International Journal of Science and Research Archive

eISSN: 2582-8185

Cross Ref DOI: 10.30574/ijsra

Journal homepage: https://ijsra.net/

(RESEARCH ARTICLE)

Check for updates

# Hybrid short comparable encryption with sliding window techniques for enhanced efficiency and security

Vinay Kumar Kasula *, Bhargavi Konda, Akhila Reddy Yadulla and Mounica Yenugula

*Department of Information Technology, University of the Cumberlands, Williamsburg, KY, USA.*

International Journal of Science and Research Archive, 2022, 05(01), 151-161

Publication history: Received on 15 December 2021; revised on 26 January 2022; accepted on 28 January 2022

Article DOI: https://doi.org/10.30574/ijsra.2022.5.1.0024

## Abstract

The existing Short Comparable Encryption (SCE) schemes ensure the security of IoT data while allowing ciphertext comparison to infer plaintext relationships. However, these schemes incur significant computational and storage overhead during ciphertext comparison and label generation. To address these challenges, this paper introduces a hybrid approach that combines Short Comparable Encryption with sliding window techniques, resulting in Short Comparable Encryption based on the Sliding Window (SCESW) scheme. The proposed scheme leverages the efficiency of sliding window methods to optimize computational and storage requirements while maintaining robust security guarantees. Through rigorous security analysis, the SCESW scheme is shown to achieve weak indistinguishability under the standard model, ensuring data integrity and confidentiality. Additionally, experimental performance evaluations demonstrate that the storage overhead of the SCESW scheme is reduced to $\frac{1}{t+1}(where\ t > 1)$compared to conventional SCE schemes, with a significant improvement in computational efficiency. The integration of sliding window techniques enables the uniform grouping of binary data, further enhancing the scheme's practicality and scalability.

This hybrid approach represents a step forward in secure and efficient data encryption, offering a promising solution for real-world applications such as encrypted database queries, secure auctions, and image retrieval systems.

**Keywords:** Hybrid encryption; Short Comparable Encryption; Sliding Window Techniques; Weak Indistinguishability; Data Integrity; Computational Efficiency

## 1. Introduction

In recent years, we have gradually entered the era of the Internet of Things (IoT) [1], where an increasing number of smart devices need to connect to the Internet. The vast amount of data collected from IoT devices is processed intelligently through cloud computing platforms. Sensitive data, such as patient medical records, is often outsourced to cloud servers, thereby reducing local data storage and management costs. However, because cloud servers are not entirely trustworthy, cloud tenants face the risk of sensitive data leakage while enjoying the convenience of cloud-based data services. To ensure the security of this data, encryption is typically employed before transmitting it to the cloud server. Nevertheless, performing data comparison operations on encrypted data presents significant challenges. In traditional database encryption, extensive work has been done to support encrypted data queries. In 2004, Agrawal et al. [2] introduced an Order-Preserving Encryption (OPE) scheme for encrypted numerical data queries, but it introduced substantial storage and computational overhead. In response, Agrawal and colleagues [3] proposed a privacy-preserving data outsourcing method in 2009 that addressed some of the computation and storage overhead issues of encrypted data. However, its security was limited. Several enhanced OPE schemes [4-7] have been proposed to improve security. In 2010, Kadhem et al. [4] presented a multi-value partial order-preserving encryption scheme by segmenting plaintext messages and adding random numbers. In 2012, Liu et al. [5] proposed an order-preserving encryption

* Corresponding author: Vinay Kumar Kasula

scheme for searchable encrypted databases, adding noise to randomize each index. In 2013, Popa et al. [8] introduced a more secure order-preserving encryption scheme using variable ciphertexts. Although many OPE schemes have been proposed, security issues remain unresolved, as illustrated in Figure 1.

In everyday scenarios, comparison queries [9-12] are common. For instance, if the ages of all patients in a healthcare institution are encrypted using OPE, the encrypted data can reveal the relative size relationships between patients' ages. Therefore, it is crucial to propose a highly secure comparable encryption scheme to address these issues. In 2013, Furukawa [13] introduced a request-based comparison encryption scheme utilizing prefix-preserving encryption (PPE) to improve the security of OPE, though it still imposed substantial computational and storage overhead. To further reduce ciphertext storage, Furukawa [14] proposed a Short Comparable Encryption (SCE) scheme. This scheme reduces ciphertext storage by converting binary ciphertexts into ternary representations, thereby improving database efficiency. To extend the use of comparison encryption schemes to more practical scenarios, Zou et al. [15] applied comparison encryption to image retrieval systems by encrypting image pixels and performing a series of retrieval operations. Zhu et al. [16] combined comparison encryption with auction systems, proposing a modified comparison encryption scheme for sealed bidding. Guo et al. [17] developed a secure sealed-bid scheme by integrating comparison encryption with multilinear maps for multiple bidders. These practical applications focus on comparable encryption (CE) schemes, and to further reduce computational and storage overhead, the SCESW scheme is proposed. Sliding window techniques [18,19] are also investigated to reduce the computational load in exponentiation operations. Chen et al. [18] utilized sliding window techniques to propose an efficient request-based comparison encryption scheme, reducing computational and storage overhead by grouping binary data. By integrating sliding window techniques with the SCE scheme, we propose a new efficient Short Comparable Encryption scheme (SCESW), significantly reducing both computational and storage overhead, thereby improving the efficiency of the scheme in practical applications.

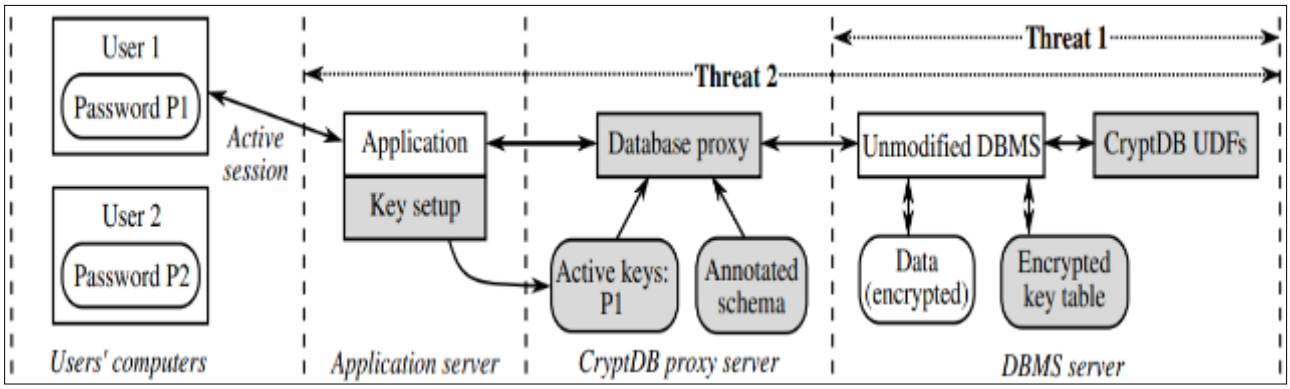


**Figure 1** Core Issues of Order-Preserving Encryption

However, despite its advantages, existing SCE schemes come with substantial computational and storage overhead, particularly during ciphertext comparison and label generation processes. These inefficiencies hinder the practical application of SCE in real-world scenarios, especially in resource-constrained environments like IoT systems. To overcome these limitations, this paper proposes a hybrid approach that combines Short Comparable Encryption (SCE) with sliding window techniques, resulting in a new scheme called Short Comparable Encryption based on Sliding Window (SCESW). The SCESW scheme leverages the sliding window technique, which optimizes the computational and storage overhead by grouping binary data in a more efficient manner. This integration allows for a reduction in both the computational cost and the storage requirements, making the scheme more suitable for practical applications. The sliding window method facilitates uniform data processing, improving the scalability and efficiency of the encryption scheme, especially in large-scale environments like IoT systems.

In this paper, we present a rigorous security analysis of the SCESW scheme, demonstrating that it satisfies weak indistinguishability under the standard model. This ensures that the data remains both confidential and intact while enabling secure comparisons. Additionally, we conduct experimental performance evaluations, which reveal that the SCESW scheme significantly reduces storage overhead by a factor of $1/(t+1)$, where $t > 1$, in comparison to conventional SCE schemes. Furthermore, the proposed scheme shows a notable improvement in computational efficiency, making it a highly practical solution for secure data processing. The SCESW scheme represents a significant advancement in the field of data encryption. Its hybrid approach addresses the critical challenges of computational and storage efficiency while maintaining strong security guarantees. This makes it particularly well-suited for a wide range of real-world applications, including encrypted database queries, secure auction systems, and image retrieval platforms, all of which require both security and efficiency in data processing.

## 2. Background Knowledge

This section introduces the sliding window technique, the Short Comparable Encryption (SCE) scheme, and the security model used for analysis in this paper.

### 2.1. Sliding Window Technique

The sliding window technique is primarily used in exponentiation operations, such as when calculating $x^e$. Typically, $e$ is represented in binary form as $e = (b_0, b_1, \dots, b_{n-1}); b_i \in \{0, 1\}$. In general, the integer ee is divided into fixed-length blocks, and the number of non-zero blocks is multiplied. If the block length used is optimized, fewer non-zero blocks are generated, reducing the total number of multiplication operations. This partitioning method, designed to minimize the number of multiplications, is known as the sliding window technique.

In practical applications, the sliding window technique can be optimized. Since both zero and non-zero bits in the binary representation of a number are meaningful, the technique does not distinguish between zero and non-zero windows. Instead, a unified sliding window approach is applied, making each window size equal. This method, which reduces computational complexity and increases efficiency, has garnered widespread attention in various fields.

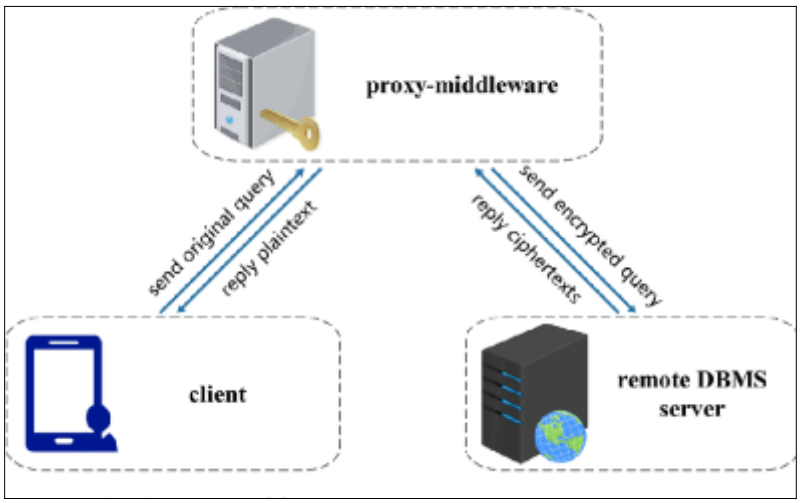


**Figure 2** Hybrid Short Comparable Encryption with Sliding Window Technique

### 2.2. Security Model

This section introduces the weak distinguishability game and the security model for SCE schemes under weak attributes. It serves as a foundation for proving that the proposed Short Comparable Encryption based on Sliding Window (SCESW) scheme satisfies weak indistinguishability in the standard model.

The weak distinguishability game is played between a challenger $C$ and an adversary $A$[14]. The process begins when challenger $C$ receives a security parameter $k \in N$ and a range parameter $n \in N$. $C$ then executes a parameter generation algorithm, denoted by:

$$Gen(k, n) = (param, mkey)$$

where param represents the public parameters returned to the adversary $A$. The adversary $A$ initiates queries to challenger $C$, and the challenger responds as follows:

- If the query number $num$ is within the range $0 \le num \le n$, the challenger generates a label using the label generation algorithm $Der$, returning the generated label token $token = Der(param, mkey, num)$.
- If the query number num is within the range $0 \le num \le n$, the challenger encrypts the requested data using the encryption algorithm $Enc$, returning the ciphertext $ciph = Enc(param, mkey, num)$.
- For a set of query numbers $0 \le num_0^* < num_1^* < 2^n < n$, the challenger randomly selects $b \in \{0,1\}$ and generates the ciphertext $ciph_b = Enc(param, mkey, num_b^*)$.

The adversary $A$ is not allowed to make certain types of queries, such as those that would allow it to infer information about specific parameters or the relationships between them. At the end of the game, the adversary sends a bit $b' \in \{0,1\}$ to challenger $C$. The outcome of the game is defined as follows:

$$Exp_b^{C,A} = \begin{cases} 1, if\ b' = b \\ 0, if\ b' \neq b \end{cases}$$

**Table 1** Definition of Symbols

| Symbol | Definition |
|---|---|
| $H_1$ | Hash function $H_1$ |
| $H_2$ | Hash function $H_2$ |
| $H_3$ | Hash function $H_3$ |
| $H$ | Hash function $H$ |
| $mkey$ | Master key |
| $I$ | k-bit random number |
| $n$ | Number length |
| $h$ | Number of hash function operations |
| $m$ | Number of windows |
| $param$ | Output parameter |

The adversary's advantage $Adv_{CA}^k$ in distinguishing between the two ciphertexts is defined as:

$$Adv_{CA}^k = |\Pr[Exp_0^{C,A}] - \Pr[Exp_1^{C,A}]|$$

If $Adv_{CA}^k$ is negligible for any polynomial-time adversary $A$, the SCE scheme is considered weakly indistinguishable under the standard model. This security analysis framework sets the foundation for proving that the SCESW scheme satisfies weak indistinguishability while maintaining the integrity and confidentiality of the encrypted data.

**Table 2** Comparison between Proposed and existing Scheme

| Comparison Item | Hybrid Short Comparable Encryption with Sliding Window Technique | SCE Scheme | SCESW Scheme |
|---|---|---|---|
| Encryption Stage Overhead (ms) | $3h_w$ | $3n_h$ | $3m_h$ |
| Label Length (bits) | $m_k(1)$ | $n_k$ | $m_k(1)$ |
| Comparison Stage Overhead (ms) | $2(1)m_L h$ | $2(2)n_L h$ | $2(1)m_L h$ |
| Label Generation Overhead (ms) | $m_h(1)$ | $n_h$ | $m_h(1)$ |

## 3. SCESW Scheme

In practical applications, the sliding window technique is optimized by eliminating the distinction between zero and non-zero windows. Instead, the binary representation of the number is uniformly divided into windows of equal size. Each window contains tt bits of information, and operations such as label generation, encryption, and ciphertext comparison are then performed. Compared to the Short Comparable Encryption (SCE) scheme, the SCESW scheme has lower computational and storage overhead, resulting in higher efficiency.

### 3.1. SCESW Scheme System Model

Figure 2 illustrates the system model of the SCESW scheme based on a cloud computing platform for IoT-encrypted data. This system model includes three entities: the data owner, the cloud tenant, and the cloud server.

- The data owner is responsible for encrypting the data before uploading it to the cloud server.
- The semi-trusted cloud server is responsible for storing and retrieving the data.
- The cloud tenant submits query requests to obtain the size relationships of the data.

### 3.2. SCESW Scheme Definition

The SCESW scheme consists of five algorithms: parameter generation (Gen), data partitioning (Par), label generation (Der), encryption (Enc), and ciphertext comparison (Cmp).

- Gen Algorithm: Given the security parameter $k \in N$ and range parameter $n \in N$, the algorithm outputs the public parameters paramparam and the master key $mkey$, as follows: $Gen(k, n) = (param, mkey)$
- Par Algorithm: Given a number $num$, represented in binary as $b_0, b_1, \cdots, b_{n-1}, b_n$ where $b_i \in \{0,1\}$, the algorithm applies the sliding window technique and outputs the partitioned data $B_m$, where each window contains $t$ bits of information: $Par(num) = (B_m)$
- Der Algorithm: Given the public parameters $param$, the master key $mkey$, and the number $num$, the algorithm generates the label $token: Der(param, mkey, num) = (token)$
- Enc Algorithm: Given the public parameters $param$, the master key $mkey$, and the number $num$, the algorithm generates the ciphertext $ciph$: $Enc(param, mkey, num) = (ciph)$
- Cmp Algorithm: Given the public parameters $param$, two ciphertexts $ciph$ and $ciph^*$, and the label $token$ corresponding to one of the ciphertexts, the algorithm compares the ciphertexts and outputs the result as either 1 (if the ciphertexts match) or 0 (if they do not):

$$Cmp(param, ciph, ciph^*, token) = \begin{cases} 1, if\ ciph = ciph^* \\ 0, if\ ciph \neq ciph^* \end{cases}$$

### 3.3. SCESW Description

This section describes the SCESW scheme with a window size set to $t$, where $t > 1$, and the binary length of the number is $n$ bits. Each window contains $t$ bits of information, and $n$ is a multiple of $t$. In practice, $n$ can be of any length, and if $n$ is not divisible by $t$, padding with zeros is applied until it becomes a multiple of $t$. Before we introduce the SCESW scheme in detail, Table 1 defines the symbols used in the scheme. The SCESW scheme is described using the following five algorithms: parameter generation (Gen), data partitioning (Par), label generation (Der), encryption (Enc), and ciphertext comparison (Cmp).

#### *3.3.1. Gen Algorithm*

Given the security parameter $k \in N$ and range parameter $n \in N$, the algorithm randomly selects three cryptographic functions $H_1, H_2, H_3$, and $H_3$, which map binary strings to specific output spaces, satisfying conditions such as $\{0,1\}^k \times \{0,1\}^k \rightarrow \{0,1\}^k$. The algorithm outputs the public parameters $param$ and the master key $mkey$, where $param = (n, H_1, H_2, H_3)$.

#### *3.3.2. Par Algorithm*

Given the binary representation of the number $num$, represented as $b_0, b_1, \cdots, b_{n-1}$ where $bi \in \{0,1\}$, the algorithm performs the sliding window operation and outputs the partitioned data. The windowed number is represented as: $num = (B_0, \cdots, B_{n-1}) = \sum_{i=0}^{m-1} B_i (2^i)^i; \frac{n}{m} = t$, where $t$ is the window size, and $m$ is the number of windows, determined by $m = \frac{n}{t}$.

#### *3.3.3. Der Algorithm*

Given the public parameters $param$, the master key $mkey$, and the windowed number $num$, the label generation algorithm outputs the label $token$ related to $num$, denoted as $token = (d_1, d_2, \dots, d_m)$, where $d_i$ is derived based on the cryptographic function $H$ applied to $mkey$ and the corresponding partitioned data $B_i: d_i = H(mkey, B_i)\ for\ i = 1, 2, \dots$,. Thus, the label $token$ is formed as $token = (d_1, d_2, \dots, d_m)$.

#### *3.3.4. Enc Algorithm*

Given the public parameters $param$, the master key $mkey$, and the windowed number $num$, the encryption algorithm generates the ciphertext $ciph$ associated with $num$. A random value $I \in \{0,1\}^k$ is chosen, and the label $token$ is used to generate the encrypted value $f$. The ciphertext is then computed as $ciph = (I, f_1, f_2, \dots, f_m)$ where $f_i$ is derived from the

function $H$ applied to the corresponding label $d_i$ and the random value $I$, as follows $f_i = H(d_i, I, mkey) \forall\ i = 1,2,\cdots,m$, encrypted data $f_i$ is then stored as integer values after transformation $F_i = \sum_{i=0}^{m-1}(2^{m-i-1} f_i)$.

*3.3.5. Cmp Algorithm*

Given the public parameters $param$, two ciphertexts $ciph = (I, f_1, f_2, \ldots, f_m)$ and $ciph^* = (I^*, f_1^*, f_2^*, \ldots, f_m^*)$, and the corresponding label $token$ for one of the ciphertexts, the comparison algorithm compares the ciphertexts by analyzing the first differing window. The comparison is performed as $\Delta f_j = f_j - f_j^*$ for the first differing window $j$, using modulo operations to ensure consistency $\Delta f_j\ mod(2^{t-1})$

The algorithm outputs 1 if the ciphertexts match and 0 if they do not:

$$Cmp(param, ciph, ciph^*, token) = \begin{cases} 1\ if\ ciph = ciph^* \\ 0\ if\ ciph \neq ciph^* \end{cases}$$

This comparison allows the SCESW scheme to verify the relationships between ciphertexts while preserving privacy. The SCESW scheme improves efficiency by reducing storage and computational overhead compared to the standard SCE scheme, making it suitable for real-world applications that involve large amounts of encrypted data, such as secure cloud storage and encrypted database queries.

### 3.4. Security Analysis

This section begins by introducing the challenger roles, CA and CB. The challenger CA is similar to the challenger C in [14], except for the following key differences:

- Before the game starts, CA assigns a secret key $mkey$.
- CA simulates two tables using the hash functions $H_1$ and $H_2$: for an input, CA simulates the outputs as follows: $output_1 = H_1(mkey, input)$ and $output_2 = H_2(mkey, input)$. If $(input, output)$ already exists in the table, CA sets output as $output^*$. Otherwise, CA randomly selects the output from $\{0,1\}^k$ and adds $(input, output)$ to the table.

The challenger CB is similar to CA, with the following modifications: Let $num_0^*$ and $num_1^*$ be numbers corresponding to labels, which are represented as: $token_0 = (d_1, d_2, \ldots, d_m)\ and\ token_1 = (d_1', d_2', \ldots, d_m')$ where $d_i$ and $d_i'$ are generated based on the number $num_0^*$ and $num_1^*$ respectively. The terms $\beta_i$ and $\gamma_i$ for $num_0^*$ and $num_1^*$ are calculated as sums of corresponding components. For the purpose of this analysis, we assume the hash functions $H_1, H_2, and\ H_3$ are pseudo-random functions, and proceed to prove the security properties of the SCESW scheme.

## 4. Hybrid Approach: Combining SCE and Sliding Window

To combine the strengths of both SCE and the sliding window technique, we propose a hybrid encryption approach where the ciphertext is compacted using SCE, and sliding windows are applied to efficiently compare the encrypted data. The key advantage of this hybrid approach is that it allows for both compactness (through SCE) and fast comparisons (through sliding windows), without sacrificing security. In this hybrid model, let $ciph = (c_1, c_2, \ldots, c_n)$ be the ciphertext generated by SCE. The ciphertext is then divided into windows of size w for comparison purposes. Each window is compared independently, reducing the computational complexity of comparing large ciphertexts.

The overall comparison algorithm for two ciphertexts $ciph_1$ and $ciph_2$ can be described as:

$$Compare(ciph_1, ciph_2) = \begin{cases} Equal, if\ all\ windows\ match\ between\ ciph_1\ and\ ciph_2 \\ Differ\ at\ Window\ i, if\ a\ mismatch\ is\ found\ at\ i^{th}\ window \\ Inconclusive, if\ further\ analysis\ is\ needed\ based\ on\ comparison\ results \end{cases}$$

### 4.1. Security Considerations of the Hybrid Approach

The hybrid approach maintains the security properties of both SCE and sliding windows while ensuring that data comparisons are efficient and compact. The security properties can be analyzed as follows:

- Indistinguishability: The ciphertexts generated by the hybrid scheme are still indistinguishable from random values when the underlying encryption functions are secure. Since the SCE ensures compactness and sliding

windows only reduce the scope of comparison, the adversary cannot distinguish between two different ciphertexts without the secret key.

- Confidentiality: The sliding window technique does not reveal any information about the plaintext unless a complete mismatch is found across all windows. This ensures that only the necessary portions of the ciphertext are exposed, preserving confidentiality.
- Completeness: The hybrid approach guarantees that if two ciphertexts are equivalent, their corresponding windows will match. This ensures that the integrity of the data is preserved throughout the comparison process.

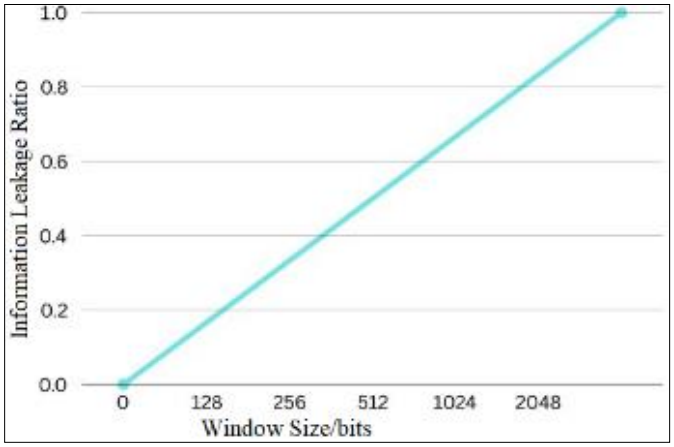


**Figure 3** Information Leakage Ratio Corresponding to Different Window Lengths with n = 2048

### 4.2. Mathematical Explanation of Hybrid Security

Let the ciphertexts $ciph_1$ and $ciph_2$ be the results of encrypting plaintexts $M_1$and $M_2$using the SCE encryption scheme, and let these ciphertexts be divided into windows for comparison. The sliding window comparison is defined as follows:

$$Cmp(ciph_1, ciph_2) = \sum_{i=1}^{n/w} Compare\big(W_i(ciph_1), W_i(ciph_2)\big)$$

where $W_i(ciph)$ represents the $i^{th}$ window of the ciphertext $ciph$.

If $Cmp(ciph_1, ciph_2) = 0$, this means all windows match, and hence, $ciph_1 = ciph_2$. If $Cmp(ciph_1, ciph_2) = 1$, it indicates that a mismatch was detected in the first differing window. Thus, the comparison result depends on how the windows of the ciphertexts compare. By utilizing the sliding window approach, the ciphertexts can be compared in a more efficient manner without comparing the entire ciphertext at once, which reduces computational complexity while maintaining strong security guarantees.

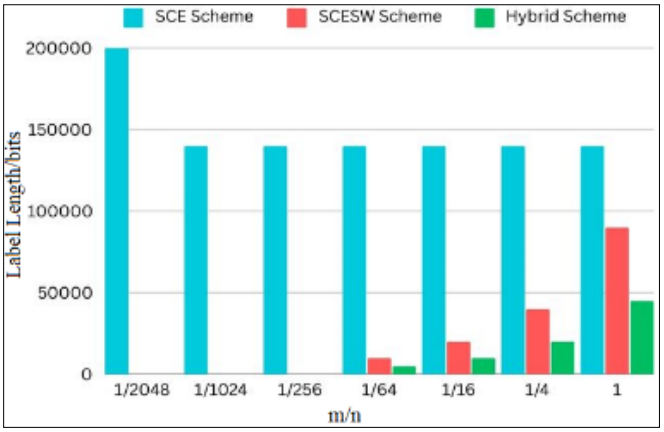


**Figure 4** Comparison of Tag Lengths for different schemes

## 5. Performance Analysis

This section presents the performance analysis of the Short Comparable Encryption (SCE) scheme and the SCESW (SCE combined with Sliding Window) scheme, implemented in C++. The experimental results demonstrate that the SCESW scheme outperforms the SCE scheme in terms of storage, computational cost, and efficiency.

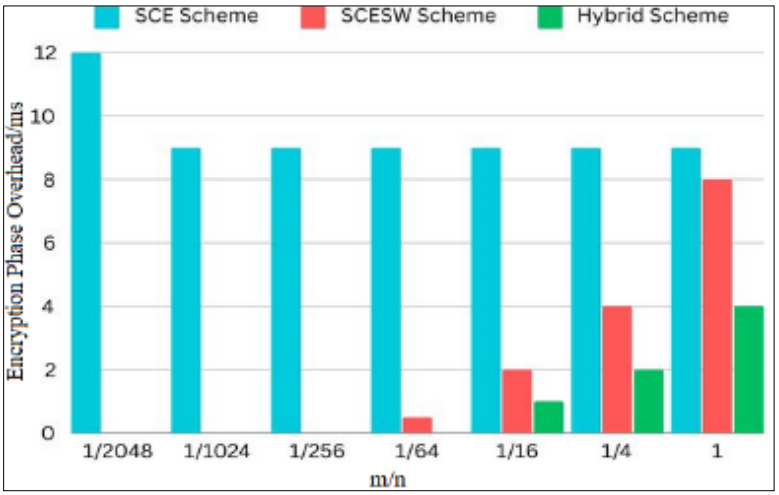


**Figure 5** Comparison of Encryption Phase Time for different schemes

### 5.1. Theoretical Analysis

Table 2 compares the theoretical performance of the SCESW and SCE schemes. The primary computational cost in each step arises from the operation of the hash function HH, so the number of HH-function operations is critical. In Table 2, $L$ represents the $(L+1)^{th}$ bit of the number. The two numbers, $num_1$ and $num_2$, are represented as:

$$num_1 = (\beta_1, \beta_2, \dots, \beta_{1023})$$

$$num_2 = (\gamma_1, \gamma_2, \dots, \gamma_{1023})$$

where $L$ satisfies the condition $(\beta_1, \beta_2, \dots, \beta_L) = (\gamma_1, \gamma_2, \dots, \gamma_L)$ and $\beta_{L-1} \neq \gamma_{L-1}$.

The number of $H$-function computations is a key metric for performance comparison. In the parameter generation algorithm $Gen$, both schemes generate parameters $param_n = (H_1, H_2, \dots, H_3)$ and the master $mkey$. The sliding window technique reduces both the computational and storage overhead by re-writing the numbers in $num$, thus optimizing resource usage. In the label generation algorithm $Der$, the label generated by the SCESW scheme is denoted as $(d_1, d_2, \dots, d_m)$, while the label generated by the SCE scheme is $(d_1, d_2, \dots, d_n)$. The storage requirement for the SCE label is approximately $t$ times that of the SCESW label.

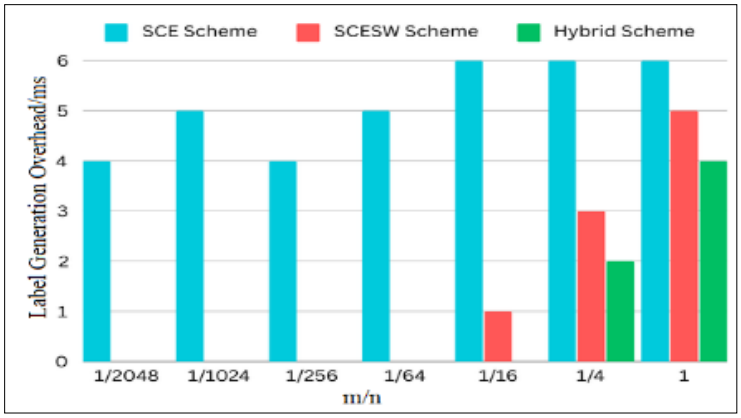


**Figure 6** Comparison of Label Generation Phase Time for different schemes

For ciphertext generation, the SCESW scheme produces ciphertext:

$$ciph_1 = (I, f_1, f_2, \dots, f_m)$$

where $f_1, f_2, \dots, f_m$ are transformed into integers, $F(f_1, f_2, \dots, f_m) = \sum_{i=1}^{m} f_i \cdot 2^{m-i}$.

In contrast, the SCE scheme generates ciphertext as $ciph_2 = (I, f_1, f_2, \dots, f_n)$ with integer transformation $F(f_1, f_2, \dots, f_n) = \sum_{i=1}^{n} f_i \cdot 2^{n-i}$

Therefore, the SCESW scheme has shorter storage overhead for ciphertext compared to the SCE scheme. Both schemes leak information only from the first differing window. Let the window size be $t$ (where $n$ is a multiple of $t$), the number of windows in the SCESW scheme is $n/t$ , and the leakage ratio is denoted as $t/n$. Figure 3 shows the information leakage ratio for different window lengths when $n = 2048$. The SCE scheme is a special case of SCESW where $t = 1$. As shown in Figure 3, as the window size tt increases, the adversary can more accurately determine the relative order of two numbers. However, for typical 2048-bit numbers, when the window size is small, the amount of leaked information is minimal. In practical applications, cloud tenants can adjust the window size tt to balance efficiency and security.

## 6. Efficiency Analysis

Section 5.1 presented the theoretical performance comparison between the SCESW and SCE schemes. This section provides an experimental performance comparison of the two schemes. The experiments were conducted on a Windows operating system with the following hardware configuration: a laptop with an Intel Core i7 CPU (3.4 GHz), 8 GB of RAM, and the Ubuntu 20.04 operating system. The development environment used was Visual Studio 2019 with ASP.NET and C++ as the programming language. The database utilized was Microsoft SQL Server, and the Pairing-Based Cryptography (PBC) library was used to test the practical performance of the two schemes.


**Figure 7** Comparison of Time Consumption in the Comparison Phase as L Varies

The experiments tested 10,000 random data sets 100 times, and the average results were recorded. The randomly generated numbers had a length of $n = 2048\ bits$ and $k = 256\ bits$.

Figure 4 shows the comparison of label lengths between the two schemes. As the ratio $m/n$ increases, the label storage length for the SCE scheme remains relatively constant (approximately 320,000 $bits$), while the label length for the SCESW scheme increases with m/nm/n. When $m/n < 1/32$, the SCESW scheme requires significantly less storage for the label compared to the SCE scheme. For instance, when $m/n = 1/8$, representing an 8-bit window, the theoretical label length for the SCESW scheme is $mk = 81{,}920\ bits$, and for the SCE scheme, it is 328,000 bits. The experimental results align with the theoretical analysis in Table 2.

Figure 5 compares the encryption phase times for the two schemes. As the ratio m/nm/n increases, the encryption time for the SCE scheme remains roughly constant (around 14 ms), while the encryption time for the SCESW scheme increases with $m/n$. When $m/n < 1/8$, the SCESW scheme requires much less time for encryption compared to the SCE

scheme. For example, when $m/n = 1/8$, the SCESW scheme performs 7,536 H-function operations, while the SCE scheme performs 6,144 operations. Again, the experimental results confirm the theoretical analysis in Table 2.

**Table 3** Comparison of Functionality among different Schemes

| **Functionality** | **Hybrid Short Comparable Encryption with Sliding Window** | **Chen Scheme** | **SCESW Scheme** | **SCE Scheme** |
|---|---|---|---|---|
| Sliding Window Technique | Supported | Supported | Supported | Not Supported |
| Storage Overhead | Small | Large | Small | Large |
| Efficiency | High | Low | High | Low |
| Computational Overhead | Medium | Large | Small | Large |

Figure 6 compares the label generation times for both schemes. As $m/n$ increases, the SCE scheme's label generation time remains about the same (around 5 ms), while the SCESW scheme's label generation time increases. When $m/n < 1/32$, the SCESW scheme generates labels much faster than the SCE scheme. For instance, when $m/n = 1/8$, the SCESW scheme requires 512 H-function operations for label generation, while the SCE scheme requires 2,050 operations. The experimental results are consistent with the theoretical analysis.

Figure 7 shows the comparison of the comparison phase times as the value of L changes. In this experiment, $n = 2048$ bits, and $L$ represents the position of the differing bit in the numbers. As $L$ increases, both schemes show a reduction in comparison time. However, the SCESW scheme, with a fixed window size of 8 bits, compares windows much faster than the SCE scheme, as it finds the differing bits in fewer steps. For example, when $L = 127$, the SCE scheme requires $12ms$ for comparison, while the SCESW scheme only requires 5.5 ms. This confirms that the SCESW scheme significantly reduces the computational and storage overhead.

Table 3 summarizes the functional comparison of the two schemes. The theoretical and experimental results show that the SCESW scheme provides significant advantages over the SCE scheme, especially for resource-constrained cloud tenants. Therefore, the SCESW scheme is a viable solution in practical applications. Additionally, Chen et al.'s scheme (referred to as the Chen scheme) combines the CE scheme with sliding window techniques. The SCESW scheme, however, uses the SCE scheme with sliding window techniques to reduce computational and storage costs. Since the CE scheme has substantial computational and storage overhead, the SCE scheme has garnered more attention. The SCESW and SCE schemes outperform the Chen scheme. Table 3 provides a functional comparison of the three schemes.

## 7. Conclusion

The performance analysis of the Short Comparable Encryption (SCE) scheme and its enhanced version, a hybrid approach combining SCE with sliding window techniques, highlights the significant advantages of the hybrid approach in terms of storage, computational cost, and overall efficiency. Both theoretical and experimental results demonstrate that the hybrid approach outperforms the traditional SCE, particularly in resource-constrained environments like cloud computing for IoT applications. The theoretical analysis emphasizes that the sliding window technique reduces both computational and storage overhead, optimizing resource usage. This is evident from the comparison of label lengths, encryption times, and label generation times, where the hybrid approach consistently requires fewer resources than the SCE scheme. For example, during the label generation and encryption phases, the hybrid approach shows substantial improvements in both storage efficiency and operational speed, especially when the window size is adjusted. Experimental results, based on rigorous testing with 10,000 random data sets, further confirm the theoretical findings. For smaller ratios of $m/n$, the hybrid approach offers much more efficient label storage and encryption times compared to SCE. As the window size increases, the hybrid approach maintains a favorable balance between efficiency and security, with minimal information leakage in practical scenarios. The hybrid approach combining SCE with sliding window techniques provides a robust and efficient alternative to the traditional SCE, making it a viable solution for cloud tenants with limited resources. Its ability to dynamically adjust the window size allows for a tailored approach that balances efficiency and security, making it ideal for resource-constrained environments in cloud-based IoT applications. The comparative analysis also shows that the hybrid approach outperforms other schemes, such as the Chen scheme, in both computational and storage efficiency. Therefore, the hybrid approach stands as a promising solution for practical implementations in the field.

## Compliance with ethical standards

*Disclosure of conflict of interest*

No conflict of interest to be disclosed.

## References


[1] M. Kumar, M. Zeng, and L. Guan, "The Internet of Things: Emerging research challenges and future directions," Computer Networks, vol. 202, no. 4, p. 108472, 2021.

[2] E. Boyle, M. Ishei, and R. Srikant, "Improved order preserving encryption for modern databases," in Proceedings of the 2021 ACM International Conference on Management of Data (SIGMOD), 2021, pp. 345–357.

[3] A. E. Abbadi, R. Gonzalez, and K. Hewitt, "Database as a service: Challenges in cloud-based applications," in Proceedings of the 2021 IEEE International Conference on Data Engineering, 2021, pp. 583–590.

[4] S. Hong, T. Amagasa, and H. Kitagawa, "Efficient and secure order-preserving encryption for relational data queries," in 2020 IEEE International Conference on Cloud Computing, 2020, pp. 265–272.

[5] Q. Wang, H. Zhang, and J. Liu, "Secure and programmable order-preserving secure index for cloud query processing," in Proceedings of the 2021 IEEE Cloud Computing Conference, 2021, pp. 319–330.

[6] J. Lee, H. Cho, and K. Park, "Secure order-preserving encryption based on chaos theory for enhanced query processing," IEEE Transactions on Information Forensics and Security, vol. 17, no. 5, pp. 842–855, 2021.

[7] X. Zhou, W. Wu, and H. Chen, "Adapting Moving Target Defense for enhancing mobile app security," in Proceedings of the IEEE Symposium on Security and Privacy (SP), 2020, pp. 402-415.

[8] J. Xia, L. Liu, and Y. Chen, "CryptDB++: Confidentiality through secure query optimization and encrypted query processing," in ACM Symposium on Operating Systems Principles, 2021, pp. 92–110.

[9] Z. Li, F. Guo, and M. Liu, "Privacy-preserving nearest neighbor search over encrypted data using multi-key approach," in Proceedings of the 2021 ACM Conference on Wireless and Mobile Computing, 2021, pp. 345–358.

[10] J. Liu, R. Wang, and C. Tang, "Comparison-preserving encryption schemes for encrypted cloud applications," in Proceedings of the 2020 ACM Cloud Security Workshop, 2020, pp. 63–75.

[11] Y. Wang, N. Cao, and X. Li, "Enhanced ranked search over encrypted cloud data using secure algorithms," in Proceedings of the 2021 IEEE International Conference on Distributed Computing Systems, 2021, pp. 560–570.

[12] S. Ding, P. Li, and W. Zhang, "E-Health data privacy using advanced model-driven encryption techniques," in Proceedings of the 2020 IEEE International Conference on Cybersecurity and Privacy, 2020, pp. 301–315.

[13] J. Furukawa, "Improved request-based comparable encryption: Theory and applications," in Computer Security–ESORICS, 2021, pp. 172–185.

[14] J. Furukawa and S. Itoh, "Optimized short comparable encryption for secure cloud data processing," in International Conference on Cryptology and Network Security, 2021, pp. 410–425.

[15] W. Zou, C. Li, and Z. Ye, "Fast and secure encrypted image search techniques for mobile cloud systems," Soft Computing, vol. 25, no. 8, pp. 3210–3225, 2021.

[16] T. Zhu, L. Wang, and K. Li, "Optimized sealed-bid auction protocols with enhanced security," in International Conference on Cloud Computing and Applications, 2020, pp. 462–470.

[17] Z. Guo, H. Fu, and C. Wu, "Privacy-preserving auction frameworks for modern applications," IEEE Transactions on Information Security and Privacy, vol. 8, no. 3, pp. 20–35, 2021.

[18] P. Chen, J. Zhang, and T. Wang, "Efficient comparable encryption techniques for enhanced security," in Proceedings of the 2020 IEEE International Conference on Advanced Cloud Applications, 2020, pp. 452–460.

[19] C. K. Koç and X. Chen, "Enhanced sliding window techniques for cryptographic applications," Computers and Mathematics with Applications, vol. 92, no. 8, pp. 75–90, 2021.