IACR Communications in Cryptology
ISSN 3006-5496, Vol. 1, No. 2, 26 pages.

https://doi.org/10.62056/a09qxrxqi

Efficient Post-Quantum Pattern Matching on Encrypted Data

Anis Bkakria[1] and Malika Izabachène[2]

[1] IRT SystemX, France

[2] Unaffiliated, France

Abstract. Pattern matching methods are essential in various applications where users must disclose highly sensitive information. Among these applications are genomic data analysis, financial records inspection, and intrusion detection processes, all of which necessitate robust privacy protection mechanisms. Balancing the imperative of protecting the confidentiality of analyzed data with the need for efficient pattern matching presents a significant challenge.

In this paper, we propose an efficient post-quantum secure construction that enables arbitrary pattern matching over encrypted data while ensuring the confidentiality of the data to be analyzed. In addition, we address scenarios where a malicious data sender, intended to send an encrypted content for pattern detection analysis, has the ability to modify the encrypted content. We adapt the data fragmentation technique to handle such a malicious sender. Our construction makes use of a well-suited Homomorphic Encryption packing method in the context of fragmented streams and combines homomorphic operations in a leveled mode (i.e. without bootstrapping) to obtain a very efficient pattern matching detection process.

In contrast to the most efficient state-of-the-art scheme, our construction achieves a significant reduction in the time required for encryption, decryption, and pattern matching on encrypted data. Specifically, our approach decreases the time by factors of 1850, 10^6, and 245, respectively, for matching a single pattern, and by factors of 115, 10^5, and 12, respectively, for matching 2^{10} patterns.

Keywords: Pattern matching over encrypted strings · streamed encryption · data fragmentation · multiple distance computation.

1 Introduction

While data encryption is undoubtedly a step forward in protecting users' privacy and confidentiality on the Internet, it raises significant security concerns. The continued increase in the use of data encryption of network traffic provides cybercriminals with new opportunities to hide malware from protection systems. This is because secure communication protocols (e.g., TLS) are generally based on standard encryption algorithms (e.g., RSA, AES) that do not allow partial processing of encrypted data, making it difficult to detect intrusion on the network. Intrusion Detection on encrypted streams is not feasible. To overcome the previous limitation, several encryption algorithms allowing a specific set of functions to be performed on encrypted data have been proposed.

E-mail: anis.bkakria@irt-systemx.fr (Anis Bkakria), malika.izabachene@gmail.com (Malika Izabachène)

Received: 2024-04-09 Accepted: 2024-06-03

1.1 Related Work

Pattern matching over encrypted data can be provided using versatile techniques such as fully homomorphic encryption (FHE) [Gen09a, Gen09b] and functional encryption (FE) [BSW11], which allow the evaluation of arbitrary functions over encrypted data. Still, these techniques are often impractical because of their very high complexities. To overcome the previous limitations, several searchable encryption techniques (e.g., [ABC$^+$08, CS15, KMO18, SWP00]) allowing to perform specific functions have been proposed. The main idea consists of (1) allowing authorized entities to search for specific predefined (i.e., chosen before the data encryption) patterns within a given encrypted data, and (2) preventing unauthorized entities from learning any information about the encrypted data. Nevertheless, since the patterns to be searched should be pre-chosen before data encryption, most existing searchable encryption techniques support only specific types of searches such as database searches where records are already indexed by the pattern to be searched, and are useless in case in which the set of patterns to be searched are not known to the data encryptor before encryption. To circumvent the previous limitation, the authors in [SLPR15] propose BlindBox, a tockenization-based approach that divides the data to be encrypted into segments of the same size ℓ, encrypts each of those segments using a searchable encryption scheme where each segment represents a pattern that can be searched. Nonetheless, BlindBox is useful only when all patterns are of the same length, which is clearly a very strong assumption. In addition, even if all the patterns to be searched have the same length, BlindBox does not provide a correct detection since it cannot detect a pattern that straddles two fragments. In [CDK$^+$17], BlindIDS, a public key variant of BlindBox has been proposed. It additionally ensures pattern indistinguishability to the entity performing the pattern matching. Yet, BlindIDS suffers from the same limitations of BlindBox.

To overcome the aforementioned limitations of BlindBox and BlindIDS, a new provably correct (i.e., without false negatives or false positives) approach allowing patterns of arbitrary length to be searched against encrypted data using constant-size trapdoor has been proposed in [DFOS18]. However, the previous improvement comes at the cost of a few weaknesses which heavily impact the practicability of the technique. First, the size of the public key required by [DFOS18] significantly increases with the size of the data to be analyzed and the considered alphabet. Second, the number of required pairings for testing the presence of a pattern at some position depends linearly on the maximum occurrence of a same symbol in the pattern which significantly increases the computational cost of the matching algorithm. Third, the security of [DFOS18] relies on a quite strong, ad-hoc interactive assumption.

In [BCC20], the authors successfully addressed some of the aforementioned limitations. The work introduces a promising fragmentation approach (see Section 2.5 for more details) that consists of splitting the data to be analyzed into two sets of non-overlapping fragments $\mathcal{F}$ and $\overline{\mathcal{F}}$ such that, $\forall F_i \in \mathcal{F}, \overline{F}_i \in \overline{\mathcal{F}}$, F_i straddles $\overline{F}_i$. Intuitively, the fragmentation is performed in such a way that, if a pattern is present in the data to be analyzed, regardless its position, it will be entirely contained by a fragment in $\mathcal{F}$ or $\overline{\mathcal{F}}$. Thanks to this fragmentation technique, [BCC20] reduces significantly the size of the required public key by transforming the problem of searching on large data to the one of searching on several small data fragments. In addition, compared to [DFOS18], the construction proposed in [BCC20] reduces the pattern matching complexity, since regardless of the length and the content of the pattern to be matched, it requires only two pairing operations to test the presence of the pattern at one position of the data fragment. Recently, relying on the aforementioned fragmentation technique, [BCS21] proposed a construction that further reduces the size of the required public key and improved the security by considering EXDH [CPST15], a static variant of the Decision Diffie-Hellman (DDH) assumption. Very recently, it has been shown in [BCS23] that the proposed fragmentation technique can be used to build a generic transformation from Hidden Vector Encryption to Stream Encryption supporting Pattern Matching leading to better features than existing ones. Unfortunately, the constructions proposed in [BCC20, BCS21, BCS23] are still suffering from the following two main limitations.

First, the data fragmentation technique used in these constructions is performed by the data sender before encryption. Hence, the fragmentation should be honestly performed by the data sender to allow the pattern matching service provider to correctly perform pattern matching on the encrypted data. Nevertheless, the considered constructions are mainly proposed to bring solutions allowing to perform deep packet inspection (DPI) on encrypted traffic. Obviously, in the DPI scenario, the traffic sender should rather be considered as a malicious entity as it may intentionally include malicious content in the traffic. As they are defined, the constructions in [BCC20, BCS21, BCS23] do not allow the pattern matching service provider to check whether the fragmentation is correctly performed by the data sender. Hence, the latter can easily disable pattern detection by making the malicious pattern straddling two fragments F_i and F_{i+1} in $\mathcal{F}$ and replacing the data in fragment $\overline{F}_i$ by random content.

Decryption would enable detection of crafted streamed content, however decrypting a malicious message without processing might be an issue, especially in scenarios where the receiver outsources the detection process to a third party. In such a case, there's a significant risk associated with retaining decrypted malicious content in the receiver's memory. Without host-based malware detection capabilities, the decrypted content poses a tangible threat in-memory execution. This could potentially exploit vulnerabilities within the receiver's operating system, leading to severe consequences. In this work, we address this concern by avoiding decrypting the message in the case that it matches the pattern.

As such, the constructions proposed in [BCC20, BCS21, BCS23, DFOS18] did not explicitly consider a decryption algorithm. The authors of these constructions argue that decryption functionality can be straightforwardly added either (1) by encrypting the exchanged data stream under a conventional encryption scheme, or (2) by issuing a trapdoor for all characters in the alphabet used to encode the exchanged data. Unfortunately, as explained above, when considering DPI scenario, the data sender should be considered as a malicious entity which makes the solution (1) useless since it can allow the sender to easily disable the detection by sending the malicious content only in the data encrypted using the conventional encryption scheme. In addition, the usage of the solution (2) to decrypt data requires a number of pairings that grows linearly with the size of the exchanged data and the size of the considered alphabet, which quickly becomes cumbersome (See Section 5).

1.2 Our Contributions

Considering the aforementioned state of the art, in this paper, we propose a construction that improves the correctness, the efficiency, and enhances the security model of the existing pattern matching on encrypted data solutions.

Correctness in the presence of malicious data sender. As discussed in section 1.1, the data fragmentation technique introduced in [BCC20] helps to reduce the size of the public key as well as detection complexity, but it could make the proposed constructions useless when a malicious sender is considered. In this work, we design a construction that can operate the data fragmentation technique while allowing to correctly perform pattern matching over encrypted data when the latter is fragmented and encrypted by a malicious sender. More specifically, our construction requires the sender to fragment and encrypt only the set of fragments $\mathcal{F}$, then allows the pattern matching service provider to build the encrypted data in the set of fragments $\overline{\mathcal{F}}$ on the fly (at detection time) to check if the pattern straddles two consecutive fragments in $\mathcal{F}$. While our goal is not to detect malicious behaviour per se, we aim to ensure detection correctness even in the presence of

Table 1: Complexity and ensured properties comparison between related work and our primitive. The scalars $m, L, |\mathcal{S}|$ denote respectively the length of the traffic to encrypt, the length of the longest pattern to be queried, and number of characters in the plaintext alphabet. As all the schemes we compare are based on the fragmentation paradigm, the comparison is done using one fragment i.e. $n = m$ where n is the length of one fragment. The "PQ" row specifies whether the construction is post-quantum or pre-quantum, the "MS" row specifies whether the construction can deal with malicious data sender, and the "PI" row specifies whether the construction is pattern indistinguishable (i.e., the entity performing the pattern matching can learn nothing about the matched patterns). In addition, we used (✓) to denote that the property is provided under specific conditions. For each scheme, we report the running time for the encryption, decryption, and pattern matching (Test) operations where $n = 1000,\ L = 100$, and $|\mathcal{S}| = 256$. All measurements are taken on the same system (see Section 5 for the details of our experimental setup).

Scheme		SEST [DFOS18] *	AS^{3}E [BCC20] *	SEPM [BCS21] *	This work †
Structure		Pairings	Pairings	Pairings	Lattices
MS		✗	✗	✗	✓
PI		✗	(✓)	✗	✓
PQ		✗	✗	✗	✓
Enc	Complexity	$2n$	$4n$	$4n + n/L$	$n/(2L)$
	Time ‡	184 ms	361 ms	370 ms	0.2 ms
Dec	Complexity.	$n\|S\|$	$n\|S\|$	$n\|S\|$	$N/(2L)$
	Time ‡	$7 \cdot 10^4$ ms	$7 \cdot 10^4$ ms	$7 \cdot 10^4$ ms	0.04 ms
Test	Complexity.	$n(L + 2)$	$2n$	$2n$	n/L
	Time ‡	3129 ms	196 ms	196 ms	0.8 ms

* For pairing-based constructions, encryption operation complexity is expressed in number of group exponentiations, while decryption and test operations complexities are expressed in number of pairings.

† For our construction, the encryption, decryption, and test operations complexities are expressed respectively in the number of homomorphic encryption, number of homomorphic decryption, and number of homomorphic multiplication.

‡ For reported experiments, we use Barreto-Naehrig curve BN254 [PSJNB11] for instantiating SEST, AS^{3}E, and SEPM, and BFV [Bra12, FV12] with $N = 2048$ for our construction.

adversarial attempts to cause false negatives by modifying the contents. Thus, our threat model addresses these misbehaviors to guarantee detection correctness, against active senders trying to evade detection.

Performance and security improvement. In contrast to [BCC20, BCS21, DFOS18] which are using pairing-based constructions, the construction we propose in Section 4 relies on Ring-Learning With Error (R-LWE) based somewhat homomorphic encryption (SHE). The benefits are threefold. First, compared to the most efficient state of the art scheme [BCS21], our construction successfully manages to reduce the time needed for encryption, decryption, and matching a pattern on encrypted data by a factors of 370, $7 \cdot 10^4$, and 200 respectively. Another appealing feature of our scheme is the ability to ensure pattern indistinguishability to the entity performing the pattern matching operation. Finally, our construction is secure against quantum attacks assuming that the ring LWE assumption is post-quantum secure, i.e., hard to break in quantum polynomial-time.

The construction we propose in this paper enables the previous contributions while providing all the advantages of [BCC20, BCS21], namely, arbitrary-length pattern matching on arbitrary ciphertexts as well as data and pattern indistinguishability to the entity performing pattern matching. To give a better understanding of the advantages of our construction compared to [DFOS18, BCC20, BCS21], we provide in Table 1 a comparative overview of their asymptotic complexities, their ability to deal with malicious data sender, and their provided security properties. We provide in Section 5 a more detailed evaluation of the efficiency of our proposed construction.

Paper outline. The paper is organized as follows. Section 2 gives the preliminaries. Section 3 introduces the system and the security model considered in this work. Section 4 details our construction and proves the security results. Section 5 details the performance of our construction. Finally, Section 6 concludes the paper.

2 Preliminaries

2.1 Notation

We use the symbols $\mathbb{Z}$ and $\mathbb{Q}$ to denote the ring of integers and the field of rational numbers respectively.

For a vector $\mathbf{V} = (v_0, v_1, \cdots, v_{k-1}) \in \mathbb{Q}^k$, we denote by $\|V\|_\infty$ the ∞-norm defined as $\max_{i=0}^{k} |v_i|$. Moreover, for $0 \leq i \leq \ell - k$, we denote by $V^{(i)} = (v_i, v_{i+1}, \cdots, v_{i+\ell-1})$ the i-th sub-vector of $\mathbf{A}$ of length $\ell \leq k$. Furthermore, we denote by $\langle V_1, V_2 \rangle$, $\mathsf{hw}(V)$ and $\mathsf{d}(V_1, V_2)$ the inner product of two equal size vectors, the Hamming weight (number of non-zero coefficients) of a vector and the Hamming distance between two binary vectors respectively.

We use N to denote the polynomial degree which is a power-of-two integer and define the base ring $R := \mathbb{Z}[X]/(X^N + 1)$. For a polynomial $a := \sum_{i=0}^{N-1} a_i X^i$, the infinite norm is defined as $\|a\|_\infty := \max_i a_i$ and the expansion factor of a ring R is defined as $\delta := \sup\{\frac{\|ab\|}{\|a\|\|b\|}, a, b \in R^\star\}$. By taking R as the cyclotomic polynomial equals to $X^N + 1$, the worst-case bound for the polynomial multiplication expansion factor is $\delta = N$[1]. In our construction, the expansion factor will be used to characterize the worst-case noise growth.

2.2 Gaussian distributions

Let $d \geq 1$ and $\sigma \geq 0$. For all $\mathbf{x} \in \mathbb{R}^d$, we denote by $\rho_\sigma(\mathbf{x}) = \exp(-\|\mathbf{x}\|^2 / \sigma^2)$ the Gaussian function centered at $\mathbf{0}$ over $\mathbb{R}^d$. We will use χ_σ to denote a distribution on R statistically close to a discrete centered Gaussian distribution of standard deviation σ. We suppose that χ_σ is B-bounded, which means that if an integer vector u is sampled from χ_σ, then $\|u\|_\infty \leq B$, and we have $B \approx \sigma\sqrt{N}$ (with high probability). In our construction, the secret and noise often follow the same B-bounded distribution, though in practice, we often take the ternary uniform distribution over $\{-1, 0, 1\}$.

2.3 FHE ciphertexts

In this section, we introduce the definitions of FHE ciphertexts encrypting either ring or integer elements.

2.3.1 RLWE encryption.

We make use of the ring LWE encryption from [FV12, Bra12], called BFV. It is defined using parameters N, q, t, and σ. In most of the BFV implementations, the modulus q is taken as a prime number satisfying $q \equiv 1 \mod 2n$, which defines the base ring $R_q = R/qR$ of the ciphertext space. The modulus $t \leq q$ is used to define a plaintext space $R_t = R/tR$. We denote $\Delta := \lfloor \frac{q}{t} \rfloor$ and the reminder on division of q by t as $r_t(q)$.

- $\mathsf{rlwe.KeyGen}$: we take $s \in R$ where each of its coefficients is drawn from χ_σ such that $s \in R$, and sample $p_1 \in R_q$ at random and an error $e \leftarrow \chi_\sigma$ such that $e \in R$. We set the public key $\mathsf{pk} = (p_0, p_1)$ with $p_0 = -(p_1 \cdot s + e) \in R_q$ and $\mathsf{sk} = s$ as

[1]We will take $\delta = N$. However, in practice, it was shown that δ is much closer to $C\sqrt{N}$, with C not higher than 2.

the secret key. It additionally defines a relinearization key that is used to perform homomorphic multiplication between polynomial ciphertexts. The public parameters contains $(N, t, q, \mathsf{pk}, \mathsf{rlk})$.

- $\mathsf{rlwe.Enc}$: given the public key $\mathsf{pk} = (p_0, p_1)$ and a plaintext $m \in R_t$, the encryption procedure samples $e_0, e_1 \in R$ with coefficients sampled from χ_σ. It then computes a "fresh" ciphertext as: $(c_0, c_1) = (p_0 \cdot u + e_0 + \Delta m,\ p_1 \cdot u + e_1) \in R_q^2$. In the following, a ring LWE (or RLWE) encryption of m under public key pk will be denoted as $\mathsf{rlwe.Enc}_{\mathsf{pk}}(m)$ or $\mathsf{rlwe.Enc}(m)$ when pk can be made implicit.

- $\mathsf{rlwe.Dec}$: given a secret key $\mathsf{sk} = s$ and (c_0, c_1), it computes $c_0 + s \cdot c_1 \in R_q$ which gives $\Delta \cdot m - e \cdot u + e_1 \cdot s + e_0$. It outputs $\lfloor \frac{m^\star}{\Delta} \rceil$. The error associated to the ciphertext (c_0, c_1) is denoted $\mathsf{Err}((c_0, c_1)) := \lfloor \frac{c_0 + s \cdot c_1 - \Delta \cdot m}{\Delta} \rceil$. In the following, a ring LWE decryption of a ciphertext c under secret key sk will be denoted as $\mathsf{rlwe.Dec}_{\mathsf{sk}}(m)$.

2.3.2 Homomorphic Operations.

For sake of simplicity, we present the homomorphic operations in an abstract way and develop the noise growth analysis later in our constructions sections. We let c and c' be two ciphertexts encrypting m and m' respectively:

- $\mathsf{Add}(c, c')$: given two ciphertexts c and c' in input, encrypting m and m' respectively, it outputs $c + c'$ which is an encryption of $m + m'$ and we have: $\|\mathsf{Err}(\mathsf{Add}(c, c'))\|_\infty \leq \|\mathsf{Err}(c)\|_\infty + \|\mathsf{Err}(c')\|_\infty$.

- $\mathsf{pMult}(a, c)$: given polynomial a and an RLWE ciphertext c encrypting m, it outputs $a \cdot c$ which is an RLWE encyption of $a \cdot m$ such that $\|\mathsf{pMult}(a, c)\|_\infty \leq \delta \|a\| \cdot \mathsf{Err}(c)$.

- $\mathsf{Mult}(\mathsf{rlk}, c, c')$: it computes the product of c and c', which gives a ciphertext of $m \cdot m'$ under secret s^2 which induces a first error; it then makes use of rlk to switch back to an encryption of $m \cdot m'$ under secret s, which adds an additional noise. Relinearization can be optional or postponed later in the computation at the price of having larger ciphertexts.

- $\mathsf{Rotate}(c, j)$: given a RLWE ciphertext $c \in \mathsf{rlwe.Enc}_{\mathsf{pk}}(m(X))$, $m(X) = \sum_{i=0}^{N-1} m_i X^i$ in $R_t[X]$ and an index $j \in [0, N-1]$, it outputs an RLWE encryption of $m'(X)$, given by:

$$\begin{aligned} X^j \cdot m(X) &= \sum_{i=0}^{N-1-j} m_i X^{i+j} + \sum_{i=N-j}^{N-1} m_i X^{i+j} \cdot X^{-N} \\ &= \sum_{k=0}^{j-1} -m_{k+N-j} X^k + \sum_{k'=j}^{N-1} m_{k'-j} X^{k'}, \end{aligned}$$

  taking $k = i + j - N$ and $k' = i + j$. In other words, it outputs a polynomial whose coefficients are rotated by j positions. Note that if $j < N$, then $\mathsf{Rotate}(c, j)$ outputs an RLWE encryption of $\sum_{i=0}^{N-1-j} m_i X^{i+j} = \sum_{k=j}^{N-1} m_{k-j} X^k$.

If (c_0, c_1) is a fresh encryption of m, note that we have:

$$\begin{aligned} \|c_0 + s \cdot c_1 - \Delta \cdot m\|_\infty &\leq \|-e \cdot u\|_\infty + \|e_1 \cdot s\|_\infty + \|e_0\|_\infty \\ &\leq 2\delta B^2 + B = B(2\delta B + 1) \end{aligned}$$

And if the norm of the noise of a fresh ciphertext is such that $\|\mathsf{Err}((c_0, c_1))\|_\infty \leq \frac{\Delta}{2}$, then $\mathsf{rlwe.Dec}$ outputs the correct message, which after calculation amounts to have $2\delta B^2 + B = B(2\delta B + 1) \leq \frac{\Delta}{2}$.

2.4 Multiple Hamming Distances computation

In [YSK$^+$14], Yasuda et al. proposed an approach allowing to homomorphically and simultaneously perform secure multiple Hamming distances computation on encrypted data items. In this section, we recall how the computation is performed, it relies on the data packing and the secure multiple inner products methods we first review in the following.

Given a data string encoded as a vector $\mathbf{A} = (a_0, a_1, \cdots, a_{k-1}) \in \mathbb{Z}_t^k$ of length $k \leq N$. The data packing method from [YSK$^+$14] produces two types of packed plaintext vectors as follows:

$$\mathsf{pm}_1(\mathbf{A}) = \sum_{i=0}^{k-1} a_i \cdot X^i \quad \text{(type 1)} \qquad \text{and} \qquad \mathsf{pm}_2(\mathbf{A}) = -\sum_{i=0}^{k-1} a_i \cdot X^{N-i} \quad \text{(type 2)}$$

A packed cipherext is then created as:

$$\mathsf{ct}_p^{(i)}(\mathbf{A}) = \mathsf{rlwe.Enc}_{\mathsf{pk}}(\mathsf{pm}_i(\mathbf{A}))$$

Based on the observation that the product $\mathsf{pm}_1(\mathbf{A}) \cdot \mathsf{pm}_2(\mathbf{B})$ of a type 1 and a type 2 packed polynomials $\mathbf{A}$ and $\mathbf{B}$ of length k and $\ell \leq k \leq N$ respectively gives:

$$\begin{aligned}\left(\sum_{i=0}^{k-1} a_i \cdot X^i\right) \cdot \left(-\sum_{j=0}^{\ell-1} b_j \cdot X^{N-j}\right) &= \sum_{j=0}^{\ell-1}\sum_{i=0}^{k-1} a_i b_j \cdot X^{i-j} \\ &= \sum_{j=0}^{\ell-1}\sum_{h=-j}^{k-1-j} a_{h+j} b_j \cdot X^h \text{ taking } h = i-j \\ &= \sum_{h=0}^{k-\ell}\sum_{j=0}^{\ell-1} a_{h+j} b_j \cdot X^h + \text{ term of degree } \geq k-\ell+1\end{aligned}$$

The packed ciphertexts defined above can then be used to perform secure multiple inner products "simultaneously". The homomorphic multiplication between $\mathsf{ct}_p^{(1)}(\mathbf{A})$ and $\mathsf{ct}_p^{(2)}(\mathbf{B})$ gives a ciphertext whose decrypted polynomial $m(X) = \sum_{i=0}^{N-1} m_i \cdot X^i$ coefficients verifies:

$$\forall i \in [0, k-\ell[, \quad m_i = \left\langle \mathbf{A}^{(i)}, \mathbf{B} \right\rangle \mod t$$

As observed in [YSK$^+$14], the Hamming distance between two-equal size vectors $\mathbf{A}^{(h)}$ and $\mathbf{B}$ can be computed as:

$$\mathsf{d}(\mathbf{A}^{(i)}, \mathbf{B}) = \mathsf{hw}(\mathbf{A}^{(i)}) + \mathsf{hw}(\mathbf{B}) - 2\left\langle \mathbf{A}^{(i)}, \mathbf{B} \right\rangle$$

In the following, we will replace Add with $+$ and Mult, pMult with $\cdot$. Applying the method from [YSK$^+$14], multiple Hamming distances can then be homomorphically computed as:

$$\mathsf{ct} = \mathsf{ct}_p^{(1)}(\mathbf{A}) \cdot C + \mathsf{ct}_p^{(2)}(\mathbf{B}) \cdot C' - 2\mathsf{ct}_p^{(1)}(\mathbf{A}) \cdot \mathsf{ct}_p^{(2)}(\mathbf{B}), \tag{1}$$

where $C := -\sum_{i=0}^{\ell-1} X^{N-i}$, $C' := \sum_{i=0}^{k-1} X^i$.

Informally, the previous states that the homomorphic operations in Equation 1 over packed ciphertexts allow to compute simultaneously $k - \ell$ Hamming distances at the expanse of two homomorphic additions and three homomorphic multiplications. The

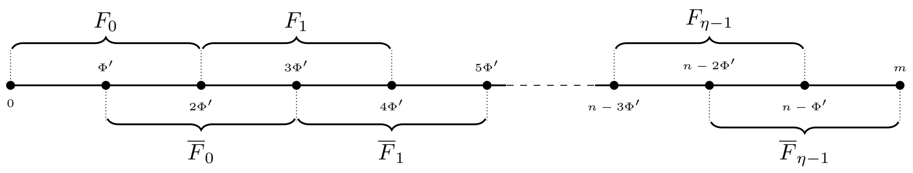

Figure 1: Fragmentation of a data string $\mathcal{B}$ of size $n = m + 1$

authors of [YSK+14] have further shown that the cost of secure multiple Hamming distances computation can be reduced by rewriting Equation (1) as

$$\mathsf{ct} = \frac{1}{2}\left[\Big((2\mathsf{ct}_p^{(1)}(\mathbf{A}) + C') \cdot (2\mathsf{ct}_p^{(2)}(\mathbf{B}) + C)\Big) + C' \cdot C\right] \tag{2}$$

Since $C' \cdot C$ can easily be pre-computed, compared to Equation 1, Equation 2 requires only one homomorphic multiplication and 3 homomorphic additions.

2.5 Data Fragmentation

Recently, [BCC20] proposed an effective fragmentation approach that allows to reduce the pattern matching over a large string to a fixed-length pattern matching method. It can be applied on any pattern matching over encrypted data stream.

We detail below the fragmentation techniques on a string $\mathcal{B}$ to be encrypted, where n the number of symbols in $\mathcal{B}$ and ℓ_m is an upper bound on the maximal length of the pattern to be searched. A fragment can be a stumbling fragment or fragment, i.e. on the top of Figure 1 or an overlined fragment, i.e. a fragment on the bottom of Figure 1, both are of size $2\Phi'$. To simplify the presentation, we take the size of a pattern ℓ equals to ℓ_m. We suppose that there exists η such that $n = (\eta - 1) \cdot 2\Phi' + \Phi' = (2\eta + 1)\Phi'$ i.e., $\mathcal{B}$ can be split into η distinct pairs of fragments and overlined framgents, each one being of size $2\Phi'$.

As in [BCC20, BCS21], for each index $i \in [0, \eta - 1]$, we call $F_i = [2\Phi' \cdot i, 2\Phi' \cdot (i + 1)[$ a *stumbling fragment* or *fragment* and $\overline{F}_i = [2\Phi' \cdot i + \Phi', 2\Phi' \cdot (i + 1) + \Phi'[$ an *overlined fragment*. Therefore, in the sequel, η will be the number of *fragments* and the number of *overlined fragments*, and Φ' is the offset between fragments and overlined ones.

This fragmentation approach has several remarkable properties. First, for any pattern of size $\ell \leq \ell_m$, if the pattern exists in the string $\mathcal{B}$ to be encrypted, then there exists an (overlined) fragment that contains the pattern. The previous property allows to perform pattern matching on a string of arbitrary length by processing each fragment independently, which makes this fragmentation technique perfectly suited to stream encryption.

As we previously mentioned, when considering malicious pattern matching over encrypted data use-cases, the data sender should be considered as a malicious entity as he/she may intentionally include malicious content inside the data to be sent to the receiver. The aforementioned fragmentation technique cannot be used as such when the data sender is considered as a malicious entity. In fact, as depicted in Figure 2, the malicious data sender can then try to bypass the detection by making the malicious content (represented in red in the figure) straddling two consecutive fragments F_i and F_{i+1} and replacing the data in fragment $\overline{F}_i$ by random content.

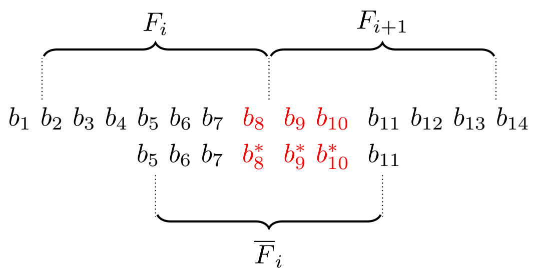
F_i
F_{i+1}
b_1 b_2 b_3 b_4 b_5 b_6 b_7 b_8 b_9 b_{10} b_{11} b_{12} b_{13} b_{14}
b_5 b_6 b_7 b_8^* b_9^* b_{10}^* b_{11}
$\overline{F}_i$

Figure 2: Bypassing the detection by making the malicious content straddling two fragments F_i and F_{i+1} and changing the content of $\overline{F}_i$.

Hence, if the pattern matching service provider does not have the ability to check whether or not the data has been correctly fragmented by the sender, which unfortunately is the case for the constructions proposed in [BCC20, BCS21], then the malicious data sender can easily bypass the pattern matching.

In the construction we propose in this paper, we overcome the previous limitation by requiring the data sender to build only the fragments $F_i, i \in [0, \eta - 1]$ and allowing the pattern matching service provider to construct the fragments $\overline{F}_i, i \in [0, \eta - 1]$ on the fly, at the detection time, without disclosing any information about the data to be analysed to the sender.

Moreover, the size of the data to be analyzed n as well as the number of data fragments η do not need to be known in practice in order to encrypt the data, they are only required for the formal definition of the construction we propose.

3 System specification and security model

In this section, we introduce the system model, system definition, threats model and security requirements of our construction.

3.1 System Model

The system we consider in this work is composed of four entities: a data sender $\mathcal{S}$, a data receiver $\mathcal{R}$, a pattern provider $\mathcal{PP}$, and a cloud service provider $\mathcal{CSP}$. $\mathcal{PP}$ is the entity that provides the detection pattern that will be searched. The $\mathcal{CSP}$ represents stakeholders that provides computation infrastructures which will be used to perform pattern matching operations on the data to be analyzed i.e., the data sent by $\mathcal{S}$ to $\mathcal{R}$. The data sender $\mathcal{S}$ is used to represent any entity that generates the data that will be processed by the receiver $\mathcal{R}$ e.g., a server that provides web contents. The data receiver $\mathcal{R}$ represents the entity that will receive and process the traffic sent by $\mathcal{S}$. As in [BCC20, DFOS18], we emphasize that the data receiver and sender roles are interchangeable as during the same secure network connection session, each entity may play both the data sender and the receiver roles. Inline with the deep packet inspection use-case we are considering, we suppose that $\mathcal{R}$ aims to analyze (e.g., perform deep packet inspection to detect malicious content) the data that are to be sent by $\mathcal{S}$ before processing it.

3.2 Adversarial model

We consider the entities $\mathcal{PP}$, $\mathcal{CSP}$ as *honest-but-curious*. That is, $\mathcal{PP}$ is expected to provide valid detection patterns allowing an effective analysis of the data exchanged between $\mathcal{S}$ and $\mathcal{R}$ that we denote $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$. This is a fairly reasonable assumption since the issuance of misleading or incorrect patterns will result in many false positives which may considerably degrade the quality of the analyses that will be provided to the $\mathcal{R}$ and may seriously affect $\mathcal{PP}$'s reputation. However, we consider $\mathcal{PP}$ to be curious as it may try to infer as much information about $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$. In addition, we expect $\mathcal{CSP}$ to perform honestly the pattern matching operations on $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$ using $\mathcal{PP}$'s analysis patterns. Nevertheless, we suppose that $\mathcal{SP}$ will try to derive information about either or both $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$ and $\mathcal{PP}$'s analysis patterns.

Moreover, we expect that $\mathcal{PP}$ and $\mathcal{CSP}$ will not collude to infer more information about $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$. As in [BCC20], we believe that this last assumption is fairly reasonable since, in a free market environment, an open dishonest behavior will result in considerable damages for involved entities.

Distinguishing itself from existing approaches, our work takes into account the sender $\mathcal{S}$ as a potential malicious entity with the intention of incorporating harmful code within the data intended for transmission to the receiver $\mathcal{R}$. Consequently, $\mathcal{S}$ possesses the capability to deviate from the prescribed protocol, thereby causing the pattern matching of the malicious code to fail. Nevertheless, it is crucial to note that any deviation employed by the sender must still enable the receiver to reconstruct the identical malicious code. Otherwise, the malicious code would be rendered useless and incapable of being executed by the receiver.

Finally, we require the approach we are proposing in this paper to provide correct pattern matching. That is, if an analysis pattern plaintext matches (resp. does not match) $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$'s plaintext, then the analysis pattern should be matched (resp. should not be matched) by our construction.

3.3 Syntax for Pattern Matching over Encrypted strings

Our pattern matching construction over encrypted data is defined using the following six algorithms that we denote `Setup`, KeyGen, `Issue`, `Encrypt`, `Match`, and `Decrypt`. The `Setup`, KeyGen, and `Decrypt` algorithms are run by the data receiver $\mathcal{R}$. The `Issue` is run by the detection pattern provider $\mathcal{PP}$, the `Encrypt` algorithm is performed by the data sender $\mathcal{S}$, and the `Match` algorithm is performed by the cloud service provider $\mathcal{CSP}$. In the following, the four entities are assumed *honest-but-curious* and not allowed to collude two at a time.

Definition 1. Pattern Matching over Encrypted strings

- $\texttt{Setup}(1^{\lambda}, \ell_m)$ is a probabilistic algorithm that takes as input a security parameter λ and a scalar ℓ_m denoting an upper bound of the length of the patterns that can be searched. This algorithm returns the public parameter pp which will be implicitly used by all the other algorithms.

- $\mathsf{KeyGen}(\mathsf{pp})$ is a probabilistic algorithm performed by $\mathcal{R}$. It returns a pair of keys $(\mathsf{sk}, \mathsf{pk})$; sk is the secret key known only to $\mathcal{R}$, while pk is public.

- $\texttt{Encrypt}(\mathsf{pk}, \mathcal{B})$ is a probabilistic algorithm performed by $\mathcal{S}$. It takes as input the public key pk of the receiver and a data string $\mathcal{B} = (b_0, \cdots, b_n)$ of any size $n = m + 1$ and returns a ciphertext ct.

- $\texttt{Issue}(\mathsf{pk}, w)$ is a probabilistic algorithm performed by $\mathcal{PP}$. It takes as input a pattern $w = (w_0, \cdots, w_{\ell-1})$ of any size $\ell \leq \ell_m$ along with the public key pk and returns a trapdoor td_w.

- $\texttt{Match}(\mathsf{ct}, \mathsf{td}_w)$ is a deterministic algorithm performed by $\mathcal{CSP}$. It takes as input a ciphertext ct and a trapdoor td_w for the pattern w and outputs an encrypted matching result ct_r.

- $\mathtt{Decrypt}(\mathsf{sk}, \mathsf{ct}, \mathsf{ct}_r)$ is a deterministic algorithm performed by $\mathcal{R}$. It takes as input a ciphertext ct, an encrypted matching result ct_r, the receiver's secret key sk and outputs the plaintext data string $\mathcal{B}$ if w is not in $\mathcal{B}$, e.g., $\forall i \in [0, n-\ell] : b_i, \cdots, b_{i+\ell-1} \neq w_0, \cdots, w_{\ell-1}$, and $\perp$ otherwise.

We emphasize that recent pattern matching on encrypted data schemes [BCC20, BCS21, DFOS18] do not consider a decryption algorithm and argue that this functionality can easily be added by (1) encrypting the stream under a conventional encryption scheme, or (2) issuing a trapdoor for all characters of the alphabet used to encode the data and running the Match algorithm on the encrypted data for each of them. However, as aforementioned, these two solutions cannot be used when a malicious data sender is considered. Our construction defines a decryption algorithm that allows the data receiver to efficiently decrypt both the encrypted pattern matching results and the ciphertext sent by a malicious data sender. We emphasize also that the decryption algorithm returns $\perp$ in case the pattern w exists in the data string $\mathcal{B}$ since we believe that in the considered deep packet inspection use-case, the ciphertext ct should not be decrypted and processed if an attack pattern is matched in the data.

3.4 Correctness

The definition of correctness is associated to the notions of false positives and false negatives. The first notion means that a pattern is detected in the stream while not being present in it and the second notion means that the pattern is not detected while being present.

- Given $(\mathsf{pk}, \mathsf{sk})$ generated by KeyGen, a data content $\mathcal{B}$ encrypted as $\mathsf{ct} \leftarrow \mathtt{Encrypt}(\mathsf{pk}, \mathcal{B})$ and $\mathsf{ct}_r := \mathtt{Match}(\mathsf{pk}, c, \mathsf{td}_w)$:

  $$w \nsubseteq \mathcal{B} \implies \mathsf{ct}_r \text{ does not decrypt to } 0$$

  with negligible probability.

- Given $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$, $\mathsf{ct} \leftarrow \mathtt{Encrypt}(\mathsf{pk}, \mathcal{B})$ and $\mathsf{ct}_r := \mathtt{Match}(\mathsf{pk}, c, \mathsf{td}_w)$:

  $$w \subseteq \mathcal{B} \implies \mathsf{ct}_r \text{ does not decrypt to } 1$$

  with negligible probability.

These two first correctness rules target the matching algorithm; we introduce a third correctness requirement targeting the decryption algorithm with respect to a malicious data sender. The objective of such a sender is to transmit undetectable malicious content to the receiver. To meet the security against malicious data sender requirement, it is crucial to ensure that the occurrence of such events is not detected with a negligible probability.

Definition 2 (Correctness in presence of a malicious Sender)**.** Given a malicious data content $\mathcal{B}_m$ that contains the detection pattern w (provided by $\mathcal{PP}$), a pair of keys $(\mathsf{pk}, \mathsf{sk})$ where sk is used by the receiver $\mathcal{R}$, a data item ct_m sent by the malicious sender $\mathcal{S}$, and a trapdoor td_w for pattern w generated by $\mathcal{PP}$, our construction is said to be correct in presence of a malicious sender $\mathcal{S}$ if:

$$\mathcal{B}_m \subseteq \mathtt{Decrypt}(\mathsf{sk}, \mathsf{ct}_m, \mathtt{Match}(\mathsf{pk}, \mathsf{ct}_m, \mathsf{td}_w))$$

occurs with neglible probability.

In other words, a malicious sender should be able to exhibit a data content $\mathcal{B}_m$ such that for any pattern w chosen by the pattern provider $\mathcal{PP}$ and contained in $\mathcal{B}$, the data set obtained by the receiver when decrypting does not match $\mathcal{B}_m \neq \perp$ at any index.

Lemma 1. *If a pattern matching protocol constructed as in Definition 1 is correct, then it verifies correctness as in Definition 2.*

Proof. If a pattern w is present in the data content $\mathcal{B}_m$, then by the correctness property, $\mathsf{ct}_r := \mathtt{Match}(\mathsf{ct}_m, \mathsf{td}_w)$ outputs 1 with high probability. In that case, $\mathtt{Decrypt}(\mathsf{sk}, \mathsf{ct}_m, \mathsf{ct}_r)$ outputs $\perp$ by construction. □

3.5 Trace Indistinguishability Property

In addition to being correct, our construction must fulfill trace indistinguishability with respect to both $\mathcal{CSP}$ and $\mathcal{PP}$ and pattern indistinguishability with respect to $\mathcal{CSP}$.

In the following definition, we formalize the data indistinguishably requirement as a security game between an adversary playing the role of a curious $\mathcal{CSP}$ or a curious $\mathcal{PP}$ for example and a challenger playing the role of an honest sender. The adversary's goal is to infer information about the content encrypted by the sender $\mathcal{S}$.
Note that the adversary is not given access to an oracle to `Issue` as the adversary can derive a trapdoor for a pattern of its choice. Also, as the answer output by `Match` is encrypted and the adversary does not know the secret key, an oracle access to `Match` would provide an encrypted output to the adversary. So we do not take into account such oracle access.

Definition 3 (Data Indistinguishability)**.** Let λ be the security parameter. We define the experiment $\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}trace}}_{\mathcal{A},\beta}$ parametrized by a bit β which is run between an adversary $\mathcal{A}$ and the challenger $\mathcal{C}$ simulating the view of $\mathcal{A}$:

```
Experiment $\mathsf{Exp}^{\mathsf{ind\text{-}cpa\text{-}trace}}_{\mathcal{A},\beta}(1^\lambda, L)$:

  1. $\mathsf{pp} \leftarrow \mathtt{Setup}(1^\lambda, L)$
  2. $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(\mathsf{pp})$
  3. $(T_0, T_1) \leftarrow \mathcal{A}(\mathsf{pk})$
  4. $\mathsf{ct} \leftarrow \mathtt{Encrypt}(\mathsf{pk}, T_\beta)$
  5. $\beta' \leftarrow \mathcal{A}(\mathsf{state}, \mathsf{pk}, \mathsf{ct})$
  6. return $(\beta == \beta')$
```

Figure 3: Security game for *data indistinguishability* with respect to $\mathcal{CSP}$ the cloud service provider and to $\mathcal{PP}$, the pattern provider.

We define $\mathcal{A}$'s advantage of winning the previous game by:

$$\mathsf{Adv}^{\mathsf{ind\text{-}cpa\text{-}trace}}_{\mathcal{A},\beta}(1^\lambda) = |\Pr[\beta = \beta'] - 1/2|$$

Our construction has *trace indistinguishability* if $\mathsf{Adv}^{\mathsf{ind\text{-}cpa\text{-}trace}}_{\mathcal{A},\beta}(1^\lambda)$ is negligible in λ.

4 RLWE-based Pattern matching protocol with fragmented substrings

4.1 Our Data Fragmentation

Recall that n is the number of symbols (e.g., bits or bytes) that composes the data string $\mathcal{B}^{\mathcal{S}\to\mathcal{R}}$, $\ell_m \geq 2$ is the upper bound on the number of symbols in a pattern, and Φ' is the offset between fragments and overlined fragments. In the following, we suppose that there exists η such that $n = 2\eta \cdot \Phi'$. We emphasize that the previous assumption is not mandatory and that the construction we are proposing provides correct pattern matching even when n is not a multiple of the offset Φ'. In contrast to the data fragmentation technique introduced in [BCC20] (see Section 2.5), the data string $\mathcal{B}^{\mathcal{S}\rightarrow\mathcal{R}}$ is fragmented into a set of η data segments $\mathcal{F} = \{F_0, F_1, \cdots, F_{\eta-1}\}$, each $F_i \in \mathcal{F}$ is composed of $2\Phi'$ symbols.

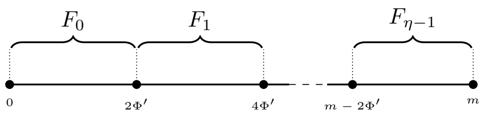

Figure 4: Fragmentation of a data string $\mathcal{B}$ of size $n = m + 1$

Compared to the fragmentation technique proposed in [BCC20], we do not need to construct the overlined fragments during the data encryption step as they will be constructed by the $\mathcal{CSP}$ on the fly during the pattern matching process, as described in Section 2.5. Hence, a malicious data sender will not have the ability to bypass the detection by making the malicious content straddling two consecutive fragments F_i and F_{i+1} and replacing the data in the overlined fragments by random contents.

4.2 The intuition

As mentioned in Section 1.2, the objective of this work is threefold. First, we aim to design a construction that can enable pattern matching on encrypted data when the latter is encrypted by a malicious sender as in the deep packet inspection use-case and tries to bypass detection while incorporating malicious content. Second, we want a more efficient construction that the one proposed in [BCC20, BCS21]. Our third goal consists of defining a construction that improves the security guarantees provided in [BCC20, BCS21] by incorporating quantum-resistant features. To meet the aforementioned three goals, the construction we present combines the homomorphic RLWE scheme from Section 2.3, the secure multiple Hamming distance computation technique from [YSK+15] and recalled in Section 2.4, and the data fragmentation technique we present in Section 4.1. First, our data fragmentation technique will allow us processing data of arbitrary size without impacting the efficiency of our construction. That is, processing each (relatively small size) couple of fragments independently allows us using FHE parameters of relatively small sizes. Second, the $\mathcal{CSP}$ will be able to detect the patterns that straddle two fragments by homomorphically concatenating each two consecutive fragments.

4.3 Description of the RLWE-based Pattern matching protocol

Let be given an RLWE scheme with a specification for rlwe.KeyGen, rlwe.Enc and rlwe.Dec. We formally describe a first protocol that satisfies all the requirements we defined in Section 3.2.

- $\texttt{Setup}(1^\lambda, \Phi)$ selects a RLWE parameters set (N, q, t, σ) such that $N = 4\Phi' + 1$, $t > 2\Phi'$, where $2\Phi'$ is the size of a fragment. The construction works for $N \geq 4\Phi' + 1$ but to simplify, we assume both are equals. We will see in Section 5 the requirements on the sizes to ensure correctness and the indistinguishability property. The Setup algorithm returns $\mathsf{pp} \leftarrow (N, q, t, \sigma, \Phi')$ which will be implicitly used by all the other algorithms and will be omitted as input.

- KeyGen performs the rlwe.KeyGen algorithm of the RLWE scheme defined in Section 2.3. It returns the public key $\mathsf{pk} = (p_0, p_1)$ and the secret key $\mathsf{sk} = s$.

- `Encrypt`$(\mathsf{pk}, \mathcal{B})$ starts by fragmenting $\mathcal{B} = (b_0, \cdots, b_{n-1})$ into a set of stumbling segment $\mathcal{F} = \{F_0, \cdots, F_{\eta-1}\}$ with $F_i = b_{i\cdot 2\Phi'}, \cdots, b_{(i+1)\cdot 2\Phi'-1}$ as described in Section 4.1. Then, it packs and encrypts each $F_i \in \mathcal{F}$ as described in Section 2.4. This algorithm returns the ciphertext $\mathsf{ct} = \{c_i\}_{i\in[0,\eta-1]}$ generated as described in Algorithm 1.

Algorithm 1 Description of the Fragment-by-Fragment encryption procedure

```
Input: $\mathcal{B} = (b_0, \cdots, b_{n-1})$ and $\mathsf{pk}$
Output: $\mathsf{ct} = \{c_0, \cdots, c_{\eta-1}\}$
for each: $F_i \in \mathcal{F}$
1: $\mathsf{pm}_1(F_i) := \sum_{j=0}^{2\Phi'-1} b_{i\cdot 2\Phi'+j} \cdot X^j$
2: $\mathsf{ct}_i = \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\mathsf{pm}_1(F_i)\right)$
```

- `Issue`(pk, w): This algorithm issues a trapdoor td_w for a pattern w of size $\ell \leq \ell_m$ as follow: $\mathsf{td}_w = \mathsf{RLWE.Enc}(\mathsf{pk}, \mathsf{pm}_2(w))$.
- `Match`$(\mathsf{pk}, \mathsf{ct} = \{c_0, \cdots, c_{\eta-1}\}, \mathsf{td}_w)$: starts by computing the data fragments:

$$\{\mathsf{ct}_{\{0,1\}}, \mathsf{ct}_{\{1,2\}}, \cdots, \mathsf{ct}_{\{\eta-2,\eta-1\}}\}$$

representing the encrypted concatenation of each of two consecutive encrypted fragments in ct for $0 \leq i < \eta - 1$ as follow:

$$\mathsf{ct}_{\{i,i+1\}} = c_i \dotplus \mathsf{Rotate}(c_{i+1}, 2\Phi'),$$

Then, it computes the encrypted multiple Hamming distances (see Section 2.4) between each $\mathsf{ct}_{\{i,i+1\}}$ for $0 \leq i < \eta - 1$ and td_w as follow:

$$\mathsf{ct}_{r,\{i,i+1\}} = \mathsf{ct}_{\{i,i+1\}} \cdot C \dotplus \mathsf{td}_w \cdot C' + \left(-2\mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}_w\right)$$

to get $\mathsf{ct}_r = \{\mathsf{ct}_{r,\{i,i+1\}} | 0 \leq i < \eta - 1\}$.

- `Decrypt`$(\mathsf{sk}, \mathsf{ct}, \mathsf{ct}_r)$ starts by decrypting ct_r using sk and parses the plaintext as $\sum_{i=0}^{N} d_i X^i$. If the correctness of the ciphertext is satisfied, then it gets the multiple Hamming distances $d_i = \mathsf{hw}(\mathcal{B}^{(i)}, w), i \in [0, N - \ell]$ between w and $\mathcal{B}$. Then if w does not occur in $\mathcal{B}$, i.e., $\forall i \in [0, N - \ell], \mathsf{d}(\mathcal{B}^{(i)}, w) \neq 0$, the data receiver performs $\mathsf{RLWE.Dec}(\mathsf{sk}, \mathsf{ct})$ to get $\mathcal{B}$; it outputs $\perp$ otherwise.

Remark 1. The implementation of our decryption algorithm is specifically designed for the DPI use-case. Nonetheless, it can be readily adapted for more general scenarios by adjusting the output to indicate the presence (1) or absence (0) of the pattern. This adjustment is guided by the results of the underlying FHE decryption algorithm, a detail that could be explicitly stated within the Decrypt algorithm itself.

4.4 Correctness

To prove that the protocol from Section 4.3 proposes a correct pattern matching over encrypted data, we rely on the correctness of the underlying RLWE packing method reviewed in Section 2.4 and the definition of the `Encrypt`, `Issue` and `Match` algorithms. The proof of the following lemma can be found in Appendix 8.1.

Lemma 2. *If* $\left\|\mathsf{Err}(\mathsf{ct}_{i,i+1}^{(r)})\right\|_\infty < \frac{q}{2t}$, *the protocol proposed in Section 4.3 provides a correct pattern matching over encrypted data.*

To prove the correctness of our pattern matching protocol over encrypted data, we show that the following two conditions hold:

- (False negative) If the pattern w exists in the data string $\mathcal{B}$ at the index i, then $\mathsf{hw}(\mathcal{B}^{(i)}, w) = 0$ with high probability.
- (False positive) If the pattern w does not exist in the string $\mathcal{B}$, then $\forall i \in [0, n - \ell], \mathsf{hw}(\mathcal{B}^{(i)}, w) \neq 0$ with high probability.

The previous two conditions by our construction is demonstrated in the following lemma, the proof of which can be found in Appendix 8.2.

Lemma 3. *Given $\mathcal{B} = b_0, b_1, \cdots, b_{n-1}$ and $w = w_0, w_1, \cdots, w_{\ell_m - 1}$. The following two conditions holds:*

(a) If $\exists i \in [0, n - \ell]$ such that $\forall j \in [0, \ell - 1]$, $b_{i+j} = w_j$ then, $\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) \neq 0] = 0$

(b) If $\forall i \in [0, n - \ell]$, $\exists t' \in [0, \ell - 1]$ such that $b_{i+t'} \neq w_{t'}$ then, $\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) = 0] = 0.$

4.5 Security properties

Theorem 1. *Assuming the* RLWE *scheme from Section 2.3 is* IND-CPA *secure, an adversary will have negligible advantage in the data-indistinguishability game for the construction presented in Section 4.3.*

The proof of Theorem 1 can be found in Appendix 8.3.

4.6 Complexity

The computation of all positions where a pattern exists in a given data fragment requires 3 homomorphic multiplications and 3 homomorphic additions. However, in malicious pattern matching solutions, such as intrusion detection or phishing detection systems, it is frequently required to examine the presence of numerous malicious patterns for each data item. Hence, when considering a collection of m patterns to be matched, transmitting the detection results of each individual pattern to the data receiver would result in multiplying the size of the data to be analyzed by m times. To address this issue, a potential solution would involve consolidating the detection results of the various patterns prior to transmitting them to the data receiver. To address this issue, a direct method involves performing homomorphic multiplication on the matching results of the given patterns. Let $\mathcal{W}$ represent the set of patterns to be matched, and let $\mathsf{ct}_{i,i+1}^{(r,w)}$ denote the matching result for each pattern $w \in \mathcal{W}$. We can compute the consolidated result as follows:

$$\mathsf{prod}_{w \in \mathcal{W}} \mathsf{ct}_{i,i+1}^{(r,w)}$$

By applying the multiplication operation, the coefficients associated with a Hamming distance of zero will be preserved, enabling the receiver to infer the presence of a particular malicious pattern or patterns in the received data. This approach requires $m - 1$ homomorphic multiplications of the results of each pattern detection, i.e., our construction needs to adhere to a multiplication depth of $log(m)$. Overall, matching m patterns in a single data fragment and consolidating the matching results in single ciphertext requires $4m - 1$ homomorphic multiplications and $3m$ homomorphic additions.

4.7 Handling Wildcards

The scheme presented in Section 4.3 enables us to conduct approximate pattern matching by providing the number of bits differentiating the pattern from the analyzed byte stream. This information empowers the data receiver to employ a threshold for obtaining an approximate pattern matching result. Nevertheless, this approach does not support the processing of patterns containing wildcards, typically denoted by $\star$ and designed to match all characters (or bits in our case). Wildcards are crucial for detecting more intricate patterns like $01\star\star\star11$, frequently encountered in practical scenarios such as signature-based intrusion detection (e.g., SNORT [SNO98]) and spam detection.

4.7.1 `Issue` and `Match` for Wildcards Detection

To facilitate the detection of patterns containing wildcards, we introduce a variant of our primary protocol. In this variation, only the `Issue` and `Match` algorithms deviate slightly from their original definitions.

- `Issue`(pk, w): Let $\mathcal{I}$ denote the set of indices corresponding to wildcards in the pattern $w = (w_0, \cdots, w_{\ell-1})$. The Issue algorithm returns $\mathsf{td}_w = (\mathsf{td}_w^{(1)}, \mathsf{td}_w^{(2)})$ such that:

$$\begin{aligned} \mathsf{td}_w^{(1)} &= \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(-\sum_{i=0, i\notin\mathcal{I}}^{\ell-1} w_i \cdot X^{N-i}\right) \\ \mathsf{td}_w^{(2)} &= \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(-\sum_{i=0, i\notin\mathcal{I}}^{\ell-1} X^{N-i}\right) \end{aligned} \tag{3}$$

  The previous `Issue` algorithm differs from the original `Issue` algorithm in two primary aspects. Firstly, it includes the additional encrypted element $\mathsf{td}_w^{(2)}$ which will be used to perform the `Match`, and secondly, it introduces the supplementary condition $i \notin \mathcal{I}$ to ensure that the encrypted polynomials in $\mathsf{td}_w^{(1)}$ and $\mathsf{td}_w^{(2)}$ will not contain any monomial of degree i for $i \in \mathcal{I}$.

- `Match`$(ct = \{c_0, c_1, \cdots, c_{\eta-1}\}, \mathsf{td}_w^{(1)}, \mathsf{td}_w^{(2)})$: it computes the data fragments

$$\{\mathsf{ct}_{\{0,1\}}, \mathsf{ct}_{\{1,2\}}, \cdots, \mathsf{ct}_{\{\eta-2,\eta-1\}}\}$$

  representing the encrypted concatenation each of two consecutive encrypted fragments in ct for $0 \le i < \eta - 1$ as follow:

$$\mathsf{ct}_{\{i,i+1\}} = c_i \dotplus \mathsf{Rotate}(c_{i+1}, 2\Phi'),$$

  Then, it computes the encrypted multiple Hamming distances (see Section 2.4) between each $\mathsf{ct}_{\{i,i+1\}}$ $(0 \le i < \eta - 1)$ and td_w as follow:

$$\mathsf{ct}_{\{i,i+1\}}^{(r)} = \mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}_w^{(2)} \dotplus \mathsf{td}_w^{(1)} \cdot \overline{C}' + \left(-2\mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}_w^{(1)}\right)$$

  to get $\mathsf{ct}_r = \{\mathsf{ct}_{\{i,i+1\}}^{(r)} | 0 \le i < \eta - 1\}$.

4.7.2 Correctness

To validate the accuracy of our wildcard-supporting construction, it is imperative to demonstrate that the probability of generating false positives or false negatives is negligible. This assertion is established through the subsequent lemma whose proof is deferred to Appendix 8.4.

Lemma 4. *Given $\mathcal{B} = b_0, b_1, \cdots, b_n - 1$ and $w = w'_0, w'_1, \cdot, w'_{\ell-1}$. Let us denote by $\mathcal{I}$ the set of indices in w that contains wildcards. The following conditions hold:*

(a) if $\exists i \in [0, n-\ell]$ such that $\forall j \in [0, \ell-1] \backslash \mathcal{I}$, $b_{i+j} = w'_j$ then $Pr[hw(\mathcal{B}^{(i)}, w) \neq 0] = 0$

(b) if $\forall i \in [0, n-\ell]$ such that $\exists j \in [0, \ell-1] \backslash \mathcal{I}$, $b_{i+j} \neq w'_j$ then $Pr[hw(\mathcal{B}^{(i)}, w) = 0] = 0$

Table 2: Performance of our construction for encrypting, detecting and decrypting detection result of a pattern of size 2^7 bits on a data of size 2^{15} bits as function of the considered Polynomial Modulus. All time measurements are denoted in milliseconds (ms).

Polynomial Degree (N)		2048	4096	8192	16384	32768
$\log(q)$		54	109	218	438	881
t		1024	2048	4096	8192	16384
Largest Pattern Size $\Phi = \ell_m$		1024 (32 fragments)	2048 (16 fragments)	4096 (8 fragments)	8192 (4 fragments)	16384 (2 fragments)
Enc. Time	Dataset	7.6	12.2	17.8	28.3	69.1
	Fragment	0.2	0.7	2.2	7	34.5
Match Time	Dataset	26	43.4	87	161.8	259.3
	Fragment	0.8	2.7	10.8	40.4	129.6
Dec. Time	Dataset	2.5	4	6.8	13.2	20.5
	Fragment	0.08	0.2	0.8	3.3	10.2
Enc. Size	per fragment	32.8 KB	131.1 KB	524.4 KB	2 MB	7.8 MB
# of consolidated results		1	2^2	2^4	2^8	2^{15}

5 Parameters and Implementation

In this section we discuss the bounds that must be verified by the parameters to ensure correctness and the security requirements. Subsequently, we evaluate the robustness of the underlying RLWE assumption by employing the Lattice Estimator method as presented in [APS15]. Finally, we conduct a comprehensive experimental analysis to gauge the efficacy of our proposed construction in comparison to existing similar solutions.

Correctness and Security

The correctness bound gives that the parameters must fulfill the following bound from condition (6) from Lemma 2 i.e.

$$5\sigma N^{3/2}(N+1)^2 + \sigma N^{3/2}(N+1) + 32N^2 \leq \frac{q}{2t}$$

For all the set of parameters for N, t and q listed in Table 2 and for $\sigma = 3.19$, inequation (6) holds except with negligible probability. In our construction, we use a secure set of parameters according to [ACC+18], which provide more than 90 bits of security according to the Lattice Estimator [APS15].

Experimental Setup

We leverage the SEAL FHE Library [SEA22] as the foundation for our construction, employing the BFV scheme for its RLWE-based FHE capabilities. Our implementation is public and can be found on the github repository https://github.com/nserser/fhe-pattern-matching. Our experiments are conducted on a Linux machine (Ubuntu 20.04) equipped with a 13th Gen Intel(R) Core(TM) i7-1355U processor and 16 GB of RAM.

Remark 2. The SEAL library is no longer actively maintained. Given that the packing method from [YSK+14] was historically implemented in SEAL and considering the simplicity of the homomorphic circuit under study, we opted not to change the library. Nonetheless, our solution is adaptable to any library that implements the BFV scheme.

Experimental Results

We conduct an empirical evaluation of our construction's performance concerning the parameters outlined in the BFV scheme, notably, the polynomial degree N, across various metrics, including:

- The maximum size of the pattern that can be matched on the data;
- The time required for encryption, matching, and decryption of both matching results and encrypted data;
- The quantity of detection results that can be consolidated in a single encrypted ciphertext;
- The provided security level.

Obtained results are depicted in Table 2. The observations derived from the results in Table 2 reveal the following insights. Elevating the degree of the polynomial N is found to enable the alignment of larger patterns, with a maximum size of $N/2$. This augmentation facilitates the consolidation of multiple pattern-matching outcomes within a single ciphertext, thereby reducing the communication overhead of our construction notably. However, this advantageous enhancement is counterbalanced by an increase in computational demands. It is evident that the temporal requirements for encryption, decryption, and matching operations exhibit a notable escalation with ascending values of N. Based on Table 2, we observe that the choice of the parameter N impacts the number of patterns to be matched and consolidated into a single ciphertext. Seeking to reduce to the best the matching results that will be forwarded to the receiver, we use the following value of N.

$$N = \begin{cases} 2048 & \text{if} \quad \#\text{ patterns} \leq 2 \\ 4096 & \text{if} \quad 2 < \#\text{ patterns} \leq 2^2 \\ 8192 & \text{if} \quad 2^2 < \#\text{ patterns} \leq 2^4 \\ 16384 & \text{if} \quad 2^4 < \#\text{ patterns} \leq 2^8 \\ 32768 & \text{otherwise} \end{cases} \tag{4}$$

Then, we use the previous instantiation of N to evaluate the efficiency of our construction in performing the match operation on a 15K bits string relative to SEST [DFOS18], AS3E [BCC20], and SEPM [BCS21], with respect to varying numbers of patterns to be matched. The comparison results are presented in Table 3, revealing significant improvements achieved by our construction. In scenarios involving a small number of patterns, our method drastically reduces matching time, with speeds up to 119 times faster compared to the most efficient existing construction, particularly evident when considering only two patterns. However, as the number of patterns increases, the speedup ratio diminishes, reaching 12 times faster when matching 2^{10} patterns. This variation is primarily attributed to the adjustment of the value of N in accordance with the number of patterns to be matched.

Note that Yasuda *et al.* [YSK+15] proposed a construction proposed in based on the same packing method utilized in our approach [YSK+14]. However, we did not include this existing construction in our comparison table (Table 3) because, unlike all the considered constructions, it does not support pattern matching on strings of arbitrary length. Nevertheless, for the sake of comparison, when the size of the considered string is fixed at 2^{15} bits, our approach still outperforms the one proposed in [YSK+15] by a factor of 10. Specifically, our construction performs the test procedure in 24 ms, whereas the method in [YSK+15] requires 255 ms.

Table 4 compares the time needed for our construction to decrypt the matching results as well as the exchanged data compared to SEST [DFOS18], AS3E [BCC20], and SEPM [BCS21], according to the numbers of patterns to be matched on 32K bits string.

Table 3: Performance comparison of our construction for performing the matching patterns on a 15K bits string compared to most relevant existing solutions.

# patterns	Match Time				Speedup Ratio
	SEST [†]	AS^{3}E [†]	SEPM [†]	This work [‡]	
2	95,6 s	5,41 s	5,38 s	45,2 ms	× 119
2^4	781,5 s	48,21 s	48,18 s	736,5 ms	× 65
2^6	2535 s	183,4 s	182,7 s	4,4 s	× 41
2^8	12019,5 s	738 s	730,5 s	12,3 s	× 59
2^{10}	48315 s	3226,5 s	3216 s	262,1 s	× 12

[†] We use Barreto-Naehrig curve BN254 [PSJNB11] for instantiating SEST, AS^{3}E, and SEPM.
[‡] We use BFV [FV12] with N determined by Equation 4.

Table 4: Performance comparison of our construction for performing decryption operation of a 32K bits encrypted string as well the pattern matching results compared to most relevant existing solutions.

# patterns	Decryption Time		Speedup Ratio
	SEST [†] / AS^{3}E [†] / SEPM [†]	This work [‡]	
2	2240 s	6,2 ms	$\times 3 \cdot 10^6$
2^4	2240 s	15,2 ms	$\times 1,4 \cdot 10^6$
2^6	2240 s	32,4 ms	$\times 6,9 \cdot 10^5$
2^{10}	2240 s	72,9 ms	$\times 3 \cdot 10^5$

[†] We use Barreto-Naehrig curve BN254 [PSJNB11] for instantiating SEST, AS^{3}E, and SEPM.
[‡] We use BFV [FV12] with N determined by Equation 4.

The table presents a comparative analysis of decryption time for pattern matching operations on a 32K-bit encrypted string, contrasting the performance of existing solutions with a proposed method. The decryption time, measured in seconds for SEST, AS^{3}E, and SEPM, alongside our proposed construction, showcases a significant disparity in efficiency. For instance, with only 2 patterns, the decryption time for the proposed method drops to a mere $6,2$ milliseconds, yielding a remarkable speedup ratio of 3×10^6 compared to existing methods. Even with a substantial increase in the number of patterns, the proposed method maintains a notable advantage, demonstrating decryption times of $15,2$ milliseconds for 2^4 patterns, $32,4$ milliseconds for 2^6 patterns, and $72,9$ milliseconds for 2^{10} patterns. Moreover, we compare the quantity of data that will be exchange between the different entities when our construction is used compared to SEST [DFOS18], AS3E [BCC20], and SEPM [BCS21], with respect to varying numbers of patterns to be matched on 32K bits string. The comparison results are reported in Table 5.

The results reveal notable performance differences among the evaluated pattern matching methods on a 32K-bit string. While the size of exchanged data remains constant at 2 MB for SEST, and 4 MB for AS^{3}E and SEPM, our construction exhibits increasing data exchange sizes with the number of patterns, reaching $22,4$ MB for 2^{10} patterns. Consequently, the ratio of exchanged data size for our construction compared to the other methods escalates significantly with more patterns, reaching 11 times larger for 2^{10} patterns.

Table 5: Communication cost comparison of our construction for performing the matching patterns on a 32K bits string compared to most relevant existing solutions.

# patterns	Size of Exchanged ata				Ratio
	SEST	AS^{3}E	SEPM	This work	
2	2 MB	4 MB	4 MB	1,57 MB	$\times 0,78$
2^4	2 MB	4 MB	4 MB	3,14 MB	$\times 1,57$
2^6	2 MB	4 MB	4 MB	6,2 MB	$\times 3,1$
2^{10}	2 MB	4 MB	4 MB	22,4 MB	$\times 11$

† We use Barreto-Naehrig curve BN254 [PSJNB11] for instantiating SEST, AS^{3}E, and SEPM.

‡ We use BFV [FV12] with N determined by Equation 4.

6 Conclusion

In conclusion, this paper presents a novel and efficient post-quantum secure construction for pattern matching over encrypted data, addressing the critical challenge of balancing privacy protection with pattern matching efficiency. Our contributions significantly advance the state of the art in this field in several key aspects. Firstly, we tackle the issue of correctness in the presence of a malicious data sender, adapting the data fragmentation technique previously proposed in [BCC20] to ensure accurate pattern matching even when the encrypted data is tampered with by the sender. This allows for reliable pattern detection even in scenarios, e.g., intrusion detection, where malicious actors attempt to disrupt the pattern matching process. Secondly, our construction leverages Ring-Learning With Error based homomorphic encryption, providing notable improvements in performance and security compared to existing pairing-based schemes. Our approach achieves substantial reductions in encryption, decryption, and pattern matching times, making it significantly more efficient than the state-of-the-art alternatives. The comparison of our construction's performance across various metrics illustrates its significant advancements in pattern matching operations. Our construction demonstrates substantial reductions in matching time, particularly evident in scenarios involving a small number of patterns, where our method achieves speeds up to 119 times faster than existing constructions. However, as the number of patterns increases, the speedup ratio diminishes, albeit remaining notably advantageous, even when matching 2^{10} patterns. This variability in efficiency is primarily attributed to the careful adjustment of the parameter N to accommodate the number of patterns to be matched. Additionally, the comparative analysis of decryption time highlights the remarkable efficiency of our proposed method, showcasing significantly faster decryption times across different pattern counts compared to existing solutions. Importantly, our construction provides post-quantum security and is secure against a malicious sender, unlike other existing constructions. Nevertheless, upon examining the sizes of exchanged data, it becomes noticeable that with a higher number of patterns, the data size increases substantially. Nonetheless, we believe this to be a reasonable trade-off for attaining all the aforementioned benefits.

Acknowledgments. We would like to thank the anonymous reviewers for their valuable feedback and constructive suggestions, which have greatly contributed to improve this work. The first author has been partially funded by the IRT SystemX Explore Project.

7 Bibliography

References

[ABC+08] Michel Abdalla, Mihir Bellare, Dario Catalano, Eike Kiltz, Tadayoshi Kohno, Tanja Lange, John Malone-Lee, Gregory Neven, Pascal Paillier, and Haixia Shi. Searchable encryption revisited: Consistency properties, relation to anonymous ibe, and extensions. *Journal of cryptology*, 21(3):350–391, 2008. doi:10.1007/S00145-007-9006-6.

[ACC+18] Martin Albrecht, Melissa Chase, Hao Chen, Jintai Ding, Shafi Goldwasser, Sergey Gorbunov, Shai Halevi, Jeffrey Hoffstein, Kim Laine, Kristin Lauter, Satya Lokam, Daniele Micciancio, Dustin Moody, Travis Morrison, Amit Sahai, and Vinod Vaikuntanathan. Homomorphic encryption security standard. Technical report, HomomorphicEncryption.org, Toronto, Canada, November 2018.

[APS15] Martin R. Albrecht, Rachel Player, and Sam Scott. On the concrete hardness of learning with errors. *Journal of Mathemtical Cryptology*, 9(3):169–203, 2015. URL: http://www.degruyter.com/view/j/jmc.2015.9.issue-3/jmc-2015-0016/jmc-2015-0016.xml, doi:10.1515/jmc-2015-0016.

[BCC20] Anis Bkakria, Nora Cuppens, and Frédéric Cuppens. Privacy-preserving pattern matching on encrypted data. In *International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2020, (Part II)*, volume 12492, pages 191–220. Springer, Springer, 2020. doi:10.1007/978-3-030-64834-3_7.

[BCS21] Elie Bouscatié, Guilhem Castagnos, and Olivier Sanders. Public key encryption with flexible pattern matching. In *Advances in Cryptology - ASIACRYPT 2021 - 27th International Conference on the Theory and Application of Cryptology and Information Security, Proceedings, Part IV*, volume 13093 of *Lecture Notes in Computer Science*, pages 342–370. Springer, 2021. doi:10.1007/978-3-030-92068-5_12.

[BCS23] Elie Bouscatié, Guilhem Castagnos, and Olivier Sanders. Pattern matching in encrypted stream from inner product encryption. In *Public-Key Cryptography - PKC 2023 - 26th IACR International Conference on Practice and Theory of Public-Key Cryptography, Atlanta, GA, USA, May 7-10, 2023, Proceedings, Part I*, volume 13940 of *Lecture Notes in Computer Science*, pages 774–801. Springer, 2023. doi:10.1007/978-3-031-31368-4_27.

[Bra12] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical GapSVP. In *Advances in Cryptology - CRYPTO 2012 - 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings*, volume 7417 of *Lecture Notes in Computer Science*, pages 868–886. Springer, 2012. doi:10.1007/978-3-642-32009-5_50.

[BSW11] Dan Boneh, Amit Sahai, and Brent Waters. Functional encryption: Definitions and challenges. In *Theory of Cryptography - 8th Theory of Cryptography Conference, TCC 2011, Proceedings*, volume 6597 of *Lecture Notes in Computer Science*, pages 253–273. Springer, 2011. doi:10.1007/978-3-642-19571-6_16.

[CDK+17] Sébastien Canard, Aïda Diop, Nizar Kheir, Marie Paindavoine, and Mohamed Sabt. Blindids: Market-compliant and privacy-friendly intrusion detection system over encrypted traffic. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security, AsiaCCS 2017*, pages 561–574. ACM, 2017. doi:10.1145/3052973.3053013.

[CPST15] Sébastien Canard, David Pointcheval, Olivier Sanders, and Jacques Traoré. Divisible e-cash made practical. In *Public-Key Cryptography - PKC 2015 - 18th IACR International Conference on Practice and Theory in Public-Key Cryptography*, volume 9020 of *Lecture Notes in Computer Science*, pages 77–100. Springer, 2015. doi:10.1007/978-3-662-46447-2_4.

[CS15] Melissa Chase and Emily Shen. Substring-searchable symmetric encryption. *Proc. Priv. Enhancing Technol.*, 2015(2):263–281, 2015. doi:10.1515/POPETS-2015-0014.

[DFOS18] Nicolas Desmoulins, Pierre-Alain Fouque, Cristina Onete, and Olivier Sanders. Pattern matching on encrypted streams. In *International Conference on the Theory and Application of Cryptology and Information Security, ASIACRYPT 2018, Proceedings, (Part I)*, volume 11272 of *Lecture Notes in Computer Science*, pages 121–148. Springer, 2018. doi:10.1007/978-3-030-03326-2_5.

[FV12] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. https://eprint.iacr.org/2012/144.

[Gen09a] Craig Gentry. *A fully homomorphic encryption scheme*. Stanford university, https://crypto.stanford.edu/craig/craig-thesis.pdf, 2009.

[Gen09b] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009*, pages 169–178. ACM, 2009. doi:10.1145/1536414.1536440.

[KMO18] Seny Kamara, Tarik Moataz, and Olya Ohrimenko. Structured encryption and leakage suppression. In *Advances in Cryptology - CRYPTO 2018 - 38th Annual International Cryptology Conference, Proceedings, Part I*, volume 10991 of *Lecture Notes in Computer Science*, pages 339–370. Springer, 2018. doi:10.1007/978-3-319-96884-1_12.

[PSJNB11] Geovandro CCF Pereira, Marcos A Simplício Jr, Michael Naehrig, and Paulo SLM Barreto. A family of implementation-friendly bn elliptic curves. *Journal of Systems and Software*, 84(8):1319–1326, 2011. doi:10.1016/J.JSS.2011.03.083.

[SEA22] Microsoft SEAL (release 4.0). https://github.com/Microsoft/SEAL, March 2022. Microsoft Research, Redmond, WA.

[SLPR15] Justine Sherry, Chang Lan, Raluca Ada Popa, and Sylvia Ratnasamy. Blindbox: Deep packet inspection over encrypted traffic. In *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2015, London, United Kingdom*, pages 213–226. ACM, 2015. doi:10.1145/2785956.2787502.

[SNO98] SNORT. Snort - network intrusion detection & prevention system, 1998. URL: https://www.snort.org/.

[SWP00] Dawn Xiaoding Song, David Wagner, and Adrian Perrig. Practical techniques for searches on encrypted data. In *Proceeding 2000 IEEE symposium on security and privacy. S&P 2000*, pages 44–55, https://people.eecs.berkeley.edu/~dawnsong/papers/se.pdf, 2000. IEEE. doi:10.1109/SECPRI.2000.848445.

[YSK+14] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. Practical packing method in somewhat homomorphic encryption. In Joaquin Garcia-Alfaro, Georgios Lioudakis, Nora Cuppens-Boulahia, Simon Foley, and William M. Fitzgerald, editors, *Data Privacy Management and Autonomous Spontaneous Security*, pages 34–50, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg.

[YSK+15] Masaya Yasuda, Takeshi Shimoyama, Jun Kogure, Kazuhiro Yokoyama, and Takeshi Koshiba. New packing method in somewhat homomorphic encryption and its applications. *Security and Communication Networks*, 8(13):2194–2213, 2015.

8 Appendix

8.1 Proof of Lemma 2

Proof. According to our construction, the ciphertext $\mathsf{ct}^{(r)}_{i,i+1}$ result of the `Match` algorithm is equal to:

$$\mathsf{ct}^{(r)}_{i,i+1} = \mathsf{ct}_{i,i+1} \cdot \overline{C} \dotplus \mathsf{td}_w \cdot \overline{C}' + \left(-2\mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}_w\right), 0 \le i < \eta - 1$$

where

$$\mathsf{ct}_{\{i,i+1\}} = \mathsf{ct}_i + \left(X^{2\Phi'} \cdot \mathsf{ct}_{i+1}\right), 0 \le i < \eta - 1$$

Let V denote the upper bound of the noise of fresh ciphertexts, we get the following inequality

$$\|\mathsf{Err}(\mathsf{ct}_{i,i+1})\|_\infty \le V(N+1)$$

From [FV12, Lemma 2], the ciphertext noise of the product of two fresh ciphertexts c and c' is bounded by:

$$2NtV(N\|\mathbf{s}\|_\infty + 1) + 2t^2N^2(\|\mathbf{s}\|_\infty + 1)^2$$

Thus, we have:

$$\begin{aligned}\left\|\mathsf{Err}(\mathsf{ct}^{(r)}_{i,i+1})\right\|_\infty &\le V(N+1)N + NV + 4NV(N+1) + 32N^2 \\ &\le 5NV(N+1) + NV + 32N^2\end{aligned}$$

where $V = B(N+1)$, which gives:

$$\left\|\mathsf{Err}(\mathsf{ct}^{(r)}_{i,i+1})\right\|_\infty \le 5\sigma N^{3/2}(N+1)^2 + \sigma N^{3/2}(N+1) + 32N^2 \tag{5}$$

To ensure the correctness of the computed ciphertexts, we need to make sure that

$$\left\|\mathsf{Err}(\mathsf{ct}^{(r)}_{i,i+1})\right\|_\infty < \frac{q}{2t} \tag{6}$$

□

8.2 Proof of Lemma 3

Proof. To prove the lemma, we perform the different steps of our constructions and show that the two conditions (a) and (b) holds. First, $\mathcal{B}$ will be encrypted fragment-by-fragment to get

$$\{c_i = \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\sum_{j=0}^{2\Phi'-1} b_{i2\Phi'+j}X^j\right) \mid i \in [0, \eta-1]\}$$

A trapdoor for w, td_w is generated such that $\mathsf{td}_w = \mathsf{rlwe.Enc}_{\mathsf{pk}}(-\sum_{j=0}^{\ell-1} w_j X^{2\Phi'-j})$.

Now, performing the `Match` algorithm between the encryption of the i_{th} section of $\mathcal{B}$ and the encryption of td_w gives:

$$\begin{aligned}\mathsf{ct}_{\{i,i+1\}} =& c_i \dot{+} \mathsf{Rotate}(c_{i+1}, 2\Phi') \\ =& \mathsf{rlwe.Enc}_{\mathsf{pk}}(\sum_{j=0}^{2\Phi'-1} b_{i2\Phi'+j}X^j) + \mathsf{rlwe.Enc}_{\mathsf{pk}}(\sum_{j=2\Phi'}^{N-1} b_{(i+1)2\Phi'+j-2\Phi'}X^j) \\ =& \mathsf{rlwe.Enc}_{\mathsf{pk}}(\sum_{j=0}^{N-1} b_{i2\Phi'+j}X^j)\end{aligned}$$

For each $0 \le i < \eta - 1$:

$$ct^{(r)}_{\{i,i+1\}} = -ct_{\{i,i+1\}} \cdot \sum_{j=0}^{\ell-1} X^{N-j} + (\mathsf{td}_w \cdot \sum_{t=0}^{N-1} X^t) + (-2ct_{\{i,i+1\}} \cdot \mathsf{td}_w)$$

which under the correctness conditions from Lemma 2 gives an RLWE encryption of:

$$\begin{aligned}&-\sum_{t=0}^{N-1} b_{i2\Phi'+t}X^t \cdot \sum_{j=0}^{\ell-1} X^{N-j} - \sum_{j=0}^{\ell-1} w_j X^{N-j} \cdot \sum_{t=0}^{N-1} X^t \\ &+ 2\sum_{t=0}^{N-1} b_{i2\Phi'+t}X^t \cdot \sum_{j=0}^{\ell-1} w_j X^{N-j} \\ &= \sum_{t=0}^{N-1}\sum_{j=0}^{\ell} X^{N+t-j}(-b_{i2\Phi'+t} - w_j + 2b_{i2\Phi'+t}w_j)\end{aligned}$$

Then under the condition of Lemma 2, we have

$$\mathsf{rlwe.Dec}_{\mathsf{sk}}(\mathsf{ct}^r_{i,i+1}) = \sum_{t=0}^{N-1}\sum_{j=0}^{\ell-1} X^{2\Phi'+t-j}(-b_{i2\Phi'+t} - w_j + 2b_{i2\Phi'+t}w_j)$$

Now, if we suppose that $\exists i \in [0, \eta-1[$ such that $\forall j \in [0, \ell-1]$, $b_{i2\Phi'+j} = w_j$, then $\forall j \in [0, \ell-1]$, $-b_{i2\Phi'+j} - w_j + 2b_{i\Phi+j}b'_j = -2w_j + 2{w_j}^2 = 0$ as w_j is a bit, which gives

$$\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) = 0$$

which gives that under the condition of Lemma 2,

$$\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) \neq 0] = 0$$

and proves that condition (a) holds.

Now, if we suppose that $\forall i \in [0, \eta-1[$, $\exists j \in [0, \ell-1]$ such that $b_{i2\Phi'+j} \neq w_j$, and considering that $-b_{i2\Phi'+j} - w_j \neq 0$, $2b_{i2\Phi'+j}w_j = 0$, and we can deduce that $\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) \neq 0$ which gives:

$$\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) = 0] = 0$$

and proves that condition (b) holds. This concludes the proof. □

8.3 Proof of Theorem 1

.

Proof. The proof follows the blueprint of the proof of [BCS23, Theorem 1]. We will proceed by contradiction assuming an adversary $\mathcal{A}$ breaks the data-indistinguishability of our construction.
We set $T_0 := (F_0^{(0)}, \dots, F_{\eta-1}^{(0)})$ and $T_1 := (F_0^{(1)}, \dots, F_{\eta-1}^{(1)})$ the fragments of T_0 and T_1 submitted by $\mathcal{A}$. We define a sequence of games where the initial game is the experiment defined by $\mathsf{Exp}_{\mathcal{A}}^{\text{ind-cpa-trace}}()$. For $i = 0, \cdots, \eta - 1$, we define $\mathbf{G_i}$ the game such that the encryption of T_β is denoted $\mathsf{ct} = (c_0, \dots, c_{\eta-1})$ and computed as:

$$c_j = \begin{cases} \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\mathsf{pm}_1(F_j^{(1))})\right) & \text{if } j \leq i \\ \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\mathsf{pm}_1(F_j^{(0))})\right) & \text{else} \end{cases}$$

Thus, there exists an index $i^\star \geq 1$ such that $\mathcal{A}$ can distinguish $\mathbf{G_i^\star}$ from $\mathbf{G_{i^\star-1}}$.

We thus construct an adversary $\mathcal{B}$ against the $\mathsf{IND\text{-}CPA}$ security of the RLWE scheme. $\mathcal{B}$ submits $F_{i^\star}^{(0)}$ and $F_{i^\star}^{(1)}$ and receives a ciphertext $C_{i^\star}$. $\mathcal{B}$ then sends $\mathsf{ct}^\star$ to $\mathcal{A}$, where $\mathsf{ct}^\star := (c_0, \dots, c_{i^\star-1}, C_{i^\star}, c_{i^\star+1}, \dots, c_{\eta-1})$ such that:

$$c_j = \begin{cases} \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\mathsf{pm}_1(F_j^{(1))})\right) & \text{if } 0 \leq j < i^\star \\ \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\mathsf{pm}_1(F_j^{(0))})\right) & \text{if } i^\star < j < \eta \end{cases}$$

Note that if $C_{i^\star}$ encrypts $\mathsf{pm}_1(F_{i^\star}^{(1)})$, $\mathcal{A}$ plays in game $\mathbf{G_{i^\star}}$ and if $C_{i^\star}$ encrypts $\mathsf{pm}_1(F_{i^\star}^{(0)})$, $\mathcal{A}$ plays in game $\mathbf{G_{i^\star-1}}$. Then $\mathcal{B}$ forwards the same output as $\mathcal{A}$ for the $\mathsf{IND\text{-}CPA}$ game of the RLWEf scheme. Let denote E_i be the event that $\mathcal{A}$ outputs 0 in game $\mathbf{G_i}$ and let $\epsilon := |\Pr[E_{i^\star}] - \Pr[E_{i^\star-1}]|$ be the advantage of $\mathcal{A}$ of distinguishing both games, then $\mathcal{B}$ wins the $\mathsf{IND\text{-}CPA}$ game with the same advantage. □

8.4 Proof of Lemma 4

Proof. We follow the same steps as in the proof of Lemma 3. First $\mathcal{B}$ will be encrypted fragment-by-fragment under an pk honestly generated from KeyGen to get:

$$\{c_i = \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(\sum_{j=0}^{2\Phi'-1} b_{i2\Phi'+j} X^j\right) \mid i \in [0, \eta - 1]\}$$

In addition, the generation of a trapdoor for w gives the following two elements td_w^1 and td_w^1 (Equation 3). Let us define b'_i and $\bar{b}_i, i \in [0, \ell - 1]$ to

$$b'_i = \begin{cases} 0 & \text{if } i \in \mathcal{I} \\ w_i & \text{otherwise} \end{cases} \qquad \bar{b}_i = \begin{cases} 0 & \text{if } i \in \mathcal{I} \\ 1 & \text{otherwise} \end{cases}$$

Then, we can write $\mathsf{td}_w^{(1)}$ and $\mathsf{td}_w^{(2)}$ as following

$$\begin{aligned} \mathsf{td}_w^{(1)} &= \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(-\sum_{i=0}^{\ell-1} b'_i \cdot X^{N-i}\right) \\ \mathsf{td}_w^{(2)} &= \mathsf{rlwe.Enc}_{\mathsf{pk}}\left(-\sum_{i=0}^{\ell-1} \bar{b}_i \cdot X^{N-i}\right) \end{aligned} \tag{7}$$

Now, performing the $\mathtt{Match}$ algorithm between the encryption of the i_{th} section of $\mathcal{B}$ and $\mathsf{td}_w^{(1)}$ and $\mathsf{td}_w^{(2)}$ gives: $\mathsf{ct}_{\{i,i+1\}} = \mathsf{rlwe.Enc}_{\mathsf{pk}}(\sum_{j=0}^{N-1} b_{i2\Phi'+j} X^j)$.

For each $0 \leq i < \eta - 1$: $\mathsf{ct}^{(r)}_{\{i,i+1\}} = \mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}^{(2)}_w + (\mathsf{td}^{(1)}_w \cdot C' + (-2\mathsf{ct}_{\{i,i+1\}} \cdot \mathsf{td}^{(1)}_w)$ which under the correctness conditions from Lemma 2 gives an encryption of:

$$\begin{aligned}\mathsf{rlwe.Dec}_{\mathsf{sk}}(\mathsf{ct}^{(r)}_{\{i,i+1\}}) &= -\sum_{t=0}^{N-1} b_{i2\Phi'+t}X^t \cdot \sum_{j=0}^{\ell-1} \bar{b}_j X^{N-j} - \sum_{j=0}^{\ell-1} b'_j X^{N-j} \cdot \sum_{t=0}^{N-1} X^t \\ &\quad + 2\sum_{t=0}^{N-1} b_{i2\Phi'+t}X^t \cdot \sum_{j=0}^{\ell-1} b'_j X^{N-j} \\ &= \sum_{t=0}^{N-1}\sum_{j=0}^{\ell-1} X^{N+t-j}(-b_{i2\Phi'+t} \cdot \bar{b}_j - b'_j + 2b_{i2\Phi'+t}b'_j)\end{aligned} \tag{8}$$

Since $\forall j \in \mathcal{I} : b'_j = \bar{b}_j = 0$, we get:

$$\mathsf{rlwe.Dec}_{\mathsf{sk}}(\mathsf{ct}^{(r)}_{\{i,i+1\}}) = \sum_{t=0}^{N-1} \sum_{j=0, j\notin\mathcal{I}}^{\ell-1} X^{N+t-j}(-b_{i2\Phi'+t} - b'_j + 2b_{i2\Phi'+t}b'_j)$$

Now, if we suppose that $\exists i \in [0, \eta-1[$ such that $\forall j \in [0, \ell-1]$, $b_{i2\Phi'+j} = w_j$, then $\forall j \in [0, \ell-1]$, $-b_{i\Phi+j} - b'_j + 2b_{i\Phi+j}b'_j = -2b'_j + 2{b'_j}^2 = 0$ as $b'_j \in \{0,1\}$, which gives $\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) = 0$ which gives that under the condition of Lemma 2, $\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) \neq 0] = 0$ and proves that condition (a) holds. In the other hand, if we suppose that $\forall i \in [0, \eta-1[$, $\exists j \in [0, \ell-1]$ such that $b_{i\Phi+j} \neq w_j$, we can deduce that $\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) \neq 0$ which gives: $\Pr[\mathsf{hw}(\mathcal{B}^{(i)}, \mathbf{w}) = 0] = 0$ and proves that condition (b) holds. This concludes the proof.

$\square$