

A Desktop Application of JAB Code for Data Security and Authentication

Rishabh Kushawah¹, Ekta Bohara²

¹ Assistant Professor, Department of Computer Science & Engineering, Dr. A.P.J Abdul Kalam Government Institute of Technology, Tanakpur, Uttarakhand, India

² Scientific Officer/ Engineer-SB, National Informatics Centre

Abstract - Use of JAB (Just Another Barcode) code, a two-dimensional representation of barcodes, for generating encrypted question papers. Traditionally, barcodes have been used for product identification, but JAB code can embed various types of information such as text, audio, video, web URLs, and more. The proposed method involves encrypting the question paper data using the AES encryption algorithm and encoding it into a JAB code. To decrypt the data, the JAB code is scanned.

Key Words: Barcode; JAB code; Encryption; Decryption; AES Algorithm.

1. INTRODUCTION

In the current digital age, ensuring data security and authentication is imperative for both individuals and businesses. Barcode technology has emerged as a reliable and efficient method for encoding and decoding information, with the potential to address these concerns. Among various 2D barcodes, JAB Code stands out as a polychrome barcode that offers higher data density and greater design flexibility, as exemplified in Figure 1. Therefore, JAB Code represents a viable option for safeguarding data and enabling authentication.

The proposed application aims to provide users with the ability to encrypt and decrypt sensitive data using JAB Code, ensuring secure transmission and storage, with access restricted to authorized individuals. This capability makes the application relevant for deployment in various domains, including finance, healthcare, and government, where safeguarding data and authentication are critical. Moreover, developing this application would augment the current understanding of JAB Code and its potential applications, paving the way for further exploration and innovation in this field.

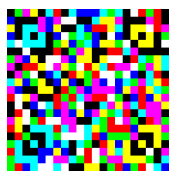


Fig. 1. JAB Code

• CURRENT APPLICATION FOR JAB CODE

JAB Code is a relatively new technology, and its current applications are still somewhat limited. However, the company that developed the technology has identified several potential applications in various industries. Here are some of the current applications of JAB Code:

Retail: JAB Code can be used for product labelling and inventory management in the retail industry. The ability to encode more information in a smaller space permits the inclusion of more descriptive product information on the label, which can enhance the consumer experience.

Logistics: JAB Code can be used for tracking and authentication of shipments in the logistics industry. The ability to include authentication mechanisms, such as digital signatures, makes JAB Code a promising option for ensuring secure transmission and storage of shipment data.

Advertising: JAB Code can be used for interactive advertising campaigns, where customers can scan the code with their smartphones to access additional information or promotional offers.

Education: JAB Code can be used for exam papers and educational materials in the education industry. The ability to encode and decode sensitive data using JAB Code can improve data security and prevent cheating.

Overall, JAB Code's current uses are limited, but its distinctive characteristics and adaptability make it a viable alternative for a range of industries. As technology continues to advance and acquire wider adoption, we can anticipate additional JAB Code applications in the future.

2. JAB CODE FOR SECURE PAPER GENERATION

A. Primer

The Application is divided into two parts i.e., JAB Code generator and JAB Code reader. Using the application, the user will be able to generate a JAB Code using two option Encrypt data and Simple.

The proposed application comprises of two modules - a JAB code generator and a JAB code reader. While generating a JAB code, the "Text" and "File" options provided by the JAB code generator are the two available choices. Users can use the "Text" option to input any desired text concealed within the JAB code by typing their selection. Users can select "File" to provide the location they want to conceal within the JAB code. If user want to encrypt the JAB code, they must input a password that will function as the encryption key.

The generated JAB code will be displayed on the screen and saved as an image file at a specific destination folder/directory within the user's PC. The JAB code reader module can be used to scan the generated JAB code by using the integrated camera of the laptop. If the JAB code is encrypted, the user will need to Decrypt the data with password for decryption. Once the correct password is entered, the text hidden inside the JAB code will be displayed. If the JAB code is not encrypted, the hidden text or be directly displayed upon scanning.

The encryption process used in this application employs the AES algorithm to encrypt the text with a password that the user

uses to decode the text. This application has various potential uses, but in this context, it can be used to securely transmit sets of question papers from one university to another. The JAB codes with the URL of the website hosting the question papers can be circulated to different faculties of different universities, who can then view the question papers by scanning the JAB code using the proposed application.

B. Methodology

Advanced Encryption Standard is the chosen algorithm for encrypting and decrypting the data stored in JAB codes in this research paper (AES). Although various encryption methods such as Data Encryption Standard, Triple Data Encryption Standard (Triple DES), and Blowfish are available, the AES technique is chosen the best suitable for our paper shown in Table 2. The steps of this method are detailed in detail below.

1) Import the necessary libraries:

The application imports the required Python libraries for AES encryption and decryption: `Crypto.Cipher.AES` for the main AES functionality, and `Crypto.Util.Padding` for padding and unpadding the data.

2) Implement the encryption function:

The encrypt function takes two parameters: text (the plaintext to be encrypted) and key (the symmetric encryption key).

Steps within the function:

- Convert the key and text to byte strings using the UTF-8 encoding.
- Create a new AES cipher object in CBC mode with the given key.
- Generate an initialization vector (IV) for the CBC mode.
- Pad the plaintext to the AES block size using the pad function.
- Encrypt the padded plaintext using the cipher object.
- Concatenate the IV with the ciphertext.
- Convert the resulting byte string to a hexadecimal string and return it.

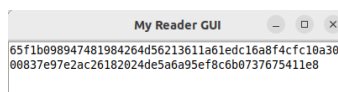


Figure 2. Encrypted data

3) Implement the decryption function:

The decrypt function takes two parameters: ciphertext (the encrypted data) and key (the symmetric encryption key).

Steps within the function:

- Convert the key to a byte string using the UTF-8 encoding.

- Convert the hexadecimal ciphertext string to a byte string.
- Extract the IV from the ciphertext.
- Remove the IV from the ciphertext.
- Create a new AES cipher object in CBC mode with the given key and IV.
- Decrypt the ciphertext using the cipher object.
- Unpad the decrypted plaintext using the unpad function.
- Decode the plaintext byte string to a UTF-8 string and return it.

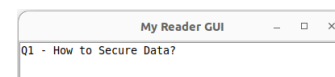


Figure 3 Decrypted data

4) Integrate the encryption and decryption functions within the applications:

The first application (for creating JAB Code images) uses the encrypt function to encrypt the input data before encoding it into a JAB Code image. The second application (for decoding JAB Code images) uses the decrypt function to decrypt the data retrieved from the JAB Code image, if necessary.

In summary, the provided applications implement the AES algorithm using Python's Crypto library to ensure data security and authentication. By integrating the encryption and decryption functions, the applications can work with encrypted data while encoding and decoding JAB Code images.

C. Pseudo Code for AES algorithm

- Import necessary libraries (`Crypto.Cipher.AES`, `Crypto.Util.Padding`)
- Function encrypt (plaintext, key):
 - Convert key and plaintext to byte strings (UTF-8 encoding)
 - Create a new AES cipher object (CBC mode) with the given key
 - Generate an initialization vector (IV) for CBC mode
 - Pad the plaintext to the AES block size
 - Encrypt the padded plaintext using the cipher object
 - Concatenate the IV with the ciphertext
 - Convert the resulting byte string to a hexadecimal string
 - Return the hexadecimal ciphertext

3) Function decrypt (hexadecimal_ciphertext, key):

- Convert key to a byte string (UTF-8 encoding)
- Convert the hexadecimal ciphertext to a byte string
- Extract the IV from the byte string

- d) Remove the IV from the ciphertext
 - e) Create a new AES cipher object (CBC mode) with the given key and IV
 - f) Decrypt the ciphertext using the cipher object
 - g) Unpad the decrypted plaintext
 - h) Decode the plaintext byte string to a UTF-8 string
 - i) Return the plaintext
- 4) Integrate the encryption and decryption functions within the applications:
- a) In the first application (JAB Code image creation):
 - i. Check if encryption is enabled and key is valid
 - ii. If yes, encrypt the input data using the encrypt function
 - iii. Encode the encrypted data into a JAB Code image
 - b) In the second application (JAB Code image decoding):
 - i. Decode the JAB Code image to retrieve the data
 - ii. Check if the data is encrypted
 - iii. If yes, decrypt the data using the decrypt function

D. Pseudo Code for JAB code generation

Step 1: Input data to be stored in JAB Code.

Step 2: Input the output file name and select the desired JAB Code configuration (U Shape, Vertical, or Horizontal).

Step 3: If the encryption option is enabled, input the encryption key and encrypt the data using the AES algorithm.

Step 4: Create an argument list (args) for the JAB Code generation command (background process).

Step 5: Generate the encoded JAB Code using the argument list (args) and the JAB Code 'writer' command.

Step 6: Display the result (Success or Fail) and the command output.

E. Pseudo Code for Reading of JAB code

Step 1: Create a graphical user interface (GUI) with a text widget to display the extracted data.

Step 2: Add a button to select a JAB Code image file using a file dialog.

Step 3: When the file is selected, read the JAB Code image using the 'reader' command from the JAB Code library.

Step 4: Display the extracted data in the text widget.

Step 5: If the extracted data appears to be encrypted (hexadecimal string), show a decryption button and an entry field to input the decryption key.

Step 6: If the user wants to decrypt the data, get the decryption key from the entry field and decrypt the data using the AES decryption algorithm.

Step 7: Replace the encrypted data in the text widget with the decrypted data.

• Analysis of QR for Secured question paper generation

A. Functional Analysis

This application can be used for various purposes such as secure transmission of sensitive documents, secure communication between organizations, and data protection in various industries. The application can be employed to secure exam papers, employee credentials, confidential reports, and other sensitive data. With the advancement of technologies, this application can be adapted to run on mobile devices, further increasing its utility. Several other applications exist to which this JAB Code may be a revolutionary replacement for existing solutions.

B. Risk Analysis

Due to the fact that risks are inherent to every application, it is necessary to examine the risks that can arise in many aspects. Risk consists of two components: the probability that the risk will occur and the repercussions if the risk does occur. Risks are hard to spot, so it increases the use of risk analysis to work effectively on mitigation plans. For analyzing the risks, it is necessary to identify the possible risks and then think of the possible mitigation strategies. Risk analysis helps in managing the application effectively.

3. CONCLUSIONS

We may hereby conclude that with the use of this JAB Code-based application, organizations can significantly enhance their data security and authentication processes. The application enables secure transmission of sensitive information, reducing risks associated with traditional methods of data exchange. By employing advanced encryption and data representation technologies, the JAB Code offers a superior alternative to existing solutions.

The JAB Codes can be customized according to the organization's needs, allowing for flexible integration into various systems and processes. The secure JAB Codes will help organizations communicate information securely, serving the dual purpose of efficient data representation and secure data exchange.

4. FUTURE WORK

The versatility of this application goes beyond just secure data transmission. But also for various purposes such as authentication, verification, and access control. This versatility makes it a one-stop solution for organizations looking to implement a comprehensive, secure, and efficient information exchange system.

By adopting this JAB Code-based application, organizations can benefit from a cost-effective, secure, and

adaptable method of data transmission and authentication, paving the way for more efficient and secure operations in the digital age.

Risk	Probability	Consequence	Mitigation Strategy
Weak encryption key	Medium	Unauthorized access to sensitive data	Enforce strong key generation policies and key management
Data tampering	Low	Corrupted or altered data	Implement data integrity checks and secure communication channels
Vulnerabilities in the application	Medium	Exploitation by malicious actors	Regular security audits and updates, fix vulnerabilities promptly
Loss of encryption key	Medium	Inability to decrypt and access data	Implement secure key storage and recovery mechanisms
Unauthorized access	Low	Leakage of sensitive data	Implement access control and authentication measures

Table 1. Risk analysis

Algorithm	Key Length (bits)	Keys Searched (per second)	Block Size (bits)	Type	Security Level	Complexity
DES	56	1 billion	64	Symmetric	Low	Complex
Triple DES	168	1×10^{23}	64	Symmetric	Low	Complex
AES	128-256	1×10^{23}	128	Symmetric	High	Complex
RSA	Variable	Variable	Less than or equal to $\log_2(n)$	Asymmetric	High	Simple
JAB Code (AES)	128-256	1×10^{23}	N/A	Symmetric	High	Complex

Table 2. Comparative study

REFERENCES

- [1] Berchtold, W., Liu, H., Steinebach, M., Klein, D., Senger, T., & Thenece, N. (2020). JAB Code-A Versatile Polychrome 2D Barcode. *Electronic Imaging*, 2020(3), 207-1.
- [2] Winter, C., Berchtold, W., & Hollenbeck, J. N. (2019, September). Securing physical documents with digital signatures. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)* (pp. 1-6). IEEE.
- [3] Yfantis, Vasileios & Kalagiakos, Panagiotis & Kouloumperi, Chrysanthi & Karampelas, Panagiotis. (2012). Quick response codes in E-learning. *2012 International Conference on Education and e-Learning Innovations, ICEELI 2012*.10.1109/ICEELI.2012.6360566.
- [4] M. F. Tretinjak, "The implementation of QR codes in the educational process," *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, Opatija, Croatia, 2015, pp. 833-835, doi: 10.1109/MIPRO.2015.7160387.
- [5] M.Filipovic Tretinjak, "The Implementation of QR codes in the Educational Process", in *2015 38th International Convention on Information and Communication Technology, Electronics and Microelectronics, MIPRO, Opatija, 2015*, pp. 833-835.
- [6] Somdip Dey and Asoke Nath, "Confidential Encrypted Data hiding and Retrieval Using QR Authentication System", in *2013 International Conference on Communication Systems and Network Technologies*, Gwalior, 2013, pp. 512-517.
- [7] Katharina Krombholz, Peter Frihwirt, Thomas Rieder, Ioannis Kapsalis, Johanna Ullrich and Edgar Weippl, "QR code Security-How Secure and Usable Apps Can Protect Users Against Malicious QR Codes", in *2015 10th international Conference on Availability, Reliability and Security, ARES, Toulouse, 2015*, pp.230-237
- [8] Lin, P. Y., & Chen, Y. H. (2017). High payload secret hiding technology for QR codes. *EURASIP Journal on Image and Video Processing*, 2017(1), 1-8.
- [9] <https://github.com/jabcode/jabcode>