International Journal of Mathematics and Computer Science
Volume **20,** Issue no. 2, (2025), 549–553
DOI: https://doi.org/10.69793/ijmcs/02.2025/nibras

Efficient Brakerski-Fan-Vercauteren Algorithm Using Hybrid-Position-Residues Number System

Nibras Hadi Jawad, Salah Abdulhadi

Department of Computer Science
Faculty of Computer Science and Mathematics
University of Kufa
Najaf, Iraq

email: nibras.hadi@qu.edu.iq, nibrash.albustani@student.uokufa.edu.iq, salah.albermany@uokufa.edu.iq

(Received January 18, 2025, Accepted February 15, 2025, Published February 19, 2025)

Abstract

Cloud services must be dealt with in a way that preserves the privacy of customer's data. In order to achieve this, homomorphic encryption is used which helps to complete business within the cloud environment while preserving the privacy of data from viewing it. In this article, we use the Brakerski-Fan-Vercauteren (BFV) algorithm, which requires high and expensive calculations, using the Hybrid-Position-Residues (HPR) number system to reduce the cost of calculations while maintaining the efficiency of the algorithm.

1 Introduction

The privacy challenge is solved by fully homomorphic encryption (FHE) using the Brakerski-Fan-Vercauteren (BFV) algorithm. BFV lets an untrusted person use encrypted data without the decryption key [1]. The BFV maintains data privacy and prevents viewing when performing cloud operations.

Keywords and phrases: BFV, Fully homomorphic encryption, RSN, HPR.
AMS (MOS) Subject Classifications: 68P25, 68P27.
ISSN 1814-0432, 2025, https://future-in-tech.net

To make the algorithm more efficient, we minimize time and memory use. In 2019, Halevi et al. [2] improved decoding and multiplication utilizing RNS residues number system. In 2021, Lara-Benítez et al. [3] analyzed BFV, CKKS, and TFHE test results using three homomorphic approaches for precision, memory utilization, and execution time. In 2022, Yates [4] discussed homomorphic encryption systems' efficiency. In 2023, Kuo and Wu [5] presented a fast procedure for secure comparison between two parties using integers and HE fundamentals.

2 BFV with HPR Proposed System

The BFV technique costs time and generates noise from repeated calculations when processing encrypted data. Suggested using the HPR system [6] in BFV to improve algorithm efficiency by parallelizing operations, which speeds execution and reduces noise. In general, the steps of modifying BFV are as follows:

Step 1: Generate parameters: Degree of polynomials (N) is determines the level of security. The value of N used is 1024. Ciphertext mod (q) must be a very large integer and the size of $q = 50$ bits. The value of q used is $(2^{50} + 1) \equiv 1 \ mod \ (2 * 1024)$. Plaintext mod ($t$) is the same as the selected q. The value of t used is $(2^{30} + 1) \equiv 1 \ mod \ (2 * 1024)$. A basis is a set (μ_a, μ_b) consisting of two pairwise coprime integers used to represent integers in RNS. Noise parameters (sp) are (a, e, e_1, e_2, u).

Step 2: Key Generation: First, generate a random number sk: Using chaotic equations as a foundation, we create a three-dimensional chaotic system. The standard algorithm relies on a binary key $sk \in R_2$ from $\{0, 1\}$ which is a weak key that can be known and guessed. Therefore, we rely on the system's chaotic maps of the Lorenz equations depending on variables x, y, and z chosen each time randomly from the period [0, 1]. As for the values of $\alpha = 10, \rho = 28$, and $\beta = 8/3$, the modified Lorenz equations are as follows: $\{x = (\alpha - 1) * (y - x),\ y = x * (\rho - z) - y, z = x * y - \beta * z + x\}$. Then the values are combined into the following equation to give the result (sk) as $sk = ((x + y + z) * 1000))$. The values of the proposed key $0 \rightarrow 2^{279}$ have a large range. In the end, it is converted to HPR.
Secondly, compute the public key with used sk, a and e as the equation $pk_{HPR} = (pk_1^{HPR} = ([-(a^{HPR}.sk^{HPR} + e^{HPR})]_M, PK_2^{HPR} = a^{HPR})$. The value of a is chosen randomly from the range $[0, Rq)$, convert a to HPR before using in pk, generate the error value e from the range $[-2, 2]$, and

convert e to HPR.

Step 3: Input Plaintext (pt): Scaling pt: Delta (Δ) from division q/t, used before encryption message (pt) to produce $pt * \Delta$.

Encoding use batch encoding $pt * \Delta$: The BFV scheme is RLWE, batch encoding based on CRT transform integer mod t into a polynomials (pt_e) $R_t = Z_t[x]/(x^n + 1)$ before encryption begins that allows encrypt many pt in a parallel way. With the condition that t is a prime number, $t \equiv 1 \ mod\ 2N$, there exists element $\mathcal{V}$ where $\mathcal{V}^{2N} \equiv 1\ (mod\ t)$ and its primitive root $2N^{th}$ of unity. We find an element h where $1 < h < 2N \rightarrow \mathcal{V}^h \neq 1\ (mod\ t)$. Then the ring $R_t = [Z_t[x]/(x^n + 1) = Z_t[x]/(x - \mathcal{V})(x - \mathcal{V}^1) \dots (x - \mathcal{V}^{2N-1})]$ $\cong [Z_t[x]/(x - \mathcal{V}) * Z_t[x]/((x - \mathcal{V}^1) * \dots * Z_t[x]/((x - \mathcal{V}^{2N-1})] \cong\ [Z_t[\mathcal{V}] * \dots * Z_t[\mathcal{V}^{2N-1}]] \cong Z_t^N$, use CRT to decode Z_t^N.

Transform pt_e (coefficients of polynomials) to $HPR \rightarrow pt_e^{HPR}$.

Step 4: Encryption: The encryption in the proposed system is done under HPR. Encrypt pt_e^{HPR} to get the ciphertext $CHPR = (C_0^{HPR}, C_1^{HPR}) \in R^2_{m_{a|b}}$. C^{HPR} consists of two polynomials (C_0^{HPR} and C_1^{HPR}), the C_0^{HPR} is computed by using $pk_0^{HPR}.\ u^{HPR} + e_1^{HPR} + pt_e^{HPR}]_M$, and C_1^{HPR} is computed by using $[pk_1^{HPR}.u^{HPR} + e_2^{HPR}]_M$.

Step 5: Evaluation: Simple addition and multiplication operations are performed within the RNS system. Compute C^{HPR3} by using ($[C_0^{HPR1}\ o\ C_0^{HPR2}]$, $[C_1^{HPR1}\ o\ C_1^{HP2}]$) where o is ($+$ *or* $*$) operation.

Step 6: Decryption: To recover the plaintext (pt) from the algorithm after obtaining the CHPR3 output, use the following equation: $\text{pt}_e^{HPR} = \left\lceil \frac{\left[C_0{}^{HPR} + C_1{}^{HPR}.sk^{HPR}\right]_M}{\Delta} \right\rceil_t$ represent the $C^{HPR3} = (C_0^{HPR3}, C_1^{HP3})$ as: $C_i^{HPR3} = \sum_{j=0}^{d-1} C_{i,j}^{HPR3} M^j$ for $i = \{0, 1\}$ where $C_{i,j}^{HPR3} \leq \mu_a.p, (0 \leq j \leq d - 2)$.

Step 7: Transform ptx_e^{HPR} from HPR to normal number: To transform pt_e^{HPR} to pt_e by using Base Extension based on MRC.

Step 8: Decoding pt: Decoding the result pt_e that are polynomials by using batch decoding to compute $[Z_t[\mathcal{V}] * \dots * Z_t[\mathcal{V}^{2N-1}]]$ to recover the plaintext.

3 Analysis and Study of the Results

3.1 Chaotic key generation of sk

The chaotic system calculates a three-dimensional secret key using three Lorenz equations with a little modification. This will give the suggested secret key a chaotic structure based on Lyapunov's criteria for chaos size (0.83004, 0.0010723, and -13.4978). The chaotic system requires different initial values for each key generation and is sensitive to even the slightest change in them. Since the key values change with any slight change in the variables, the resulting keys are different in both cases. The computer generates the key using six parameters $(x_0, y_0, z_0, \alpha, \rho, \ and\ \beta)$ with a precision of 10^{14}. Multiplying $(10^{14})^6 \cong 2^{279}$ yields a key $> 2^{128}$, making it reliable for encryption and resistant to brute force attacks. Use NIST for analysis of sk as follows: (Run: 0. 87057708, Frequency: 0. 07658140, Serial: 0. 19613940, Random excursion: 0.913788, Linear complexity: 0. 69479678, Binary matrix rank: 0. 05650373, Overlapping template matching: 0. 32417062, Entropy: 0. 20726256).

3.2 Noise

The BFV system uses multiple random values. If noise values are above the threshold limit, decoding may be affected. Operations on ciphertexts mixed with noise increase noise size and growth. It must be rounded to the nearest integer and error parameters limited to $\frac{t}{2q}$. to eliminate errors. Compared to the original technique, the new algorithm returns smaller noise coefficients due to decreased coefficient values. $c_i(y) = \sum_{j=0}^{d-1} c_{ij} y^i$ where $\|c_i\| \leq \mu * p$ and error. limited by $\|e\| \leq \frac{p}{p-1} * \mu$, without causing a large error when representing MSD mod μ. The error in the result of original BFV is 0.000816000625 while the error resulting from the Proposed BFV is 0.0000099.

3.3 Consumed memory

The consumed memory original BFV (Key: 63.1, Encryption: 63.3, Decryption: 62.2), and consumed memory proposed BFV (Key: 61.4, Encryption: 60.1, Decryption: 59.5).

3.4 Consumed Time

Table 1 shows the time taken to generate the key with encryption and decryption.

Table 1: Time consumed

BFV	q	t	Time in second		
			key	encryption	decryption
original	2^{50}	2^{30}	0.00433	0.00035	0.00023
proposed	2^{50}	2^{30}	0.00804	0.000073	0.000036

4 Conclusion

Using the HPR system which is a hybrid algorithm between RNS and PNS in the BFV algorithm leads to faster execution of operations and reduced noise in addition to saving space in memory as the work is sequential. Using HPR instead of just RNS, modular multiplications are more effective.

References

[1] Inferati Inc., Introduction to the BFV FHE Scheme, (2021). Available at: https://www.inferati.com/blog/fhe-schemes-bfv

[2] S. Halevi, Y. Polyakov, V. Shoup, An Improved RNS Variant of the BFV Homomorphic Encryption Scheme. In Topics in Cryptology–CT-RSA, Lecture Notes in Computer Science, 11405, Springer, Cham., (2019), 183–201.

[3] P. Lara-Benítez, M. Carranza-García, J. C. Riquelme, An experimental review on deep learning architectures for time series forecasting, International Journal of Neural Systems, **31,** no. 3, (2021), 2130001.

[4] K. Yates, Efficiency of Homomorphic Encryption Schemes, M. Sc. Thesis, Clemson University, Clemson, SC, USA, (2022).

[5] T. H. Kuo, J. L. Wu, A High Throughput BFV-Encryption-Based Secure Comparison Protocol, Mathematics, **11,** no. 5, (2023), 1227.

[6] K. Bigou, A. Tisserand, Hybrid position-residues number system. In IEEE 23rd Symposium on Computer Arithmetic, IEEE, (2016), 126–133.