Separable Reversible Data Hiding Scheme in Homomorphic Encrypted Domain Based on NTRU

NENG ZHOU, MINQING ZHANG, HAN WANG, YAN KE, AND FUQIANG DI
Key Laboratory of Network and Information Security Under the Chinese People Armed Police Force (PAP), College of Cryptography Engineering, Engineering University of PAP, Xi’an 710086, China

Corresponding authors: Minqing Zhang (api_zmq@126.com) and Han Wang (aca_wang@163.com)

This work was supported by the National Natural Science Foundation of China under Grant 61872384 named “Research on Theory and Method of Reversible Data Hiding in Encrypted Domain Based on Encryption Process”. Through excavation and effective use of redundancy in the encryption process, we compromise the information hiding technology and encryption technology, construct the theoretical model of the fusion of cryptography and information hiding technology, and design secure and efficient reversible information hiding methods.

ABSTRACT NTRU (Number Theory Research Unit) has the characteristics of resistance to quantum computing attacks, fast encryption and decryption, and high security. It is very suitable for wireless confidential data networks and authentication systems. Combined with reversible data hiding technology, a separable reversible data hiding scheme in homomorphic encrypted domain based on NTRU is proposed. The image owner directly divides the cover-image into groups of a reference pixel and $T$ adjacent pixels. Then, the grouped image is encrypted by NTRU. Finally, the encrypted image is uploaded to the data hider. The encrypted image is divided into groups by the same grouping method as the image owner. After that, the data hider calculates $T$ absolute differences of adjacent pixels in each group to obtain histogram of the absolute differences. The additional data can be embedded into the encrypted image by shifting histogram of the absolute differences. After receiving the encrypted image with hidden data, the receiver can either use the data hiding key to directly extract the additional data from encrypted domain to obtain the encrypted image, or use the private key and data hiding key to extract the additional data from plaintext domain to get the cover-image. Experimental results show that our scheme has higher security and better embedding performance comparing with other state-of-the-art works.

INDEX TERMS Reversible data hiding, encrypted domain, public key cryptosystem, homomorphic property, number theory research unit (NTRU).

I. INTRODUCTION

Information hiding technology is a cross discipline in the field of information security [1], covering cryptography, mathematics, computer vision, artificial intelligence, and computer application technology. The key difference between information hiding and cryptography is that cryptography emphasizes the confidentiality of “communication content”. Based on mathematical problems and strict derivation on mathematical logic, an encryption algorithm is constructed to transform and disguise the plaintext content of communication. In the case of unknown decryption key, the amount of effective information that ciphertext data can provide is equivalent to random noise. Information hiding technology emphasizes the confidentiality of “communication existence”. The research of information hiding technology mainly includes three aspects: steganography, digital watermarking and reversible data hiding (RDH) technology. Among them, steganography and digital watermarking technology are irreversible data hiding. The embedding process will bring permanent distortion to the original carrier. This is unacceptable for application organizations that require complete extraction of information and also need to recover the original carrier without distortion, such as annotation of encrypted data in the cloud environment, remote medical diagnosis, and judicial forensics. Bartron first proposed the concept of reversible data hiding [2]. Because RDH takes into account both information hiding and distortion-free recovery of the original carrier, it has a broader application prospect. Therefore, it has been widely studied. Currently, RDH mainly includes difference expansion (DE) [3]–[6] and histogram shifting (HS) [7]–[10].

In 1978, Rivest et al. first proposed the concept of homomorphic encryption in [11]. Since then, various homomorphic cryptosystems have emerged. First, single homomorphic
cryptosystems have appeared, such as the Paillier cryptosystem [12], the ElGamals cryptosystem [13], and so on. Paillier is a typical multiplicative homomorphic encryption. Stream cipher and RC4 in symmetric cryptosystem also satisfy the addition homomorphism. In [14], Gentry proposed a fully homomorphic cryptosystem, which provided theoretical support for performing any number of addition and multiplication operations in encrypted domain.

With the continuous development of network technology, network services have covered many aspects of people’s lives. The user uploads the data to cloud service provider (CSP) for saving and processing the data through the network. In order to achieve the confidentiality of data, users must encrypt the data before uploading. The CSP needs to embed the time stamp, user identity, and other information directly in encrypted domain for management. Reversible data hiding in encrypted domain (RDH-ED) came into being, and has become the latest research hotspot [15]–[20]. The RDH-ED requires that the carrier used for embedding is encrypted. After embedding the information, the original carrier can still be decrypted without errors. The RDH-ED in [15]–[20] uses stream ciphers to encrypt images. There are problems such as a large number of keys and difficult key management. Though the stream cipher encrypts the image into unreadable encrypted data, it will destroy the algebraic structure of the encrypted data. Afterwards, this will make it impossible to process the encrypted image. However, in a realistic cloud computing environment, we not only require CSP to protect user privacy and data security, but also need to perform traditional data processing operations on encrypted data. Traditional cryptographic algorithms can hardly adapt to the requirements in the context of cloud computing. Fig. 1 is an application scenario of RDH-ED. Without decrypting the image, the CSP can directly perform operations on the encrypted image. The algorithm of RDH-ED is an important combination of signal processing technology in encrypted domain and information hiding technology. It has the dual functions of privacy protection and secret information transmission. Therefore, constructing some algorithm of RDH-ED by using public key cryptography with homomorphic properties is a key issue that needs to be addressed in big data and cloud computing environments. In this way, we can achieve effective management and security protection of encrypted data.

Chen et al. [21] proposed the reversible data hiding in encrypted domain with public key cryptosystem (RDHED-P), which uses the characteristics of public key cryptography to overcome the shortcomings of symmetric cryptography that requires a secure channel to pass the key in advance. The algorithm embeds a bit data into a pair of adjacent encrypted pixels. According to the homomorphic characteristics of the Paillier cryptosystem, the receiver obtains secret information by comparing all decrypted pixel pairs. The disadvantage is that there is an inherent overflow problem. Subsequently, Shiu et al. [22] and Wu et al. [23] improved Chen’s method by solving the overflow problem. Because the inseparable algorithms mentioned above have fewer application scenarios, the separable algorithms have higher research value. The receiver’s key determines which operation it can perform: 1) the receiver only extracts additional data with the data hiding key. 2) the receiver only decrypts the image with the private key. 3) the receiver both decrypts the cover-image and extracts additional data with the private key and data hiding key. Therefore, Zhang et al. [24] proposed a separable RDH-ED. The algorithm first vacates room before encryption (VRBE). After encryption, additional data can be embedded through the histogram shifting, so that the hidden data can be extracted from plaintext domain. Combined with the feature of homomorphic encryption, wet paper code (WPC) is used to embed additional data into the encrypted image. This makes it possible to extract the hidden data from encrypted domain. The algorithm can realize the separation of data extraction and image decryption. Besides, multiple bits are used for embedding, and the maximum embedding rate is 1 bpp (bit per pixel). However, due to the use of WPC technology, it is necessary to use Gaussian elimination to solve a system of linear equations with k unknowns. The algorithm actually performs two data hiding operations. Literature [25] proposes a RDH-ED by using the homomorphism and probability characteristics of the Paillier cryptosystem. The image owner self-embeds the target pixel into the original image by VRBE. The data hider uses the data to be embedded to form dummy pixels. It can be used to replace the target pixels to complete the embedding procedure. The receiver with the corresponding key can extract the hidden data from plaintext domain or encrypted domain, respectively, and restore the cover-image completely after decryption. Though it is a separable scheme, the embedding rate is not high. In summary, although the method of VRBE also has a good embedding performance, it requires the image owner to do a lot of extra work before encryption. It must have a secure and reliable interaction with the data hider to prevent data leakage. Compared to VRBE, vacating room after encryption (VRAE) is more practical.

Xiang et al. [26] proposed that the absolute differences are recovered with groups of two selected encrypted pixels by exploiting the homomorphic property in encrypted domain. After that, the additional data can be embedded into the encrypted image by shifting histogram of the absolute differences. However, the maximum embedding rate of Xiang et al.’ scheme is 0.5 bpp. In [27], Li et al. use the histogram shifting technique to embed additional data in the DGHV encryption domain. Compared with [26], the algorithm proposed in [27] has lower computational cost. Each additional data is embedded into two pixels in [26] and [27] so that the maximum embedding rate is only 0.5 bpp. In [28], Xiong et al. proposed two additional data are embedded into groups of three pixels, which means the embedding rate can increase to 0.67 bpp in the best case. Considering above algorithm with VRAE, this paper proposes a separable reversible data hiding scheme in homomorphic encrypted domain by using NTRU instead of Paillier or DGHV (Dijk, Gentry, Halevi
Shor [29] gave the first quantum algorithm to solve integer factorization in polynomial time. Quantum algorithms have seriously shaken the security of typical schemes in classic cryptography, such as Paillier cryptography, elliptic curve cryptography, etc. To solve this problem, the focus on cryptographic research has shifted to two important areas: quantum cryptography and post-quantum cryptography. Among them, the post-quantum cryptographic scheme runs in traditional computer and network systems and can resist the attacks of traditional algorithms and quantum algorithms. Data encrypted with a post-quantum cryptographic algorithm can achieve confidentiality requirements. In actual applications, the encrypted data often needs to be processed. Therefore, post-quantum cryptographic algorithms with homomorphic property will have huge applications in the future. Among all post-quantum cryptographic algorithms, lattice-based fully homomorphic encryption algorithm is the best choice. NTRU (Number Theory Research Unit) full homomorphic encryption algorithm can resist quantum and traditional algorithm attacks, and its security can be reduced to the shortest vector problem (SVP) on the lattice [30]–[32].

The main contributions of this paper are as follows.

1) The security of the algorithm in this paper can be reduced to the SVP problem on the lattice. It has the characteristics of resistance to quantum computing attacks, fast encryption and decryption, and high security.

2) The image owner does not need to perform any preprocessing operations on the cover-image, and only needs to encrypt the cover-image. More importantly, the proposed scheme is more scalable in the context of big data and cloud computing due to the avoidance of preprocessing operations.

3) Each additional data is embedded into two pixels in [26] and [27] so that the maximum embedding rate is only 0.5 bpp. Two additional data are embedded into groups of three pixels in [28], which means the embedding rate can increase to 0.67 bpp at most. Generally speaking, \( T \) additional data are embedded into groups of a reference pixel and \( T \) adjacent pixels, which the embedding rate can be calculated as \( T/(T + 1) \). \( T \in \{1, 2, 3, 4\} \). That is to say, the maximum embedding rate of this paper can increase to 0.8 bpp at most. Additionally, when \( T = 3 \), that is, the embedding rate is 0.75 bpp, the algorithm has the best performance on PSNR.

The remainder of this paper is organized as follows. A brief introduction to NTRU cryptosystem is given in Section II. Then, the details of the proposed scheme are elaborated in Section III. Security Analysis is introduced in Section IV. Simulation experiment and theoretical analysis are shown in Section V. Finally, this paper is concluded in Section VI.

II. NTRU CRYPTOSYSTEM

NTRU is a ring-based public key cryptosystem. It has the following advantages: 1) it is easy to generate short keys; 2) it has high-speed encryption and decryption; 3) it can be operated under low memory conditions. In summary, NTRU
is particularly suitable for applications with high security and efficiency requirements, such as encrypted image management, encrypted data annotation, judicial forensics, and military information processing in big data and cloud computing environments. Fig. 1 is also an example of an application environment for encrypted image management. The cloud can embed data such as user identity and integrity authentication information into the encrypted image without obtaining the original image to achieve the management function of encrypted image.

In all process, we use $\Phi = x^n + 1$ with $n \geq 8$ a power of $2$, $R = \mathbb{Z}[x]/\Phi$ and $R_q = R/qR$ with $q \geq 5$ prime such that $\Phi = \prod_{k=1}^{n} \Phi_k$ in $R_q$ with different $\Phi_k$'s.

1) SETTING PARAMETERS

In this section, we present the variant of the NTRU-Encrypt algorithm. We define the NTRUEncrypt with parameters $n, q, p, \alpha, \sigma$ as follows, which is NTRUEncrypt $(n, q, p, \alpha, \sigma)$. The rings $R$ and $R_q$ are defined by the parameters $n$ and $q$. The plaintext message space $P = R/pR$ is defined by the parameter $p \in R_q$. The coefficients of the polynomial $P$ are small with respect to $q$, but meanwhile we require $N(p) = |P| = 2^{\Omega(n)}$ in order to encode many bits immediately. By taking the absolute minimum residue of the modulus $p$, any element of $P$ can be written as $\sum_{0 \leq i < n} e_i x^i p_i$ with $e_i \in \{-1/2, 1/2\}$. Since $R = \mathbb{Z}[x]/(x^n + 1)$, any element of $P$ is an element of $R$ with infinity norm $\leq (\deg(p) + 1)/|p|$. The parameter $\alpha$ is the R-LWE (Ring-Learning With Errors) noise distribution parameter. At last, the parameter $\sigma$ is the standard deviation of the discrete Gaussian distribution used in key generation.

2) NTRU.KEYGEN $(1^\ast)$

Selecting parameter $n, q \in \mathbb{Z}, p = 2 \in R_q^\times$, $\alpha \in \mathbb{R}$, we can sample $f'$ from $D_{2n}^{\ast}$. Let $f' = 2f' + 1$, if $f'(\mod q) \notin R_q^\times$ sample. In addition, we can sample $g$ from $D_{2n}^{\ast}$. If $g(\mod q) \notin R_q^\times$, sample. Finally, we can return secret key $sk = f \in R_q^\times$ with $f = 1$ (mod 2) and public key $pk = h = 2g/f \in R_q^\times$, which is a key pair $(sk, pk) \in R \times R_q^\times$.

3) NTRU.ENC $(pk, m)$

Given plaintext $m \in P$, sample $s, e \leftrightarrow \gamma_a$ ($\gamma_a$ is a distribution over absolute values) and return ciphertext $c = hs + 2e + m \in R_q$ with $pk = h$.

4) NTRU.DEC $(sk, c)$

Given ciphertext $c$ and $sk = f$, compute $c' = f \cdot c \in R_q$ and return $m$, which is calculated by

$$m = c'(\mod 2) = f \cdot c(\mod 2) = (2(gs + ef) + fm)(\mod 2) \in P$$

(1)

5) NTRU.ADD $(pk, c_1, c_2)$ AND NTRU.MULT $(pk, c_1, c_2)$

In particular, consider ciphertexts $c_1 = hs_1 + 2e_1 + m_1 \in R_q$ and $c_2 = hs_2 + 2e_2 + m_2 \in R_q$ that encrypt plaintexts $m_1$ and $m_2$ under public key $h$ respectively, with noise terms $s_1, s_2, e_1$ and $e_2$. The algebraic manipulation shows that we can encrypt the addition and multiplication of plaintexts $m_1$ and $m_2$ to get ciphertexts $c_{addd} = c_1 + c_2$ and $c_{mul} = c_1c_2$, respectively. Namely, decrypting $c_1 + c_2$ and $c_1c_2$ with the secret key $f$ and joint secret key $f^2(f^2)$ is a product of the two secret keys $f$ and $f$ gives us:

$$f(c_1 + c_2) = 2(f(e_1 + e_2) + g(s_1 + s_2)) + f(m_1 + m_2) \triangleq 2E_{Add}$$

(2)

That is, the addition of two ciphertexts $c_1$ and $c_2$ is equal to the addition of plaintexts $m_1$ and $m_2$ using the secret key $f$ after decryption. Likewise, we get:

$$f^2(c_1c_2) = 2(2g^2s_1s_2 + gs_1f(2e_2 + m_2) + gsf(2e_1 + m_1) + f^2(e_1m_2 + e_2m_1 + 2e_1e_2)) + f^2(m_1m_2) \triangleq 2E_{Mult}$$

(3)

This shows that the multiplication of two ciphertexts $c_1$ and $c_2$ is equal to the multiplication of plaintexts $m_1$ and $m_2$ using the joint secret key $f^2$ after decryption.

III. PROPOSED SCHEME

A. GENERAL FRAMEWORK

The proposed scheme is shown in Fig. 2. The image owner first divides the cover-image into groups of a reference pixel and $T$ neighboring pixels, and then uses the $pk$ to encrypt the grouped image. The data hidder divides the encrypted image according to the same grouping method as the image owner, and calculates the absolute difference $d$ of each group. Then, the data hiding key $K_d$ is used to embed the additional data by shifting histogram of the absolute differences. The receiver divides the encrypted image with hidden data by the same grouping method as the image owner. With the data hiding key $K_d$, additional data can be extracted from the encrypted image. Besides, the encrypted image can be obtained by inverse operations of histogram shifting. With the $sk$ and data hiding key $K_d$, the receiver can use the $sk$ to decrypt the encrypted image with hidden data. Later, the data hiding key $K_d$ can be used to extract additional data from the directly decrypted image with hidden data. Finally, the cover-image is obtained by inverse operations of histogram shifting.

B. IMAGE ENCRYPTION

Before encryption, we divide image into groups of a reference pixel and $T$ adjacent pixels. As shown in Fig. 3, $P_t(k)$ is the reference pixel and $P_l(k), \ldots, P_r(k)$ are the adjacent pixels, where $T \in \{1, 2, 3, 4\}$. It is noting that we use the case of $T = 1$ shown in Fig. 3 (a) for a better description in the following sections. More importantly, the parameter $T$ has a greater impact on the PSNR versus embedding performance.
FIGURE 2. The sketch of the proposed scheme.

FIGURE 3. Embedded groups including a reference pixel and T adjacent pixels, T = 1, 2, 3 or 4. (a) T = 1. (b) T = 2. (c) T = 3. (d) T = 4.

since the additional data is embedded into image by shifting histogram of absolute difference between the reference pixel and T adjacent pixels. Section V will discuss the details.

To ensure the semantic security, two parameters $s, e \leftarrow \gamma_{\alpha}$ should be selected randomly for each pixel. However, to preserve the relevance of the plaintext pixels to the ciphertext, we select the same $s, e \leftarrow \gamma_{\alpha}$ to encrypt pixels in each group to reserve the difference between the reference pixel and adjacent pixels for data hiding in encrypted domain.

In detail, the image owner divides image into groups of a reference pixel and an adjacent pixel according to Fig. 3 (a). Mark the reference pixel and adjacent pixel in the $k$th group as $P_1(k)$ and $P_r(k)$, respectively. Each group is encrypted with public key $h$ and parameter $s, e \leftarrow \gamma_{\alpha}$.

$$c_1(k) = hs + 2e + P_1(k)$$
$$c_r(k) = hs + 2e + P_r(k)$$ (4)

where $c_1(k)$ and $c_r(k)$ are the ciphertexts of $P_1(k)$ and $P_r(k)$. The image owner sends the encrypted image to the data hider after encryption.

C. DATA HIDING

The data hider divides the encrypted image by the same grouping method as the image owner, and calculates the absolute difference $d$ of each group to obtain histogram of the absolute differences. Then, the data hiding key $K_d$ is used to embed the additional data by shifting histogram of the absolute differences to obtain the encrypted image with hidden data. The process of specific data hiding is as follows.

1) First, the encrypted image is divided into groups of two adjacent pixels by the same grouping method as the image encryption process. Mark the two encrypted pixels as $\{c_1(k), c_r(k)\}$ in the $k$th group.

2) Then, absolute difference $d$ is obtained by two adjacent encrypted pixels in each group, which is equal to the absolute difference between two adjacent plaintext pixels. It is calculated by

$$d(k) = |c_1(k) - c_r(k)|$$

$$= |(hs + 2e + P_1(k)) - (hs + 2e + P_r(k))|$$

$$= |P_1(k) - P_r(k)|$$ (5)

At the same time, the data hider can obtain the magnitude relationships between $P_1(k)$ and $P_r(k)$. If $c_1(k) \geq c_r(k)$, we have $P_1(k) \geq P_r(k)$; If not, we have $P_1(k) < P_r(k)$.

3) Finally, the data hider can embed additional data into the encrypted image by shifting histogram of the absolute differences in several rounds. If the additional data is totally embedded in certain round, we can mark the final pixel’s location as $z$. The data hiding key $K_d$ includes the embedding rounds $n_z$, the embedding bin $EP$ in each round, and the final pixel’s location $z$. In $k$th group, for $c_1(k)$ and $c_r(k)$,
if \( P_1(k) \geq P_r(k) \)

\[
c_1^w(k) = \begin{cases} 
\text{NTRU}.Add(c_1(k), w) = c_1(k) + w, & \text{if } d(k) = EP \\
\text{NTRU}.Add(c_1(k), 1) = c_1(k) + 1, & \text{if } d(k) \geq EP + 1 \\
c_1(k), & \text{else}
\end{cases}
\]

\[ c_r^w(k) = c_r(k) \]

\[ c_1^w(k) = c_1(k) \]

where \( c_1^w(k) \) and \( c_r^w(k) \) are the encrypted pixels with additional data in the \( k \)th group and \( w \) is a bit of the additional data. According to Eq. (7) and Eq. (9), \( c_1^w(k) \) remains unchanged as a reference for image recovery and data extraction. Mark the directly decrypted versions of \( \{c_1^w(k)\}, \{c_r^w(k)\} \) as \( \{P_1^w(k)\}, \{P_r^w(k)\} \), respectively. Due to the homomorphic addition of the NTRU, the effect of data hiding on plaintext pixels is to change \( \{P_1(k), P_r(k)\} \) to \( \{P_1^w(k), P_r^w(k)\} \). In \( k \)th group, for \( \{P_1(k), P_r(k)\} \),

\[
P_1^w(k) = \begin{cases} 
P_1(k) + w, & \text{if } d(k) = EP \\
P_1(k) + 1, & \text{if } d(k) \geq EP + 1 \\
P_1(k), & \text{else}
\end{cases}
\]

\[ P_r^w(k) = P_r(k) \]

\[ P_1^w(k) = \begin{cases} 
P_1(k) - w, & \text{if } d(k) = EP \\
P_1(k) - 1, & \text{if } d(k) \geq EP + 1 \\
P_1(k), & \text{else}
\end{cases}
\]

\[ P_r^w(k) = P_r(k) \]

where \( P_1^w(k) \) and \( P_r^w(k) \) are the marked versions of \( P_1(k) \) and \( P_r(k) \). Obviously, the relative magnitude of two pixels in each group is kept unchanged after data hiding. Namely, if \( P_1(k) \geq P_r(k) \), we have \( P_1^w(k) \geq P_r^w(k) \); If not, we have \( P_1^w(k) < P_r^w(k) \).

By employing NTRU cryptosystem to encrypt each group of two adjacent pixels, we can successfully retain the absolute differences between plaintexts in encrypted domain. Absolute differences can be restored by the data hider, which embeds additional data directly into the encrypted image by shifting histogram of the absolute differences. In summary, the data hider can produce the encrypted image with hidden data.

D. DATA EXTRACTION AND IMAGE RECOVERY

In this section, data extraction and image recovery are the inverse operations of data hiding. There are two methods for a legitimate receiver to extract the hidden data and recover the encrypted image or cover-image.

1) EXTRACT HIDDEN DATA AND RECOVER ENCRYPTED IMAGE FROM ENCRYPTED DOMAIN

If the receiver has only the data hiding key \( K_d \), the additional data \( w \) and directly encrypted image can be retrieved from the encrypted image with hidden data. The process of specific extraction and recovery are described as follows.

1) The receiver divides the encrypted image with hidden data into groups of two adjacent pixels. Mark the \( k \)th group as \( \{c_1^w(k), c_r(k)\} \) by the same grouping method as the data hiding process.

2) Similar to calculate the absolute difference in the process of data hiding, the absolute differences \( d^w(k) \) in the \( k \)th group can be obtained by referring to Eq. (14).

\[ d^w(k) = |c_1^w(k) - c_r(k)| \]

The embedded histogram can be generated by the embedded absolute differences of all groups.

3) With the data hiding key \( K_d \), the receiver can obtain the embedding rounds \( n_e \) and the embedding bin \( EP \) in each round. By referring to \( EP \), the receiver can extract additional data \( w \) from the histogram of the absolute differences,

\[
w = \begin{cases} 
0, & \text{if } d^w(k) = EP \\
1, & \text{if } d^w(k) = EP + 1
\end{cases}
\]

Besides, the directly encrypted image can be obtained by the receiver without any error. With the property of NTRU and the magnitude relationship between pixels in each group, the receiver can retrieve the encrypted pixels \( \{c_1(k), c_r(k)\} \) correctly from Eq. (16) to Eq. (19).

If \( P_1^w(k) \geq P_r(k) \)

\[
c_1(k) = \begin{cases} 
\text{NTRU}.Add(c_1^w(k), -1) = c_1^w(k) - 1, & \text{if } d^w(k) \geq EP + 1 \\
c_1^w(k), & \text{else}
\end{cases}
\]

\[ c_r(k) = c_r^w(k) \]

else

\[
c_1(k) = \begin{cases} 
\text{NTRU}.Add(c_1^w(k), 1) = c_1^w(k) + 1, & \text{if } d^w(k) \geq EP + 1 \\
c_1^w(k), & \text{else}
\end{cases}
\]

\[ c_r(k) = c_r^w(k) \]

It is noting that, when scanning the final location \( z \), the data extraction and encrypted image recovery should stop in the \( n_e^{th} \) round. After extracting the additional data \( w \) in the \( n_e^{th} \) round, the receiver can turn to the \((n_e - 1)^{th}\) round for data extraction. Finally, after \( n_e \) rounds of data extraction and inverse operations of histogram shifting, the receiver can extract the additional data and retrieve the encrypted image completely.
2) EXTRACT HIDDEN DATA AND RECOVER COVER-IMAGE AFTER DECRYPTION

Differing from the previous method, the receiver has both the private key $sk$ and the data hiding key $kd$, which can extract the additional data and recover the cover-image after decryption.

1) With the private key $sk$, the receiver directly decrypts the received encrypted image with hidden data.

$$ P^w_r(k) = f \cdot c^w_r(k)(\text{mod } 2) $$

$$ P^w_r(k) = (2gs + ef) + f \cdot P^w_r(k)(\text{mod } 2) $$

$$ P^w_r(k) = (2gs + ef) + f \cdot P^w_r(k)(\text{mod } 2) $$

where \{P^w_r(k), P^w_r(k)\} are the corresponding plaintexts of \{c^w_r(k), c^w_r(k)\} in the $k^{th}$ group. When decrypting all the groups, the receiver can obtain the directly decrypted image with hidden data.

2) With the data hiding key $kd$, the absolute differences in each group are calculated by

$$ d^w(k) = |P^w_r(k) - P_r(k)| $$

Then, the receiver can completely extract the additional data $w$ by referring to Eq. (15). Meanwhile, the cover-image can be retrieved by the following formulations.

If $P^w_r(k) \geq P_r(k)$

$$ P_1(k) = \begin{cases} P^w_r(k) - 1, & \text{if } d^w(k) \geq EP + 1 \\ P^w_r(k), & \text{else} \end{cases} $$

$$ P_r(k) = P^w_r(k) $$

else

$$ P_1(k) = \begin{cases} P^w_r(k) + 1, & \text{if } d^w(k) \geq EP + 1 \\ P^w_r(k), & \text{else} \end{cases} $$

$$ P_r(k) = P^w_r(k) $$

After $n_e$ rounds of data extraction and inverse operations of histogram shifting, the additional data $w$ can be extracted completely and the cover-image can be retrieved correctly.

3) It is well known that the pixel values of a grayscale image are in the range [0, 255]. In most smooth images, in general, embedding additional data $w = 0/1$ does not cause overflow / underflow problems. However, in some textured images, overflow / underflow problem of the histogram shifting method should be discussed. The pixel value in our scheme may be 256 or -1. We can change the pixel value 256 to 255 and -1 to 0, and preserve the location map on these pixels. The location map is the binary bit stream BS where the minimum length of the BS is $L$. $L$ is sent to the receiver. In order to ensure that the cover-image can be restored without loss, the binary stream BS will also be first embedded in the ciphertext as auxiliary information. In this case, the additional data $w$ actually includes two parts: the auxiliary information BS and the secret information to be embedded.

For the receiver, the location map can be extracted from the directly decrypted image. After extracting the location map, the receiver can recover the cover-image without any error.

E. EXPLAIN THE PROCESS OF SCHEME WITH PRACTICAL EXAMPLES

1) NTRU IN IMAGE ENCRYPTION

The image owner divides image into groups of a reference pixel and an adjacent pixel according to Fig. 3 (a). Mark the reference pixel and adjacent pixel in the 1$^{th}$ group as $P_1(1) = 162$ and $P_r(1) = 160$, respectively. According to Eq. (4), the ciphertext of $P_1(1)$ and $P_r(1)$ is

$$ c_1(1) = hs + 2e + 162 $$

$$ c_r(1) = hs + 2e + 160 $$

2) DATA HIDING

Let the embedding bin $EP = 2$ and additional data $w = 1$. According to Eq. (5), the absolute difference $d(1)$ is obtained by

$$ d(1) = |c_1(1) - c_r(1)| $$

$$ d(1) = |hs + 2e + P_1(1) - (hs + 2e + P_r(1))| $$

$$ d(1) = |P_1(1) - P_r(1)| = 2 $$

At the same time, the data hider can obtain the magnitude relationships between $P_1(1)$ and $P_r(1)$. We have $P_1(1) \geq P_r(1)$ because of $c_1(1) \geq c_r(1)$. Finally, according to Eq. (6) and Eq. (7),

$$ c^w_1(1) = NTRU.Add(c_1(1), 1) = c_1(1) + 1 $$

$$ c^w_r(1) = c_r(1) = hs + 2e + 160 $$

Mark the directly decrypted versions of \{c^w_1(1), c^w_r(1)\} as \{P^w_1(1), P^w_r(1)\}, respectively. Due to the homomorphic addition of the NTRU, the effect of data hiding on plaintext pixels is to change \{P_1(1), P_r(1)\} to \{P^w_1(1), P^w_r(1)\}. In 1$^{th}$ group, according to Eq. (10) and Eq. (11),

$$ P^w_1(1) = P_1(1) + w = 162 + 1 = 163 $$

$$ P^w_r(1) = P_r(1) = 160 $$

3) DATA EXTRACTION AND IMAGE RECOVERY

1) Data extraction from encrypted domain

Similar to calculate the absolute difference in the process of data hiding, the absolute differences $d^w(1) = |c^w_1(1) - c^w_r(1)| = |163 - 160| = 3$ in the 1$^{th}$ group can be obtained by referring to Eq. (14). With the data hiding key $kd$, the receiver can obtain the embedding bin $EP$. By referring to $EP = 2$, the receiver can extract additional data $w = 1$ according to Eq. (15).

2) Data extraction and image recovery after decryption

Firstly, with the private key $sk$, the receiver directly decrypts the received encrypted image with hidden data
according to Eq. (20).
\[ P_i^{w_1}(1) = f \cdot c_i^{w_1}(1) \text{(mod 2)} \]
\[ = (2(g_s + e_f) + f \cdot P_i^r(1)) \text{(mod 2)} \]
\[ = P_i^r(1) = 163 \]
\[ P_i^{w_2}(1) = f \cdot c_i^{w_2}(1) \text{(mod 2)} \]
\[ = (2(g_s + e_f) + f \cdot P_i^r(1)) \text{(mod 2)} \]
\[ = P_i^w(1) = 160 \]
\[ (32) \]

where \( \{P_i^{w_1}(1), P_i^{w_2}(1)\} \) are the corresponding plaintexts of \( \{c_i^{w_1}(1), c_i^{w_2}(1)\} \) in the \( i^{th} \) group. Then, with the data hiding key \( K_d \), the absolute differences in the \( i^{th} \) group are calculated by \( d_i^w(1) = |P_i^{w_1}(1) - P_i^r(1)| = |163 - 160| = 3 \). Similarly, by referring to \( EP = 2 \), the receiver can extract additional data \( w = 1 \) according to Eq. (15). Finally, the original \( P_i(1) = P_i^{w_1}(1) - 1 = 162 \) and \( P_i(1) = P_i^{w_2}(1) = 160 \) can be retrieved by Eq. (22) and Eq. (23).

In order to better display the above example, the demonstration of NTRU execution on two pixels of cover-image in our example is depicted in Fig. 4.

**FIGURE 4.** Practical examples.

**FIGURE 5.** Four test images. (a) Lena. (b) Man. (c) Plane. (d) Lake.

**IV. SECURITY ANALYSIS**

The security of using NTRU to encrypt images is the basic requirement of the scheme in this paper. This section mainly analyzes the security of the proposed scheme. Since the correctness and security of the NTRU algorithm have been demonstrated in detail in the literature [31] from the perspective of numerical geometry and algebraic number theory, its security can be reduced to a SVP problem on the lattice [33] and the BDD (Bounded Distance Decoding) problem [34], which is a variant of a SVP problem. It is worth noting that the analysis features used in the existing steganalysis algorithms are mainly derived from the model based on the correlation of the plaintext image. Therefore, for the ciphertext data with random noise state, the traditional steganalysis technology is less effective. Anyway, the security of the RDH-ED algorithm, including this paper, is not analyzed from the perspective of ensuring the concealment of the embedded information [35]. Therefore, this section only discusses the potential attacks that the proposed scheme may encounter. We first list several difficult problem definitions used in security analysis, and then mainly consider known plaintext attack (KPA) and ciphertext only attack (COA).

**Definition 1 (SVP Problem):** Given any set of bases of lattice \( L \). If a non-zero lattice vector can be found that satisfies the following conditions: \( v \in L \), such that \( |v| = \lambda_1(L) \), this problem is called SVP problem.

**Definition 2 (B_1’s Security Is Equal to \( \alpha - \text{BDD} \)):** Given the lattice \( L \) and vector \( m_i \) (in the distance of \( \alpha \cdot \lambda_1(L) \)). Finding the lattice point \( B_i \in L \) where the distance \( m_i \) satisfies the distance condition \( \alpha \cdot \lambda_1(L) \) in the distance of \( \alpha \cdot \lambda_1(L) \) is a \( \alpha - \text{BDD} \) problem.

Let \( B_i \) and \( m_i \) be two different coefficient vectors, \( B_i \in L \). \( m_i \) is very close to \( L \). According to the \( \alpha - \text{BDD} \) problem, this problem is to find \( B_i \in L \) that satisfies \( ||B_i - m_i|| \) minimization. The BDD problem is a variant of a SVP problem. In the case that \( \lambda_1(L) \) is minimized, \( m_i \) must be closest to the lattice. For any given constant factor \( \alpha > 1/\sqrt{2} \), the \( \alpha - \text{BDD} \) problem has no solution in polynomial time.

**A. KNOWN PLAINTEXT ATTACK**

Under the KPA assumption, the attacker can obtain a large number of plaintext and ciphertext pairs. The attacker may obtain the parameters \( s_i, e_i \). Therefore, the attacker can perform the following calculations:

\[ c_{j0} - c_{j1} = E(m_i, 2, s_i, e_i) - E(m_i, 2, s_j, e_j) \]
\[ = h(s_i - s_j) + 2(e_i - e_j) \]
\[ (33) \]

However, according to the NTRU problem assumption, it is impossible for an attacker to obtain \( s_i \) and \( e_i \), which is
N. Zhou et al.: Separable RDH Scheme in Homomorphic Encrypted Domain Based on NTRU

FIGURE 6. PSNR versus embedding performance with different $T$. (a) Lena. (b) Man. (c) Plane. (d) Lake.

equivalent to solving the SVP problem on the lattice. In $h = 2g/f$, $g$ and $f$ are sampled from the Gaussian distribution $D_{2g,f}$. From the perspective of the attacker, the ciphertext only contains pseudo-random sampling, so the algorithm in this paper is safe under the assumption of KPA.

B. CIPHERTEXT ONLY ATTACK

Under the COA assumption, the attacker can obtain ciphertext, including encrypted boundaries. The plaintext $m_i$ and $B_i$ are encrypted using the same public key $h$ and parameter $s_i$, $e_i$. Different $m_i$ corresponds to different $B_i$. The attacker may derive the boundary $B_i$. Therefore, the attacker performs the following calculations:

\[ c_{j0} - c_{j1} = E(m_i, 2, s_i, e_i) - E(B_i, 2, s_i, e_i) = m_i - B_i \]  

(34)

However, according to the difficulty of the BDD problem, at the same time $m_i$ is encrypted, so the attacker cannot obtain $B_i$. That is to say, no other information can be obtained from the ciphertext. Therefore, the algorithm in this paper is safe under the assumption of COA.

V. EXPERIMENTAL RESULTS

In this section, in order to test the performance of the algorithm in this paper, 4 standard test images with a size of 512 × 512 in the USC-SIPI image library are selected for experiments as shown in Fig. 5. The hardware and software environment used in the experiment is as follows.

CPU: Intel (R) Core (TM) i7-5500U @ 3.60 GHz
RAM: 8 GB
OS: Windows 10
Programming: C ++ and MATLAB R2015b

A. EMBEDDING DISTORTION

Peak signal-to-noise ratio (PSNR) is the most commonly used criterion for evaluating quality between the cover-image and the directly decrypted image (plaintext image with hidden data).

\[ \text{MSE} = \frac{1}{MN} \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} (P_{ij} - C_{ij})^2 \]

\[ \text{PSNR} = 10 \times \log_{10} \left( \frac{255^2}{\text{MSE}} \right) \]  

(35)
where $M \times N$ is the size of the test images, $P_{ij}$ and $C_{ij}$ are the pixel value of the cover-image and the directly decrypted image respectively, MSE is the mean square error between the cover-image and the directly decrypted image.

According to the analysis of the section III-B, parameter $T$ has a greater impact on the PSNR versus embedding performance. For all the four tested images, the best embedding performance can be achieved by choosing proper $T$ values, as plotted in Fig. 6. From the comparison of the line chart, when $T = 1, 2, 4$ and the embedding rate (ER) is less than 0.1 bpp, the PSNR values are almost the same under the same embedding rate. Besides, when ER $> 0.1$ bpp, PSNR value under $T = 2$ is higher than $T = 1$ or 4, but still lower than the PSNR value under $T = 3$. In general, $T = 3$ is the best parameter in our scheme. This is because histogram of the absolute differences under $T = 3$ has the least pixel values between the peak point and zero point. In view of this, the number of invalid shifting pixels [36] is the least when embedding information by shifting histogram of the absolute differences. In order to further determine the optimal parameters of the proposed scheme, 100 gray images in the USC-SIPI image database were selected for further experiments. The comparison of average PSNR performance at different $T$ values is shown in Fig. 7. Obviously, the average PSNR value is highest at $T = 3$. This shows that the optimal parameter $T = 3$ in our scheme is reliable.

In the following experiments, we use experimental results under $T = 3$ to compare embedding distortion with other
TABLE 1. Comprehensive comparison between the proposed scheme and other state-of-the-art works.

| Method                        | Separable in theory | Maximum embedding rate (bpp) | How to generate and use redundancy | Protection mechanism of the hidden data |
|-------------------------------|---------------------|------------------------------|-------------------------------------|----------------------------------------|
| Method in [22]                | No                  | 0.5                          | VRBE                                | Homomorphic property of Paillier       |
| Method in [24]                | Yes                 | 1                            | VRBE                                | WPC mechanism                         |
| Method in [25]                | Yes                 | 0.249                        | VRBE                                | Homomorphic property of Paillier       |
| Method in [26]                | Yes                 | 0.5                          | VRAE                                | Homomorphic property of Paillier       |
| Method in [28]                | Yes                 | 0.67                         | VRAE                                | Homomorphic property of R-LWE          |
| The proposed scheme           | Yes                 | 0.8                          | VRAE                                | Homomorphic property of NTRU           |

state-of-the-art works. The proposed scheme is compared with the other state-of-the-art works of Shiu et al. [22], Zhang et al. [24], Xiang et al. [25], [26] and Xiong et al. [28] using four test images (in Fig. 5). Since the above 5 references and our paper are representative documents encrypted with public key cryptography, they are comparable. The results are presented in Fig. 8, which illustrates that our scheme shows excellent embedding performance. At the same embedding rate, the PSNR of our scheme is higher than that of the literatures [22], [24]–[26]. When the embedding rate is small, the PSNR in [28] is higher than our paper. This is because the absolute difference in [28] is sorted so that the pixels near the peak value are embedded first. It is an effective way to reduce histogram translation and reduce image distortion. However, the highest embedding rate, the more obvious the advantages of the proposed scheme.

B. PERFORMANCE COMPARISON

At last, we conduct a detailed analysis and comparison from the perspective of encryption methods and embedding rates. Comparison on other features is also listed in Tab. 1.

1) Literature [22] combined the Paillier cryptosystem with the DE algorithm, which is an inseparable algorithm. The receiver can only decrypt the image first and then extract the information. Method in [24] first vacates room before encryption. After encryption, additional data can be embedded through the histogram shifting, so that the hidden data can be extracted from plaintext domain. Combined with the feature of homomorphic encryption, WPC is used to embed additional data into the encrypted image. This makes it possible to extract the hidden data from encrypted domain. Likewise, the receiver with the corresponding key in [25] can extract the hidden data from plaintext domain or encrypted domain, respectively. In [26], the additional data is embedded into the encrypted image by shifting histogram of the absolute differences. Due to the homomorphic addition of the Paillier, the additional data can be extracted not only from encrypted domain, but also from plaintext domain. The separable method in [28] is similar to [26]. In summary, apart from the literature [22], all methods in Tab. 1 can realize the separation of data extraction and image decryption.

2) In RDH-ED, embedding rate is the index that needs to be improved. The embedding capacity of [22] can reach the same as the DE algorithm in the spatial domain, which is 0.5 bpp. Zhang et al. [24] used multiple bits for embedding so that the maximum embedding rate is 1 bpp. Xiang et al. [25] used the data to be embedded to form pseudo pixels, and replaced the target pixels to complete the embedding procedure. However, the embedding rate in [25] is only 0.249 bpp. Each additional data is embedded into two pixels in [26], and the maximum embedding rate is 0.5 bpp. Literature [28] proposed two additional data are embedded in a set of three pixels, which means the embedding rate will increase to 0.67 bpp in the best. The maximum embedding rate of this paper can increase to 0.8 bpp at most. Additionally, when \( T = 3 \), that is, the maximum embedding rate is 0.75 bpp, the algorithm has the best performance on PSNR.

3) In RDH-ED, the algorithms in Tab. 1 can be classified from the perspective of how to generate and use redundancy. Among them, literature [22], [24], and [25] use VRBE. In contrast, literature [26], [28] and the proposed scheme use VRAE.

4) Paillier cryptography is used for encryption in [22] and [24]–[26]. Literature [28] encrypted images using R-LWE. This paper uses NTRU to encrypt image in order to resist quantum computing attacks in the context of big data and cloud computing.

VI. CONCLUSION

This paper proposes a separable reversible data hiding scheme in homomorphic encrypted domain based on NTRU. Differing from VRBE, the VRAE used in this paper does not require any preprocessing of the cover-image. For RDH-ED, the encryption operation has maximized the information entropy of the encrypted image. From the perspective of information theory, it is very difficult to hide information in encrypted image, which is also a difficult problem in this field. The VRAE method in this paper not only solves this difficult problem, but also improves the embedding capacity in encrypted domain and corresponding PSNR value of the algorithm for the hiding process. Our scheme is devoted to finding the \( T \) value, which can make the best use of the correlation between adjacent pixels. The experimental results show that the proposed scheme has the best embedding performance when \( T = 3 \). More importantly, compared with other state-of-the-art works, the PSNR value of our scheme is the highest at the same embedding rate. At present, with the growing demand for network security and privacy protection, a large number of public key cryptographic technologies will be used in the future. The embedding and extraction technology in encrypted domain of the proposed scheme can be widely applied to further enrich the application scenarios of RDH-ED. In the future, our research work will focus on further improving the security of the scheme and the embedding performance.
YAN KE received the B.S. degree in information research and security and the M.S. degree in cryptography from the Engineering University of Chinese People Armed Police Force, Xi’an, China, in 2014 and 2016, respectively, where he is currently pursuing the Ph.D. degree in cryptography. His research interests include information hiding, lattice cryptography, and deep learning.

FUQIANG DI received the M.S. and Ph.D. degrees in cryptography from the Engineering University of Chinese People Armed Police Force, Xi’an, China, in 2015 and 2019, respectively. He is currently with the Key Laboratory of Network and Information Security Under Chinese People Armed Police Force as a Lecturer. His research interests include reversible data hiding and steganalysis.