{W}$ where $\mathscr{X}_j \sim U[-\frac{\mathsf{Bg}}{2}, \frac{\mathsf{Bg}}{2}) \cap \mathbb{Z}, \mathscr{Y}_j \sim \Phi_{\mathtt{CB},0}, \mathscr{Z}_j \sim \Phi_{\mathtt{CB},1}, \mathscr{W} \sim \Phi_{\mathtt{EP},0}$. $\Phi_{\mathtt{CB},0}$ can be expressed as $\sum_{j=0}^{(n+1)d'-1} \mathscr{X}_j \cdot \mathscr{Y}_j$ where $\mathscr{X}_j \sim U[-\frac{Bg'}{2}, \frac{\mathsf{Bg}'}{2}) \cap \mathbb{Z}, \mathscr{Y}_j \sim \chi_{\sigma_{\mathsf{KSK}}}$. $\Phi_{\mathtt{CB},1}$ can be expressed as $\mathscr{X} + \sum_{j=0}^{n} \mathscr{Y}_j + \sum_{j=0}^{n-1} \mathscr{Z}_j$ where $\mathscr{X} \sim \Phi_{\mathtt{CB},0}, \mathscr{Y}_i \sim U[-\frac{q}{2\mathsf{Bg}'^{d'}}, \frac{q}{2\mathsf{Bg}'^{d'}}) \cap \mathbb{Z}, \mathscr{Z}_j \sim \Phi_{\mathtt{EP}}$. $\Phi_{\mathtt{EP}}$ can be expressed as $\sum_{j=0}^{2nd-1} \mathscr{X}_j \cdot \mathscr{Y}_j + \mathscr{Z}$ where $\mathscr{X}_j \sim U[-\frac{\mathsf{Bg}}{2}, \frac{\mathsf{Bg}}{2}) \cap \mathbb{Z}$, $\mathscr{Y}_j \sim \chi_{\sigma_{\mathsf{BK}}}$, $\mathscr{Z} \sim \Phi_{\mathtt{EP},0}$. $\Phi_{\mathtt{EP},0}$ can be expressed as $\sum_{j=0}^{\frac{n}{2}} \mathscr{X}_j + \sum_{j=0}^{\frac{n}{2}-1} \mathscr{Y}_j$ where $\mathscr{X}_j \sim U[-\frac{q}{2\mathsf{Bg}^d}, \frac{q}{2\mathsf{Bg}^d}) \cap \mathbb{Z}$, $\mathscr{Y}_j \sim U[-\mathfrak{e}_{\mathtt{FFT}}, \mathfrak{e}_{\mathtt{FFT}}) \cap \mathbb{Z}$ where $\mathfrak{e}_{\mathtt{FFT}}$ is the bound on the noises caused by the FFT *operations. $z_{1-\xi}$ is the quantile on χ_1 where the CDF equals $1-\xi$. $\mathtt{CCB}_{\Phi,\mathsf{n}}$ is the bound of the sum of* n *independent random variables from Φ distribution. $\mathtt{SDU}^{-1}_{\mathsf{a},\mathsf{b},\mathsf{c}}$ is the inverse of the cumulative distribution function (CDF) for the sum of* c *discrete uniform distributions over $[\mathsf{a}, \mathsf{b}) \cap \mathbb{Z}$ where $\mathsf{a}, \mathsf{b}, \mathsf{c} \in \mathbb{Z}$.*

Due to space limitation, the full proof for Theorem V.1. can be found in the supplemental document.

VI. Experiment

In the experiment, we first compare the performance of FHECAP on a set of micro-benchmarks. Then, we apply FHECAP to an end-to-end spacecraft rendezvous system to study the latency and stability impacts of encrypted control over real-world applications.

A. Experiment Setup

The entire FHECAP framework is implemented in C++17 and compiled using g++-10. We develop FHECAP using the low-level HE operators provided in [61], [62] and [63]. The performance figures are recorded on a single thread of the Intel Xeon Gold 6226R processor with 503 GB of RAM.

In Table III, we summarize the instantiated system parameters for FHECAP and comparative microbenchmarks. The parameters are selected to meet three goals: 128-bit security under lwe-estimator [64], stable system states with correct control values, and small parameter sizes to enhance system performance. The details on specific parameter selections can be found on Section B. In particular, to verify the correctness of the parameters listed in Table III, we employ 10K Monte-Carlo simulations to simulate the noise of the final LWE ciphertext and compare it with the theoretical noise estimated in Theorem V.1. As shown in the probability density plot in Section VI-A, we observe two key facts. First, the theoretical noise distribution closely aligns with the simulated noise. However, discrepancies arise due to two primary factors: (1) the inherent looseness of the bounds derived from Lemma II.1 (Supplementary Material), which provides conservative estimates to ensure generality, and (2) the numerical optimization errors introduced when applying the Chernoff-Cramér inequality [65], particularly during operations such as argmin search. Second, the bound we set differ insignificantly from the simulation results, corresponding to a $\xi = 2^{-30}$ failure probability. In Section VI-C, we show that such ξ is adequate for ensuring the practical system stability of practical encrypted control systems.

TABLE III
Parameter Instantiation for FHECAP and Comparative Microbenchmarks

Design	Ciphertext	Parameters
Paillier-based	-	$\log_2 q = \log_2 p = 3\,072$
Leveled HE	RLWE	$N = 4\,096$, $\lfloor \log_2 Q \rfloor = 111$
PA-based	RLWE	$N = 16\,384$, $\lfloor \log_2 Q \rfloor = 440$
LUT-based	LWE	$n = 2\,048$, $q = 2^{64}$
FHECAP (ours)	RLWE	$N = 4\,096$, $\lfloor \log_2 Q \rfloor = 111$
	LWE	$n = 2\,048$, $q = 2^{64}$

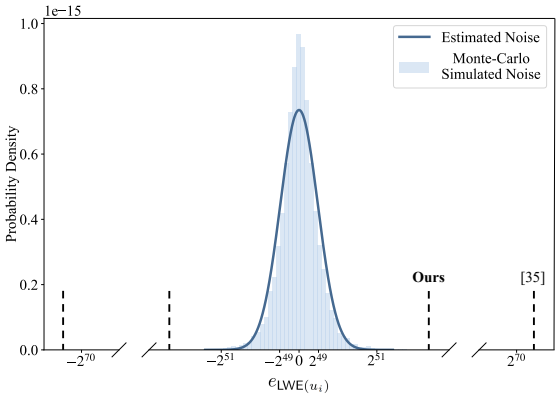


Figure 5. The probability distributions of our theoretical noise and the simulated noise via 10 000 Monte-Carlo tests. Our noise bound based on ξ is about 6 times the standard deviation (σ) of the theoretical noise, while the noise bound of [40] is about 500 000× larger than ours.

B. Microbenchmarks

Here, we test FHECAP on a set of microbenchmarks over both linear and non-linear control sub-tasks.

Linear Transformation: The most common operation in linear control systems is matrix-vector multiplication of the linear control law matrix K and the system state $\boldsymbol{x}$. As shown in Figure 6, the encrypted controller based on FHECAP is 4–1000× faster than that based on the Paillier-based techniques [13], and up to 4× faster than the existing leveled

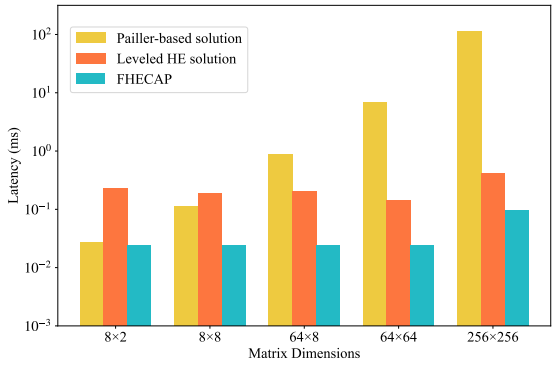


Figure 6. The micro-benchmarks of the linear transform evaluation. We select [24] as the referenced design for leveled-HE-based linear control systems as all of these systems rely on the multiply-and-rotate approach for encrypted linear transformations.

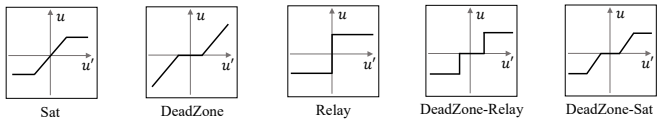


Figure 7. Different types of control-orient non-linear actuation functions.

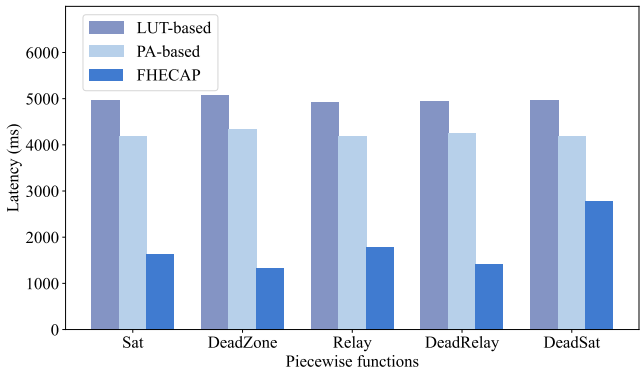


Figure 8. The micro-benchmarks of the piecewise non-linear evaluation.

HE solution [19], [21]–[24]. Notably, owing to the batching capability of RLWE, the performance of FHECAP and [24] are independent of the dimension of a matrix (up to the lattice dimension n), while the runtime of [13] is proportional to the dimension of the linear transformation matrix. In addition, since FHECAP adopts the coefficient encoding technique [56], [57], the latency of linear transformation of FHECAP is significantly reduced when compared to the multiply-and-rotate approach in [24].

Piecewise Non-linear Function Evaluation: Because most of the existing encrypted controllers do not support non-linear operations, we compare the performance of FHECAP to related literature on HE operator designs that can handle general non-linear operations, namely, the PA-based method [66], and the LUT-based method [40]. As illustrated in Figure 7, the experiments are carried out using different kinds of non-linear functions that widely exist in types of actuators, including the saturation function, the dead zone function, the relay function, the relay function nested with dead zones, and the saturation function nested with dead zones. In Figure 8, we show that the latency of FHECAP in homomorphically evaluating non-linear functions can be reduced by at least 1.4× when compared to the PA-based methods and by at least 1.8× when compared to the LUT-based methods.

Remark: In addition to the performance gap, the PA-based and LUT-based solutions exhibit two major drawbacks when evaluating piecewise non-linear functions:

(a) The PA-based solution requires significantly larger parameters for N and Q in RLWE. This is because evaluating the approximation polynomial necessitates a larger modulus Q and ring dimension N. However, larger N and Q lead to a substantial reduction in the efficiency of the linear transformation.

(b) Both the PA-based and LUT-based solutions can only compute Γ when Γ is public. When Γ is private, it is impossible to construct the corresponding approximation polynomial or look-up table, making the evaluation of the corresponding non-linear operations infeasible.

C. End-to-End Application

To put FHECAP under the test of real-world control tasks, we consider the scenario of spacecraft navigation and rendezvous [67]. Assume that, we have two spacecrafts, $\mathcal{A}$ and $\mathcal{B}$. Initially, $\mathcal{A}$ is situated in its own orbit. Then, due to the need of a collaborative mission, it needs to change its orbit to rendezvous and dock with $\mathcal{B}$. Thus, $\mathcal{B}$ will navigate and control $\mathcal{A}$. Clearly, spacecraft $\mathcal{A}$ corresponds to the plant $\mathcal{P}$ while spacecraft $\mathcal{B}$ plays the role of the controller $\mathcal{C}$ under the FHECAP context. The spacecraft rendezvous system is characterized by the following equations:

$\dot{\boldsymbol{x}}(t) = A \cdot \boldsymbol{x}(t) + B \cdot \boldsymbol{u}(t)$, where

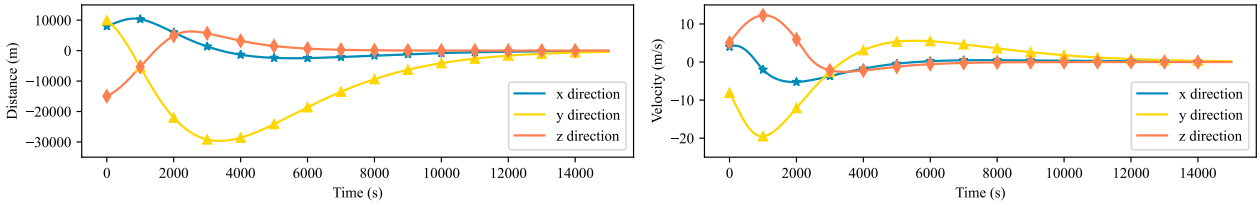
$$A = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 3\omega^2 & 0 & 0 & 0 & 2\omega & 0 \\ 0 & 0 & 0 & -2\omega & 0 & 0 \\ 0 & 0 & -\omega^2 & 0 & 0 & 0 \end{bmatrix}, B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (20)$$

Here, the system state $\boldsymbol{x}$ is a six-element vector containing the distance and velocity in the x, y, z axis directions. Whereas, the control value $\boldsymbol{u}$ is a vector of length three containing the acceleration values in the same three directions, and ω is the orbital angular velocity of $\mathcal{B}$. Limited by the engine specification of $\mathcal{A}$, the rendezvous system is a typical control system with non-linear actuation (i.e., saturation on the control input $\boldsymbol{u}$). Substituting the orbital parameters, the overall control system is characterized as $\boldsymbol{u}(t) = \mathrm{sat}(K \cdot \boldsymbol{x}(t))$, where

$$K = \begin{bmatrix} \mathsf{k}_{11} & -\frac{\omega^2 \mathsf{f}_1}{2} & 0 & -\omega(\mathsf{f}_1 + \mathsf{f}_3) & \mathsf{k}_{15} & 0 \\ -2\omega^2 \mathsf{f}_2 & 0 & 0 & 0 & -\omega \mathsf{f}_2 & 0 \\ 0 & 0 & 0 & 0 & 0 & -2\omega \end{bmatrix}. \quad (21)$$

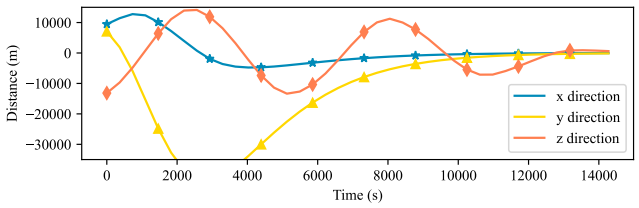
Here, we have that $\mathsf{k}_{11} = \frac{\omega^2(3\mathsf{f}_1 f_2^2 + 3\mathsf{f}_1 + 4\mathsf{f}_2^2 \mathsf{f}_3)}{f_2(f_2^2+1)}$ and $\mathsf{k}_{15} = -\frac{\omega(3\mathsf{f}_1 f_2^2 + 3\mathsf{f}_1 + 4\mathsf{f}_2^2 \mathsf{f}_3)}{2\mathsf{f}_2(f_2^2+1)}$, where $\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3$ are positive constants characterizing the target control system.

To simulate the behaviors of the above control system, we set $\omega = 1.1068 \times 10^{-3}$, $\mathsf{f}_1 = \frac{\sqrt{3}}{9}$, $\mathsf{f}_2 = 1$, $\mathsf{f}_3 = \frac{8\sqrt{3}}{9}$, $\boldsymbol{x}_0$=[8 000, 10 000, -15 000, 4, -8, 5], Γ=[8×10^{-3}, 8×10^{-3}, 6×10^{-3}] according to [67]. Figure 9 records the system state trajectory of the closed-loop system based on FHECAP under two different interaction intervals dt, namely, dt=5 s, and 366 s. The results in Figure 9 show that using this bounded linear feedback method and our cryptographic construction of FHECAP, the tracking spacecraft $\mathcal{A}$ and the target spacecraft $\mathcal{B}$

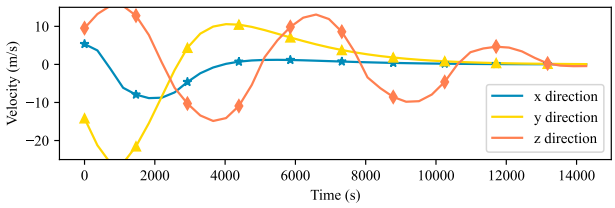


(a) Distance value when $dt = 5s$

(b) Velocity value when $dt = 5s$



(c) Distance value when $dt = 366s$



(d) Velocity value when $dt = 366s$

Figure 9. Control trajectories of $\mathcal{A}$ under different interaction intervals.

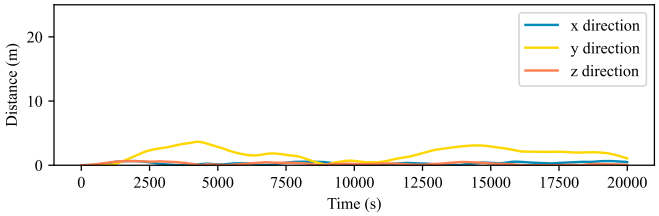


Figure 10. Noises observed in the decrypted system states at each control step under $dt = 5s$.

can successfully meet, without $\mathcal{A}$ and $\mathcal{B}$ knowing the secret information of each other. Taking a closer look, we draw the difference between the system states in the plaintext control system and the encrypted control system in Figure 10. We can see that the error caused by employing an encrypted control system is extremely small (the distance between spacecrafts can easily range to 10^4 m). Furthermore, due to the self-correcting property of the closed-loop system, the error also does not accumulate over time, resulting in a correct and stable system behavior. However, comparing Figure 9(a) and (c), we see that the interaction interval plays a critical role in the self-correcting process of the system, where a longer interaction time worsens the capability of the system to correct itself from erroneous states.

VII. Conclusion

In this work, we propose FHECAP, an encrypted control protocol for closed-loop linear systems with non-linear actuation functions. We develop new techniques for applying both linear and non-linear functions over quantized control values with high accuracy and efficiency. In the experiments, we show that FHECAP can be $4\times$–$1000\times$ faster than existing solutions on linear-control systems, while being able to handle high-precision non-linear actuation functions in real-time systems with stable control outputs.

References

[1] L. Shumei and F. Wei, "Two close encounters of starlink satellites possibly aimed to test china's sensibility in space: expert," globaltimes, December 27 2021. [Online]. Available: https://www.globaltimes.cn/page/202112/1243527.shtml

[2] J. Zhang, Y. Cai, C. Xue, Z. Xue, and H. Cai, "Leo mega constellations: Review of development, impact, surveillance, and governance," *Space Sci. Technol.*, 2022.

[3] J. Bai, X. Zhang, L. Qi, W. Liu, X. Zhou, Y. Liu, X. Lv, B. Sun, B. Duan, S. Zhang, and X. Che, "Survey on application of trusted computing in industrial control systems," *Electronics*, vol. 12, no. 19, p. 4182, 2023.

[4] A. Ayodeji, M. Mohamed, L. Li, A. Di Buono, I. Pierce, and H. Ahmed, "Cyber security in the nuclear industry: A closer look at digital control systems, networks and human factors," *Progress in Nuclear Energy*, vol. 161, p. 104738, 2023.

[5] J. Beerman, D. Berent, Z. Falter, and S. Bhunia, "A review of colonial pipeline ransomware attack," in *CCGRID*, 2023, pp. 8–15.

[6] S. C. Patel, G. D. Bhatt, and J. H. Graham, "Improving the cyber security of scada communication networks," *Commun. ACM*, vol. 52, pp. 139–142, 2009.

[7] D. Ding, Q.-L. Han, X. Ge, and J. Wang, "Secure state estimation and control of cyber-physical systems: A survey," *IEEE Trans. Syst. Man Cybern.: Syst.*, vol. 51, no. 1, pp. 176–190, 2021.

[8] P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes," in *EUROCRYPT*, 1999, pp. 223–238.

[9] A. B. Alexandru, M. Morari, and G. J. Pappas, "Cloud-based mpc with encrypted data," in *IEEE CDC*, 2018, pp. 5014–5019.

[10] M. Kishida, "Encrypted control system with quantizer," in *HSCC*. Association for Computing Machinery, 2019, pp. 274–275.

[11] A. B. Alexandru, K. Gatsis, Y. Shoukry, S. A. Seshia, P. Tabuada, and G. J. Pappas, "Cloud-based quadratic optimization with partially homomorphic encryption," *IEEE Trans. Autom. Control*, vol. 66, no. 5, pp. 2357–2364, 2021.

[12] H. Wang, D. Li, Z. Guan, Y. Liu, and J. Liu, "Consensus control based on privacy-preserving two-party relationship test protocol," *IEEE Control Syst. Lett.*, vol. 7, pp. 2185–2190, 2023.

[13] Q. Hu, Y. Shi, and E. Nekouei, "Secure motion control of micro-spacecraft using semi-homomorphic encryption," *Secur. Saf.*, 2023.

[14] M. Schulze Darup, A. Redder, and D. E. Quevedo, "Encrypted cooperative control based on structured feedback," *IEEE Control Syst. Lett.*, vol. 3, no. 1, pp. 37–42, 2019.

[15] W. Fang, M. Zamani, and Z. Chen, "Secure and privacy preserving consensus for second-order systems based on paillier encryption," *Systems & Control Letters*, vol. 148, p. 104869, 2021.

[16] A. V. Suryavanshi, A. Alnajdi, M. S. Alhajeri, F. Abdullah, and P. D. Christofides, "Encrypted model predictive control of nonlinear systems," in *MED*, 2023, pp. 904–911.

[17] Y. Shi and E. Nekouei, "Secure adaptive control of linear networked systems using paillier encryption," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 71, no. 11, pp. 5271–5284, 2024.

[18] J. Kim, D. Kim, Y. Song, H. Shim, H. Sandberg, and K. H. Johansson, "Comparison of encrypted control approaches and tutorial on dynamic systems using learning with errors-based homomorphic encryption," *Annu. Rev. Control*, vol. 54, pp. 200–218, 2022.

[19] J. Lee, D. Lee, J. Kim, and H.-S. Shim, "Encrypted dynamic control exploiting limited number of multiplications and a method using ring-lwe based cryptosystem," *ArXiv*, vol. abs/2307.03451, 2023.

[20] J. Kim, "Further methods for encrypted linear dynamic controllers utilizing re-encryption," *IEEE Trans. Autom. Control*, pp. 1–6, 2024.

[21] N. Schlüter, J. Kim, and M. S. Darup, "A code-driven tutorial on encrypted control: From pioneering realizations to modern implementations," *ArXiv*, vol. abs/2404.04727, 2024.

[22] J. Kim, H. Shim, and K. Han, *Comprehensive Introduction to Fully Homomorphic Encryption for Dynamic Feedback Controller via LWE-Based Cryptosystem*. Springer Singapore, 2020, pp. 209–230.

[23] J. Kim, H. Shim, and H. Kyoohyung, "Dynamic controller that operates over homomorphically encrypted data for infinite time horizon," *IEEE Trans. Autom. Control*, vol. 68, no. 2, pp. 660–672, 2023.

[24] K. Teranishi, T. Sadamoto, and K. Kogiso, "Input–output history feedback controller for encrypted control with leveled fully homomorphic encryption," *IEEE Trans. Control Netw. Syst.*, 2021.

[25] J. Pan, T. Sui, W. Liu, J. Wang, L. Kong, Y. Zhao, and Z. Wei, "Secure control of linear controllers using fully homomorphic encryption," *Applied Sciences*, vol. 13, no. 24, 2023.

[26] T. Sui, J. Wang, W. Liu, J. Pan, L. Wang, Y. Zhao, and L. Kong, "Optimizing encrypted control algorithms for real-time secure control," *Journal of the Franklin Institute*, vol. 361, no. 5, p. 106677, 2024.

[27] Y. Jang, J. Lee, S. Min, H. Kwak, J. Kim, and Y. Song, "Ring-lwe based encrypted controller with unlimited number of recursive multiplications and effect of error growth," *ArXiv*, vol. abs/2406.14372, 2024.

[28] J. Fan and F. Vercauteren, "Somewhat practical fully homomorphic encryption," *Cryptology ePrint Archive*, p. 144, 2012.

[29] J. H. Cheon, A. Kim, M. Kim, and Y. Song, "Homomorphic encryption for arithmetic of approximate numbers," in *ASIACRYPT*, 2017.

[30] N. P. Smart and F. Vercauteren, "Fully homomorphic simd operations," *DESIGN CODE CRYPTOGR.*, vol. 71, pp. 57–81, 2012.

[31] E. Sontag and H. Sussmann, "Nonlinear output feedback design for linear systems with saturating controls," in *IEEE CDC*, vol. 6, 1990, pp. 3414–3416.

[32] L. Y. Xiong Z, Liu Z and X. J, "An adaptive and bounded controller for formation control of multi-agent systems with communication break," *Appl. Sci.*, vol. 12, no. 11, p. 5602, 2022.

[33] H. Sussmann, E. Sontag, and Y. Yang, "A general result on the stabilization of linear systems using bounded controls," *IEEE Trans. Autom. Control*, vol. 39, no. 12, pp. 2411–2425, 1994.

[34] J. Guerrero-Castellanos, N. Marchand, A. Hably, S. Lesecq, and J. Delamare, "Bounded attitude control of rigid bodies: Real-time experimentation to a quadrotor mini-helicopter," *Control Eng. Pract.*, vol. 19, no. 8, pp. 790–797, 2011.

[35] E. Serpelloni, M. Maggiore, and C. Damaren, "Bang–bang hybrid stabilization of perturbed double-integrators," *Automatica*, vol. 69, pp. 315–323, 2016.

[36] D. Recker, P. Kokotovic, D. Rhode, and J. Winkelman, "Adaptive nonlinear control of systems containing a deadzone," in *IEEE CDC*, 1991, pp. 2111–2115.

[37] H. Li, S. Zhao, W. He, and R. Lu, "Adaptive finite-time tracking control of full state constrained nonlinear systems with dead-zone," *Automatica*, vol. 100, pp. 99–107, 2019.

[38] J. Kreiss, M. Jungers, A. Pierron, G. Millérioux, J. Dupont, and M. Martig, "Control design for linear systems with asymmetric input backlash and dead-zone through lmi conditions," *IEEE Control Syst. Lett.*, pp. 1–1, 2024.

[39] C. Gentry, A. Sahai, and B. Waters, "Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based," *IACR Cryptol. ePrint Arch.*, vol. 2013, p. 340, 2013.

[40] I. Chillotti, N. Gama, M. Georgieva, and M. Izabachène, "Tfhe: fast fully homomorphic encryption over the torus," *J. Cryptology*, vol. 33, no. 1, pp. 34–91, 2020.

[41] L. Ducas and D. Micciancio, "Fhew: bootstrapping homomorphic encryption in less than a second," in *EUROCRYPT*, 2015, pp. 617–640.

[42] Z. Li, D. Micciancio, and Y. Polyakov, "Large-precision homomorphic sign evaluation using fhew/tfhe bootstrapping," in *ASIACRYPT*, 2022.

[43] I. Chillotti, D. Ligier, J.-B. Orfila, and S. Tap, "Improved programmable bootstrapping with larger precision and efficient arithmetic circuits for tfhe," in *ASIACRYPT*, 2021.

[44] A. Guimarães, E. Borin, and D. F. Aranha, "Revisiting the functional bootstrap in tfhe," *IACR Trans. Cryptogr. Hardw. Embed. Syst.*, pp. 229–253, 2021.

[45] R. Mayans, "The chebyshev equioscillation theorem," *Journal of Online Mathematics and Its Applications*, vol. 6, 2006.

[46] A. Tasissa, "Function approximation and the remez algorithm," 01 2021.

[47] J. H. Cheon, D. Kim, and D. Kim, "Efficient homomorphic comparison methods with optimal complexity," in *ASIACRYPT*, 2020, pp. 221–256.

[48] S. Panda, "Polynomial approximation of inverse sqrt function for fhe," in *Cyber Security, Cryptology, and Machine Learning*, 2022, pp. 366–376.

[49] R. Canetti, "Security and composition of multiparty cryptographic protocols," *J. Cryptology*, vol. 13, pp. 143–202, 2000.

[50] R. Lindner and C. Peikert, "Better key sizes (and attacks) for lwe-based encryption," in *Topics in Cryptology–CT-RSA*, 2011, pp. 319–339.

[51] J. Katz and Y. Lindell, *Introduction to Modern Cryptography. 2nd ed.* CRC Press, 2014.

[52] J. H. Cheon, K. Han, A. Kim, M. Kim, and Y. Song, "Bootstrapping for approximate homomorphic encryption," *IACR Cryptol. ePrint Arch.*, vol. 2018, p. 153, 2018.

[53] W. jie Lu, Z. Huang, C. Hong, Y. Ma, and H. Qu, "Pegasus: Bridging polynomial and non-polynomial evaluations in homomorphic encryption," in *S&P*, 2021, pp. 1057–1073.

[54] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, "GAZELLE: A low latency framework for secure neural network inference," in *USENIX Security*, 2018, pp. 1651–1669.

[55] S. Halevi and V. Shoup, "Algorithms in helib," *IACR Cryptol. ePrint Arch.*, vol. 2014, p. 106, 2014.

[56] S. Bian, D. E. S. Kundi, K. Hirozawa, W. Liu, and T. Sato, "Apas: Application-specific accelerators for rlwe-based homomorphic linear transformations," *IEEE Trans. Inf. Forensics Secur.*, vol. 16, pp. 4663–4678, 2021.

[57] Z. Huang, W. jie Lu, C. Hong, and J. Ding, "Cheetah: Lean and fast secure Two-Party deep neural network inference," in *USENIX Security*, 2022, pp. 809–826.

[58] J. H. Cheon, D. Kim, and D. Kim, "Efficient homomorphic comparison methods with optimal complexity," in *ASIACRYPT*, 2020, pp. 221–256.

[59] S. Bian, M. Hiromoto, and T. Sato, "Darl: Dynamic parameter adjustment for lwe-based secure inference," in *DATE*, 2019, pp. 1739–1744.

[60] Q. Lou, S. Bian, and L. Jiang, "Autoprivacy: automated layer-wise parameter selection for secure neural network inference," in *NeurIPS*, 2020, p. 10.

[61] "Microsoft SEAL (release v4.1.0)," microsoft Research, Redmond, WA. [Online]. Available: https://github.com/Microsoft/SEAL

[62] "Tfhepp (release v8)," virtualsecureplatform. [Online]. Available: https://github.com/virtualsecureplatform/TFHEpp

[63] "He^{3}db," zhou Zhang. [Online]. Available: https://github.com/zhouzhangwalker/HE3DB

[64] M. R. Albrecht, R. Player, and S. Scott, "On the concrete hardness of learning with errors," *J. Math. Cryptol.*, vol. 9, pp. 169–203, 2015.

[65] E. Alkim, L. Ducas, T. Pöppelmann, and P. Schwabe, "Post-quantum key Exchange—A new hope," in *USENIX Security*, 2016, pp. 327–343.

[66] E. Lee, J.-W. Lee, Y.-S. Kim, and J.-S. No, "Optimization of homomorphic comparison algorithm on rns-ckks scheme," *IEEE Access*, vol. 10, pp. 26 163–26 176, 2022.

[67] B. Zhou and J. Lam, "Global stabilization of linearized spacecraft rendezvous system by saturated linear feedback," *IEEE Trans. Control Syst. Technol.*, vol. 25, no. 6, pp. 2185–2193, 2017.

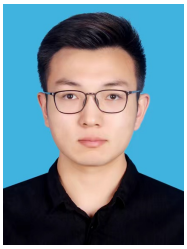
[68] "Tfhe-rs," zama. [Online]. Available: https://github.com/zama-ai/tfhe-rs

[69] M. S. Paterson and L. J. Stockmeyer, "On the number of nonscalar multiplications necessary to evaluate polynomials," *SIAM Journal on Computing*, vol. 2, no. 1, pp. 60–66, 1973.

Song Bian (Member, IEEE) is currently an associate professor at Beihang University. His main areas of interest include fully homomorphic encryption, privacy-preserving computing and domain-specific hardware accelerator. He received B.S. from University of Wisconsin-Madison in 2014. He received M.S. and Ph.D. from Kyoto University, in 2017 and 2019, respectively. He was an assistant professor at Kyoto University from 2019 to 2021. He served as technical committee members/reviewers for top-tier international conferences/journals across different fields of studies, including CVPR, IEEE TIFS and IEEE TCAD. He is a member of ACM and IEEE.

Yunhao Fu received his B.E. in Beihang University, China, in 2023, and is now pursuing a Ph.D degree at the School of Cyber Science and Technology, Beihang University, Beijing, China. His current research interests include encrypted control system and homomorphic encryption.

Dong Zhao (Senior Member, IEEE) received the B.E. degree in automation and the Ph.D. degree in control science and engineering from Beijing University of Chemical Technology, Beijing, China, in 2011 and 2016, respectively. From 2017 to 2018 and in 2021, he worked as a postdoctoral research fellow with the Institute for Automatic Control and Complex Systems (AKS), University of Duisburg-Essen, Germany. From 2018 to 2020, he joined KIOS Research and Innovation Center of Excellence at the University of Cyprus as a postdoctoral research fellow. Since 2022, he has been a professor of School of Cyber Science and Technology, Beihang University, Beijing, China. His research interests are fault diagnosis, fault-tolerant control, cyber-physical systems, and cyber security.



Haowen Pan received his B.S. from Beihang University, China, in 2023, and is now pursuing a Ph.D. degree at the School of Cyber Science and Technology, Beihang University, Beijing, China. His current research interests include applied cryptography and homomorphic encryption.

Yuexiang Jin received his Master's degree from the University of Edinburgh, Scotland, in 2022, and is currently pursuing a Ph.D. at the School of Cyber Science and Technology, Beihang University, Beijing, China. His current research interests include encrypted control systems and homomorphic encryption.

Jiayue Sun (Member, IEEE) received the Ph.D. degree in power electronics and power transmission from Northeastern University in 2021. Now she is a Professor at Northeastern University. Her current research interests include optimization of complex industrial processes, intelligent adaptive learning, distributed control of multi-agent systems and its applications.

She is a Member of the Chinese Association of Automation (CAA) and Chinese Association for Artificial Intelligence (CAAI), and works as the Intelligent Adaptive Learning Committee's secretary of CAAI. She has authored or co-authored 50 peer-reviewed international journal papers. She serves as an Associate Editor of *IEEE Transactions on Cybernetics*, *IEEE Transactions on Neural Networks* and *Learning Systems and IET Electronics Letters*.

Hui Qiao received the BE degree in automation from Tsinghua University, Beijing, China, in 2013, and the PhD degree in control science and engineering from Tsinghua University, Beijing, China, in 2019. From 2019 to 2021, he was a postdoc in the Department of Automation at Tsinghua University, Beijing, China. He is currently an associate professor at Tsinghua University. His research interests include computational imaging, biomedical engineering, and artificial intelligence.

Zhenyu Guan (Member, IEEE) received the Ph.D. degree in electronic engineering from Imperial College London, the United Kingdom, in 2013. Now, he is a professor of the School of Cyber Science and Technology at Beihang University. His current research interests include image processing and high performance computing. He has published more than 45 technical papers in international journals and conference proceedings.

APPENDIX

A. FULL NOTATIONS

We summarize the notations and operators used in this work in Table A1.

B. Details of Parameter Selection

The parameter selection for FHECAP is guided by the following steps to address the three goals outlined in Section VI-A: The size of the plaintext space is first set to $p = 2^{12}$, which directly corresponds to the precision of the system and input upper bound. (as discussed in Section V-A). We apply Theorem V.1 to derive secure LWE parameters: n=2,048, q=2^{64} to ensure the correctness of decryption. Subsequently, by analyzing the number of ciphertext levels consumed in a single multiplication step of the linear transformation, we determine the RLWE modulus $\lfloor \log_2 Q \rfloor$=111. Finally, based on the desired security level, the corresponding polynomial degree N=4,096 for the RLWE scheme is selected to ensure compliance with the security requirements.

In the microbenchmark for linear transformations, the parameter details of the baseline methods are as follows: For the Pailler-based solutions, a 3,072-bit prime modulus was adopted to satisfy 128-bit security requirements under the Pailler cryptosystem. For the Leveled HE solutions, we maintained identical RLWE cryptographic parameters as specified in Table III.

In the microbenchmark for piecewise non-linear function evaluation, we adjust the parameters to ensure consistent accuracy across the three solutions, enabling a fair comparison of efficiency. In the LUT-based solutions, we employ the 12-bit LUT algorithm from [68]. For the PA-based solutions, the evaluation error depends on the degree of the approximation polynomial. As shown in Figure A1, when the degree is approximately 1,000, the maximum error is around 0.017, and the average error is 0.001, which aligns with the rounding error introduced by FHECAP during encoding quantization.

The evaluation of a 1,000-degree polynomial utilizing the Paterson-Stockmeyer method [69] necessitates 11 levels. Based on the lwe-estimator [64], to maintain an equivalent level of security, the RLWE dimension must be set to 16,384.

C. Concrete Parameter Derivation

For practical considerations, we derive Theorem V.1 by substituting the parameters listed in Table A2, an extended version of Table III.

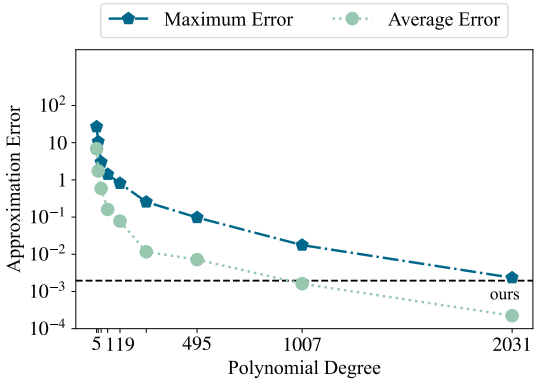
Based on the analysis in step ③ and step ④ of Section V-B, when we set σ_{fresh}=3.2, $\lfloor \log_2 q \rfloor$=64 and $\lfloor \log_2 Q' \rfloor$=60 and $\|F\|_1$=2 (note that $\|F\|_1$=2 can vary depending on the exact

Appendix Table A1
SUMMARY OF NOTATIONS

Notation	Description
$\boldsymbol{a}$	A vector
$\boldsymbol{a}_i$	The i-th element of $\boldsymbol{a}$
A	A matrix
A_i	The i-th row of A
$\{a_i\}_n$	A list of a_i where $1 \leq i \leq n$
$A_{i,j}$	The element of A in the i-th row, j-th column
$\widetilde{a}$	A polynomial
$\widetilde{a}_i$	The i-th coefficient of $\widetilde{a}$
$\|\boldsymbol{a}\|_i$	The i-th norm of $\boldsymbol{a}$
$\|\widetilde{a}\|_i$	The i-th norm of the coefficient vector of $\widetilde{a}$
$\boldsymbol{a} \bigodot \boldsymbol{b}$	Hadamard Product of $\boldsymbol{a}$ and $\boldsymbol{b}$
$A \otimes B$	Kronecker product of A and B
$\langle \boldsymbol{a}, \boldsymbol{b} \rangle$	Inner product of $\boldsymbol{a}$ and $\boldsymbol{b}$
$\mathcal{L}$	The LCNA system
$\boldsymbol{u}$	The system input
$\boldsymbol{x}$	The system state
r	The dimension of the system state
l	The dimension of the system input
K	The gain matrix
Γ	The actuation parameter list
G_Γ	The actuation function family under Γ
$G_{\boldsymbol{\gamma}_i}$	The piecewise actuation function separated by $\boldsymbol{\gamma}_i$
k	The number of segments of the piecewise function
$[\zeta_{l,i}, \zeta_{r,i}]$	The threshold of $G_{\boldsymbol{\gamma}_i}$
ϵ	The system state error of $\mathcal{L}$
$\eta(\epsilon)$	The system input error of $\mathcal{L}$ corresponding to η
n / N	The lattice dimensions for LWE / RLWE / RGSW
$\boldsymbol{s}$	The secret key for LWE
$\widetilde{s}$	The secret key for the RLWE and RGSW
Δ	The scaling factor
Q	The ciphertext modulus for a fresh RLWE
Q'	The ciphertext modulus for an RLWE after rescaling
q	The ciphertext modulus for LWE / RGSW
χ_σ	A Gaussian distribution $\mathcal{N}(0, \sigma)$
$\mathbb{Z}_q^n$	The set of n-length vectors over $\mathbb{Z}_q$
R_q	The cyclotomic ring $\mathbb{Z}_q[\tau]/(\tau^n + 1)$
H	The gadget matrix of RGSW
d	The dimension of H
Bg	The decompose radix of RGSW
SK	The secret key including $\boldsymbol{s}$ and $\widetilde{s}$
BK	The bootstrapping key
KSK	The key switching key
$\mathsf{LWE}_{\boldsymbol{s}}^{n,q}(m)$	An LWE ciphertext encrypting m with parameters (n, q) and key $\boldsymbol{s}$
$\mathsf{RLWE}_{\widetilde{s}}^{N,Q}(\widetilde{m})$	An RLWE ciphertext encrypting $\widetilde{m}$ with parameters (N, Q) and key $\widetilde{s}$
$\mathsf{RGSW}_{\widetilde{s}}^{n,q,\mathsf{Bg},d}(\widetilde{m})$	An RGSW ciphertext encrypting $\widetilde{m}$ with parameters (n, q, Bg, d) and key $\widetilde{s}$
$\mathcal{C}$	The Server Controller
$\mathcal{P}$	The Plant
ℓ_q	The bit length of the LWE phase
ℓ_p	The bit length of the plaintext space of LWE
ℓ_e	The bit length of the bound of LWE noise
ℓ_d	The bit length of the fractional part of the plaintext
δ	The quantizer
ξ	The failure probability

control applications) as prescribed in Table III, we get that the standard deviation of the noise $e_{\mathsf{LWE}(u')}$ in the ciphertext $\mathsf{LWE}(u')$ is $3.2 \times 2 \times \frac{q}{Q'}$, which is about 7 bits.

Subsequently, considering the RLWEToLWE process, we set $q = 2^{64}$, Bg'=32,d'=8 and substitute them into Equation (16): $\mathfrak{e}_{\mathtt{conv}} = \mathfrak{e}_{\mathtt{MS}} + \mathtt{SDU}^{-1}_{-2^{23},2^{23},2048}(2^{-30}) + 2^{12} \times \sqrt{-2\ln(2^{-30})\mathtt{CCB}_{\Psi_{32},2048\times 8}(2^{-30})}$ is approximately 28 bits, where $t_{\Phi_{\mathsf{Bg}'}} \approx 0.051$ using by Lemma II.3 in the supplemental document.



Appendix Figure A1. Approximation error versus polynomial degree for the saturation function. The maximum error ($\max(|f(x) - P(x)|)$ and average error ($E[|f(x) - P(x)|]$) are shown, where $f(x)$ is the saturation function and $P(x)$ is the approximating polynomial.

Appendix Table A2
CONCRETE PARAMETER INSTANTIATION FOR FHECAP

Ciphertext / Key	Parameters
RLWE	N=4,096, σ_{fresh}=3.2 $\lfloor \log_2 Q \rfloor$=111, $\lfloor \log_2 Q' \rfloor$=60
LWE	n=2,048, $q = 2^{64}$
RGSW	n=2,048, Bg=512, d=4, $q = 2^{64}$
BK	$\underline{n}$=672, $\sigma_{\mathsf{BK}} = 2^{12}$
KSK	Bg'=32, d'=8, $\sigma_{\mathsf{KSK}} = 2^{12}$

According to the analysis in step ⑤ of Section V-B, we estimate the noise of EP by setting $\sigma_{\mathsf{BK}} = 2^{12}$, Bg=512, d=4, $\mathfrak{e}_{\mathtt{FFT}} = 2^{24}$: $\mathfrak{e}_{\mathtt{EP}} = \mathtt{SDU}^{-1}_{-2^{27},2^{27},1025}(2^{-30}) + 2^{12} \times \sqrt{-2\ln(2^{-30})\mathtt{CCB}_{\Psi_{512},(2048)\times 2\times 4}(2^{-30})} + \mathtt{SDU}^{-1}_{-2^{24},2^{24},1024}(2^{-30})$ is approximately 33 bits. Thus, the HomComp noise is obtained by utilizing Lemma II.3 in the supplemental document combining with $\mathfrak{e}_{\mathtt{EP}}$ and $\underline{n}$=672: $\mathfrak{e}_{\mathtt{HomLUT}} = \mathfrak{e}_{\mathtt{HomComp}} = \mathtt{CCB}_{\Phi_{\mathtt{EP}},672}(\xi) \approx \sqrt{672}\mathfrak{e}_{\mathtt{EP}}$ is about 38 bits. Finally, we proceed to analyze the CB step. The encrypted polynomial coefficients of RGSW after CB follow two distinct distributions, where $\mathfrak{e}_{\mathtt{CB},0} = \sqrt{-2\ln(2^{-30})\mathtt{CCB}_{\Psi_{32},(2049)\times 8}(2^{-30})}$ is about 26 bits and $\mathfrak{e}_{\mathtt{CB},1} = \mathfrak{e}_{\mathtt{HomComp}} + \mathtt{SDU}^{-1}_{-2^{23},2^{23},2049}(2^{-30}) + 2^{12} \times \sqrt{-2\ln(2^{-30})\mathtt{CCB}_{\Psi_{32},2049\times 8}(2^{-30})}$ is about 38 bits. Finally, in the CMUX gate, $\mathfrak{e}_{\mathtt{CMUX}} = \mathfrak{e}_{\mathtt{conv}} + \mathtt{CCB}_{\Phi_{\mathtt{CB},0},3\times 2048\times 4-2\times 4}(\xi) + \mathtt{CCB}_{\Phi_{\mathtt{CB},1},2048\times 4+2\times 4}(\xi) \approx \frac{1}{4}\mathtt{CCB}_{\Phi_{\mathtt{CB},1},2\times 2048\times 4}(\xi)$ is about 51 bits. Through the concrete parameter derivation, we obtain that when $l_q = 64$, $l_e = 51$, the failure probability is 2^{-30}, which means that under this set of parameters, our plaintext space has 12 bits.