Rust Implementation Suite for *k*-Resilient ID-Based Cryptosystem

Poh-Wen Kho1, a), Syh-Yuan Tan1, b), Swee-Huay Heng 2, c), Koon-Ming Chan1, d), Evon Yee-Ting Ng1, e), Anik-Sen1, f)

1Faculty of Information Science and Technology, Multimedia University, Jalan Ayer Keroh Lama, 75450 Bukit Beruang, Melaka, Malaysia.

2Centre for Intelligent Cloud Computing, CoE for Advanced Cloud, Faculty of Information Science and Technology, Multimedia University, Jalan Ayer Keroh Lama, 75450 Bukit Beruang, Melaka, Malaysia.

*c) Corresponding author: shheng@mmu.edu.my   
a) pohwenkho@gmail.com*

*b) sytan@mmu.edu.my*

*d) koonming1996@gmail.com*

*e) evonngyt1602@gmail.com*

*f) aniksen360@gmail.com*

**Abstract.** Many identity-based and searchable encryption schemes rely on bilinear pairings, which can be computationally expensive and impractical for lightweight or real-time applications. In contrast, *k*-resilient schemes constructed over finite fields offer a performance-friendly alternative. The computational cost of exponentiation in finite fields is significantly lower than that of bilinear pairings, allowing these schemes to achieve competitive runtime efficiency. Thus, in this paper, we present the *k*-resilient suite, a unified cryptographic application that integrates four *k*-resilient schemes: *k*-resilient Identity-Based Encryption (KR-IBE), *k*-resilient Identity-Based Identification (KR-IBI), *k*-resilient Public Key Encryption with Keyword Search (KR-PEKS), and *k*-resilient Public-key Authenticated Encryption with Keyword Search (KR-PAEKS). To the best of our knowledge, these *k*-resilient primitives have not been implemented in practice, and existing works remain conceptual, without real deployment or interface evaluation. We address the lack of practical implementations by developing a cross-platform desktop application using Rust and Tauri to demonstrate the practical usability of these schemes. Our implementation includes intuitive user interfaces for cryptographic operations such as setup, key extraction, encryption, decryption, and keyword search, enabling hands-on evaluation and benchmarking. We also conduct a detailed performance analysis of each scheme, measuring the total execution time under certain parameters. Our results show that all schemes operate within practical time bounds, confirming their feasibility for real-world use cases involving interactive cryptographic applications. The analysis provides insights into the trade-offs and feasibility of deploying *k*-resilient cryptographic primitives in real-world applications.

# Introduction

In recent years, secure data sharing has become a fundamental requirement in distributed, cloud-based, and multi-user environments. As increasing volumes of sensitive information are exchanged and stored online, cryptographic mechanisms that offer both strong security guarantees and practical deployment have gained significant attention. Identity-Based Cryptography (IBC), originally introduced by Shamir [1], simplifies key management by enabling public keys to be generated from user identities, thereby removing the dependency on conventional certificate authorities. A practical realisation of IBC was later introduced by Boneh and Franklin [2], whose Identity-Based Encryption (IBE) scheme utilises bilinear pairings to achieve chosen ciphertext security in the random oracle model.

However, pairing-based schemes are computationally intensive and may present challenges for real-time or resource-constrained environments. Besides, most of the schemes achieved chosen ciphertext security in the random oracle model only instead of standard model. To address this, Heng and Kurosawa [3] introduced the notion of *k*-resilience. Their *k*-resilient IBE scheme is constructed based on exponentiation in finite fields, thus it is significantly more efficient than pairing computations, and the security is proved without relying on random oracles

The concept of *k*-resilience was later extended to identification and searchable encryption primitives. Chin and Heng [4] introduced a *k*-resilient Identity-Based Identification (KR-IBI) scheme that withstands active attacks, making it well-suited for applications such as access control and identity authentication. Meanwhile, searchable encryption has become increasingly important as it enables search functionality over encrypted data. Boneh et al. [5] introduced Public Key Encryption with Keyword Search (PEKS), enabling users to perform keyword searches on encrypted data through the use of trapdoors. However, PEKS schemes are vulnerable to offline Keyword Guessing Attacks (KGA) , particularly when trapdoors or ciphertexts are exposed. The construction of many PEKS schemes is based on bilinear pairing techniques. To reduce this reliance, Khader [6] proposed a *k*-resilient PEKS (KR-PEKS) that achieves indistinguishability under chosen keyword attacks (IND-CKA). Yau et al. [7] further improved the scheme by reducing reliance on strong IBE security assumptions while maintaining the security level, improving efficiency. However, their schemes lack trapdoor privacy and is vulnerable to KGA [8]. To counter KGA and ensure both confidentiality and sender authenticity in keyword search, Chan et al. [9] proposed a *k*-resilient Public-key Authenticated Encryption with Keyword Search (KR-PAEKS). These advancements enable authenticated and privacy-preserving keyword searches over encrypted data, addressing a critical requirement in secure multi-party communication systems.

Despite theoretical maturity, there is no unified, practical implementation of these *k*-resilient schemes, preventing broader adoption and performance validation of these schemes in real-world environments. Building on these foundations, we address that gap by presenting the *k*-resilient suite, the first unified desktop application implements and evaluates the integrated use of *k*-resilient primitives: *k*-resilient Identity-Based Encryption (KR-IBE), *k*-resilient Identity-Based Identification (KR-IBI), *k*-resilient Public Key Encryption with Keyword Search (KR-PEKS), and *k*-resilient Public-key Authenticated Encryption with Keyword Search (KR-PAEKS). Built using Rust and the Tauri framework, our cross-platform system features an interactive interface to demonstrate each scheme’s functionality. All cryptographic operations are executed locally through the MIRACL Core library using the Ed25519 curve. We also present detailed performance benchmarks and assess each scheme’s practicality, highlighting the trade-offs and feasibility of deploying *k*-resilient primitives in secure data sharing scenarios.

# Review of k-resilient schemes

## KR-IBE Scheme

We recall the KR-IBEscheme from Heng and Kurosawa [3]. The scheme comprises four algorithms, namely, Setup, Extract, Encrypt, and Decrypt. We outline their scheme construction as follows:

## Setup: Let be a multiplicative group of prime order and choose two generators . Define six random *k*-degree polynomials over :

Next, the algorithm computes: for and let be a collision-resistant hash function. The system then outputs the system parameters: The master key, known only to the PKG is: .

Extract: Given a public identity , compute: .

Encrypt: To perform encryption on a message using the public identity , and choose to compute and output the ciphertext.

Decrypt: Given a ciphertext and the private key , compute then test if abort the process if the condition fails to hold, then compute to decrypt the message.

## KR-IBI Scheme

We recall the KR-IBI scheme from Chin and Heng [4]. The scheme comprises three algorithms, namely, Setup, Extract, and Identification Protocol. We reiterate their scheme construction as follows:

Setup: Let be a cyclic group of prime order where and is prime. Select a random generator and choose a random *k*-degree polynomial over . The system parameters are defined as . The polynomial serves as the master key and is kept as secret.

Extract: Given a public identity (hash function can be used to hash it to the appropriate length), use the master key to compute .

Identification Protocol: The prover selects a random value , computes and sends it to verifier. Then, the verifier responds with a random challenge and sends it back to the prover. The prover computes the value and returns it to the verifier. The verifier accepts if .

## KR-PEKS Scheme

We recall the KR-PEKS scheme from Yau et al. [7]. The scheme comprises four algorithms, namely, KeyGen, PEKS, Trapdoor, and Test. We outline their scheme construction as follows:

KeyGen: To generate a public and private key pair: a group of order is selected along with a generator ∈ . A random is chosen and compute = . Next, two random polynomials and of -degree polynomials over are selected. Then, compute , for . Finally, return the private key and public key .

PEKS: To encrypt a keyword under the receiver’s public key , the sender choose to compute Then output the ciphertext,.

Trapdoor: To generate a trapdoor for a keyword , the receiver (who holds the private key) evaluates the private polynomials at .

Test: The gateway will determine whether a ciphertext encrypts by determining whether . It will output “1” if the result matched, else output “0”. To prove the correctness of the equation:

## KR-PAEKS Scheme

We recall the KR-PAEKS scheme from Chan et al. [9]. The scheme comprises six algorithms, namely, Setup, SenderKeyGen, ReceiverKeyGen, PAEKS, Trapdoor, and Test. We outline their scheme construction as follows:

Setup: Let be a cylic group of prime order , with generator ∈ . Select random and compute = . Choose a random . Compute the common parameter .

SenderKeyGen: Choose two random -degree polynomials over : ; . Then, compute , for . Set the private key  and the public key .

ReceiverKeyGen: Select two random -degree polynomials over : ; .

Then, compute , for . Set the private key and the public key .

PAEKS: Choose a random . For a chosen keyword compute , , and . Lastly, output the searchable ciphertext .

Trapdoor: Choose a random . For a chosen keyword , compute , , and . Lastly, output the trapdoor .

Test: Upon receiving a trapdoor , the server will test for a matching searchable ciphertext by performing . It will output “1” if the result matched, else output “0”. The correctness of the equation is as follows: .

# SYStem design and Implementation

Our system is a desktop application developed for performance analysis of four k-resilient cryptographic schemes: KR-IBE, KR-IBI, KR-PEKS, and KR-PAEKS. Built using the Tauri framework, it combines a React-based frontend for user interaction with a Rust backend that executes all cryptographic operations through the MIRACL Core library, employing the Ed25519 curve for underlying cryptographic functions.

As shown in Figure 1, the context diagram illustrates how the system operates in isolation, the sole user (performance tester) interacts directly with the application to initiate tests and view results, without reliance on external services such as databases or network connectivity. The user interface, depicted in Figure 2, presents an intuitive layout that allows performance testers to configure the k security parameter and execute benchmarks across the four cryptographic primitives.

All cryptographic operations are performed locally, with results processed in-memory and displayed through the React UI. The Rust backend ensures high-performance execution of cryptographic computations, while the Tauri framework enables seamless and efficient communication between the frontend and backend components. This closed-system design supports precise measurement and comparison of the schemes’ performance characteristics under controlled conditions, making it an ideal evaluation tool for researchers investigating the practical efficiency of k-resilient cryptosystems.

# Performance analysis

The performance evaluation of our system was conducted on a standard consumer-grade hardware configuration featuring an Intel Core i5-8265U processor running at 1.60 GHz base frequency (with turbo boost up to 1.80 GHz) and 12 GB of RAM. To ensure consistent benchmarking conditions across all tests, we fixed the identity parameter as ID = "*alicebob@gmail.com*" and used the string "*keyword*" as both the searchable keyword for PEKS/PAEKS schemes and the plaintext input for IBE encryption operations. This controlled testing environment provided realistic measurements of computational efficiency while eliminating variables that could affect comparative analysis.

A diagram of a performance tester

AI-generated content may be incorrect.

**FIGURE 1.** Context Diagram

A screenshot of a computer

AI-generated content may be incorrect.

**FIGURE 2.** User Interface

The results (Figure 3) demonstrate clear performance characteristics across different security parameter values (*k*= 20 to 100), with all schemes showing predictable growth in total execution time as *k* increases. KR-IBE exhibited the highest computational overhead among the four schemes, with total execution times growing from 33 ms for encrypting/decrypting the "*keyword*" plaintext at *k* = 20 to 163 ms at *k* = 100. This performance profile reflects the inherent complexity of its identity-based encryption operations under the fixed identity parameter.

KR-IBI showed more efficient performance characteristics when processing the same identity ("*alicebob@gmail.com*"), completing all cryptographic operations in approximately half the total execution time of KR-IBE across all tested *k* values, ranging from 17 ms to 77 ms. The keyword search schemes demonstrated particularly interesting results when querying for the fixed keyword string — KR-PEKS maintained the fastest total execution times throughout the analysis (15 ms to 61 ms), while KR-PAEKS, with its additional authentication features for the same "*keyword*" search functionality, showed intermediate performance between KR-PEKS and KR-IBI, ranging from 38 ms to 146 ms.

A graph of a number of execution times

AI-generated content may be incorrect.

**FIGURE 3.** Performance Analysis

These measurements reveal several important patterns. The near-linear growth in total runtime across all schemes confirms their designed *k*-resilience approach even with fixed input parameters. The performance hierarchy indicates that although the *k*-resilient schemes serve different purposes, their performance are on a par to each other and the *k*-resilience techniques provide stable performance characteristics. This consistency suggests that future variants of *k*-resilient schemes, if constructed using similar cryptographic primitives and finite field operations, would likely demonstrate comparable performance characteristics without significant deviation from the current results. Execution times below 200 ms indicate that these schemes are efficient enough for deployment in interactive applications, such as secure messaging or keyword-based search systems. The consistent use of "*alicebob@gmail.com*" as the identity parameter and "*keyword*" as both plaintext and search term ensured fair comparison between schemes by eliminating input-size variability as a performance factor. The test hardware's moderate clock speeds likely influenced the absolute performance numbers, suggesting that more powerful processors could significantly reduce total execution times while maintaining the same relative performance relationships between the schemes. This analysis provides valuable insights for practitioners considering implementation of these *k*-resilient cryptosystems in real-world applications where such fixed-length identity strings and common keyword searches are typical use cases.

# Conclusion

In this work, we have developed the *k*-resilient suite, a cross-platform desktop application using the Rust and Tauri framework to demonstrate and evaluate the performance of four *k*-resilient cryptosystems: KR-IBE, KR-IBI, KR-PEKS, and KR-PAEKS. By integrating these primitives into a unified and user-friendly platform, we provide a practical tool for testing, benchmarking, and showcasing the efficiency of *k*-resilient cryptographic schemes in real-world environments. Our performance analysis confirms that these primitives exhibit predictable and reasonable execution times under varying security parameters, supporting their feasibility for practical use. Looking ahead, this platform lays the groundwork for broader applications such as secure data sharing and searchable encryption. In particular, KR-IBI can serve as a resilient login mechanism, while KR-IBE enables secure communication, KR-PEKS and KR-PAEKS can further enhance privacy-preserving search over encrypted content. Future enhancements may include user experience refinements, integration with decentralised identity systems, and benchmarking against pairing-based schemes to highlight the efficiency of our pairing-free constructions. We also plan to conduct a performance analysis based on practical datasets, such as the Enron email dataset, to simulate realistic workloads as our immediate future work.

# Acknowledgements

This work was supported by the Ministry of Higher Education of Malaysia’s Fundamental Research Grant Scheme (FRGS/1/2023/ICT07/MMU/01/1).

# References

1. A. Shamir, “Identity-Based Cryptosystems and Signature Schemes,” in *Advances in Cryptology*, edited by G.R. Blakley and D. Chaum, (Springer Berlin Heidelberg, Berlin, Heidelberg, 1985), pp. 47–53.
2. D. Boneh, and M. Franklin, “Identity-Based Encryption from the Weil Pairing,” SIAM J. Comput. 32(3), 586–615 (2003).
3. S.-H. Heng and K. Kurosawa, "*k*-Resilient identity-based encryption in the standard model," in *Topics in Cryptology–CT-RSA 2004: The Cryptographers’ Track at the RSA Conference 2004, San Francisco, CA, USA, February 23–27, 2004, Proceedings*, edited by T. Okamoto (Springer, Berlin, Heidelberg, 2004), pp. 67–80.
4. J.-J. Chin and S.-H. Heng, "Security upgrade for a *k*-resilient identity-based identification scheme in the standard model," in *The 3rd International Conference on Cryptology and Computer Security 2012 (Cryptology 2012)* (2012).
5. D. Boneh, G. Di Crescenzo, R. Ostrovsky, and G. Persiano, “Public Key Encryption with Keyword Search,” in *Advances in Cryptology - EUROCRYPT 2004*, edited by C. Cachin and J.L. Camenisch, (Springer Berlin Heidelberg, Berlin, Heidelberg, 2004), pp. 506–522.
6. D. Khader, “Public Key Encryption with Keyword Search Based on *k*-Resilient IBE,” in *Computational Science and Its Applications - ICCSA 2006*, edited by M. Gavrilova, O. Gervasi, V. Kumar, C.J.K. Tan, D. Taniar, A. Laganá, Y. Mun, and H. Choo, (Springer Berlin Heidelberg, Berlin, Heidelberg, 2006), pp. 298–308.
7. W.-C. Yau, S.-H. Heng, S.-Y. Tan, B.-M, Goi, and R.C.‐W. Phan, “Efficient encryption with keyword search in mobile networks,” Security Comm Networks 5(12), 1412–1422 (2012).
8. K.-M. Chan, S.-H. Heng, W.-C. Yau, and S.-C. Tan, “Trapdoor Privacy in Public Key Encryption with Keyword Search: A Review,” IEEE Access 10, 21584–21598 (2022).
9. K.-M. Chan, S.-H. Heng, S.-Y. Tan, and S.-C. Tan, “*k*-Resilient Public Key Authenticated Encryption with Keyword Search,” in *Proceedings of the 21st International Conference on Security and Cryptography*, (SCITEPRESS - Science and Technology Publications, Dijon, France, 2024), pp. 381–388.