ABSTRACT The Rasta cipher, proposed by Dobraunig et al. (CRYPTO 2018), is an HE-friendly cipher enjoying the fewest ANDs per bit and the lowest AND-depth among the existing ciphers. A novel feature of Rasta is that its affine layers are freshly and randomly generated for every encryption. In this paper, we propose a new variant of Rasta, dubbed Masta. Similarly to Rasta, Masta takes as input a (master) secret key and a nonce, and generates a keystream block for each counter. On the other hand, Masta has two main differences from Rasta: Masta uses modular arithmetic to support HE schemes over a non-binary plaintext space, and it uses a smaller number of random bits in the affine layers by defining them with finite field multiplication. In this way, Masta outperforms Rasta in a transciphering framework with BGV/FV-style HE schemes. Our implementation shows that Masta is 505 to 592 times faster in terms of the throughput on the client-side, while 4792 to 6986 times faster on the server-side.

INDEX TERMS Homomorphic encryption, stream cipher, transciphering framework.

I. INTRODUCTION
Recently, effective manipulation of large amount of data has become one of the most important issues in IT-related industries. Smart devices measure and record immense data every day, and novel insights are being drawn from these data. However, in light of scalability, small companies have concerned the cost of server maintenance. Cloud computing might mitigate their scalability concerns by outsourcing data and computation instead of maintaining their own servers. However, cloud computing has risks in privacy and security including data exposure and misuse by the service provider. Conventional encryption schemes might efficiently address such security issues, while any computation over encrypted data would be impossible. If the encryption scheme is homomorphic, then the cloud would be able to perform meaningful computations on the encrypted data, supporting a wide range of applications such as machine learning over a large amount of encrypted data.

Homomorphic Encryption. An encryption scheme that enables addition and multiplication over encrypted data without decryption key is called a homomorphic encryption (HE) scheme. Since the emergence of Gentry's blueprint [30], there has been a large amount of research in this area [6], [8]–[11], [13], [15], [17]–[19], [26], [31]–[34], [43], [52]. Various applications of HE to medical, genomic, and financial tasks have also been proposed [7], [14], [16], [37], [40], [48].

Unfortunately, HE schemes commonly have two technical problems: speed and ciphertext expansion. The encryption/decryption time and the evaluation time of HE schemes are relatively slow compared to conventional encryption schemes. In particular, ciphertext expansion seems to be an intrinsic problem of homomorphic encryption due to the noise used in the encryption algorithm. Although the ciphertext expansion has been significantly reduced down to the order of hundreds in terms of the ratio of a ciphertext size to its plaintext size since the invention of the batching technique [32], it does not seem to be acceptable from a practical viewpoint. Furthermore, this ratio becomes even worse when it comes to encryption of a short message; encryption of a single bit might result in a ciphertext of a few megabytes.

Transciphering Framework. To address the issue of the ciphertext expansion and the client-side computational
overload, a hybrid framework, also called a transciphering framework, has been proposed [48]. In the client-server model, a client encrypts a message $m$ using a symmetric cipher $E$ with a secret key $k$; this secret key is also encrypted using an HE algorithm $Enc_{HE}$. The resulting ciphertexts $c = E_k(m)$ and $Enc_{HE}(k)$ are stored by the server.

When the server wants to compute $Enc_{HE}(m)$ (for computation over encrypted data), it will first compute $Enc_{HE}(c)$ for the corresponding ciphertext $c$. Then the server homomorphically evaluates $E^{-1}$ over $Enc_{HE}(c)$ and $Enc_{HE}(k)$, securely obtaining $Enc_{HE}(m)$.

Given a symmetric cipher with low multiplicative depth and complexity, this framework has the following advantages on the client side.

- A client does not need to encrypt all its data using an HE algorithm (except the symmetric key). All the data can be encrypted using only a symmetric cipher, significantly saving computational resources in terms of time and memory.
- Symmetric encryption does not result in ciphertext expansion, so the communication overload between the client and the server will be significantly low compared to using any homomorphic encryption scheme alone.

All these merits come at the cost of computational overload in the server side. That said, this trade-off would be worth considering in practice since servers are typically more powerful than clients.

**HE-friendly Cipher.** Symmetric ciphers are built on top of linear and non-linear primitives, and in a conventional environment, there have been no significant difference between the two types of primitives in terms of their implementation cost. However, when combined with BGV/FV-style HE schemes in a transciphering framework, the situation changes. Homomorphic addition is way cheaper than homomorphic multiplication in terms of computation time and noise growth.

With this observation, an HE-friendly cipher is evaluated by its multiplicative complexity and depth. In an arithmetic circuit, its multiplicative complexity is represented by the number of multiplications (ANDs in the binary case). Since homomorphic multiplication is much slower than homomorphic addition, smaller number of multiplications are desirable. Multiplicative depth is the depth of the tree that represents the arithmetic circuit, closely related to the noise growth in the HE-ciphertexts. These two metrics bring a new direction in the design of symmetric ciphers: to use simple non-linear layers at the cost of highly randomized linear layers.

**Rasta and Its Drawbacks.** The Rasta stream cipher [23] is one of the latest HE-friendly ciphers, which offers the fewest AND/bits and the lowest ANDdepth among the existing ciphers. It features an ASASA-like stream cipher, where affine layers are freshly generated for every encryption. Although this novel construction provides good results in terms of the cost metrics for HE-friendly ciphers, there are two drawbacks that limit its practical relevance.

The first drawback is the use of a binary plaintext space. Most of HE schemes operate on a specific class of rings, basing their security on the hardness assumption of the Ring Learning-With-Errors (RLWE) problem. The choice of the ring affects the overall security and efficiency of the HE scheme. For example, one can use a batching technique—an encoding technique enabling SIMD-like operations—only by carefully choosing the ring and the plaintext space. With a binary plaintext space, the maximum number of slots (i.e., bits which can be packed in a single ciphertext) is significantly less than the maximum number of slots for a non-binary case. In particular, the batching technique is not available in a power-of-two cyclotomic ring, which is one of the most widely-used rings in RLWE-based cryptography. Practically, it would lead to a loss of efficiency in the server-side computation.

The second drawback is that Rasta requires a heavy computational cost in the client-side due to the generation of a huge number of pseudorandom bits for its affine layers. To efficiently prevent algebraic attacks with a small number of rounds, Rasta generates fresh affine layers for every keystream generation. Numerically, a 128-bit secure Rasta (with $n = 525$ and $r = 5$) generates pseudorandom bits of 202 KB to encrypt a plaintext of 525 bits. It costs millions of cycles, increasing computational cost in the client-side.

### A. OUR CONTRIBUTION

The main contribution of this paper is to propose a new variant of Rasta, dubbed Masta. The Masta cipher takes as input a (master) secret key, a nonce, and generates a keystream. The design principle behind our construction is two-fold: using modular arithmetic to support HE schemes over a non-binary plaintext space, and reducing random bits used in the affine layers by defining them with finite field multiplication. In this way, we aim at improving its performance in a transciphering framework with BGV/FV-style HE schemes.

**Using Modular Arithmetic.** The plaintext space of Masta is given as $\mathbb{Z}_m^t$, where $t$ is chosen such that it allows full batching on the power-of-two cyclotomic ring of the HE scheme. The encryption of Masta is defined by using modular operations (such as multiplication and addition modulo $t$). This arithmetic makes Masta outperform Rasta, when combined with BGV/FV-style HE schemes. Based on comprehensive analysis of our non-binary cipher, we will carefully specify the parameters, and compare the performance of Masta to Rasta for such parameters.

**Reducing Randomness in the Affine Layers.** Constant multiplication in a finite field $\mathbb{F}_p$ can be represented as a linear map on $\mathbb{Z}_m^n$. In Masta, affine layers are generated by choosing a random element in $\mathbb{F}_p$. In this way, a single encryption requires only $O(\log M)$ pseudorandom bits, significantly smaller than $O(\log^2 M)$ for Rasta, where $M$ denotes the size of the plaintext space for each cipher. We will show that this technique does not degrade the security of the
cipher against conventional attacks such as differential and linear cryptanalysis as well as algebraic attacks.

**Implementation Results.** We implemented Masta on both the client-side (with arithmetics on plaintexts) and the server-side (with arithmetics on HE-ciphertexts), and compared it to Rasta. We put our focus on speeds in both environments, and the results are summarized in Table 1.

In this table, client-side latency and throughput refer to latency and reciprocal of throughput on the client-side environment, respectively. Server-side latency and throughput refer to latency and throughput on the server-side environment, respectively. Ratio refers to the ratio of the speed of Masta to that of Rasta, so that ratio larger than 1 implies that Masta is faster than Rasta. Overall, Masta outperforms Rasta over all the sets of parameters. We note that Masta has not been fully optimized in order to make a fair comparison of Masta to Rasta, implemented with its public source code, which is not a fully optimized version. We provide fully optimized benchmarks of Masta in Section IV.

### B. RELATED WORK

The transciphering framework has first been proposed in [48].

In this framework, the circuit of the AES block cipher is homomorphically evaluated [32]. This work was followed by the implementation of lightweight block ciphers SIMON [41] and PRINCE [25]. Since these ciphers have not been designed for the transciphering framework, the performance of the homomorphic evaluation was not satisfactory. In this line of research, low multiplicative complexity and depth become an important design principle, and LowMC is the first construction based on this design principle. However, it turned out that LowMC-80 and LowMC-128 are vulnerable to algebraic attacks and their variants [21, 24, 51].

Canteaut et al. claimed that stream ciphers might be advantageous in terms of online complexity compared to block ciphers, and proposed a new stream cipher Kreyvium [12]. However, its practical relevance is limited since the multiplicative depth (with respect to the secret key) keeps growing as keystreams are generated. A new stream cipher FLIP [47] is based on a novel design strategy that its permutation layer is randomly generated for every encryption without increasing the algebraic degree of the secret key. Rasta [23] is a stream cipher aiming at higher throughput at the cost of lower latency using random affine layers defined by an extendable output function (XOF).

### II. PRELIMINARIES

#### A. NOTATION

Throughout the paper, bold lowercase letters (resp. bold uppercase letters) denote vectors (resp. matrices). Usual dot product is denoted by \( \cdot \). For two vectors (strings) \( a \) and \( b \), their concatenation is denoted \( [a|b] \). For an integer \( q \), we identify \( \mathbb{Z}_q \) with \( \mathbb{Z} \cap \{ -q/2, q/2 \} \); for any integer \( z \), \( \lfloor z \rfloor_q \) denotes the mod \( q \) reduction of \( z \) into this interval. The notation \( [-]_q \) is extended to vectors (resp. polynomials) to denote their component-wise (resp. coefficient-wise) reduction. Throughout the paper, \( \xi \) denote a 2\( N \)-th primitive root of unity over the finite field \( \mathbb{Z}_t \), for fixed parameters \( N \) and \( t \). We assume that \( t \equiv 1 \pmod{2N} \).

For a set \( S \), we will write \( a \leftarrow S \) to denote that \( a \) is chosen from \( S \) uniformly at random. For a probability distribution \( D \), \( a \leftarrow D \) will denote that \( a \) is sampled according to the distribution \( D \). Unless stated otherwise, all logarithms are to the base 2.

#### B. BGV SCHEME

We will briefly review the BGV homomorphic encryption scheme with a special case: power-of-two cyclotomic ring. For more details, we refer to [9].

For a positive integer \( q \), an integer \( M \) which is a power of two, and \( N = M/2 \), BGV uses

\[
\mathcal{R}_q = \mathbb{Z}_q[X]/(\Phi_M(X))
\]

as its ciphertext space, where \( \Phi_M(X) = X^N + 1 \). We begin with its underlying hard problem.

**LWE and RLWE.** Let \( n \) and \( q \) be positive integers, and let \( D \) be a probability distribution over \( \mathbb{Z} \). For an unknown vector \( s \in \mathbb{Z}_q^n \), the LWE (Learning with Errors) distribution \( A_{n,q,D}(s) \) over \( \mathbb{Z}_q^n \times \mathbb{Z}_q \) is obtained by sampling a vector \( a \) uniformly at random from \( \mathbb{Z}_q^n \) and an error \( e \) according to \( D \), and outputting

\[
(a, b = [\langle a, s \rangle + e]_q) \in \mathbb{Z}_q^n \times \mathbb{Z}_q.
\]

The search-LWE problem is to find \( s \in \mathbb{Z}_q^n \) when independent samples \( (a_i, b_i) \) are obtained according to \( A_{n,q,D_i}(s) \). The decision-LWE problem is to distinguish the distribution \( A_{n,q,D}(s) \) from the uniform distribution over \( \mathbb{Z}_q^n \times \mathbb{Z}_q \).

Lyubashevsky et al. [45] introduced the ring version of the LWE problem, which is also called Ring-LWE (RLWE).

For a positive integer \( M \), let \( \Phi_M(X) \) be the \( M \)-th cyclotomic polynomial of degree \( N = \phi(M) \). Let \( \mathcal{R} = \mathbb{Z}[X]/(\Phi_M(X)) \) and let \( \mathcal{R}_q = \mathcal{R}/q\mathcal{R} = \mathbb{Z}_q[X]/(\Phi_M(X)) \). The (decisional) RLWE problem is to distinguish the distribution of \((a, b = [\langle a \cdot s + e \rangle_q]_q) \in \mathbb{R}_q^2 \) from the uniform distribution over \( \mathbb{R}_q^2 \), where \( s \in \mathcal{R} \) is a secret polynomial, \( a \) is sampled uniformly at random from \( \mathcal{R} \), and \( e \) is sampled according to a certain error distribution over \( \mathcal{R} \). The security of BGV is based on the hardness assumption of the RLWE problem.

**Encoding and Decoding.** The encoder \( \mathsf{Enc} : \mathbb{Z}_q^N \rightarrow \mathcal{R}_1 \) of the BGV scheme is the inverse of the decoder \( \mathsf{Dec} \), where

| Cipher | \( n \) | \( t \) | Client-side latency (cycles) | Client-side throughput (C/B) | Server-side latency (s) | Server-side throughput (bits/s) |
|--------|------|------|-----------------------------|-----------------------------|-------------------------|-----------------------------|
| Masta  | 5    | 16   | 58228                       | 1819.63                     | 54.38                   | 964.2                      |
|        | 64   | 327  | 37566084                   | 89066.95                    | 236.28                  | 1.38                       |
| Ratio  |      |      | 645.15                     | 505.07                      | 4.345                   | 6956.96                    |
|        | 728  | 7    | 76203                       | 2341.34                     | 96.15                   | 5511                       |
|        | 6    | 351  | 61959798                  | 1412102.06                  | 1626.01                 | 1.15                       |
| Ratio  |      |      | 813.06                     | 592.99                      | 19.19                   | 4792.17                    |

**TABLE 1.** Performance comparison between Masta and Rasta for some selected parameters. \( r \) is the number of rounds, \( n \) is the word-wise blocksize and \( t \) is the plaintext modulus.
for $p(X) \in \mathcal{R}_r$,
\[
\text{Dec}(p(X)) = [(p(a_0), \cdots, p(a_{N-1}))]_r \in \mathbb{Z}_q^N,
\]
where $a_i = \xi^{q-1} \pmod{t}$ for $0 \leq i \leq N - 1$.

**Key generation.** Given a security parameter $\lambda > 0$, fix integers $N$, $P$, and $q_0, \ldots, q_t$ such that $q_i$ divides $q_{i+1}$ for $0 \leq i \leq L - 1$, and distributions $\mathcal{D}_{\text{key}}$, $\mathcal{D}_{\text{err}}$ and $\mathcal{D}_{\text{enc}}$ over $\mathcal{R}$ in a way that the resulting scheme is secure against any adversary with computational resource of $O(2^\lambda)$.

1. Sample $a \leftarrow \mathcal{R}_{q_t}$, $s \leftarrow \mathcal{D}_{\text{key}}$, and $e \leftarrow \mathcal{D}_{\text{err}}$.
2. The secret key is defined as $sk = (s, 1) \in \mathcal{R}$, and the corresponding public key is defined as $pk = (a, b) \in \mathcal{R}_{q_t}^2$, where $b = [-a \cdot s + te]_{q_l}$.

**Encryption.** Given a public key $pk$ and a plaintext $m \in \mathcal{R}_t$, $\mathcal{D}_{\text{enc}}$ and $e_0, e_1 \leftarrow \mathcal{D}_{\text{err}}$.

1. Sample $r \leftarrow \mathcal{D}_{\text{enc}}$ and $e_0, e_1 \leftarrow \mathcal{D}_{\text{err}}$.
2. Compute $\text{Enc}(pk, 0) = [r \cdot pk + t(e_0, e_1)]_{q_t}$.

- $\text{Enc}(pk, m) = [\text{Enc}(pk, 0) + (0, m)]_{q_l}$.

**Decryption.** Given a secret key $sk \in \mathcal{R}$ and a ciphertext $ct \in \mathcal{R}_{q_t}^2$,
\[
\text{Dec}(sk, ct) = [(sk, ct)]_{q_l}.\]

**Addition.** Given ciphertexts $ct_1$ and $ct_2$ in $\mathcal{R}_{q_t}^2$, their sum is defined as
\[
ct_{\text{add}} = [ct_1 + ct_2]_{q_t}.
\]

**Multiplication.** Given ciphertexts $ct_1 = (a_1, b_1)$ and $ct_2 = (a_2, b_2)$ in $\mathcal{R}_{q_t}^2$, their product is defined as
\[
ct_{\text{mult}} = [(a_1a_2, a_1b_2 + a_2b_1, b_1b_2)]_{q_t}.
\]

After the basic multiplication, the resulting ciphertext has 3 ring elements. For fully functional BGV, two additional procedures are required. The first procedure is key switching, which reduce the ciphertext to have 2 ring elements. The second procedure is modulus switching. After multiplication, errors in ciphertext is multiplied by $t^2$. Modulus switching reduce the multiplier back again. For the details, we refer the reader to [9].

### C. RASTA

The Rasta stream cipher of $r$ rounds, denoted Rasta', takes as input a secret key $k \in \mathbb{Z}_2^n$ and a nonce $nc$, and returns a keystream block $k_{nc} \in \mathbb{Z}_2^n$, where $nc$ is fed to an extendable output function XOF, and the output binary sequence from XOF defines invertible affine layers $\text{Affine}_i$, $i = 0, \ldots, r$.

In the non-linear layers of Rasta, the $\chi$-transformation [20] is used. So Rasta computes its keystream block by applying $\text{Affine}_i$ and $\chi$ in an interleaved manner, followed by the final key addition step, denoted $\text{AddKey}$ (see Figure 1):

| Rasta'[k, nc] = AddKey[k] \circ \text{Affine}_r[nc] \circ \chi \circ \cdots \circ \chi \circ \text{Affine}_0[nc]. |

A primitive root of unity $\xi$ exists if the characteristic $t$ of the message space is an odd prime such that $t = 1 \pmod{M}$.

By using internal counters, multiple keystream blocks can be generated as follows. An $\ell$-block keystream $z = z[0]|z[1]|\ldots|z[\ell - 1]$ is defined by
\[
z[i] = \text{Rasta'}[k, nc][i](k)
\]
for $i = 0, \ldots, \ell - 1$.

**Affine Layers.** Each affine layer $\text{Affine}_i$, $i = 0, \ldots, r$, consists of an $n \times n$ binary matrix $A^{(i)}$ followed by constant addition by $b^{(i)} \in \mathbb{Z}_2^n$.

\[
\text{Affine}_i(nc)(x) = A^{(i)}x \oplus b^{(i)},
\]
for $x \in \mathbb{Z}_2^n$. All the bits of $A^{(i)}$s and $b^{(i)}$s are defined by the sequence from the extendable output function XOF using $nc$.

**Non-linear Layers.** Rasta uses a single non-linear layer of the $\chi$-transformation [20]. Let $x = (x_1, \ldots, x_n)$, where $x_i \in \mathbb{Z}_2$ for $i = 1, 2, \ldots, n$. Then the $i$-th component of $\chi(x)$, denoted $\chi(x)$, is defined as follows.

\[
\chi(x)_i = x_i \oplus x_{i+2} \oplus x_{i+4} \oplus x_{i+6}
\]
for $i = 1, 2, \ldots, n$, taking the indices modulo $n$.

**Final Key Addition.** The secret key $k \in \mathbb{Z}_2^n$ is added to the output from the final affine layer. This step can be defined as follows.

\[
\text{AddKey}[k](x) = x \oplus k
\]
for $x \in \mathbb{Z}_2^n$.

### III. A NEW STREAM CIPHER: MASTA

In this section, we propose a $\mathbb{Z}_2$-variant of Rasta [23], dubbed Masta, and analyze its security.

### A. DESIGN RATIONALE

**Why Non-binary Rings?** Server-side efficiency heavily depends on careful selection of HE-parameters since the server will spend most of the time in homomorphic evaluation. A homomorphic encryption scheme is typically based on an RLWE problem over a ring $\mathcal{R}/q\mathcal{R}$, where $\mathcal{R} = \mathbb{Z}[X]/(f(X))$ and $f(X)$ is a monic irreducible polynomial over $\mathbb{Z}$. If $f(x)$ is a cyclotomic polynomial whose degree is a power of two, then $\mathcal{R}/q\mathcal{R}$ is called a power-of-two cyclotomic ring, and such rings are widely used in

![FIGURE 1. The r-round Rasta construction. Operations in the box with dotted (resp. thick) lines are public (resp. secret).](image)
TABLE 2. Detailed performance of affine layer generation in full-round Rasta using the AVX2-optimized SHAKE256 algorithm. When generating affine layers, lower and upper triangular (invertible) matrices are generated and multiplied together to make uniformly random invertible matrices. We check the time spent in initialization of SHAKE256, generation of LU matrices, multiplication, generation of round constants, and encryption. All the units for time are the number of cycles.

| Cipher | r | n | t | Initialization | LU matrices generation | LU matrix multiplication | Round constant generation | Encryption | Total time |
|--------|---|---|---|---------------|-------------------------|-------------------------|--------------------------|------------|------------|
| 80-bit |   |   |   |               |                         |                         |                          |            |            |
| Rasta  | 4 | 327 | 2 | 4382          | 12058527                | 19037397                | 36846                    | 35301      | 31202453   |
| Rasta  | 6 | 219 | 2 | 4379          | 7603284                 | 10672950                | 34544                    | 53448      | 18368605   |
| 128-bit|   |   |   |               |                         |                         |                          |            |            |
| Rasta  | 5 | 525 | 2 | 4470          | 37191471                | 70361948                | 75358                    | 191270     | 107793217  |
| Rasta  | 6 | 351 | 2 | 4468          | 19142428                | 31312877                | 54941                    | 109202     | 50616634   |

RLWE-based schemes due to their merits in efficiency and security (e.g., [2], [5], [9], [27], [29]).

A plaintext space \( \mathbb{Z}_q^n \) and a modulus polynomial \( f(X) \) determine the number of distinct roots in \( \mathbb{Z}_q \), and hence the number of slots in the HE scheme. However, over a power-of-two cyclotomic ring with characteristic two (i.e., \( t = 2 \)), one cannot use the batching technique since

\[
X^N + 1 \equiv (X + 1)^N \pmod{2}.
\]

For this reason, we opted for non-binary rings and the corresponding modular operations in the design of our cipher.

**Generating Affine Layers with Smaller Randomness.**

In Rasta, there are two major bottlenecks with respect to its efficiency: to generate large \( n \times n \) binary matrices and to check if they are invertible (or in a better way, one can generate lower and upper triangular matrices and multiply them). This part amounts to over 99\% of the total encryption time as shown in Table 2.

In order to address this issue, we identify the plaintext space of Masta with a finite field \( \mathbb{F}_r^n \), and define each affine layer by choosing a random nonzero element from \( \mathbb{F}_r^n \) and multiplying it to the state. Any nonzero element ensures the invertibility of the affine layer. This approach quadratically speeds up the generation of affine layers. By sufficient cryptanalysis, we show that weak randomness used in our design does not significantly degrade the overall security of our cipher.

**B. SPECIFICATION**

The Masta stream cipher of \( r \) rounds, denoted Masta\(^r\), takes as input a secret key \( k \in \mathbb{Z}_q^n \) and a nonce \( nc \), and returns a keystream block \( k_{nc} \in \mathbb{F}_q^n \), where \( nc \) is fed to an extendable output function XOF, and the output binary sequence from XOF defines invertible affine layers Affine\(i_0\), \( i = 0, \ldots, r \). So the structure of Masta is similar to Rasta; Masta computes its keystream block by applying Affine; and the \( \chi \)-transformation in an interleaved manner, followed by the final key addition step AddKey:

\[
\text{Masta}^r[k, nc] = \text{AddKey}[k] \circ \text{Affine}_0[nc] \circ \chi \cdots \circ \chi \circ \text{Affine}_0[nc].
\]

By using internal counters, multiple keystream blocks can be generated as follows. An \( \ell \)-block keystream \( z = z[0] || z[1] || \ldots || z[\ell - 1] \) is defined by

\[
z[i] = \text{Masta}^r[k, nc][i](k)
\]

for \( i = 0, \ldots, \ell - 1 \).

**Affine Layers.** An extendable output function XOF will be used in the generation of Masta. In practice, it can be instantiated with AES in the counter mode or a sponge-type hash function. Since XOF does not use any secret key, it is allowed to have a high algebraic degree; it is the degree of the cipher with respect to \( k \) that affects the overall performance of the cipher in the hybrid framework. We also note that the sequence from XOF can be precomputed in the client side.

Each affine layer is decomposed as Affine\(i_0 = \text{ARC}_i \circ \text{FMul}_i \), \( i = 0, 1, \ldots, r \), each of which represents a linear transformation \( A^{(i)} : \mathbb{Z}_q^n \rightarrow \mathbb{Z}_q^n \) and constant addition by \( b^{(i)} \in \mathbb{Z}_q^n \), respectively. Namely,

\[
\begin{align*}
\text{FMul}_i(x) &= A^{(i)}(x), \\
\text{ARC}_i(x) &= x + b^{(i)}, \\
\text{Affine}_i(x) &= A^{(i)}(x) + b^{(i)},
\end{align*}
\]

for \( x \in \mathbb{Z}_q^n \) and \( i = 0, 1, \ldots, r \). Similarly to Rasta, both \( A^{(0)} \) and \( b^{(0)} \) are generated by XOF, while, for Masta, \( A^{(0)} \) is defined as finite field multiplication by a random element \( a^{(0)} \) of \( \mathbb{F}_q^n \), being identified with \( \mathbb{Z}_q^n \), i.e., \( A^{(0)}(x) = a^{(0)} \cdot x \) (see Figure 2).

The finite field \( \mathbb{F}_q^n \) is constructed by extending \( \mathbb{F}_q \) using an irreducible polynomial \( f(X) \) of degree \( n \) over \( \mathbb{F}_q \). In order to make the field multiplication as simple as possible, we will use an irreducible polynomial with the smallest number of nonzero coefficients, namely, \( f(X) = X^n - \alpha \) for some \( \alpha \in \mathbb{Z}_q \). The following theorem guarantees existence of such irreducible polynomials.

**Theorem 1 (Theorem 3.75 in [42]):** For a prime power \( q \), let \( \alpha \in \mathbb{F}_q^\times \) with order \( e \). Then \( X^n - \alpha \) is irreducible in \( \mathbb{F}_q[X] \) if and only if the integer \( n \geq 2 \) satisfies the following conditions:

(i) each prime factor of \( n \) divides \( e \), but not \( (q - 1)/e \),

(ii) if \( 4 \) divides \( n \) then \( 4 \) divides \( q - 1 \).

Let \( t \) be a prime of the form \( 2m + 1 \) for some positive integer \( m \), and let \( n \) be a power-of-two. Then a generator \( \alpha \) of a multiplicative group \( \mathbb{Z}_q^\times \) satisfies all the above conditions.

Once \( \mathbb{Z}_q^n \) and \( \mathbb{F}_q^n \) are identified with respect to an irreducible polynomial \( f(X) = X^n - \alpha \), field multiplication by \( a = (a_1, a_2, \ldots, a_n) \) can be represented by an \( n \times n \) matrix
Let $\chi \in \mathbb{Z}_t$ element, and add 1 to the element modulo our implementation), then one can randomly choose a 16-bit the vectors independent.

For simplicity of analysis, we will assume that all entries of the vectors $a(i)$'s and $b(i)$'s are defined by the sequence from XOF, so one can assume that all the vectors are independent at random over the rounds. In this paper, we will not specify any correspondence between the pseudorandom sequence and the affine layers; all the analyses are based on the assumption that all the vectors are truly random and independent.

For simplicity of analysis, we will assume that all entries of the vectors $a(i)$ and $b(i)$ are nonzero; if $t = 2^{16} + 1$ (as used in our implementation), then one can randomly choose a 16-bit element, and add 1 to the element modulo $t$. The result will always be a nonzero element of $\mathbb{Z}_t$.

**Non-linear Layers.** Masta uses a $\mathbb{Z}_t$-variant of the $\chi$-transformation [20] as a single non-linear layer. Let $x = (x_1, \ldots, x_n)$, where $x_i \in \mathbb{Z}_t$ for $i = 1, 2, \ldots, n$. Then the $i$-th component of the secret key $k \in \mathbb{Z}_t^n$. Since multiplication is more expensive than addition in HE schemes, most HE-friendly ciphers have been designed to have a low multiplicative depth. This property might possibly make such ciphers vulnerable to algebraic attacks. Indeed, some of recent constructions have been analyzed by algebraic attacks due to their low algebraic depth [1], [21], [28]. In this section, we will consider two different types of algebraic attacks: trivial linearization and the Gröbner basis attack.

**Trivial Linearization.** Trivial linearization is to make the system of polynomial equations linear by replacing all monomials by new variables. When the cipher is represented by a system of polynomial equations of degree $d$ over $\mathbb{Z}_t$ in $n$ unknowns (and $d < t$), the number of monomials appearing in this system is upper bounded by

$$S = \sum_{i=0}^{d} \binom{n + i - 1}{i}. $$

Therefore, at most $S$ equations will be enough to solve this system of equations. If the system is sparse, then it would require less equations to solve the system. We will show that almost all the monomials appear after $r$ rounds of Masta, and hence this attack requires $O(S)$ data and $O(S^\omega)$ time, where $\omega$ is the linear algebra constant such that $2 \leq \omega \leq 3$.

An adversary might also try the guess and determine attack before trivial linearization. By guessing $g$ variables, the number of possible monomials is reduced down to

$$S_g = \sum_{i=0}^{d} \binom{n - g + i - 1}{i}. $$

This approach will be useful in particular when almost every monomial appears in the system. In this case, the overall time complexity becomes $O(t^5 S_g^\omega)$.

**Gröbner Basis Attack.** The Gröbner basis attack is to solve a system of equations by computing a Gröbner basis of the system. If such a Gröbner basis is found, then the variables can be eliminated one by one. Gröbner basis can be computed with low data unlike the trivial linearization. However, its computation is slower than the trivial linearization with a small amount of data. For this reason, the Gröbner basis
attack will be useful (compared to the trivial linearization) when either the data is limited or the number of monomials grows faster than the number of equations. For an example of the second case, we refer to [1]. From the perspective of the cipher design, it is enough to set the parameters such that $O(S^n)$ is large enough.

**Number of Monomials.** If the number of monomials is small, then algebraic attacks may work well with low data and time complexities. Here, we estimate the average number of monomials appearing in an $r$-round Masta.

Consider a single round $\chi \circ \text{Affine}(x)$ of Masta with input $x = (x_1, x_2, \ldots, x_n)$ and $\text{Affine}(x) = a \cdot x + b$. Finite field multiplication by $a = (a_1, a_2, \ldots, a_n)$ is equivalent to multiplication by an $n \times n$ matrix $L$ as defined in (1). Let $L_{i,j}$ be the entry in the $i$-th row and the $j$-th column of the matrix $L$. Then, the $i$-th component of the output $\chi \circ \text{Affine}(x_i)$ is given as

$$
\chi \circ \text{Affine}(x_i) = \sum_{j=1}^{n} u_{i,j}^2 \cdot x_i^2 + \sum_{j=1}^{n} \sum_{i,j,l} v_{i,j,l}^2 \cdot x_i x_l + \sum_{j=1}^{n} w_{i,j}^2 \cdot x_j + g(i),
$$

where

$$
\begin{align*}
 u_{i,j}^2 & = L_{i+1,j} \cdot L_{i+2,j}, \\
 v_{i,j,l} & = L_{i+1,j} \cdot L_{i+2,l} + L_{i+2,j} \cdot L_{i+1,l}, \\
 w_{i,j} & = \alpha + b_{i+2} \cdot L_{i+1,j} + (1 + b_{i+1}) \cdot L_{i+2,j}, \\
 g(i) & = b_i + b_{i+2} + b_{i+1} \cdot b_{i+2}.
\end{align*}
$$

Since every $a_j$ is nonzero, $u_{i,j}^2 \neq 0$ for every $i = 1, 2, \ldots, n$. This implies that every monomial $x_i^2$ appears in the output $\chi \circ \text{Affine}(x)$.

When it comes to a monomial $x_i x_j$ of degree two with $j < i$, it does not appear in the output if and only if $v_{i,j,l} = 0$ for every $i = 1, 2, \ldots, n$. This is equivalent to the following system of equations:

$$
\begin{align*}
a_{j-1} a_n & + a_{j-1} a_n = 0 \\
a_{j-1} a_n & + a_{j-1} a_n = 0 \\
\vdots & \\
a_{n-i} a_{n-i} + a_{n-i} a_{n-i-1} & = 0 \\
\alpha \cdot a_{n-i} a_{n-i+1} + a_{n-i} a_{n-i+1} & = 0 \\
a_1 a_{n-i+2} + a_2 a_{n-i+1} & = 0 \\
a_2 a_{n-i+3} + a_3 a_{n-i+2} & = 0 \\
\vdots & \\
a_1 a_{n-i+1} + a_{n-i+1} a_n & = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0 \\
a_{n-i} a_n & + a_{n-i} a_n = 0
\end{align*}
$$

Once $a_{n-j}$ and $a_{n-l} \in \mathbb{Z}_n^*$ are fixed, $a_{n-j+1}$ is uniquely determined by $a_{n-j+1}$ in the first equation. Given $a_{n-j+1}$ and $a_{n-l+1}$, $a_{n-j+2}$ is uniquely determined by $a_{n-l+2}$ in the second equation. By repeating this process, one will find the smallest $k$ such that every $n-l+k = n-j$ or $n-l+k = n-l \mod n$. From here, each equation uniquely determines the remaining one variable if any contradiction does not occur. Therefore, there are at most $(t - 1)^2 + \min(l-1, n-j-l-1) \leq (t - 1)^2 + n/2$ solutions and hence,

$$
\Pr \left[ \forall i, w_{i,j} = 0 \right] \leq \frac{1}{(t - 1)^{n/2-2}},
$$

which is close to 0 for sufficiently large $n$ and $t$. In other words, all quadratic terms appear after a single round of Masta except with a negligible probability.

A monomial $x_j$ does not appear in the output if and only if $w_{i,j} = 0$ for every $i = 1, 2, \ldots, n$, which is equivalent to the following system of equations:

$$
\begin{align*}
\alpha \cdot a_{n-j+1} + \alpha \cdot b_{3} a_{n-j+2} + \alpha \cdot b_{j} a_{n-j+3} & = 0 \\
\alpha \cdot a_{n-j+2} + \alpha \cdot b_{4} a_{n-j+3} + \alpha \cdot b_{j} a_{n-j+4} & = 0 \\
\vdots & \\
\alpha \cdot a_{n-2} + \alpha \cdot b_{j} a_{n-1} + \alpha \cdot b_{j}' a_{n} & = 0 \\
\alpha \cdot a_{n-1} + b_{1} a_{n} + b_{2} a_{n-j+1} & = 0 \\
\alpha \cdot a_{n} + b_{j+2} a_{n} + b_{j+2} a_{n-j+2} & = 0 \\
\alpha \cdot a_{n} + b_{j+3} a_{n} + b_{j+3} a_{n-j+3} & = 0 \\
\vdots & \\
\alpha \cdot a_{n-1} + b_{n-j+2} a_{n-j+1} + b_{n-j+2} a_{n-j+2} & = 0
\end{align*}
$$

where $b_{j}' = b_{j} + 1$ for $j = 1, 2, \ldots, n$. Once $a_{n-j+1}$ and $a_{n-j+2}$ are fixed, all monomials appear after a single round except with a negligible probability. Since all the monomials of degree at most 2 appear after a single round with overwhelming probability, one can conclude that almost all monomials of degree at most $2^r$ appear after $r$ rounds of Masta.
### Table 3. Parameter $n$ secure against algebraic attacks when $t = 2^{16} + 1.$

| Security (bit) | 80  | 128 | 192 | 256 |
|---------------|-----|-----|-----|-----|
| Round         |     |     |     |     |
| 4             | 32  | 128 | 512 | 2048|
| 5             | 16  | 64  | 128 | 256 |
| 6             | 16  | 32  | 64  | 128 |
| 7             | 8   | 16  | 32  | 64  |

**Parameters.** With respect to the algebraic attacks, recommended parameters are given in Table 3 for various security levels. These parameters have been computed using the estimation of $S^\omega$ with $\omega = 2$. On the other hand, guessing variables will not affect the security of Masta when $t = 2^{16} + 1$. For $t = 2^{16} + 1$, we can use $\alpha = 3$ as the smallest generator of the multiplicative group $\mathbb{Z}_t^\times$.

#### 2) LINEAR ATTACKS

Linear cryptanalysis [46] is based on a linear approximation of a fixed (keyed) permutation using a sufficient number of plaintext-ciphertext pairs, while this attack does not apply to Masta since this cipher uses independent random affine layers for every encryption.

In [23], a variant of the linear attack has been considered to be applied to Rasta; an adversary constructs an overdefined system of linear approximations that are valid with a certain probability, and recovers the secret key by using the LPN-problem solver. Considering this attack, the block size and the algebraic depth of Rasta have been newly estimated for each security level.

One might want to apply this type of attack to Masta (operating on a vector space $\mathbb{Z}_t^\ell$) by using an LWE-problem solver instead of an LPN-problem solver. However, in the LWE-problem, an error $e$ is sampled according to the Gaussian distribution, while the error introduced in the linear approximations of Masta will follow the uniform random distribution. For this reason, a straightforward application of the LWE-problem solver does not seem to work.

#### 3) DIFFERENTIAL ATTACKS

The resistance of a substitution-permutation-cipher to differential cryptanalysis is typically estimated by analyzing the maximum probability of differential trails [4]. Such a trail is detected by obtaining ciphertexts for multiple pairs of plaintexts with a fixed difference. However, the affine layers of Masta are generated independently at random for every encryption. For this reason, similarly to Rasta, the classical differential attack will not be applicable to Masta.

#### 4) OTHER ATTACKS

There are many variants of differential and linear attacks such as truncated differential attacks, higher-order differential attacks, impossible differential attacks, boomerang attacks and differential-linear attacks [3], [38], [39], [53]. Cube attacks and interpolation attacks can be seen as variants of algebraic attacks and integral attacks, respectively [22], [36], [44].

All these attacks are based on the evaluation of a fixed permutation on a large number of plaintexts and ciphertexts. However, Masta uses different affine layers for every encryption, so such attacks would not be applicable to Masta.

### IV. IMPLEMENTATION

In this section, we implement Masta and Rasta, and compare their performance. For selected parameter sets, we evaluate their performance in terms of speed on both the client-side and the server-side.

Our source codes are developed in C++17 with HElib ver. 1. 0. 2 [35]. Our experiments are done in AMD Ryzen 7 2700X @ 3. 70 GHz single-threaded with 64 GB memory, using GNU C++ 7. 5. 0 compiler. XOF is instantiated with SHAKE256 [49] in XKCP [54]. In our tables, we will use the following parameters.

- $r$: the number of rounds;
- $n$: the blocksize;
- $t$: the plaintext modulus;
- $q$: the ciphertext modulus of the BGV schemes at first;
- $N$: the degree of the polynomial modulus in the BGV schemes;
- $\ell$: the multi-dimensional array structure of the slot in a BGV plaintext;
- $\lambda'$: the security level of the BGV schemes with given parameters.

#### A. BENCHMARKS

**Client-side Computation.** Comparison between Masta and Rasta on the client-side is summarized in Table 4. The source code of Rasta is brought from the github repository [50] which is officially written in the paper [23].

The total time consists of the time for affine layer generation and the encryption time, all measured in the number of cycles. All the affine layers and the round constants are generated by SHAKE256. The encryption time covers all the computation except generation of affine layers. Throughput is the total time divided by the number of bits per block (i.e., $n \log t$), measured in cycles per byte. ‘Optimization level’ specifies which part has been optimized. The first one is for SHAKE256, where ‘AVX2’ stands for the use of AVX2-optimized code in XKCP while ‘Naive’ stands for the use of SHAKE256 code in Rasta github repository (which is not AVX2-optimized). The second one is for the encryption part, where ‘Opt’ stands for a fully optimized encryption code (including AVX2) and ‘Naive’ stands for a non-optimized code. For Rasta, we use the code in the repository as it is (without any further optimization).

Our result shows that Masta outperforms Rasta in any set of parameters. For a “fair” comparison with the “Naive-Naive” optimization level in the 80-bit security level, the latency of Masta is 214 to 645 times faster than Rasta, and the throughput is 443 to 571 times faster. When it comes to fully optimized codes, we can find the benchmark...
TABLE 4. Performance of Masta and Rasta on the client-side.

| Cipher | $r$ | $n$ | $t$ | $\log q$ | $N$ | $\ell$ | $\lambda'$ | Affine (cycles) | Encryption (cycles) | Total time (cycles) | Throughput (C/B) | Optimization level |
|--------|-----|-----|-----|----------|-----|-------|-----------|---------------|-------------------|-------------------|----------------|------------------|
| 80-bit |
| Masta 4 | 32 | 65537 | 80965 | 22040 | 103005 | 1609.45 | Naïve - Naïve |
| Masta 5 | 16 | 65537 | 12333 | 3667 | 16000 | 250 | AVX2 - Opt |
| Masta 5 | 16 | 65537 | 48794 | 9434 | 58228 | 1819.63 | Naïve - Naïve |
| Masta 5 | 16 | 65537 | 7408 | 1519 | 8927 | 278.97 | AVX2 - Opt |
| Rasta 4 | 327 | 2 | 37500346 | 65698 | 37566044 | 191046.95 | Naïve - Naïve |
| Rasta 4 | 327 | 2 | 59380357 | 65301 | 32000878 | 782896.10 | AVX2 - Naïve |
| Rasta 4 | 327 | 2 | 22021067 | 53038 | 22074105 | 806360.00 | AVX2 - Naïve |
| Rasta 6 | 219 | 2 | 18331024 | 53448 | 18384472 | 671578.89 | AVX2 - Naïve |

128-bit

| Cipher | $r$ | $n$ | $t$ | $\log q$ | $N$ | $\ell$ | $\lambda'$ | Affine (cycles) | Encryption (cycles) | Total time (cycles) | Throughput (C/B) | Optimization level |
|--------|-----|-----|-----|----------|-----|-------|-----------|---------------|-------------------|-------------------|----------------|------------------|
| Masta 6 | 32 | 65537 | 110718 | 24857 | 135575 | 2118.36 | Naïve - Naïve |
| Masta 6 | 32 | 65537 | 17117 | 4891 | 22008 | 343.88 | AVX2 - Opt |
| Masta 7 | 16 | 65537 | 65922 | 12281 | 76203 | 2381.34 | Naïve - Naïve |
| Masta 7 | 16 | 65537 | 7971 | 1881 | 11672 | 364.75 | AVX2 - Opt |
| Rasta 5 | 525 | 2 | 128500430 | 167992 | 128668422 | 1950661.67 | Naïve - Naïve |
| Rasta 5 | 525 | 2 | 110215322 | 161270 | 110376592 | 1681929.02 | AVX2 - Naïve |
| Rasta 6 | 351 | 2 | 61853372 | 102606 | 61959578 | 1411202.06 | AVX2 - Naïve |
| Rasta 6 | 351 | 2 | 51975234 | 101920 | 52077154 | 1186943.68 | AVX2 - Naïve |

1. In the tables, the columns ‘Affine’ and ‘Non-linear’ refer to the total time to evaluate affine and non-linear layers, respectively. Both affine and non-linear layers are implemented using addition, multiplication and rotation. The column ‘Eval’ refers to the total time to evaluate the given cipher, recording 900,000 cycles for 219-bit 6-round Rasta (with the 80-bit security level) in [23]. Although microarchitectures are different, we can simply compare it to 16-word 5-round Masta, which is about 100 times faster than Rasta in latency.

Server-side Computation. The server-side is implemented by using the HElib library. The performance of Masta and Rasta is summarized in Table 5: (a) has tight BGV parameters for evaluating Masta and Rasta, and (b) has BGV parameters allowing additional 7 multiplications after evaluating Masta and Rasta, as considered in [23].

As mentioned in the design rationale of Masta, power-of-two cyclotomic rings are used in our implementation of Masta. The ciphertext modulus $q$ is chosen to guarantee at least a security level of $\lambda'$ bits and enough noise capacity to evaluate Masta. In case of Rasta, the slot structure $\ell$ is also important in the choice of the BGV parameters. When $\ell$ is $s \times d$, $\lfloor s/n \rfloor \times d$ blocks are packed in each BGV plaintext, so $s$ should be at least $n$.2

In the tables, the columns ‘Affine’ and ‘Non-linear’ refer to the total time to evaluate affine and non-linear layers, respectively. Both affine and non-linear layers are implemented using addition, multiplication and rotation. The column ‘Eval’ refers to the total time to evaluate the given cipher.

2. In this paper, we consider slot structures of at most 2 dimensions.
including time not only for affine and non-linear layers but also for the final key addition.

Our implementation shows that Masta is also better than Rasta in homomorphic evaluation. For the BGV parameters with tight capacity, the throughput of Masta is 3729 to 11758 times larger than Rasta with the 80-bit security level, while 3317 to 8531 times larger with the 128-bit security level. For the BGV parameters with additional capacity, the throughput of Masta is 3806 to 9608 times larger than Rasta.

V. CONCLUSION
In this paper, we proposed a new HE-friendly cipher Masta by modifying the Rasta cipher. Masta uses modular arithmetic in its encryption in order to support HE schemes over a non-binary plaintext space; it allows a larger number of slots and smaller number of multiplication per bits on the server side. The affine layers of Masta are defined by finite field multiplication, which requires a smaller number of random bits compared to Rasta. By comprehensive analysis, we showed that this modification does not degrade the overall security of the cipher. We also implemented Masta and Rasta on the client and the server side to compare their performance. Compared to Rasta, Masta turns out to be hundreds of times faster on the client-side, and thousands of times faster on the server-side.

REFERENCES
[1] R. Martin Albrecht, C. Cid, L. Grassi, D. Khovratovich, R. Lüftenegger, C. Rechberger, and M. Schönfegger, “Algebraic cryptanalysis of STARK-friendly designs: Application to MARVELLous and MiMC,” in Advances in Cryptology-ASIACRYPT, vol. 1192, S. D. Galbraith and S. Moriai, Eds. Cham, Switzerland: Springer, 2013, pp. 371–379.
[2] E. Alkim, L. Ducas, T. Poppelmann, and P. Schwabe, “Post-quantum key exchange: A new hope,” in Proc. SEC, 2016, New York, NY, USA, 2016, pp. 327–343.
[3] E. Biham, A. Biryukov, and A. Shamir, “Miss in the Middle Attacks on IDEA and Khufu,” in Fast Software Encryption, vol. 1636, L. Knudsen, Ed. Berlin, Germany: Springer, 1999, pp. 124–138.
[4] E. Biham and A. Shamir, “Differential cryptanalysis of DES-like cryptosystems,” in Advances in Cryptology-CRYPTO, vol. 537, A. J. Menezes and Scott A. Vanstone, Eds. Berlin, Germany: Springer, 1991, pp. 2–21.
[5] J. Bos, L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, J. M. Schanck, P. Schwabe, G. Seiler, and D. Stehle, “CRYSTALS-kyber: A CCA-secure Module-Lattice-Based KEM,” in Proc. IEEE Euro. Symp. Secur. Privacy (EuroS&P), Apr. 2018, pp. 353–367.
[6] W. Joppe Bos, K. Lauter, J. Loftus, and M. Naehrig, “Improved security for a ring-based fully homomorphic encryption scheme,” in Cryptography and Coding (IMACC), vol. 8308, M. Stam, Ed. London, U.K.: Springer, 2013, pp. 45–64.
[7] J. W. Bos, K. Lauter, and M. Naehrig, “Private predictive analysis on encrypted medical data,” J. Biomed. Inform., vol. 50, pp. 234–243, Aug. 2014.
[8] Z. Brakerski, “Fully homomorphic encryption without modulus switching from classical GapsVP,” in Advances in Cryptology-CRYPTO, vol. 7417, R. Safavi-Naini and R. Canetti, Eds. Berlin, Germany: Springer, 2012, pp. 886–886.
[9] Z. Brakerski, C. Gentry, and V. Vaikuntanathan, “(Leveled) fully homomorphic encryption without bootstrapping,” in Proc. 3rd Innov. Theor. Comput. Sci. Conf. (ITCS), 2012, p. 309.
[10] Z. Brakerski and V. Vaikuntanathan, “Fully homomorphic encryption from ring-LWE and security for key dependent messages,” in Proc. 43rd Annu. ACM Symp. Symp. Theory Comput. (STOC), Bethesda, MD, USA, 2009, pp. 169–178.
[11] Z. Brakerski and V. Vaikuntanathan, “Lattice-based FHE as secure as PKE,” in Proc. 5th Conf. Innov. Theor. Comput. Sci. (ITCS), 2014, pp. 1–12.
[12] A. Canteaut, S. Carpo, C. Fontaine, T. Lepoint, M. Naya-Plasencia, P. Paillier, and R. Sirius, “Stream ciphers: A practical solution for efficient homomorphic ciphertext compression,” J. Cryptol., vol. 31, no. 3, pp. 885–916, Jul. 2018.
[13] J. H. Cheon, J.-S. Coron, J. Kim, M. S. Lee, T. Lepoint, M. Tibouchi, and A. Yun, “Batch Fully Homomorphic Encryption over the Integers,” in Advances in Cryptology–EUROCRYPT, vol. 7881, T. Johansson and P. Q. Nguyen, Eds. Berlin, Germany: Springer, 2013, pp. 315–335.
[14] J. H. Cheon, J. Jeong, J. Lee, and K. Lee, “Privacy-preserving computations of predictive medical models with minimax approximation and non-adapted form,” in Financial Cryptography and Data Security, vol. 10323, M. Brenner, K. Rohloff, J. Boneau, A. Miller, P. Y. A. Ryan, V. Teague, A. Bracciali, M. Sala, F. Pintore, M. Jakobsson, Eds. Cham, Switzerland: Springer, 2017, pp. 53–74.
[15] J. H. Cheon, J. Kim, M. S. Lee, and A. Yun, “CRT-based fully homomorphic encryption over the integers,” Int. J. Info. Sci., vol. 310, pp. 149–162, Jul. 2015.
[16] J. H. Cheon, M. Kim, and K. Lauter, “Homomorphic computation of edit distance,” in Proc. Int. Conf. Financial Cryptogr. Data Secur. Berlin, Germany: Springer, 2015, pp. 194–212.
[17] J. H. Cheon and D. Stehle, “Fully homomorphic encryption over the integers revisited,” in Advances in Cryptology–EUROCRYPT, vol. 9056, E. Oswald and M. Fischlin, Eds. Berlin, Germany: Springer, 2015, pp. 513–536.
[18] J.-S. Coron, A. Mandal, D. Naccache, and M. Tibouchi, “Fully Homomorphic Encryption over the Integers with Shorter Public Keys,” in Advances in Cryptology–CRYPTO, vol. 6841, P. Rogaway, Ed. Berlin, Germany: Springer, 2011, pp. 487–504.
[19] J.-S. Coron, D. Naccache, and M. Tibouchi, “Public key compression and modulus switching for fully homomorphic encryption over the integers,” in Advances in Cryptology–EUROCRYPT, vol. 7237, D. Pointcheval and T. Johansson, Eds. Berlin, Germany: Springer, 2012, pp. 446–464.
[20] J. Daemen, “Cipher and hash function design strategies based on linear and differential cryptanalysis,” Ph.D. dissertation, Dept. Elect. Eng., KU Leuven, Leuven, Belgium, 1995.
[21] I. Dinur, Y. Liu, W. Meier, and Q. Wang, “Optimized interpolation attacks on LowMC,” in Advances in Cryptology–ASIACRYPT, vol. 9453, T. Iwata and J. H. Cheon, Eds. Berlin, Germany: Springer, 2015, pp. 535–560.
[22] I. Dinur and A. Shamir, “Cube attacks on tweakable black box polynomials,” in Advances in Cryptology–EUROCRYPT, vol. 5479, A. Joux, Ed. Berlin, Germany: Springer, 2009, pp. 278–299.
[23] C. Dobrnagin, M. Eichelseder, L. Grassi, V. Lallemand, G. Leander, E. List, F. Mendel, and C. Rechberger, “Rasta: A cipher with low ANDdepth and few ANDs per bit,” in Advances in Cryptology–CRYPTO, vol. 10991, H. Shacham and A. Boldyreva, Eds. Cham, Switzerland: Springer, 2018, pp. 662–692.
[24] C. Dobrnagin, M. Eichelseder, and F. Mendel, “Higher-order cryptanalysis of LowMC,” in Information Security and Cryptology–ICISC, vol. 9558, S. Kwon and Y. Aun, Eds. Cham, Switzerland: Springer, 2016, pp. 87–101.
[25] Y. Doroz, A. Shahverdi, T. Eisenbarth, and B. Sunar, “Towards Practical Homomorphic Evaluation of Block Ciphers Using Prince,” in Financial Cryptography Data Secur., vol. 8438, R. Böhme, M. Brenner, T. Moore, and M. Smith, Eds. Berlin, Germany: Springer, 2014, pp. 208–220.
[26] L. Ducas and D. Micciancio, “FHWE: Bootstrapping homomorphic encryption in less than a second,” in Advances in Cryptology–EUROCRYPT, vol. 9056, E. Oswald and M. Fischlin, Eds. Berlin, Germany: Springer, 2015, pp. 617–640.
[27] L. Ducas, E. Kiltz, T. Lepoint, V. Lyubashevsky, P. Schwabe, G. Seiler, and D. Stehle, “CRYSTALS-dilithium: A lattice-based digital signature scheme,” IACR Trans. Cryptograph. Hardw. Embedded Syst., vol. 2018, no. 1. Bochum, Germany, Feb. 2018, pp. 238–268.
[28] S. Duval, V. Lallemand, and Y. Rotella, “Cryptanalysis of the FLIP Family of Stream Ciphers,” in Advances in Cryptology–CRYPTO, vol. 9814, M. Robshaw and J. Katz, Eds. Berlin, Germany: Springer, 2016, pp. 457–475.
[29] J. Fan and F. Vercauteren, “Somewhat practical fully homomorphic encryption,” IACR Cryptol. ePrint Arch., Las Vegas, NV, USA, Tech. Rep. 2012/144, 2012, p. 144. [Online]. Available: https://eprint.iacr.org/2012/144
[30] C. Gentry, “Fully homomorphic encryption using ideal lattices,” in Proc. 41st Annu. ACM Symp. Symp. Theory Comput. (STOC), Bethesda, MD, USA, 2009, pp. 169–178.
JINCHEOL HA received the B.S. degree in mathematical science from KAIST, Daejeon, in 2019, where he is currently pursuing the master’s degree in cryptography. His research interest includes HE-friendly symmetric ciphers.

SEONGKWANG KIM received the B.S. degree in mathematics from POSTECH, Pohang, in 2016, and the M.S. degree in mathematical science from KAIST, Daejeon, in 2018, where he is currently pursuing the Ph.D. degree in cryptography. His main research interest includes symmetric cipher design for privacy-preserving primitives.

WONSEOK Choi received the B.S. and M.S. degrees in mathematical science from KAIST, Daejeon, in 2012 and 2018, respectively, where he is currently pursuing the Ph.D. degree in cryptography. His main research interest includes security and analysis of symmetric ciphers.

JOYOUNG LEE received the master’s and Ph.D. degrees in information security from Korea University. He was a Research Engineer with the Electronics and Telecommunications Research Institute (ETRI), South Korea. He is currently working as an Associate Professor with the School of Computing, KAIST.

DUKJAE MOON received the B.S. and Ph.D. degrees in mathematics from Seoul National University. He is currently working as a Senior Engineer with the Security Algorithm Team, Samsung SDS.

HYOJIN YOON received the master’s and Ph.D. degrees in mathematics and cryptography from Seoul National University. She is currently a Team Leader of the Security Algorithm Team, Samsung SDS.

JIHOON CHO received the M.Math. degree in cryptography from the University of Waterloo and the Ph.D. degree in information security from the Royal Holloway University of London. He worked as a Security Architect for mobile devices with LG Electronics, South Korea. He is currently the Director of the Security Research, Samsung SDS, South Korea.

---

[31] C. Gentry, S. Halevi, and P. Nigel Smart, “Better bootstrapping in fully homomorphic encryption,” in *Public Key Cryptography–PKC*, M. Fischlin, J. Buchmann, and M. Manulis, Eds. Berlin, Germany: Springer, 2012, pp. 1–16.

[32] C. Gentry, S. Halevi, and P. Nigel Smart, “Homomorphic evaluation of the AES circuit,” in *Advances in Cryptology–CRYPTO*, vol. 7417, R. Safavi-Naini and R. Canetti, Eds. Berlin, Germany: Springer, 2012, pp. 850–867.

[33] C. Gentry, A. Sahai, and B. Waters, “Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,” in *Advances in Cryptology–CRYPTO*, vol. 8042, R. Canetti and J. A. Garay, Eds. Berlin, Germany: Springer, 2013, pp. 75–92.

[34] S. Halevi and V. Shoup, “Bootstrapping for HElib,” in *Advances in Cryptology–EUROCRYPT*, vol. 9056, E. Oswald and M. Fischlin, Eds. Berlin, Germany: Springer, 2015, pp. 641–670.

[35] (Aug. 2020). *HElib (release 1.0.2).* [Online]. Available: https://github.com/homenc/HElib

[36] T. Jakobsen and R. Lars Knudsen, “The interpolation attack on block ciphers,” in *Fast Software Encryption–FSE*, vol. 1267, E. Biham, Ed. Berlin, Germany: Springer, 1997, pp. 28–40.

[37] C. Juvekar, V. Vaikuntanathan, and A. Chandrakasan, “GAZELLE: A low latency framework for secure neural network inference,” in *Proc. 27th USENIX Conf. Secure Syst.*, New York, NY, USA, 2018, p. 1651–1668.

[38] R. Lars Knudsen, “Truncated and higher order differentials,” in *Fast Software Encryption*, vol. 1008, B. Preneel, Ed. Berlin, Germany: Springer, 1995, pp. 196–211.

[39] K. Susan Langford and E. Martin Hellman, “Differential-linear cryptanalysis,” in *Advances in Cryptology–CRYPTO*, vol. 839, Y. G. Desmedt, Ed. Berlin, Germany: Springer, 1994, pp. 17–25.

[40] K. Lauter, A. López-Alt, and M. Naehrig, “Private computation on encrypted genomic data,” in *Progress in Cryptology–LATINCRYPT*, vol. 8895, D. F. Aranha and A. Menezes, Eds. Cham, Switzerland: Springer, 2015, pp. 3–27.

[41] T. Lepoint and M. Naehrig, “A comparison of the homomorphic encryption schemes FV and YASHE,” in *Progress in Cryptology–AFRICACRYPT*, vol. 8469, D. Pointcheval and D. Vergnaud, Eds. Cham, Switzerland: Springer, 2014, pp. 318–355.

[42] R. Lidl and H. Niederreiter, “Finite fields,” vol. 20. Cambridge, U.K.: Cambridge Univ. Press, 1997.

[43] A. López-Alt, E. Tromer, and V. Vaikuntanathan, “On-the-fly multiparty computation on the cloud via multikey fully homomorphic encryption,” in *Proc. 44th Symp. Theory Comput. (STOC)*, 2012, p. 1219.

[44] S. Lucks, “The saturation attack—A bait for Twofish,” in *Fast Software Encryption–FSE*, vol. 2355, M. Matsui, Ed. Berlin, Germany: Springer, 2002, pp. 1–15.

[45] V. Lyubashevsky, C. Peikert, and O. Regev, “On ideal lattices and learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based,” in *Advances in Cryptology–EUROCRYPT*, vol. 7656, B. Preneel and A.bossaerts, Eds. Berlin, Germany: Springer, 2012, pp. 850–867.

[46] M. Matsui, “Linear cryptanalysis method for DES cipher,” in *Advances in Cryptology–EUROCRYPT*, vol. 765, T. Helleseth, Ed. Berlin, Germany: Springer, 1994, pp. 386–397.

[47] P. Méaux, A. Journault, F.-X. Standaert, and C. Carlet, “Towards stream ciphers for efficient FHE with low-noise ciphertexts,” in *Advances in Cryptology–EUROCRYPT*, vol. 9665, M. Fischlin and C. Jean-Sébastien, Eds. Berlin, Germany: Springer, 2016, pp. 311–343.

[48] M. Naehrig, K. Lauter, and V. Vaikuntanathan, “Can homomorphic encryption be practical?” in *Proc. 3rd ACM Workshop Cloud Comput. Secure Workshop*, New York, NY, USA, 2011, p. 113–124.

[49] SHA-3 Standard: Permutation-Based Hash and Extendable-Output Function Standard FIPS 202, National Institute of Standards and Technology, 2015.

[50] Rasta Implementation. Accessed: Feb. 2018. https://github.com/aikkrypto/rasta

[51] C. Rechberger, H. Soleimany, and T. Tiessen, “Cryptanalysis of low-data instances of full LowMCv2,” *IACR Trans. Symmetric Cryptol.*, vol. 2018, no. 3, pp. 163–181, Sep. 2018.

[52] M. V. Dijk, C. Gentry, S. Halevi, and V. Vaikuntanathan, “Fully homomorphic encryption over the integers,” in *Advances in Cryptology–EUROCRYPT*, vol. 6100, H. Gilbert, Ed. Berlin, Germany: Springer, 2010, pp. 24–43.

[53] D. Wagner, “The Boomerang attack,” in *Fast Software Encryption–FSE*, vol. 1636, L. Knudsen, Ed. Berlin, Germany: Springer, 1999, pp. 156–170.

[54] (Aug. 2020). *XKCP: Extended Keccak Code Package.* [Online]. Available: https://github.com/XKCP/XKCP