IMCE: An In-Memory Computing and Encrypting Hardware Architecture for Robust Edge Security

Hanyong Shao[1,2], Boyi Fu[1], Jinghao Yang[1], Wenpu Luo[1], Chang Su[1], Zhiyuan Fu[1], Kechao Tang[1,3*] and Ru Huang[1,3*]
[1]School of Integrated Circuits & [2]Academy for Advanced Interdisciplinary Studies, Peking University, Beijing 100871, China
[3]Beijing Advanced Innovation Center for Integrated Circuits, Beijing 100871, China.
*E-mail: {tkch, ruhuang}@pku.edu.cn

***Abstract*—Edge devices deployed in unsupervised scenarios employ Physical Unclonable Functions (PUFs) for identity authentication and embedded XOR encoding for data encryption. However, on the one hand, the existing strong PUFs such as CMOS-based XOR Arbiter PUFs and NVM-based RRAM PUFs are vulnerable to various machine learning (ML) modeling attacks. On the other hand, the transmission of keys for embedded XOR encoding also faces the risk of being eavesdropped in unsecured channels. In response to these challenges, this paper proposes a high-security In-Memory Computing and Encrypting (IMCE) hardware architecture based on a FeFET macro, featuring both a PUF mode for identity authentication and an encrypted CIM mode with *in-situ* decryption. The PUF mode ensures a prediction accuracy close to 50% (equivalent to random guessing attack) under various ML models due to the proposed Hamming distance comparison used in challenge-response pairs (CRPs) generation. In addition, by utilizing the CRPs generated in PUF mode as encryption keys, the CIM mode of IMCE achieves robust security through public-key cryptography via CRPs-masked key transfer, preventing the leakage of keys and data. Therefore, by applying a novel CRPs generation scheme and reusing the generated CRPs for *in-situ* CIM decryption, the security of both PUF and encrypted CIM mode is enhanced concurrently. In addition, IMCE significantly reduces the power overhead thanks to the high energy efficiency of ferroelectric FETs (FeFETs), making it highly suitable for secure applications in edge computing devices.**

***Keywords*—Hardware architecture, Physical Unclonable Functions (PUFs), Encrypted in-memory computing, Machine learning security, Ferroelectric FET (FeFET)**

I. Introduction

In the era of the Internet of Everything (IoE), the rise of edge computing has risen the demand for secure data interactions from cloud to edge. Deployed in unsupervised environments and connected to cloud data, edge devices are prime targets for identity and asset attacks [1]. Identity attacks involve masquerading as legitimate users to deceive the cloud and illegally access sensitive data, whereas asset attacks involve stealing data stored in non-volatile memories (NVM) or intercepting transmitted keys. Therefore, it is imperative to assign unique identifiers for authentication and encrypt local storage and data transfer on resource-limited edge devices to safeguard assets like privacy and pre-trained weight parameters.

As hardware security primitives, Physical Unclonable Functions (PUFs) exploit manufacturing variations to generate unique challenge-response pairs (CRPs), thereby facilitating lightweight on-chip authentication [2]. Additionally, the use of embedded XOR encoding in encrypted Compute-in-Memory (CIM) architecture safeguards the data stored in the memories, particularly for those CIM based on NVM [3-4].

However, due to the rapid advancement of AI technology and computing power, existing strong PUFs such as XOR Arbiter PUF, double Arbiter PUF (DAPUF), and RRAM-based strong PUFs are increasingly at risk from a variety of newly developed machine learning (ML) algorithms [5-8]. This vulnerability arises because these PUFs only introduce non-linear XOR operation when generating responses. Therefore, the coupling between PUF cells still employ linear accumulation, making them susceptible to modeling attacks. On the other hand, encrypted CIM architectures like XOR-CIM still rely on a critical key string to perform XOR encryption on stored data. Consequently, the keys required for decryption operations are at risk of eavesdropping when transmitted over insecure channels, leading to the potential leakage of stored data [9]. Therefore, the security of edge devices necessitates a more comprehensive and robust hardware security architecture.

To address these issues, this paper presents an In-Memory Computing and Encrypting (IMCE) hardware architecture based on a FeFET macro. As an emerging memory technology, FeFET offers non-volatile and high-energy efficiency, thereby reducing the energy consumption of the IMCE architecture.

The major contributions of this paper are as follows:

1) Propose a secure hardware architecture featuring both a PUF mode for authentication and an encrypted CIM mode with *in-situ* decryption. The modeling resistance capability of the PUF mode is significantly enhanced by the Hamming Distance Comparison (HDC) used in the generation of CRPs. The HDC introduces non-linear operations for each PUF cell, resulting in robust security compared to XOR APUF and RRAM PUFs.

2) Introduce a public-key cryptography based protocol named CRPs-Masked Key Transfer (CMKT) to prevent eavesdropping. This protocol utilizes the challenges and responses generated in PUF as the masked-keys and real keys, thereby enhancing the security during key transmission in the encrypted CIM mode. Additionally, the encrypted CIM mode enables high parallelism *in-situ* decryption, avoiding the data transfer associated with traditional decryption methods.

3) Develop a ML algorithm based on variation-matrix modeling. This algorithm demonstrates an accuracy exceeding 85% against XOR APUF and RRAM PUFs, but only 50% against our PUF, thereby proving the superior security of IMCE.

The rest of this paper is organized as follows: Section II presents the relevant background. Section III details the IMCE hardware architecture, the CMKT protocol, and describes the workflow of its PUF and CIM modes. Section IV introduces the proposed ML algorithms for PUF attacking and conducts the evaluation of the security and performance of the IMCE architecture. Finally, Section V provides the conclusion.

979-8-3503-4859-0/DATE24/© 2024 EDAA

II. Background and Related Works

In this section, we mainly discuss the current research on PUFs and encrypted CIM, including mainstream PUF designs and encrypted CIM architecture. Additionally, we provide an introduction to the FeFET utilized in this paper.

A. Existing strong PUFs

As outlined in Sec. I, numerous strong PUFs such as DAPUF and XOR APUF, have been developed to bolster security [10-11]. **Fig. 1(a)** demonstrates that DAPUF expands the CRPs space by increasing the number of MUX chains. Despite this, it only increases the interlinking complexity between delay chains, while the core mechanism is still the linear accumulation of time delay. Therefore, DAPUF's multiple responses are sequentially XORed to introduce non-linearity, similar to the process in XOR APUF shown in **Fig. 1(b)**. As both PUFs are mathematically equivalent [5], we mainly focus on XOR APUF.

Besides CMOS-based PUFs, there are PUF designs based on non-volatile memory like RRAM [12-13]. **Fig. 1(c)** depicts a common RRAM PUF design. The I-V non-linearity variations caused by RRAM filaments result in random sense line (SL) currents when different rows and columns are selected in the array. Devices corresponding to M rows and N columns are selected through input challenges, and SL currents are split into two segments using a column selector. These two parts are combined and compared to generate responses. However, even when XOR operations are introduced after the comparison, this current accumulation-based CRPs generation also faces the risk of ML attacks due to the linear coupling of PUF cells.

Additionally, existing PUF designs are dedicated, resulting in additional circuit overhead. To cut this area cost, it is beneficial to combine PUFs with the CIM architecture [14-15].

B. Encrypted CIM architecture

Edge devices deployed in unsupervised scenarios pose a risk of data leakage, such as directly reading memory cell or reverse engineering [16], especially for NVM-based CIM architectures. Therefore, it has been proposed to use XOR encoding for lightweight encryption protection of stored 0/1 data. However, during decryption and computation, there is a power and time cost associated with data transfer. Conventional decryption requires reading encrypted weights, off-chip decrypting, and then writing back to CIM, severely lower computing efficiency. Researches have been conducted on complementarily storing data through paired units and dual word line (WL), such as the dual-WL 6T SRAM [3], 2R-RRAM [17], and 2T-FeTFET [18] encrypted CIM architectures, to achieve *in-situ* decryption. As shown in **Fig. 2**, the unit structure of dual-WL 6T SRAM first preprocesses the input to the two WLs, and then, based on the complementary weights at the two nodes, *in-situ* decryption and multiply-and-accumulate (MAC) operations can be performed:

$$BL(\mathrm{Out}) = (\mathrm{In}\cdot\mathrm{Key})\cdot\overline{\mathrm{eW}} + (\mathrm{In}\cdot\overline{\mathrm{Key}})\cdot\mathrm{eW}$$
$$= \mathrm{In}\cdot(\mathrm{Key}\oplus\mathrm{eW}) = \mathrm{In}\cdot\mathrm{W} \quad (1)$$

here, $eW = Key \oplus W$ is the pre-stored encrypted weight. Since each row shares the same input In, the weights in the same row share the same Key.

However, during the actual *in-situ* decryption process, since data encryption is usually completed in the cloud, it is necessary to obtain the corresponding keys from the cloud. For edge devices, there is a risk of eavesdropping in the communication channel with the cloud, and the same data leakage problem still exists even after encryption [9].

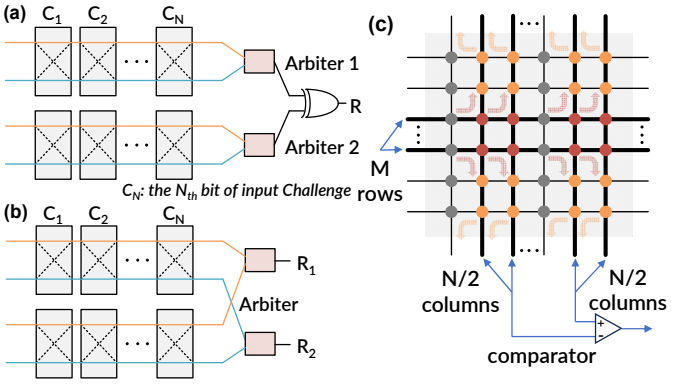

Fig. 1. Circuit designs of (a) double Arbiter PUF and (b) XOR Arbiter PUF. (c) Circuit and sneak paths of commonly used RRAM PUF; the M rows and N columns are selected by challenges, leaving the rest of the nodes floating.

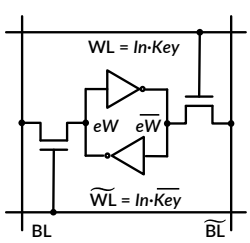

Fig. 2. Circuits of a dual-WL 6T SRAM cell; two nodes on the left and right can perform AND operation separately.

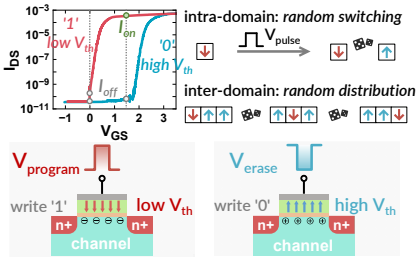

Fig. 3. Structure and I_{DS}-V_{GS} curve of FeFETs. The pos./neg. gate pulses for programming will cause an equivalent shift in the curve. I_{on} is only obtained at high V_{GS} and low V_{th}.

C. Basics of the FeFET

HfO_2-based FeFET is a strong candidate for edge computing hardware due to its non-volatility, low power consumption, and CMOS compatibility [19]. The structure of FeFET is similar to that of MOSFET but includes an additional ferroelectric (FE) layer in the gate stack. The polarization direction of the FE layer can be altered by applying a program or erase voltage to the gate, enabling the modulation of the I_{DS}-V_{GS} curve, which is simulated in HSPICE as illustrated in **Fig. 3**. The curves with low and high threshold voltage (V_{th}) correspond to the stored weights of '1' and '0' in the FeFET, respectively. Since the polarization state of the FE layer remains stable even after the external electric field is removed, the information it stores is non-volatile and may last >10 years [19]. To read the FeFET, one only needs to apply a small pulse to the gate, which does not disrupt the polarization state. The stored weight (polarization state) can then be determined based on the I_{DS}. For example, the high I_{DS} is read when the FE is polarized towards the channel. FeFET has lower power consumption and is suitable for edge computing.

Furthermore, because the FE layer includes numerous FE domains, when the orientations of the domains are not aligned, the FeFET is in an intermediate threshold state and exhibits significant random V_{th} variation. This is because both the spatial distribution and the stochastic switching of the domains contribute to the entropy of the FeFET [20], as illustrated in the schematic in **Fig. 3**. The fluctuation of V_{th} consequently leads to variations of the I_{DS} when device is read. Therefore, FeFETs can generate random weights (rW) through cycle-to-cycle I_{DS} variations at the intermediate V_{th} state. In this work, this feature is utilized as the entropy source to generate true random numbers.

III. The In-Memory Computing and Encrypting Secure Hardware Architecture

In this section, we elaborate on the proposed IMCE secure hardware architecture, encompassing its design, workflow, and security optimization strategies for both the PUF and encrypted CIM modes.

A. Architecture design and FeFET macro

Fig. 4 shows the proposed IMCE architecture design, including the FeFET-based macro and corresponding circuit structure. By re-using the FeFET array, the IMCE architecture can be operated under two modes: PUF and encrypted CIM. As shown in **Fig. 4(a)**, the compute data-paths related to the PUF and CIM modes are represented by distinct colors. Inputs (or challenges) to the IMCE are preprocessed through dual signal encoder and inputted into the array, eventually serving as the WL voltage for encrypted MAC (or CRPs generation). The selected sense line (SL) current is outputted via the SL selector, and depending on the circuit of the different modes, the encrypted MAC result (or response) is obtained.

The circuit structure of the FeFET array is described in **Fig. 4(b)**. It is an AND-type array, with its basic cell consisting of two complementary FeFETs with opposite polarization states, as indicated by the grey dashed lines. The WLs connected to the gates of the complementary FeFETs during operation are also paired. The voltage levels are determined by the operating mode of the IMCE, which will be detailed in the following sections. The computation results of the array are outputted to the SL selector (MUX) via the current on the SLs.

To enable the IMCE to operate in both PUF and CIM modes, we improve the input signal generator in [3]. The circuit design of the dual signal encoder proposed in this paper is shown in **Fig. 4(c)**. Its output is connected to the WLs of the FeFET array and is responsible for preprocessing the input signals. This encoder consists of two NOR gates and two inverters, aiming to realize the Boolean logic of $\overline{AB} = \overline{A} + \overline{B}$. The workflows of the two modes of the IMCE will be explained in details below.

B. PUF mode: Hamming distance comparison

The PUF mode of IMCE involves two steps during operation: enrollment and authentication. During enrollment, the PUF uses the random fluctuations of the FE domains of the FeFET as an entropy source to generate true random numbers (see Sec. II-C), then complementarily stores them as rW and $\overline{rW}$ in the cell, respectively. Alternatively, it is feasible to use random numbers generated by external TRNG as rW. Therefore, compared to APUFs and other CMOS-based PUFs, the FeFET-based PUF in IMCE can erase and regenerate rW randomly as needed, thereby changing the CRPs of PUF, and reconfiguring it. This is beneficial for the PUF mode of IMCE to avoid privacy leaks in scenarios such as user change [2].

The authentication step of the PUF mode utilizes Hamming distance comparison (HDC) to generate CRPs. In the PUF mode, the gate of the two complementary FeFETs depicted in **Fig. 4(b)** are always connected to the WLs at complementary voltages, and the source of the FeFETs in the same column are connected to the same SL. The challenge inputted into the PUF is divided into two parts after vector coding: the first part of challenge (the first n bits) is processed in the dual signal encoder. As shown in **Fig. 4(c)**, the inputs are C_i and "1" (always high voltage), while the results outputted by NOR gates are C_i and $\bar{C}_i$, respectively. The results of encoder are inputted into the array complementary, which correspond to the 2n rows of WL voltage. Therefore, the ith 1/0 value of the C_i is the high/low voltage of the ith WL_i. Meanwhile, the second part of the challenge is inputted into the SL decoder, which connects to the MUX and determines which two SLs are selected as the output to the comparator.

Because FeFET only outputs I_{on} at low V_{th} state and high V_G, it performs an AND operation. As the WLs voltage (gate voltage) and the weights rW (threshold state) within the same cell are complementary, this achieves a nonlinear XOR in the ith cell: $C_i \cdot \overline{rW}_i + \bar{C}_i \cdot rW_i = C_i \oplus rW_i$. Therefore, the SL current of the jth column is $I_{SL,j} = \Sigma_i (C_i \oplus rW_{i,j}) \cdot I_{on} = HD(C \oplus rW_j) \cdot I_{on}$, meaning the SL current of the jth column is proportional to the Hamming distance between the input vector $\boldsymbol{C}$ and the weight vector $\boldsymbol{rW_j}$ of the jth column. Further, the SL selector is utilized to select two SLs (decided by the challenge) for current comparison and finally generate 1-bit response.

The operation of bitwise XOR is an essential process of column-wise computing the Hamming distance between the challenge vector $\boldsymbol{C}$ and the random weight matrix $\boldsymbol{rW}$, thus comparing the above result is equivalent to Hamming distance comparison. Introducing the nonlinear XOR is the most critical process of CRPs generation (authentication) to enhance the

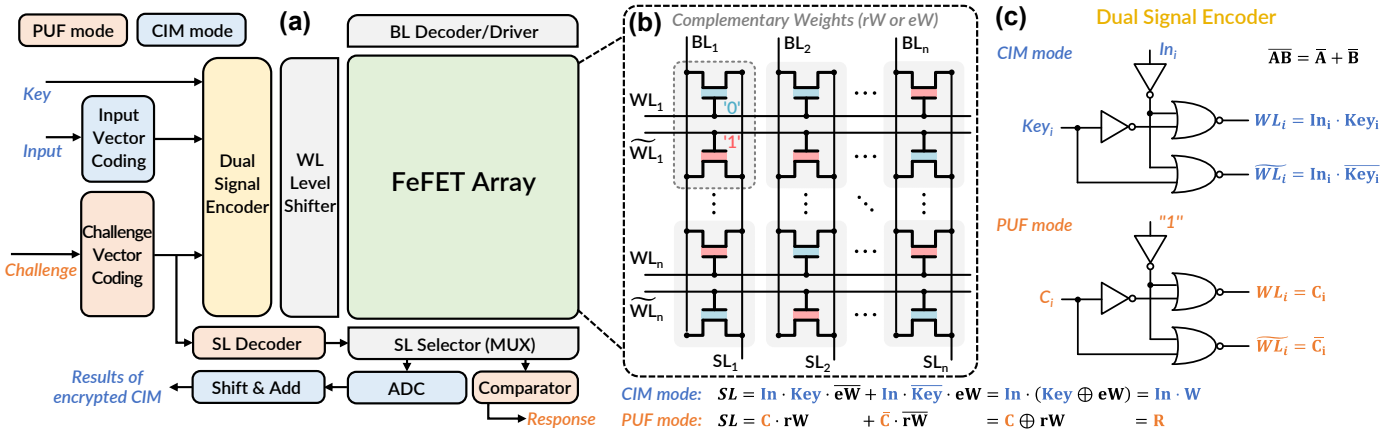

Fig. 4. Illustration of the IMCE architecture and FeFET macro. (a) Macro design based on the FeFET array, including the peripheral circuits of IMCE for PUF and CIM modes, represented by distinct colors. (b) Circuit structure of the FeFET array, consisting of cells composed of two complementary state FeFETs, with adjacent WLs being pair-coupled. (c) Dual signal encoder in the macro, generating the logic levels of adjacent WLs in (b) through the input of In and Key (or challenge bit).

coupling complexity between PUF cells, making it difficult for attackers to model the CRPs relationship or directly map the random weights. At the same time, because there are 2^n possible n-bit challenge signals, combining with the possibility of selected SL pairs, there are a total of $2^n \cdot \binom{m}{2}$ possible CRPs. With the increase of the array size, the exponential growth of CRPs ensures the number of CRPs that cannot be exhaustively enumerated within finite time, thus serving as the strong PUF.

C. *Encrypted CIM mode: In-situ decryption*

The encrypted CIM mode of IMCE mainly involves the preprocessing of the input vector and the *in-situ* decryption of MAC operations. In the encrypted CIM architecture, the original weights W are stored as ciphertext, typically using lightweight XOR encryption, hence the encrypted weights eW satisfy $eW = Key \oplus W$.

As shown in **Fig. 4(b)**, the expected result of vector-matrix multiplication (VMM) is $In \cdot W$. Since each cell in the array simultaneously stores eW and $\overline{eW}$, the decryption process shown in equation (1) can be realized through paired FeFETs. Therefore, by preprocessing the input and key vectors through the dual signal encoder, we obtain the $In \cdot Key$ and $In \cdot \overline{Key}$ necessary for decryption. The input and output in this mode are shown in **Fig. 4(c)**. Thus, based on the complementary FeFETs cell, IMCE avoids weights transfer in the encrypted CIM mode, completing decryption and VMM through MAC operations at the same time, thereby improving computational parallelism while protecting the original weights.

D. *Workflow and protocol*

We develope a complete workflow and protocol for edge devices and cloud interaction based on the IMCE, utilizing its features and security enhancement strategies (Hamming distance comparison, embedded XOR encoding) in both PUF and CIM modes, as shown in **Fig. 5(a)**.

The three objectives of the protocol are: 1) To verify the legal identity of edge devices; 2) To securely transfer weights and data to verified devices; 3) To encrypt and protect local data with high computing efficiency. The following describes the protocol and the detailed workflow of the IMCE.

In a trusted environment:

(1-1) Random weights generation. When the IMCE is in a trusted environment, the cloud enrolls its PUF mode. The true random numbers generated by entropy source of FeFET or external rW are written in the array as random weights rW.

(1-2) PUF enrollment. The cloud sends a large amount of challenges to the IMCE, then the responses generated in PUF mode are returned to the cloud and recorded, thus establishing the CRPs library of IMCE.

In an unsupervised environment:

(2-1) PUF authentication. Before starting communication between edge devices and the cloud, it is essential to first verify the identity of IMCE. The cloud randomly selects challenges from the library and sends them to the array of the PUF mode. The returned responses are then compared with those in the library to verify the IMCE.

(2-2) Encrypted weights writing. After passing the identity verification, the cloud sends encrypted weights eW to the IMCE through an unsecured channel. The IMCE will complementarily write eW in the array of the CIM mode.

(2-3) *In-situ* decryption. When CIM is needed, the cloud will send masked-keys to the IMCE, avoiding the risk of eavesdropping on the keys used to decipher the eW. The protocol is given below. The IMCE uses the CRPs generation of the PUF mode to decrypt the masked-keys into keys, then the CIM results can be obtained with the input and keys.

The protocol effectively ensures the security of the cloud and data, which is reflected in three aspects: 1) the XOR operation inside the PUF's unit-cell makes it difficult for attackers to model CRPs relationships via collected CRPs; 2) because the keys transferred in the channel are masked, eavesdroppers cannot decrypt the eW based on masked-keys; 3) since the weights stored in the IMCE are encrypted, thieves cannot obtain the original weights by reading the memory.

E. *CRPs-masked key transfer*

To ensure the security of weights over an unsecured channel, it is essential that the key and ciphertext are not accessible at the same time. Therefore, we propose a public-key cryptography-based protocol using CRPs of PUF, as shown in **Fig. 5(b)**. The cloud randomly selects challenge-response pairs from the CRPs library, where the responses serve as the keys for XOR encryption, and the challenges serve as the masked-keys.

Consequently, when the edge device receives the challenges, the corresponding responses of the PUF are the keys for eW. These keys are then input into CIM mode to yield the correct result for *in-situ* decryption. Since CRPs for encryption only exist in two places: the cloud library and the local PUF itself, this CRPs-Masked Key Transfer (CMKT) strategy effectively protects the keys in an unsecured communication channel.

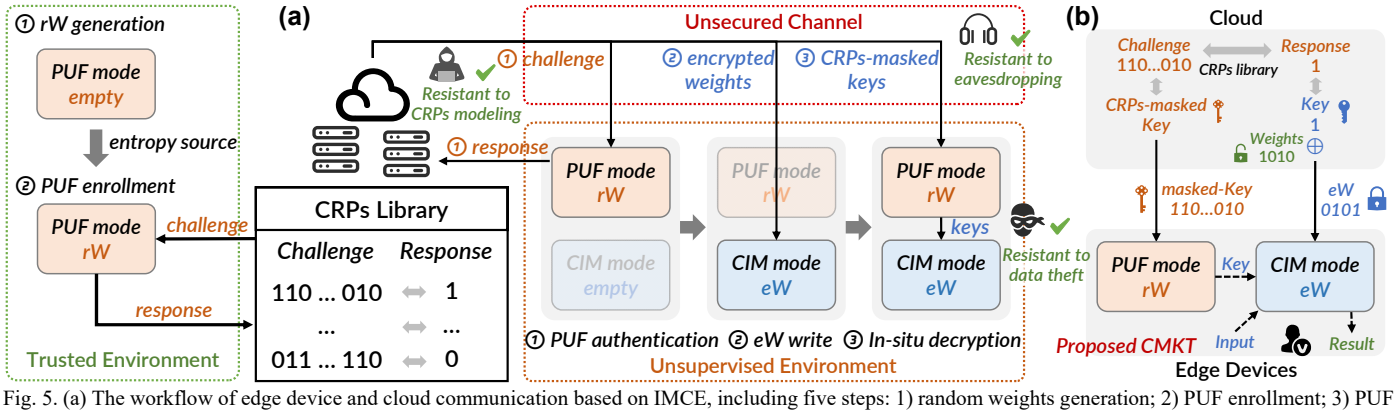

Fig. 5. (a) The workflow of edge device and cloud communication based on IMCE, including five steps: 1) random weights generation; 2) PUF enrollment; 3) PUF authentication; 4) encrypted weights transfer and write; 4) CRPs-masked keys transfer and in-situ decryption. (b) Schematic of CRPs-Masked Key Transfer (CMKT). The cloud uses the CRPs library to encrypt data and keys, and then the edge device uses its PUF and CIM mode to implement key-generation and *in-situ* decryption.

IV. Evaluation of IMCE

In this section, we conduct a comprehensive evaluation of the security and performance of IMCE architecture. We also propose a ML algorithm based on variation-matrix modeling.

A. Security of PUF mode and the proposed algorithm

First, we evaluate the security of the IMCE's PUF mode using different ML algorithms employed in previous studies. These algorithms consider challenges from CRPs as input layers or samples, with the corresponding responses as output layers or labels. This paper employs eight conventional ML models to evaluate PUF security, as shown in **Fig. 6(a)**. These adversary models cover a broad spectrum of ML algorithms [8].

Reports indicate that the XOR APUF and RRAM PUF mentioned in Sec. II-A also exhibit resistance to ML modeling under the conventional algorithms mentioned above. This is because these algorithms treat the PUF as a black box without considering its circuit structure. However, the actual PUF design is public to attackers, and combining ML models with the PUF design will significantly increase the attack success rate [21].

To better utilize the PUF circuits for simulating more realistic attack scenarios, we propose a modeling method based on the variation-matrix. Its core is to directly attack the random fluctuations inside the PUF. In the actual CRPs generation, the PUF uses the random variations brought by its entropy source and the challenges $\boldsymbol{C}$ to generate responses $\boldsymbol{R}$ through the function $puf()$. The variations introduced by the entropy source are constant for the same PUF, such as the delay τ_i of each MUX in APUF or the conductance value $G_{i,j}$ of each device in RRAM PUF. These random constants can be recorded as a variation-matrix $\boldsymbol{M}$, which is not visible to the attacker. So, we have:

$$\boldsymbol{R} = puf(\boldsymbol{C}, \boldsymbol{M}) \tag{2}$$

Because the PUF circuit structure and the corresponding function $puf()$ are known, modeling CRPs can be simplified to modeling the random variation-matrix $\boldsymbol{M}$ in the PUF. Hence, the modeling of the variation-matrix comes to an optimization NP problem, with the objective function (loss) being:

$$Loss\left(\boldsymbol{C}, \widetilde{\boldsymbol{M}}, \boldsymbol{R}\right) = \left|\boldsymbol{R} - puf\left(\boldsymbol{C}, \widetilde{\boldsymbol{M}}\right)\right|^2 \tag{3}$$

where $\widetilde{\boldsymbol{M}}$ is the guessed matrix. Another function that can be used to assess the accuracy during the optimization is:

$$Diff\left(\boldsymbol{M}, \widetilde{\boldsymbol{M}}\right) = \frac{\sum\left|\widetilde{\boldsymbol{M}} - \boldsymbol{M}\right|}{\sum|\boldsymbol{M}|} \in [0,1] \tag{4}$$

Simulated Annealing (SA) is a heuristics global optimization algorithm that probabilistically converges to the global optimum, which is effective in solving combinatorial optimization NP problems [22]. In this work, the variation-matrix $\boldsymbol{M}$ of PUF is guessed and obtained through SA. In **Algorithm 1**, elements in the matrix are randomly modified, and the acceptance of this modification is jointly determined by the optimization of the loss and the temperature probability $e^{-\Delta E/T}$. After multiple iterations, $\boldsymbol{M}$ is well-guessed and attack accuracy increases.

Algorithm 1 Simulated Annealing for PUF modeling

Input: Challenges $\boldsymbol{C}$ and responses $\boldsymbol{R}$ of PUF instance, where $\boldsymbol{R} = \mathrm{puf}(\boldsymbol{C}, \boldsymbol{M})$. N is the cell number of PUF. Initial temperature T_0, cooling rate $\kappa \in (0,1)$.

Output: The final variation matrix $\widetilde{\boldsymbol{M}}$ of target PUF.

$\widetilde{M} \leftarrow M_0$; $\quad T \leftarrow T_0$; $\quad loss \leftarrow \mathrm{Loss}(C, M_0, R)$;
for $k \leftarrow 1$ **to** max_iterations **do**
  $new\_\widetilde{M}[\mathrm{Rand}(i), \mathrm{Rand(j)}] \leftarrow \widetilde{M}[\mathrm{Rand}(i), \mathrm{Rand(j)}] + \delta$;
  $new\_loss \leftarrow \mathrm{Loss}(C, new\_\widetilde{M}, R)$;
  if $(new\_loss < loss)$ **then**
    $\widetilde{M} \leftarrow new\_\widetilde{M}$;
  else
    if $(\mathrm{Rand}() < \exp(-(new\_loss - loss)/T)$ **then**
      $\widetilde{M} \leftarrow new\_\widetilde{M}$;
    end if
  end if
  $T \leftarrow \kappa * T$;
end for

Fig. 6(a) shows that as the number of CRPs used for training increases, the success rate of attack hovers around 50%, close to the accuracy of random guessing, thus proving the security of our PUF mode under conventional algorithms. **Fig. 6(b)** illustrates that as the number of CRPs used for training increases under the variation-matrix modeling, the attack accuracy for APUF, XOR APUF, and RRAM PUFs increases. The attack accuracy of XOR APUF exceeds 85%. Meanwhile, the IMCE's PUF remains at 50%, which demonstrates high security. **Fig. 6(c)** delineates the attack of RRAM PUF in [12]. During the greedy algorithm, the $Diff$ rapidly drops from 0.5 to 0.2; whereas, in SA, the accuracy ascends from 50% to 90%. However, for the IMCE' PUF, it remains around 0.5, indicating its robust security.

The proposed IMCE's PUF has high security when attacked in conjunction with actual circuit design. This is due to each cell of our PUF utilizing nonlinear XOR, rather than just performing a single XOR on the linear sum of results like other PUFs.

B. Security of CIM mode

The security of the encrypted CIM architecture can be effectively demonstrated by the degradation of inference accuracy under encryption [3,4,18]. We employ a widely used lightweight CNN model, MobileNet V3 [23], pre-trained on the CIFAR-10 dataset to achieve an high accuracy of over 90%, which is the baseline in **Fig. 7(a)**. Due to the large parameter size of network models, encrypting all weights is costly, so we

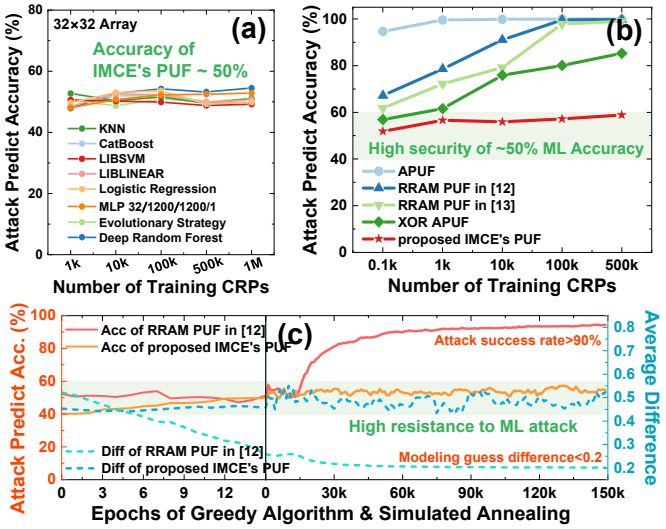

Fig. 6. (a) Attack accuracy and guess difference with the GA and SA epochs. (b) Accuracy of existing PUF designs and our IMCE's PUF with trained CRPs. (c) ML attack accuracy of IMCE's PUF under eight common attack models.

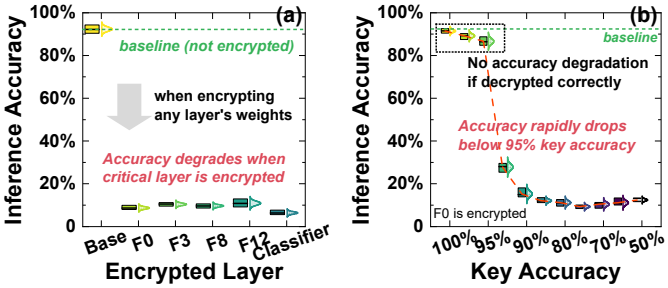

Fig. 7. (a) The impact on inference accuracy decay after encrypting different layers, where "Base" represents no encryption; (b) Decrypting with incorrect key results in a swift drop in inference accuracy as key accuracy decreases.

assessed the security improvement from encrypting parts of the layers. The figure shows the inference accuracy quickly drops below 10% when encrypting the network's feature 0, 3, 8, 12, and classifier layers individually. Therefore, encrypting critical layers will protect the entire network at a relatively low cost.

Fig. 7(b) further evaluates the improvement in CIM security brought by the proposed CMKT. Since the attacker cannot obtain the entirely correct key from the masked-key, the inference accuracy of the *in-situ* decryption will drastically decline. When the key accuracy is below 95%, the inference accuracy drops below 30%, effectively avoiding security risks from guessing attacks and eavesdropping attacks. Meanwhile, when the key is 100% accurate, the accuracy is almost the same as the baseline accuracy (without encryption), indicating that the decryption process does not affect inference accuracy.

C. *Area and energy-efficiency*

Fig. 8 evaluates the area and energy efficiency of IMCE, comparing it with existing XOR-CIM and PUF implementations under the BSIM model in HSPICE with the same node. The IMCE is based on a compact 2T-FeFET array, resulting in a significantly reduced area and energy overhead compared to Arbiter PUF and SRAM. Compared to RRAM, it also shows an advantage in energy consumption. During the evaluation, energy consumption is normalized with same array size (per cell) that perform 1-bit response generation or single MAC operation.

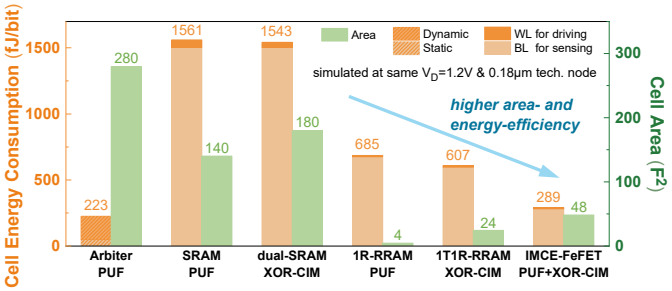

Fig. 8. The area and energy consumption overhead of different PUF and XOR-CIM designs, and the area values are typical ones reported in previous research.

V. Conclusion

In this work, we proposed and verified a secure hardware architecture that incorporates both PUF and encrypted CIM mode for edge security. The ML resistance of the PUF mode is enhanced by introducing non-linear XOR in each PUF cell, while the security of the keys in the encrypted CIM mode is safeguarded by the CRPs-Masked Key Transfer. Additionally, we also developed the variation-matrix modeling algorithm to enable a more realistic scenario for PUF security evaluation. We believe that the FeFET-based IMCE will serve as a foundation for the development of secure edge computing.

Acknowledgment

This work was supported by National Key R&D Program of China (2022YFB4400300), NSFC (62274003, 61927901 and 92164203), 111 Project (B18001) and Peking Nanofab.

References

[1] Xiao, Yinhao, et al. "Edge computing security: State of the art and challenges." *Proceedings of the IEEE* 107.8 (2019): 1608-1631.

[2] Gao, Yansong, Said F. Al-Sarawi, and Derek Abbott. "Physical unclonable functions." *Nature Electronics* 3.2 (2020): 81-91.

[3] Huang, Shanshi, et al. "XOR-CIM: Compute-in-memory SRAM architecture with embedded XOR encryption." *Proceedings of the 39th International Conference on Computer-Aided Design*. 2020.

[4] Huang, Shanshi, et al. "Secure XOR-CIM engine: Compute-in-memory SRAM architecture with embedded XOR encryption." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* 29.12 (2021): 2027-2039.

[5] Khalafalla, Mahmoud, and Catherine Gebotys. "PUFs deep attacks: Enhanced modeling attacks using deep learning techniques to break the security of double arbiter PUFs." *2019 Design, automation & test in Europe conference & exhibition (DATE)*. IEEE, 2019.

[6] Wang, Sying-Jyan, et al. "Adversarial attack against modeling attack on PUFs." *Proceedings of the 56th Annual Design Automation Conference*. IEEE, 2019.

[7] Chatterjee, et al. "SACReD: An attack framework on SAC resistant delay-PUFs leveraging bias and reliability factors." *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021.

[8] Ruhrmair, Ulrich, and Jan Solter. "PUF modeling attacks: An introduction and overview." *2014 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. 2014.

[9] Rottenberg, François, et al. "CSI-based versus RSS-based secret-key generation under correlated eavesdropping." *IEEE Transactions on Communications* 69.3 (2020): 1868-1881.

[10] Machida, Takanori, et al. "A new arbiter PUF for enhancing unpredictability on FPGA." *The Scientific World Journal* 2015 (2015).

[11] Zhang, Jiliang, et al. "DA PUF: dual-state analog PUF." *Proceedings of the 59th ACM/IEEE Design Automation Conference*. 2022.

[12] Yang, Jianguo, et al. "A machine-learning-resistant 3D PUF with 8-layer stacking vertical RRAM and 0.014% bit error rate using in-cell stabilization scheme for IoT security applications." *2020 IEEE International Electron Devices Meeting (IEDM)*. IEEE, 2020.

[13] Nili, Hussein, et al. "Hardware-intrinsic security primitives enabled by analogue state and nonlinear conductance variations in integrated memristors." *Nature Electronics* 1.3 (2018): 197-202.

[14] Chen, Zhuojun, et al. "PUF-CIM: SRAM-Based Compute-In-Memory With Zero Bit-Error-Rate Physical Unclonable Function for Lightweight Secure Edge Computing." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems* (2023).

[15] Kim, Jeong-Hyeon, et al. "Reliable and lightweight PUF-based key generation using various index voting architecture." *2020 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, 2020.

[16] Zuo, Pengfei, et al. "Sealing neural network models in encrypted deep learning accelerators." *2021 58th ACM/IEEE Design Automation Conference (DAC)*. IEEE, 2021.

[17] Li, Wantong, et al. "Secure-RRAM: A 40nm 16kb compute-in-memory macro with reconfigurability, sparsity control, and embedded security." *2021 Custom Integrated Circuits Conference (CICC)*, 2021.

[18] Luo, Jin, et al. "Novel ferroelectric tunnel FinFET based encryption-embedded computing-in-memory for secure AI with high area-and energy-efficiency." *2022 International Electron Devices Meeting (IEDM)*, IEEE, 2022.

[19] Khan, Asif Islam, et al. "The future of ferroelectric field-effect transistor technology." *Nature Electronics* 3.10 (2020): 588-597.

[20] Mulaosmanovic, Halid, et al. "Random number generation based on ferroelectric switching." *IEEE Electron Device Letters* (2017): 135-138.

[21] Alamro, Meznah A., et al. "Robustness and unpredictability for double arbiter pufs on silicon data: Performance evaluation and modeling accuracy." *Electronics* 9.5 (2020): 870.

[22] Lin, Feng-Tse, et al. "Applying the genetic approach to simulated annealing in solving some NP-hard problems." *IEEE Transactions on systems, man, and cybernetics* 23.6 (1993): 1752-1767.

[23] Howard, Andrew, et al. "Searching for mobilenetv3." *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2019.