JSSE
JOURNAL OF SCIENCE & SCIENCE EDUCATION
Faculty of Science, Ubon Ratchathani University
Vol. 7 No. 2 (Jul. – Dec. 2024): 282 - 294
http://doi.org/10.14456/jsse.2024.22

Research Article in Science

# Strengthening Database Privacy: A Comprehensive Approach Using One Time Pad

**Alain Jean[1] and Tossaporn Alherbe[1,*]**

*[1]Department of Mathematics, Statistics and Computer, Faculty of Science, Ubon Ratchathani University, Ubon Ratchathani, Thailand*

**Email: tossaporn.c@ubu.ac.th*

Received <4 July 2024>; Revised <26 August 2024>; Accepted <28 August 2024>

## Abstract

Securing sensitive data stored in database environments is a critical challenge faced by organizations across various industries. Inadequate database security can lead to devastating data breaches, compromising the privacy and confidentiality of crucial information. This research focuses on the importance of robust database security measures and the role of the Gid Crypto tool in addressing these concerns. Gid Crypto, a stand-alone application, efficiently encrypts and decrypts data using private, public, and shared keys. The previous version stored encryption keys and contact names directly in the MongoDB database, posing a security risk if attackers managed to hack the password. This research introduces a new mechanism leveraging One Time Pad (OTP)-based encryption to enhance security. By implementing OTP, the tool ensures that contact lists, private keys, and secret keys are encoded before storage, making the data incomprehensible without the appropriate decryption keys, even if unauthorized access occurs. The enhanced Gid Crypto application maintains data integrity, confidentiality, and authenticity, effectively protecting sensitive information in databases. Comprehensive testing confirmed the tool's robustness, accuracy, and efficiency, establishing it as a practical and cost-effective option for database security.

**Keywords**: Encryption, Decryption, One Time Pad (OTP), Database security, Data encryption

บทความวิจัยทางวิทยาศาสตร์

# การเพิ่มความปลอดภัยให้ฐานข้อมูล: วิธีการของ One Time Pad

**อลัน จีน[1] และทศพร อเลิร์ป[1,*]**

*[1]ภาควิชาคณิตศาสตร์ สถิติ และคอมพิวเตอร์ คณะวิทยาศาสตร์ มหาวิทยาลัยอุบลราชธานี*

**Email: tossaporn.c@ubu.ac.th*

รับบทความ: <4 กรกฎคม 2567>; แก้ไขบทความ: <26 สิงหาคม 2567>; ยอมรับตีพิมพ์: <28 สิงหาคม 2567>

## บทคัดย่อ

การรักษาความปลอดภัยของข้อมูลที่มีความละเอียดอ่อนซึ่งถูกเก็บไว้ในฐานข้อมูลถือเป็นความท้าทายที่สำคัญที่องค์กรในหลากหลายอุตสาหกรรมต้องเผชิญ ถ้าฐานข้อมูลมีการรักษาความปลอดภัยที่ไม่เพียงพอ จะนำไปสู่การละเมิดข้อมูลหรือการรั่วไหลของข้อมูลที่รุนแรง ซึ่งส่งผลกระทบต่อความเป็นส่วนตัวโดยเฉพาะข้อมูลสำคัญที่เป็นความลับ งานวิจัยนี้มุ่งเน้นไปที่ความสำคัญของมาตราการรักษาความปลอดภัยบนฐานข้อมูล และบทบาทของการใช้เครื่องมือกิดคริปโทในการรักษาข้อมูลให้เป็นความลับ กิดคริปโทเป็นแอปพลิเคชันแบบสแตนอโลนที่สามารถทำการเข้ารหัสและถอดรหัสข้อมูลได้อย่างมีประสิทธิภาพโดยใช้กุญแจส่วนตัว กุญแจสาธารณะ และกุญแจร่วม ก่อนหน้านี้กิดคริปโทจะทำการเก็บกุญแจที่ใช้ในการเข้ารหัสและรายชื่อผู้ติดต่อกันลงในฐานข้อมูลมองโกลดีบีโดยตรง ซึ่งอาจก่อให้เกิดความเสี่ยงด้านความปลอดภัยหากผู้โจมตีระบบสามารถเข้าถึงรหัสผ่านได้ งานวิจัยนี้ได้นำเสนอวิธีการใหม่ที่ใช้ประโยชน์จากการนำวิธีการเข้ารหัสแบบวันไทม์แพด (โอทีพี) เข้ามาร่วม เพื่อยกระดับความปลอดภัยให้กับการใช้งานกิดคริปโท การใช้โอทีพีทำให้ข้อมูลรายชื่อผู้ติดต่อ กุญแจส่วนตัว และกุญแจลับ มีการเข้ารหัสก่อนนำไปจัดเก็บลงฐานข้อมูล ทำให้ข้อมูลไม่สามารถเข้าใจได้หากไม่มีกุญแจที่ใช้ในการถอดรหัสที่ถูกต้อง แม้ว่าจะมีการเข้าถึงข้อมูลเหล่านั้นโดยผู้ที่ไม่ได้รับอนุญาตก็ตาม แอปพลิเคชันกิดคริปโทที่ได้รับการปรับปรุงนี้สามารถรักษาความสมบูรณ์ของข้อมูล เก็บข้อมูลไว้เป็นความลับและรักษาความถูกต้องของข้อมูลเอาไว้ โดยสามารถปกป้องข้อมูลที่มีความละเอียดอ่อนในฐานข้อมูลได้อย่างมีประสิทธิภาพ จากการทดสอบการทำงานของกิดคริปโท ผลการทดสอบยืนยันถึงความทนทานต่อการพยายามโจมตี สามารถทำงานได้อย่างถูกต้องและมีประสิทธิภาพ ทำให้เป็นตัวเลือกในการนำไปใช้งานได้จริงและมีความคุ้มค่าในการนำไปรักษาความปลอดภัยให้กับฐานข้อมูล

**คำสำคัญ**: การเข้ารหัส การถอดรหัส วันไทม์แพด (โอทีพี) ความปลอดภัยฐานข้อมูล การเข้ารหัสข้อมูล

## Introduction

The rapid advancement of technology makes it easier for anyone to store, transmit, and access data. However, this convenience also brings significant risks, particularly regarding sensitive information that malicious individuals can intercept, alter, or steal. Encryption plays a critical role in protecting databases from both inside and outside attackers by converting sensitive data into a format that is unreadable without the proper decryption key. This ensures that even if an attacker gains unauthorized access, the data remains secure and unusable. For inside attackers, encryption prevents misuse of legitimate access, while for outside attackers, it serves as a formidable barrier against hacking attempts. By safeguarding data integrity and confidentiality, encryption helps maintain trust, compliance, and the overall security of the organization's information assets.

Several studies have been conducted to enhance the understanding and application of encryption and decryption techniques. A research work by Ibrahim *et. al* (2023) examined database security by employing ASCII code for data encryption in conjunction with their proprietary algorithms. Shingari and Mago (2024) investigated the protection of sensitive medical records sourced from Internet of Things (IoT) technologies. With the growth of cloud technology, numerous researchers have turned their attention toward cloud security. For instance, Fatima *et al.* (2022) suggested the implementation of the Advanced Encryption Standard (AES) for encrypting data within cloud-based applications and storage solutions. Similarly, Hammami, Obaidat and Yahia (2020) concentrated on safeguarding personal data during authentication processes in cloud environments.

Currently, various commercial and online cryptography services are utilized across diverse systems. For instance, VeraCrypt (Evkan *et al.*, 2020; KakaSoft, 2021) is a tool compatible with Linux, macOS, and Windows operating systems, capable of creating hidden drives for data encryption that remain invisible in the file system. However, its complexity can hinder effective use for some users, potentially affecting system performance. Miguel (2024) conducted a comprehensive review and evaluation of several popular encryption software options, including IPassword, Boxcryptor, NordLocker, Kruptos2 Professional, and CryptoForge. These tools typically require subscriptions. Among email encryption services, Cisco Secure Email Encryption Service (TrustRadius, 2024) offers an inexpensive plug-in, but it only supports Outlook and Microsoft 365. Proton Mail (James, 2024; Preveil, 2024; Rubenking, 2024) provide robust encryption services to secure email messages; however, their free versions offer limited features, with additional functionalities and unlimited storage available only through paid plans. Addressing these limitations, we have developed a tool named Gid Crypto (Jean and Alherbe, 2024), designed to encrypt and decrypt data in emails, text messages, and CSV files. This tool was designed to be user-friendly and does not require any additional subscription fees. Gid Crypto was also utilized by Ngaogate *et al.* (2024) to safeguard patient data at rest. The tool encrypts data before storing it in a database, with decryption occurring before entering the data classification process.

One Time Pad (OTP) encryption uses a random key that is only used once to encrypt plaintext. It ensures the security of sensitive information from unauthorized access. The OTP process generates a random key that is the same length as the plaintext. Each bit or character of the plaintext is then encrypted by combining it with the corresponding bit from the OTP key using modular addition. Besides being an uncrackable encryption method, OTP is cost-effective as it usually does not require additional hardware or software, making it a practical option for enhancing database security.

Certain research efforts have explored the use of OTPs to secure data, for instance, Budiman, Zarlis and Hafirzah (2021) combined the Rabin-*p* algorithm and OTP to protect data images. Their research result shows a conclusion that the larger the size of the images, the longer the time is used for encryption and

decryption. Indra and Nabila (2023) conducted a study on securing text messages by first using RSA to encrypt the plaintext, followed by using OTP with the same plaintext to maintain data authenticity. Kumari, Balaji and Lyengar (2023) used OTP for secure data in cloud computing and explained the OTP process.

In this research, we proposed a new security mechanism to enhance the efficiency of the Gid Crypto tool in securing data protection and privacy within database environments. By implementing OTPs, we strengthen the security of data by encoding contact lists, private keys, and secret keys before their direct storage in the database, as was observed in the previous Gid Crypto version. Consequently, even if unauthorized individuals gain access to the database, they will be unable to comprehend the stored information without possessing the appropriate decryption keys.

## Research Objectives

1. To encrypt data while storing and retrieving from the database.
2. To keep the data confidential so that those not involved cannot understand that information.
3. To secure data from inside and outside attackers.

## Research Method

In this section, we describe the current Gid Crypto system, detail the implementation of OTP, and explain how applying OTP can significantly enhance Gid Crypto's security. We also provide testing details to ensure the robustness of Gid Crypto’s encryption and decryption capabilities. The following information will explain these aspects comprehensively.

### Gid Crypto Application

Our development tool, Gid Crypto (Jean and Alherbe, 2024), is a stand-alone application designed to encrypt and decrypt data efficiently and rapidly using private, public and shared keys. The tool was developed using Diffie-Hellman, Mersenne Twister, and AES techniques to ensure secure data. Speed tests indicate that the encryption and decryption processes are performed within a reasonable time frame, and attempts to attack the system were unsuccessful in obtaining the keys. Figure 1 shows the current Gid Crypto encryption and decryption workflow where Bob and Alice represent a pair on the contact lists.

To establish secure communication, each communication pair must create a contact name. As shown in Figure 1, Bob created Alice as his contact name, while Alice did the same for Bob. Subsequently, Bob generates a private key and a public key. These keys are used to create a shared key for Bob. Bob then sends the public key and shares it over the internet with Alice, enabling her to generate her secret key.

Similarly, Bob’s secret key is generated using Alice’s public and shared key. The contact name, private key, and secret key are stored in the database on each side. During communication, Bob and Alice use their keys to encrypt and decrypt messages, ensuring their conversations are secure. In this way, the current version of Gid Crypto ensures that communication messages are protected and secure enough, as described and tested in our work (Jean and Alherbe, 2024). Even if eavesdroppers intercept their encrypted messages, they cannot read the content without the proper keys.

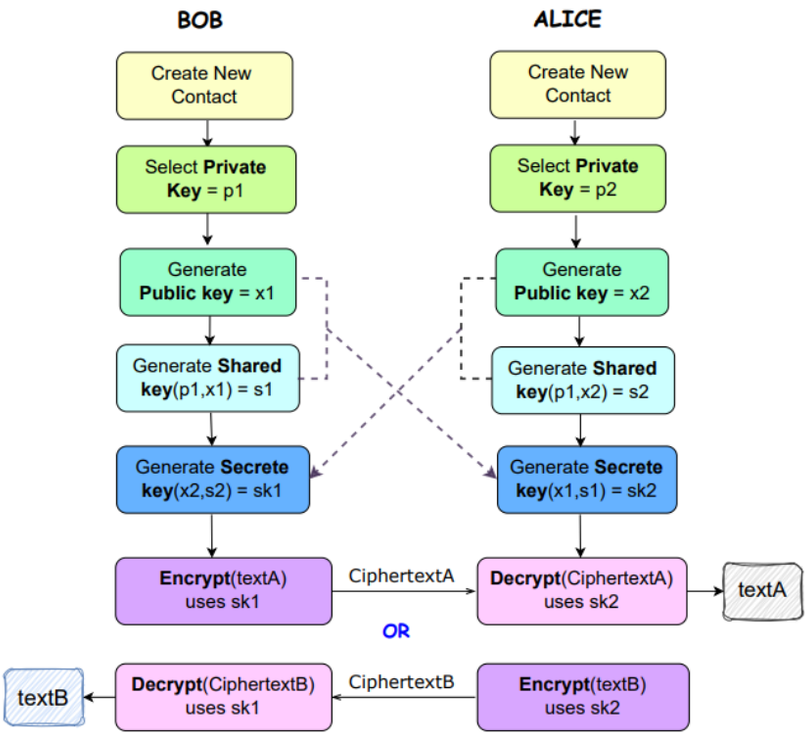



**Figure 1** The current Gid Crypto encryption and decryption workflow

To enhance Gid Crypto's efficiency, we have pinpointed specific areas for improvement. In the current version, we found that the private keys, secret keys, and contact names are stored directly in the MongoDB database. Despite Gid Crypto's implementation of password-based access control, if inside or outside attackers hack the password, they could access the database and view the stored encryption keys, enabling them to decrypt the protected data.

**Enhanced Gid Crypto Application**

In this research, we have enhanced the capabilities of Gid Crypto to encrypt the contact names, private keys, and secret keys before storing them in the database. The plain text will be encrypted using random sequence OTP keys and stored in the datastore. For decryption, the data will be retrieved from the datastore and then decrypted. This process ensures that if an unauthorized person were to hack into the database, they would be unable to discern the contact names or keys, thereby increasing the data's privacy and security. In doing that, we developed the application using Python, leveraging the PyQt5, PyMongo, and Cryptodome libraries, among others, and utilized MongoDB for the database.

We have utilized the OTP methods to assist in encrypting data for security enhancement. The OTP encryption and decryption technique are applied to both sides of the communication, as demonstrated in our example with Bob and Alice. Figure 2 illustrates the enhancement process of the Gid Cryptosystem, specifically on Bob's side.

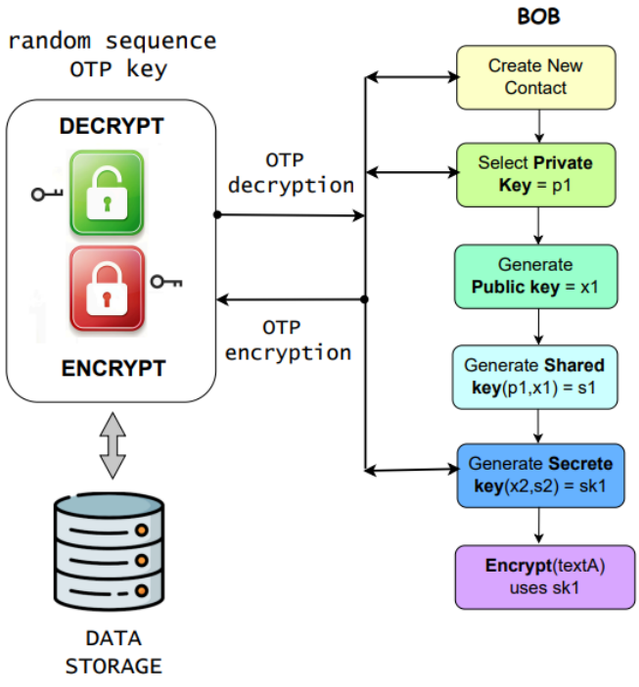



**Figure 2** The Gid Crypto Enhancement

**One Time Pad**

The OTP is an encryption technique that only uses a predefined secret key once. The key length is equal to the size of the message being encrypted, and we use this key in an algorithm to modify each character of the text that we are encrypting and store the resulting cipher in the database; when that data is needed, the algorithm, is used to decrypt the cipher to plaintext. Our proposed OTP-generating method is described as follows:

1. In Gid Crypto, contact names are generated as strings that may include spaces and underscore, while private and secret keys are solely numeric. However, there are no restrictions on the length of contact names, private keys, or secret keys. Users can make them as long as they want.

2. According to the condition of the contact name and the keys generated above, we would use [A-Z], space, [0-9], underscore (_), and [a-z] (a total of 64 alphabets) for encryption and decryption. Special characters such as @, #, and ! are excluded since they are not used in naming conventions and cannot be used to create public and private keys. The numerical value of each alphabet is shown in Figure 3.

3. The OTP random key (K) value is uniquely generated using our developed function Random_Sequence(). This key is the same length as the plaintext used to modify each text character. Even if the plain text characters are identical, the generated key will differ each time. This key is only used once to encrypt the plaintext and decrypt the ciphertext, ensuring unique encryption and decryption for each instance.

| Alphabet | A | B | C | ... | Y | Z | space | 0 | 1 | ... | 8 | 9 | _ | a | b | c | ... | y | z |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Numerical Value | 0 | 1 | 2 | ... | 24 | 25 | 26 | 27 | 28 | ... | 35 | 36 | 37 | 38 | 39 | 40 | ... | 62 | 63 |

**Figure 3** Numerical value of each alphabet

**Key Generation Function**

We developed a Random_Sequence() function to generate OTP keys as follows:

**For contact name**:

1. Specify the Plain Text: The length of the OTP key sequence will equal the length of the plain text.
2. Map Characters to Numerical Values: Each character in the plain text is **randomly mapped** to a numerical value ranging from 0 to 63, resulting in a sequence of integers making the OTP keys.

**For private and secret keys**:

1. Specify the Plain Numbers: The length of the OTP key sequence will equal the length of the plain numbers.
2. Map Numbers to Numerical Values: Each plain number is mapped to a **256-bit** length number from 0 to 9.

The length of the plaintext and plain numbers can be adjusted as needed. If the communication pair wishes to use a new set of OTP keys for the contact name, they can delete the current contact and regenerate a new contact using a new plain text to issue a new random sequence. This process is similar to updating the OTP keys for private and secret keys. The encryption and decryption processes are explained below:

**Algorithm for OTP Encryption**

Step 1: Identify the plaintext to encrypt (contact name, private key, or secret key).

Step 2: Convert each character, number, space, or underscore into a numerical value ranging from 0 to 63.

Step 3: Use the developed function Random_Sequence() to generate OTP keys of the same length as the plaintext.

Step 4: Add each numerical plaintext value to the corresponding OTP key.

Step 5: Perform modulo 64 on each resulting value from Step 4 to obtain the ciphertext.

**Algorithm for OTP Decryption**

Step 1: Subtract each ciphertext numerical value from its corresponding OTP key value.

Step 2: Perform modulo 64 on each resulting value from Step 1 to obtain the plaintext.

Figure 4 illustrates the encryption and decryption process using OTP for a contact name “Alain Jean.” In this example, the contact name has two “a” and two “n.” The developed algorithm uses a different key for each occurrence of the same character. For instance, the OTP key for the first “a” is 44, while the OTP key for the second “a” is 53. Consequently, the ciphertexts for “a” are different.

Figure 5 illustrates the OTP encryption and decryption process for a private key. It demonstrates that, with the developed algorithm, the ciphertext can be identical even if the keys are different. For instance, the numbers 2, 3, and 9 all have the same cipher representation as “Q.” This means that if an unauthorized person accesses the database and sees the "Q" values, they might incorrectly assume they originate from the same initial plaintext. Therefore, attempts to decrypt the data, considering it comes from the same source, will be unsuccessful.

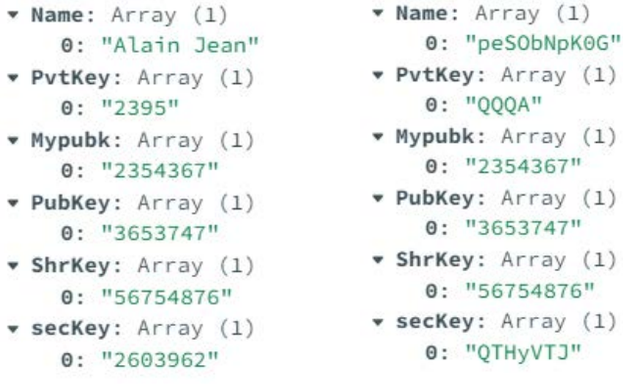
| **ENCRYPTION** | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Plaintext** | **A** | **l** | **a** | **i** | **n** | | **J** | **e** | **a** | **n** |
| Numerical value of Plaintext | 0 | 49 | 38 | 46 | 51 | 26 | 9 | 42 | 38 | 51 |
| OTP Key (K) | 53 | 57 | 44 | 32 | 52 | 51 | 44 | 32 | 53 | 19 |
| pk = Numerical value of Plaintext + K | 53 | 106 | 82 | 78 | 103 | 77 | 53 | 74 | 91 | 70 |
| pk MOD 64 | 53 | 42 | 18 | 14 | 39 | 13 | 53 | 10 | 27 | 6 |
| Cipher text | p | e | S | O | b | N | p | K | 0 | G |
| **DECRYPTION** | | | | | | | | | | |
| Cipher text | p | e | S | O | b | N | p | K | 0 | G |
| Numerical value of Ciphertext | 53 | 42 | 18 | 14 | 39 | 13 | 53 | 10 | 27 | 6 |
| OTP Key (K) | 53 | 57 | 44 | 32 | 52 | 51 | 44 | 32 | 53 | 19 |
| ck = Numerical value of Ciphertext - K | 0 | -15 | -26 | -18 | -13 | -38 | 9 | -22 | -26 | -13 |
| ck MOD 64 | 0 | 49 | 38 | 46 | 51 | 26 | 9 | 42 | 38 | 51 |
| **Plain text** | **A** | **l** | **a** | **i** | **n** | | **J** | **e** | **a** | **n** |

**Figure 4** OTP Encryption and Decryption for a Contact Name

| **ENCRYPTION** | | | | |
|---|---|---|---|---|
| **Private key (Plaintext)** | **2** | **3** | **9** | **5** |
| Numerical value of Plaintext | 29 | 30 | 36 | 32 |
| OTP Key (K) | 51 | 50 | 44 | 32 |
| pk = Numerical value of Plaintext + K | 80 | 80 | 80 | 64 |
| pk MOD 64 | 16 | 16 | 16 | 0 |
| Cipher text | Q | Q | Q | A |
| **DECRYPTION** | | | | |
| Cipher text | Q | Q | Q | A |
| Numerical value of Ciphertext | 16 | 16 | 16 | 0 |
| OTP Key (K) | 51 | 50 | 44 | 32 |
| ck = Numerical value of Ciphertext - K | -35 | -34 | -28 | -32 |
| ck MOD 64 | 29 | 30 | 36 | 32 |
| **Private key (Plaintext)** | **2** | **3** | **9** | **5** |

**Figure 5** OTP Encryption and Decryption for a Private Key

Figure 6 illustrates the difference between the data stored in the database using the previous Gid Crypto method and the data stored using the new Gid Crypto version with OTP encryption. The figure shows that the contact name, private key, and secret key are encrypted before being stored in the database.




(a) Previous version (b) Enhanced version with OTP

**Figure 6** Comparison of Gid Crypto Datastore: (a) Previous version (b) Enhanced version with OTP

**Evaluation**

Detailed tests were conducted to ensure the robustness of Gid Crypto's encryption and decryption capabilities, focusing on three primary areas: accuracy, security, and performance. Samples of data used in the experimental evaluation are shown in Table 1.

**Table 1** Samples of data used in the experimental evaluation

| File types | Data examples |
|---|---|
| Clear text data | As Moore's Law marches on, the Internet of Things continues to gain traction, and personal privacy is an above the fold news story, the importance of cryptography on the modern Internet continues to increase. While once the realm of mathematicians and hardcore computer scientists, cryptography has gone mainstream ...<br><br>(1946 Characters, 302 Words, 24 Lines) |
| CSV file | dex,Customer Id,First Name,Last Name,Company,City,Country,Phone 1,Phone 2,Email,Subscription Date,Website<br>1,EB54EF1154C3A78,Heather,Callahan,Mosley-David,Lake Jeffborough,Norway,043-797-5229,915.112.1727,urangel@espinoza-francis.net,2020-08-26,http://www.escobar.org/<br>2,10dAcafEBbA5FcA,Kristina,Ferrell,"Horn, Shepard and Watson",Aaronville,Andorra,932-062-1802,(209)172-7124x3651,xreese@hall-donovan.com,2020-04-27,https://tyler-pugh.info/<br>3,67DAB15Ebe4BE4a,Briana,Andersen,Irwin-Oneal,East Jordan,Nepal,8352752061,(567)135-1918,haleybraun@blevins-sexton.com,2022-03-22,https://www.mack-bell.net/ ...<br><br>(1699947 Characters, 119670 Words, 19998 Lines) |
| Programming | '------------------------------------------------------------<br>' Date Developer Comments |<br>' ---------- ------------------- ----------------------------|<br>' 02/25/2010 Jack Original code |<br>'------------------------------------------------------------|<br>' rms ( Room Management Services) application/web service | |<br>'------------------------------------------------------------<br>Imports MySql.Data.MySqlClient<br>Imports System.Text<br>Imports System.Math<br>Imports System.IO ...<br><br>(7334 Characters, 548 Words, 267 Lines) |

The specifics of each test are explained as follows.

**1. Accuracy tests** verify the correctness of encryption and decryption of various data types, including private keys, contact names, and secret keys, to ensure that the process preserves data integrity and accuracy without any loss or alteration. Table 2 shows the results of the encryption and decryption accuracy testing.

**Table 2** Encryption and Decryption Accuracy

| Test Case | Description | Expected Outcome | Actual Outcome | Pass/Fail |
|---|---|---|---|---|
| Encrypt Private Keys | Encrypting private key with OTP random key | Encrypted cipher modified with a mix of letters and numbers | Encrypted cipher modified with a mix of letters and numbers | Pass |
| Decrypt Private Keys | Decrypting private key with OTP random key | Decrypted cipher to plaintext matches the originals | Decrypted cipher to plaintext matches the originals | Pass |
| Encrypt Contact Names | Encrypting contact names with OTP random key | Encrypted names modified in a cipher with a mix of letters and numbers | Encrypted names modified in a cipher with a mix of letters and numbers | Pass |
| Decrypt Contact Names | Decrypting contact names with OTP random key | The decrypted name was restored to the original plaintext name | The decrypted name was restored to the original plaintext name | Pass |
| Encrypt Secret Keys | Encrypting Secret key with OTP random key | Encrypted cipher modified with a mix of letters and numbers | Encrypted cipher modified with a mix of letters and numbers | Pass |
| Decrypt Secret Keys | Decrypting secret key with OTP random key | Decrypted cipher to plaintext matches the originals | Decrypted cipher to plaintext matches the originals | Pass |

**2. Security tests** evaluated the system's capability to protect data against unauthorized access and tampering. These tests focused on generating true random digits to fill the pad, with a length of 256 bits sufficient to cover the most extended key or name. We also employed multiple pads, ensuring that if an attacker managed to decrypt a name using a brute-force attack, they could not use the same sequence for the secret or private key. The objective was to determine if the encryption would hold long enough for an attack to be detected or for the cipher's utility to be completed. The results demonstrated that the encryption was robust enough to withstand a brute-force attack for over a week. Simultaneous attacks on the three ciphers (name, private, and secret key) were unsuccessful, as none of the number sequences matched the keys or names. We also conducted the Frequency Analysis test to determine if any patterns were discernible in the name cipher. No discernible patterns were found, leading to the termination of the frequency test. Table 3 provides detailed results of the security tests.

**Table 3** Security tests

| Test Case | Description | Expected Outcome | Actual Outcome | Pass/Fail |
|---|---|---|---|---|
| Integrity Check | Attempt at decrypting cipher without the proper key in the reasonable time frame | Unable to decrypt name or keys in 24 hours | Not even 1 character was decrypted in 24 hours using a “brute-force” attack method with a word dictionary | Pass |

**3. Performance tests** focused on evaluating the system's efficiency and scalability. This included measuring the system performance during the encryption and decryption processes and determining whether these operations affected the system. Table 4 compares CPU and memory usage between the previous and enhanced version of Gid Crypto with OTP encryption, while Table 5 details performance tests. The summary results of the tests are depicted in Table 6.

**Table 4** Comparison of performance test examples between the previous and enhanced versions of Gid Crypto with OTP encryption, based on the samples provided in Table 1

| Performance Test Category | Clear text | | CSV | | Programming | |
|---|---|---|---|---|---|---|
| | Previous Version | Enhanced version with OTP | Previous Version | Enhanced version with OTP | Previous Version | Enhanced version with OTP |
| CPU (%) | 8 | 8 | 17 | 17 | 9 | 9 |
| Memory (%) | 36 | 36 | 36 | 36 | 36 | 36 |

The performance tests consistently showed efficient resource usage, with no significant changes observed in CPU and memory usage.

**Table 5** Performance tests

| Test Case | Description | Expected Outcome | Actual Outcome | Pass/Fail |
|---|---|---|---|---|
| Resource Utilization for encryption | Measure the load on the system to encrypt data | Efficient and no notable load on CPU, resources, or time constraint | No discernible load on the CPU or resources or time constraint | Pass |
| Resource Utilization for decryption | Measure the load on the system to decrypt data | Efficient and no notable load on CPU, resources, or time constraint | No discernible load on the CPU or resources or time constraint | Pass |

**Table 6** Summary of Results

| Test Category | Total Tests | Passed | Failed |
|---|---|---|---|
| Encryption and Decryption Accuracy | 30 | 30 | 0 |
| Security Tests | 10 | 10 | 0 |
| Performance Tests | 16 | 16 | 0 |
| Overall | 56 | 56 | 0 |

As shown in the "Test Category" column, we conducted various tests. For the security test, we used brute-force attacks, while for the contact name encryption and decryption accuracy test, we employed both brute-force attacks and Frequency Analysis. All tests passed, demonstrating that our application provides a high degree of privacy. Additionally, the performance tests consistently indicated very reasonable resource usage, with no noticeable changes in CPU and memory usage.

## Results and Discussion

Our proposed methodology significantly enhances the security of data storage and retrieval in databases, effectively addressing the research objectives in the following:

1. Enhanced data security**:** We effectively developed a method to encrypt data while storing and retrieving from the database. Compared to the work of Ibrahim *et al.* (2023), encrypting text-type data resulted in text-type ciphertext, and encrypting number-type data produced number-type ciphertext. Their approach addresses a significant security risk since an attacker could infer that numeric ciphertext likely represents numeric data, such as age or salary. In contrast, our improved method makes it impossible to determine whether the original data was text or numeric, enhancing security and protecting the data more effectively.

2. Dynamic encryption with OTP**:** Utilizing OTP for data encryption means that each instance of encryption produces a different outcome, even for the exact text. This is because a unique key is generated for each encryption instance, making it significantly harder for attackers to decrypt the data through repeated attempts. Our work keeps the data confidential, so those not involved need help understanding that information.

3. Mitigation of outsider and insider threats: Our approach ensures that even if a malicious insider or outsider can entirely bypass and access our database, they can only understand the encrypted data with the proper decryption keys. This robust encryption mechanism mitigates the risk posed by unauthorized access.

4. Efficiency in encryption and decryption: The comparison between the previous version of Gid Crypto and the enhanced version with OTP encryption shows that the difference in performance is negligible. Consequently, this improved tool can be used to encrypt data efficiently while maintaining high security.

## Conclusion and Future Work

This research successfully enhances the Gid Crypto tool by incorporating OTP-based encryption, significantly improving the security of sensitive data stored in databases. The tests confirmed that OTP encryption effectively protects contact names, private keys, and secret keys, ensuring that even if unauthorized access occurs, the data remains incomprehensible without the correct decryption keys. This enhancement addresses the critical issue of securely storing sensitive data, thereby ensuring the confidentiality, integrity, and authenticity of data stored in databases, which is a valuable addition to the tool's initial version.

Despite the significant advancements achieved, there are still areas for further improvement. Future research should focus on expanding the accessibility and ease of integration of OTP encryption, making it more straightforward to incorporate as a plug-in or add-on to other solutions. Additionally, exploring further cryptographic techniques will be essential to enhance security even more. Moreover, integrating real-time monitoring and automated threat detection systems could provide proactive security measures, enabling identifying and mitigating potential threats before they cause harm.

## References

Budiman, M. A., Zarlis, M. and Hafirzah (2021). Implement hybrid cryptosystem using Rabin-*p* algorithm and One Time Pad to secure images. **Journal of Physics: Conference Series**, 1898(1), 012037.

Evkan, H., Lahr, N., Niederhagen, R., Petri, R., Poller, A., Roskosch, P. and Troger, M. (2020). **Security Evaluation of VeraCrypt**. Bonn: Fraunhofer Institute for Secure Information Technology.

Fatima, S., Rehman, T., Fatima, M. Khan S. and Ali, M. A. (2022). Comparative analysis of Aes and Rsa Algorithms for Data Security in Cloud Computing. **Journal of Engineering Proceedings**, 20(1), 14.

Hammami, H., Obaidat, M. S. and Yahia, S. B. (2020). An Enhanced Lightweight Authentication Scheme for Secure Access to Cloud Data. In **Proceedings of the 17th International Conference on Security and Cryptography (SECRYPT 2020)** (pp. 102-109).

Ibrahim, S., Zengin, A., Hizal, S., Akhter, A. F. M. S. and Altunkaya, C. (2023). A novel data encryption algorithm to ensure database security. **Acta Infologica**, 7(1), 1-16.

Indra, Z. and Nabila, R. C. (2023). Implementing the RSA Algorithm and the One Time Pad Algorithm for Text Message Security. **Formosa Journal of Science and Technology**, 2(1), 379-388.

James, T. (2024). Proton Mail Review 2024 – How is it in Reality?. Retrieved 5 May 2024, from **Travelsecurely** https://travelsecurely.com/protonmail-review/

Jean, A. and Alherbe, T. (2024). Gid Crypto: Application for End-to-End Encrypt and Decrypt E-mail and Data. **Journal of Scientific and Technological Reports**, 27(2), 90-102.

KakaSoft (2021). VeraCrypt Review 2021: Price, Feature and Alternatives. Retrieved 12 March 2024, from **Kakasoft** https://www.kakasoft.com/review/veracrypt-review/

Kumari, A. B., Balaji, H., and Lyengar, N. Ch. S. N. (2023). One Time Pad Encryption Technique in Cryptography. **International Journal of Computational Learning & Intelligence**, 2(1), 1-7.

Miguel, P. G. (2024). 18 Best Encryption Software. Reviewed For 2024. Retrieved 28 May 2024, from **Thectoblub** https://thectoclub.com/tools/best-encryption-software/

Ngaogate, W., Jean, A., Wattanataweekul, R., Janngam, K. and Alherbe, T. (2024). Hybrid Machine Learning Algorithm with Fixed Point Technique for Medical Data Classification Problems Incorporating Data Cryptography. **Thai Journal of Mathematics**, 22(2), 295-310.

Preveil (2024). Simple, Encrypted Email and File Collaboration. Retrieved 10 June 2024, from **Preveil** https://www.preveil.com/

Rubenking, N. J. (2024). The Best Email Encryption Services for 2024. Retrieved 3 June 2024, from **Pcmag** https://www.pcmag.com/picks/the-best-email-encryption-services

Shingari, N. and Mago, B. (2024). The Importance of Data Encryption in Ensuring the Confidentiality and Security of Financial Records of Medical Health. In **2024 IEEE International Conference on Interdisciplinary Approaches in Technology and Management for Social Innovation (IATMSI)** (pp. 1-6). Gwalior: IEEE.

TrustRadius (2024). Make confident technology decision. Retrieved 7 March 2024, from **TrustRadius** https://www.trustradius.com/