Grenze International Journal of Engineering and Technology, June Issue

Innovative use of Noise Images for File Encryption and Honeywords within a Cloud-based Environment

E.Rajkumar[1], Muthaiah V[2], Pranav S[3] and Praveen Kumar A[4]
[1]Assistant Professor, Department of Cyber Security, SRM Valliammai Engineering College
Email: raju.becs39@gmail.com
[2-4]UG Scholar, Department of Cyber Security, SRM Valliammai Engineering College
Email: muthaiah55555@gmail.com, pranavawm@gmail.com, pk2851764@gmail.com

***Abstract*— In today's digital landscape, securing sensitive data is paramount, particularly in web-based applications where encryption techniques play a vital role. This paper introduces a novel approach to enhancing data security through a combination of noise image conversion and honeyword concepts. The proposed method focuses on encrypting AES keys using noise image conversion, thereby adding an additional layer of protection to the encryption process. Furthermore, honeywords are employed to generate fake AES keys, adding complexity and confusion for potential attackers. In the event of a compromise, an alert mechanism is triggered, promptly notifying administrators of potential breaches. This approach not only strengthens data encryption but also provides proactive measures against unauthorized access, ultimately bolstering overall system security.**

***Index Terms*— Encryption, Advanced Encryption Standard, Noise image conversion, Honeywords, Unauthorized access, Data breaches**

I. INTRODUCTION

It is critical to safeguard sensitive data from unwanted access in the constantly changing field of cybersecurity. Furthermore, the integration of honeywords introduces an additional layer of deception and complexity to the encryption process. Honeywords are false cryptographic keys that resemble genuine keys, but their compromise triggers alerts, signaling potential unauthorized access attempts. By deploying honeywords alongside genuine AES keys, our approach not only deters attackers but also provides administrators with early detection capabilities, enabling swift response to security incidents.The technical aspects of creating noisy image conversions and honeywords and discuss how they could be applied to improve data security in web applications. We also offer a framework that explains how to actually implement these techniques, including steps for encrypting AES keys and generating honeywords. Using empirical analysis and case studies, we demonstrate how well our method supports data encryption and thwarts possible threats. All things considered, our method offers a comprehensive approach to enhancing data security in online environments, empowering companies to safeguard confidential information and reduce the dangers associated with unauthorized access and data breaches.

II. RELATED WORKS

[1] This research presents a novel encryption technique using two 2D modification models, offering enhanced control over data values and enabling encryption of multiple photos with the same key. [2] This research introduces

Grenze ID: 01.GIJET.10.2.225

a high-capacity image hiding technique using the Improved Product Code over Z4 (Z4MPC) encoder, achieving an embedding rate of 1.78bpp and an efficiency of [3] It combines wavelet transform and grey difference features for improved block matching, achieving greater embedding rates with low distortion in LSB layers and resistance to high-dimensional steganography attacks. [4] The CSNTSteg model enhances steganographic capacity without compromising invisibility, using RGB coding and character spacing in normalization and pre-embedding phases. [5] This technique addresses challenges in picture hiding by enhancing visual security, restoration quality, and antidetection performance for large, spatially-featured secret images. [6] This paper introduces Two privacy-preserving biometric identification protocols cater to different needs: one balances security and efficiency, while the other prioritizes efficiency with strong security.

III. Material Implementation

A. AES Encryption

There are several steps involved in using the AES (Advanced Encryption Standard) to encrypt a file using a programming language like Python. Import the necessary libraries first (cryptography, for instance). After that, RAM is filled with the contents of the file that has to be encrypted. A secret key is created for AES encryption, and its length changes depending on the AES mode that is chosen (e.g., AES-128, AES-192, AES-256). An initialization vector (IV) is also generated to ensure that the same plaintext does not encrypt to the same ciphertext. Padding is added if needed to ensure that the plaintext is a multiple of the block size (e.g., 128 bits for AES). This process ensures that the original file's contents are securely encrypted and that they can only be accessed with the correctkey.

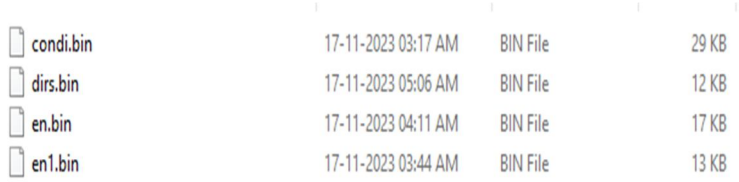

Fig 1 – Generating Adversarial Image

B. Conversion of Key to Image

The AES key is converted into a noise image by translating its binary data into pixel values in an image format.E ach byte in the key might represent one of 256 possible grayscale values.By reshaping the crucial data to match t he intended image proportions, the integrity of thedata is maintained.The corresponding byte of the AES key det ermines the intensity of each pixel in the raw noise image created by this process.Depending on the length of the key and the intended image size, additional processing or resizing may be required to obtain a visually acceptabl e noise image.

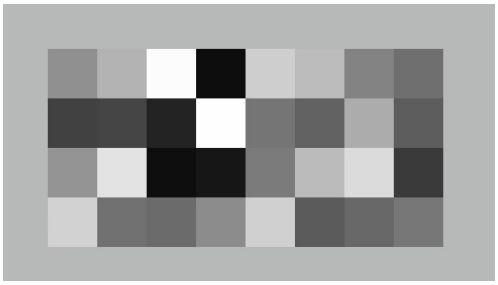

Fig 2- Conversion of key to image.

C. Fake Key Creation

To further boost security, a bogus key can be created in addition to the original AES key. This fake key must be the same size as the genuine one in order to maintain consistency. The fake key is converted into a noise image in the same way as the real key. After generation, the noise images corresponding to the fake and actual keys are stored in adatabase. The purpose of the fake key and related noise image is to act as a decoy, adding another layer of complexity and obfuscation for potential attackers. An unauthorized party trying to access the

database and extract the noise image in order to identify the AES key would be faced with several noise images, making it difficult for them to figure out which one is the real key.

Fig3-Fake keys generated as same as of the original key

D. Authentication

Using a fake key, similar to honeywords, strengthens the authentication process even further. Similar to how honeywords are fake passwords used to detect unauthorized access attempts, the bogus key serves as a decoy to identify illicit attempts to access the encrypted material. During the authentication procedure, both the genuine and fake keys are shown as potential authentication tokens. Both keys are meant to be accepted by the authentication system, just as honeywords are regarded as legitimate passwords in honeyword-based authentication systems. Any attempt to access fake or dummy data by using the fake key to decrypt the data would fail. The actual key would correctly decrypt the real data.If an unauthorized user manages to get hold of the fake key and tries to use it for decryption thereby tricking them into accessing false information the system will be notified of a possible security breach.

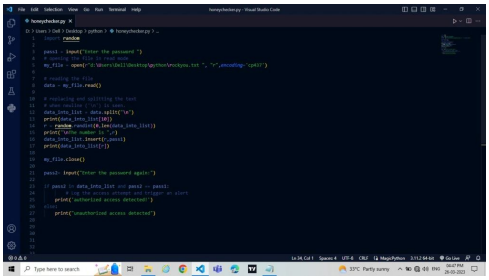

Fig 4-Honeychecker which is used to check for the fake passwords

F. Alerting the Admin

As soon as an intrusion attempt is discovered, as when someone uses the fake key to obtain illegal access, an alert is sent to the system administrator. The alert system is designed to promptly notify the administrator via email notifications, SMS messaging, and interaction with centralized logging and monitoring systems, among other channels.Notable details about the intrusion attempt are typically included in the warning, including the time of the incident, the kind of access attempt (for example, decryption using a fake key), and any further information discovered during the authentication process. With the use of this information, the administrator may quickly assess the incident's seriousness and take the required action to reduce the risk.

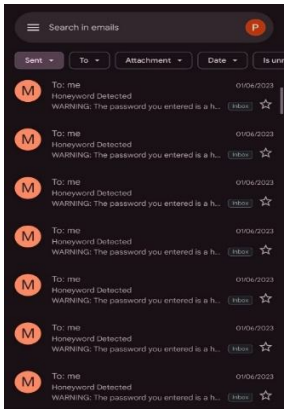

Fig 5- The output for defense model

IV. Conclusion & Future Work

In conclusion, incorporating a fake key—a type of honeyword—into an AES encryption scheme provides an additional layer of security by deceiving potential attackers. The system may detect and alert administrators to any attempt to obtain unauthorized access by producing noise pictures for both actual and false keys and keeping them in a database. This increases the security of encrypted data and lowers the possibility of security breaches. Future projects could benefit from the integration of threat intelligence feeds, the implementation of more complex alerting mechanisms, the investigation of additional authentication techniques, ongoing system log monitoring and analysis, performance and scalability optimization, and compliance with relevant regulatory standards.These considerations enable the AES encryption system to successfully safeguard sensitive data while maintaining the integrity and confidentiality of information assets, and to adjust to emerging security risks.

References

[1] Singh, K. N., & Singh, A. K. (2022). Towards integrating image encryption with compression: A survey. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 18(3), 1-21.

[2] Tiwari, P. K., Kannan, K., Veeraiah, D., Ranjan, N., Singh, J., Alshammri, G. H., & Halifa, A. (2022). Security Protection Mechanism in Cloud Computing Authorization Model Using Machine Learning Techniques. Wireless Communications and Mobile Computing, 2022.

[3] Vedaraj, M., & Ezhumalai, P. (2022). A Secure IoT-Cloud Based Healthcare System for Disease Classification Using Neural Network. Computer Systems Science & Engineering, 41(1).

[4] Al-Khasawneh, M. A., Uddin, I., Shah, S. A. A., Khasawneh, A. M., Abualigah, L., & Mahmoud, M. (2022). An improved chaotic image encryption algorithm using Hadoop-based MapReduce framework for massive remote sensed images in parallel IoT applications. Cluster Computing, 1-15.

[5] Gupta, I., Singh, A. K., Lee, C. N., & Buyya, R. (2022). Secure data storage and sharing techniques for data protection in cloud environments: A systematic review, analysis, and future directions. IEEE Access.

[6] Abd-El-Atty, B., ElAffendi, M., & El-Latif, A. A. A. (2023). A novel image cryptosystem using Gray code, quantum walks, and Henon map for cloud applications. Complex & Intelligent Systems, 9(1), 609-624.

[7] Suganya, M., & Sasipraba, T. (2023). Stochastic Gradient Descent long short-term memory based secure encryption algorithm for cloud data storage and retrieval in cloud computing environment. Journal of Cloud Computing, 12(1), 1-17.

[8] Adee, R., & Mouratidis, H. (2022). A dynamic four-step data security model for data in cloud computing based on cryptography and steganography. Sensors, 22(3), 1109.

[9] Yin, H. (2022). Public security video image detection system construction platform in cloud computing environment. Computational Intelligence and Neuroscience, 2022.

[10] Karthikeyan, A. N., Yuvaraj, N., Gowsic, K., Islam, M. J., & Ramasamy, M. (2023, June). Innovative Image Privacy Preservation Applied to Data Perturbation in Cloud. In 2023 International Conference on Applied Intelligence and Sustainable Computing (ICAISC) (pp. 1-5). IEEE.