HONEY2FISH - A HYBRID ENCRYPTION APPROACH FOR IMPROVED PASSWORD AND MESSAGE SECURITY

Rohit Verma
(rohit.verma@ncirl.ie)
School of Computing
National College of Ireland

CONTENTS

INTRODUCTION

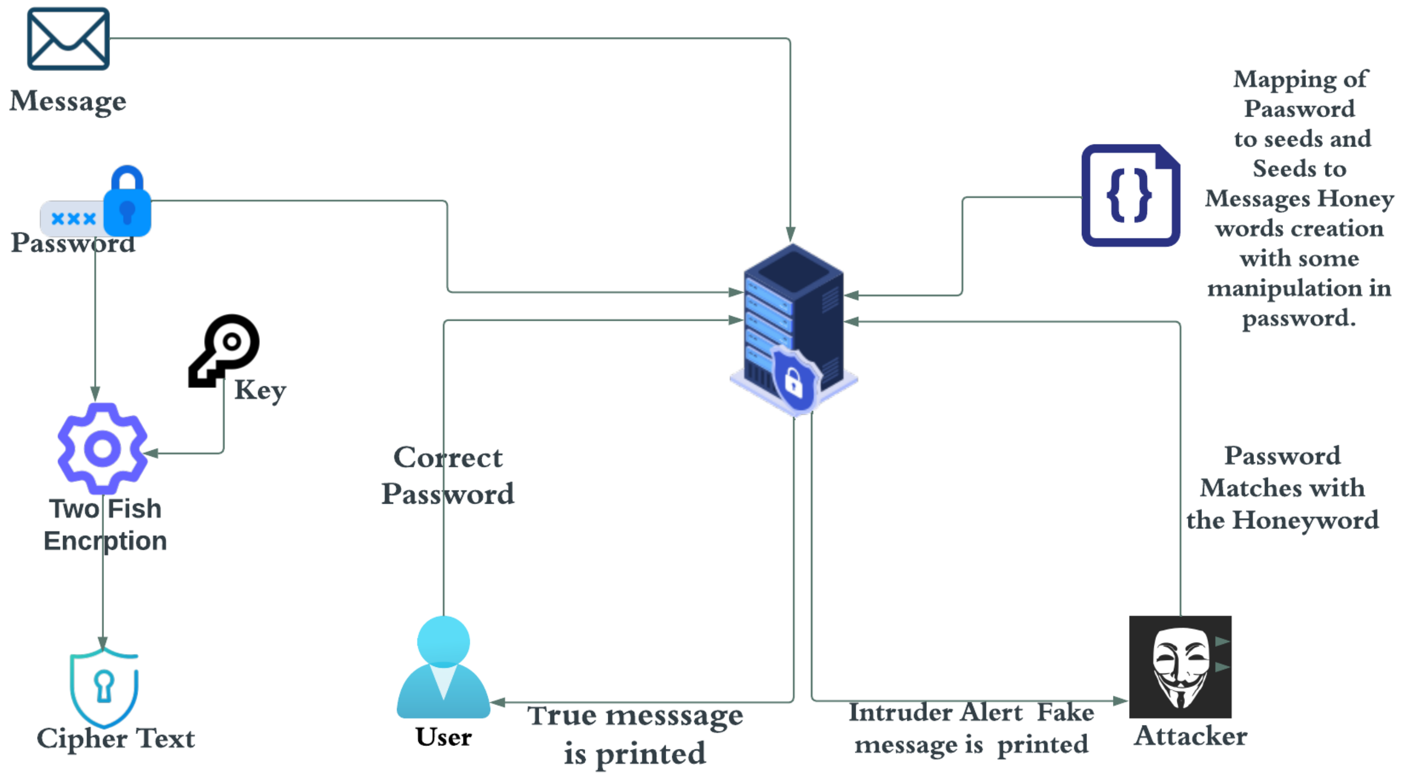
- Safeguarding user credentials' privacy, validity, and security has become imperative, especially passwords
- Password Based Encryption - to protect confidential information
- **Low entropy** in passwords -> easily vulnerable to brute force attacks and social engineering
- Honey Encryption (HE) - secondary layer of protection to enhance system security
- Honey2Fish - a novel low-complexity hybrid encryption approach to enhances the security of passwords and messages.

HONEY2FISH

- The problem:

  Address the challenges in password-based encryption (PBE) systems, especially the vulnerability to brute-force attacks due to weak user-generated passwords.
- Solution:
  - **Honey2Fish, a hybrid encryption approach combining Honey Encryption (HE) and Twofish to enhance password and message security.**
  - Honey2Fish approach:
    - Combines Honey encryption with TwoFish encryption
      - Step 1: Apply Honey Encryption (HE): **(Decoy Generation and Confusion Factor)**
        - Generates decoy plaintexts (honeywords) for every incorrect key, misleading attackers.
        - Protects against brute-force attacks by making it hard to distinguish the correct password.
      - Step 2: Apply Twofish Algorithm: (**Robust Encryption and High Performance**)
        - A symmetric key block cipher known for its robustness and efficiency.
        - Provides high security with 128-bit block size and variable key sizes (128, 192, 256 bits).
      - Step 3: Store Both honeywords and encrypted passwords/messages.
  - Enhances performance
  - Provides improved security
  - Maintains low system complexity

HONEY2FISH ALGORITHM



HONEY2FISH ENCRYPTION ALGORITHM

Input: UserName, Password, Message
Output: Encrypted Password and Message

Procedure HONEY ENCRYPTION:

- **Apply Honey Encryption Algorithm:**
  - Initiates the process by generating fake yet plausible plaintexts (honeywords) for incorrect keys to confuse attackers.
- **Generate Random Seed Value:**
  - Ensures unpredictable mapping of passwords to honeywords and messages.
- **Map Password-Seeds and Seeds-Messages:**
  - **Password-Seeds Mapping:** Each password gets a unique seed.
  - **Seeds-Messages Mapping:** Each seed is mapped to a specific message to ensure realistic decoys.
- **Create Honeywords using Digit Tweaking and Tailing:**
  - **Digit Tweaking:** Modify digits in passwords to create realistic variants.
  - **Tailing:** Add random but plausible tails to passwords for more variations.
- **Apply TwoFish Encryption (with required padding):**
  - Encrypt the original password and message using TwoFish, ensuring proper block size with padding.
- **Generate Encrypted Password and Message:**
  - Produces the final encrypted password and message, including honeywords, to store securely.

HONEY2FISH DECRYPTION ALGORITHM

Input: Username, Password
Output: Retrieved Message (Genuine or Decoy)

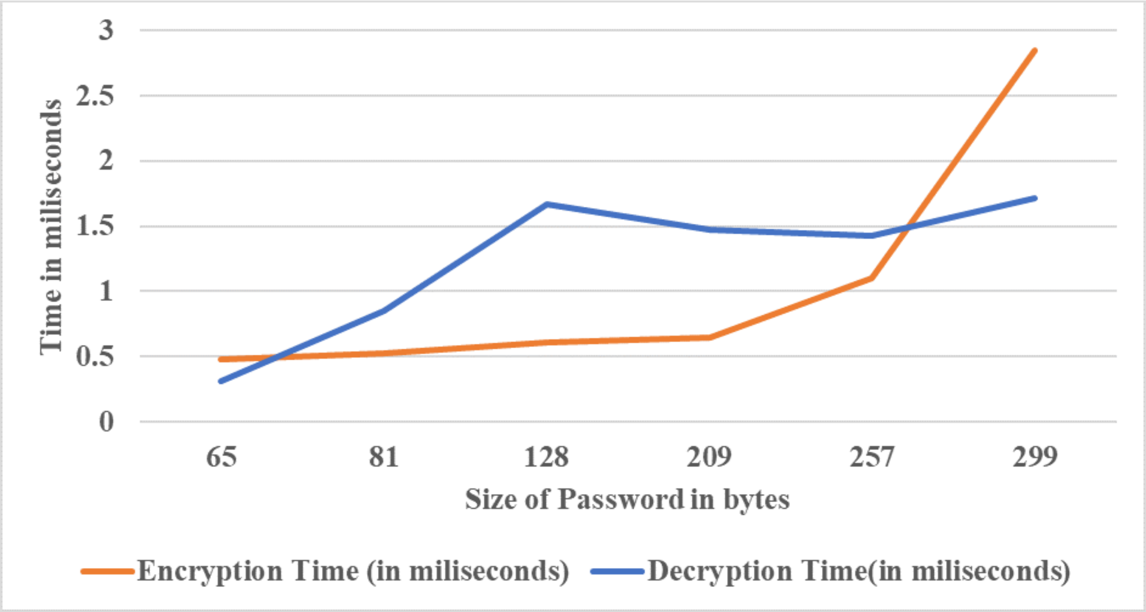
Procedure DECRYPTION:

- **Input Handling:**
  - User supplies their username and password.
- **TwoFish Decryption:**
  - **Decrypt Encrypted Data:**
    - Use TwoFish algorithm to decrypt the encrypted password and message.
  - **Padding Removal:**
    - Remove any padding added during encryption to restore original data format.
- **Honeyword Verification:**
  - **Check Against Honeywords:**
    - Verify if the decrypted password matches the stored original password or any honeywords.
  - **Scenarios Based on Password Matching:**
    - **Case 1: Valid Password:**
      - Retrieve and display the genuine message.
    - **Case 2: Honeyword Match:**
      - Trigger alert and display a decoy message.
    - **Case 3: Invalid Password:**
      - No valid decryption; deny access.
- **Message Retrieval:**
  - Retrieve the corresponding message based on the verification outcome.

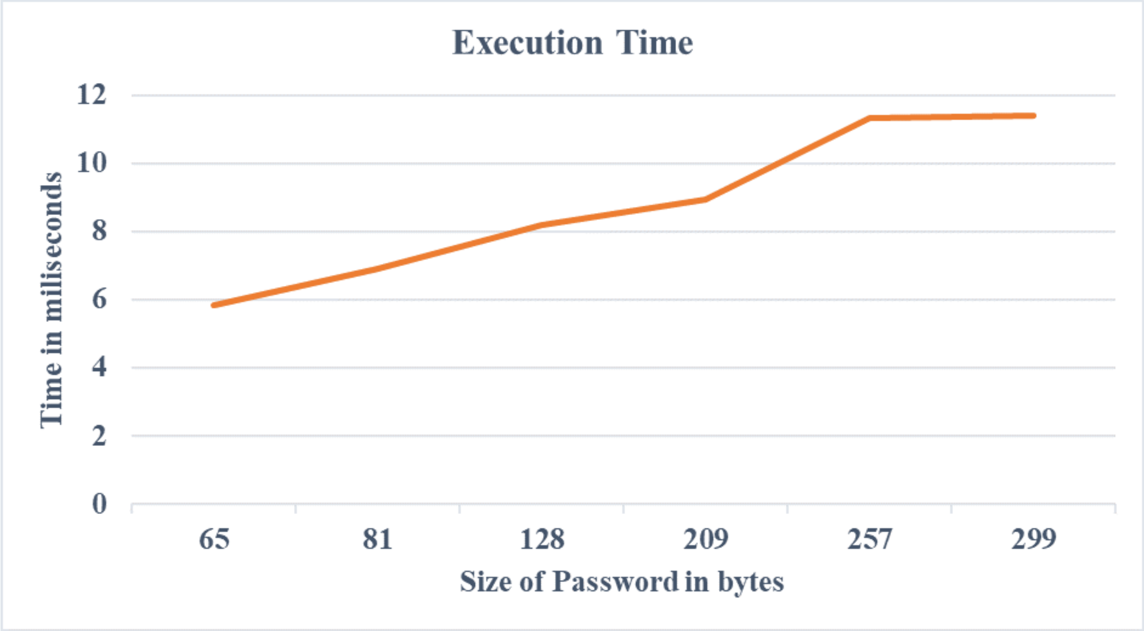
PERFORMANCE ANALYSIS

1. Encryption and decryption time vs. password size
2. Total execution time vs. size of password
3. Encryption time vs. password size for AES and Twofish
4. Decryption time vs. password size for AES and Twofish
5. Avalanche Effect - HONEY2FISH

Encryption and decryption time vs. password size



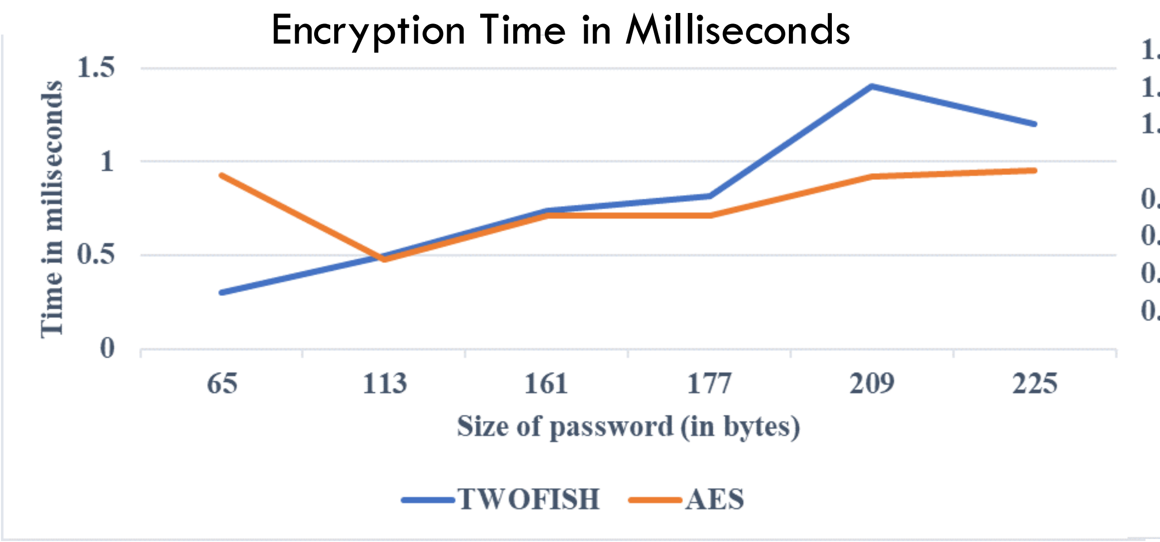
Total execution time vs. size of password

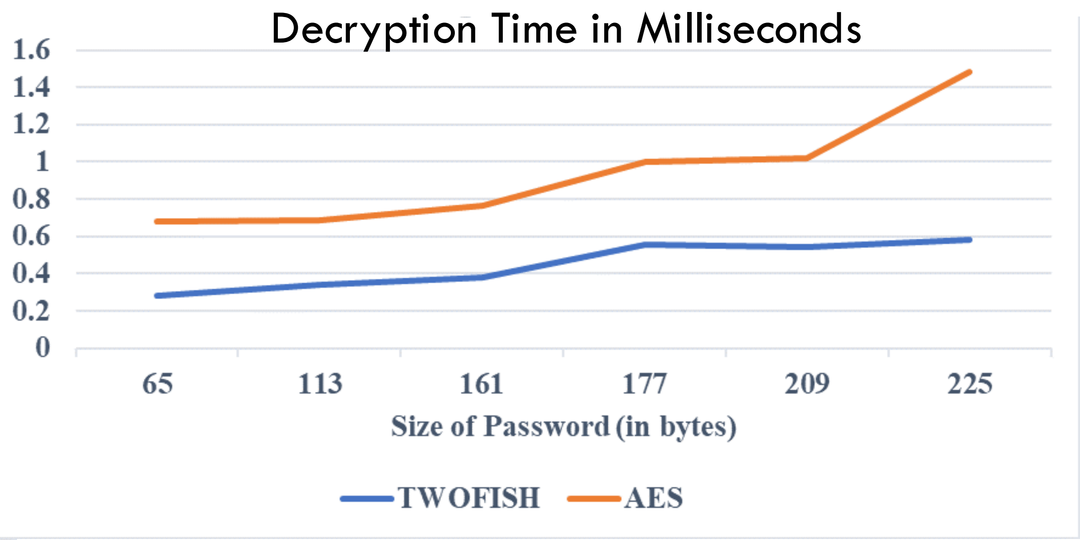


Throughput with Varied Password Length

Size (in bytes)	Size (in MB)	Execution Time (in seconds)	Throughput
65	0.000065	0.0058213	0.011165891
81	0.000081	0.0068987	0.011741343
128	0.000128	0.00817	0.015667075
209	0.000209	0.0089488	0.023355087
257	0.000257	0.0113062	0.022730891
299	0.000299	0.0113884	0.026254786

Encryption & Decryption time vs. password size for AES and Twofish





Avalanche Effect - Honey2Fish

Password	Size in bytes	Changed Plaintext	Avalanche Effect
Tiger	54	Tiges	**48.54%**
Mangoapple	59	Mangoappld	**50.58%**
spiderman@batman	65	spiderman@batmao	**53.09%**
jdhfbodvbnkdjvbdjkwd	69	jdhfbodvbnkdjvbdjkwe	**53.61%**

RELATED WORK

- Juels and Rivest [5] introduced Honey Encryption, which used fake passwords called honeywords to enhance password security.
- Chakraborty et al. [7] proposed a Honey Circular model that utilized a circular list to store passwords.
- Pagar et al. [8] explored various honeyword creation methods, such as chaffing with tough nuts and tweaking.
- Erguler's use of chaffing with a password [4], which manipulates the password to create honeywords.
- Shen et al. [9], Chakraborty et al. [7], Mohammed et al. [10], and Bangera et al. [11], have utilized the concept of DTE (Distribution Transforming Encoder) in combination with other security measures.
- Tan et al. [15] utilized honey encryption, grid-based passwords, and OTP techniques.
- Almuhanna et al. [12] proposed a method in which credentials are stored using hashes, and if the provided passcode is valid, a specific honey word is generated and saved for future use.

RELATED WORK

- Hybrid Models:
  - Moe et al. [20] added salting and hashing to honey encryption to enhance security and time complexity.
  - Burgess et al. [21] utilised RSA to improve security and provide better encryption and brute-force techniques.
  - Sahu et al. [22] combined honey encryption with Blowfish and AES to protect the system from brute-force attacks.
  - Dibas et al. [6] demonstrated that Blowfish's performance improves when encrypting small, sparsely packed information.

Method	Key Features	Advantages	Limitations	Comparison with Honey2Fish
Honey Encryption (HE)	Generates decoy plaintexts (honeywords) for incorrect keys	Misleads attackers with plausible decoys, enhancing security against brute-force attacks	Storage overhead, increased computational time	Honey2Fish combines HE with Twofish, providing enhanced security with low complexity
Honey Circular Model	Uses a circular list to store passwords	Reduces storage requirements	Algorithm for generating honeywords needs improvement	Honey2Fish offers a more sophisticated honeyword creation and robust encryption
Chaffing with Tough Nuts	Manipulates passwords to create honeywords	Provides additional layer of security	Limited effectiveness, storage and management challenges	Honey2Fish addresses these challenges with efficient Twofish encryption
Dynamic Keypad Scheme	Uses dynamic keypads to delay password extraction	Difficult for attackers to quickly crack passwords	Increased user complexity, potential usability issues	Honey2Fish balances security and usability effectively
Hybrid Model (DNA Cryptosystem)	Combines honey encoding with DNA cryptosystem for key generation	Enhanced resilience against brute-force attacks, innovative DNA coding strategy	High computational requirements, complex implementation	Honey2Fish offers similar security with lower computational overhead
Grid-Based Honey Encryption	Uses grid-based passwords and OTP techniques for mHealth applications	Strong defense against brute-force and man-in-the-middle attacks	Complex user interactions, potential for false positives	Honey2Fish simplifies user interactions while maintaining high security
AES Encryption	Widely used symmetric key encryption algorithm	High security, well-established, fast for large data sets	Less efficient for smaller data sets, susceptible to side-channel attacks	Honey2Fish uses Twofish, which is more efficient for varied data sizes
Blowfish Encryption	Symmetric key block cipher with flexible key size	Efficient for small, sparsely packed information	Less efficient for larger data sets	Honey2Fish with Twofish provides better performance for larger and varied data sizes
Twofish Encryption	Symmetric key block cipher with 128-bit block size, variable key sizes	High security, resistant to various cryptographic attacks, efficient	May require more processing power than simpler algorithms	Honey2Fish leverages Twofish for robust and efficient encryption
RC5 with Honey Encryption	Combines RC5 encryption with honey encryption for key sharing	Increased security through hybrid approach	Higher computational complexity, potential performance issues	Honey2Fish achieves similar security with streamlined performance
Blowfish and AES Hybrid	Combines Blowfish and AES to protect against brute-force attacks	Enhanced security through multiple encryption layers	Increased computational requirements, complexity in implementation	Honey2Fish provides similar security with a simpler, more efficient hybrid approach

CONCLUSIONS & FUTURE WORKS

- Honey2Fish approach to enhance security and protect against leaks and abuse.
- Used honey encryption for messages and Twofish encryption for passwords.
- Suitability for real-life applications, particularly those using password-based authentication.
- Performance evaluation of the approach, including its suitability for varied lengths of passwords and real-world applications.
- Future plans to evaluate and expand the application of Honey2Fish, including in credit card OTP security, fraud detection websites, IoT, Edge, and cloud environments.

REFERENCES

[1] M. Jakobsson, M. Jakobsson, and D. Liu, “Your password is your new pin,” Mobile Authentication: Problems and Solutions, pp. 25–36, 2013.

[2] M. Rambabu and N. Ramana, “Improved Honey Cipher Structure Using Least Significant Bit Techniques,” in ICDSMLA 2019 (A. Kumar, M. Paprzycki, and V. K. Gunjan, eds.), Lecture Notes in Electrical Engineering, (Singapore), pp. 671–677, Springer, 2020.

[3] P. Dixit, A. K. Gupta, M. C. Trivedi, and V. K. Yadav, “Traditional and hybrid encryption techniques: a survey,” in Networking Communication and Data Knowledge Engineering: Volume 2, pp. 239–248, Springer, 2018.

[4] I. Erguler, “Achieving Flatness: Selecting the Honeywords from Existing User Passwords,” IEEE Transactions on Dependable and Secure Computing, vol. 13, pp. 284–295, Mar. 2016. Conference Name: IEEE Transactions on Dependable and Secure Computing.

[5] A. Juels and R. L. Rivest, “Honeywords: making password-cracking detectable,” in Proceedings of the 2013 ACM SIGSAC conference on Computer & communications security, CCS ’13, (New York, NY, USA), pp. 145–160, Association for Computing Machinery, Nov. 2013.

[6] H. Dibas and K. E. Sabri, “A comprehensive performance empirical study of the symmetric algorithms:AES, 3DES, Blowfish and Twofish,” in 2021 International Conference on Information Technology (ICIT), pp. 344–349, July 2021.

[7] N. Chakraborty and S. Mondal, “On designing a modified-UI based honeyword generation approach for overcoming the existing limitations,” Computers & Security, vol. 66, pp. 155–168, May 2017.

[8] V. R. Pagar and R. G. Pise, “Strengthening password security through honeyword and Honeyencryption technique,” in 2017 International Conference on Trends in Electronics and Informatics (ICEI), pp. 827– 831, May 2017.

[9] C. Shen, T. Yu, H. Xu, G. Yang, and X. Guan, “User practice in password security: An empirical study of real-life passwords in the wild,” Computers & Security, vol. 61, pp. 130–141, Aug. 2016.

[10] S. mohammed, S. KURNAZ, and A. H. Mohammed, “Secure Pin Authentication in Java Smart Card Using Honey Encryption,” in 2020 International Congress on Human-Computer Interaction, Optimization and Robotic Applications (HORA), pp. 1–4, June 2020.

[11] S. Bangera, P. Billava, and S. Naik, “A Hybrid Encryption Approach for Secured Authentication and Enhancement in Confidentiality of Data,” in 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), pp. 781–784, Mar. 2020.

[12] A. AlMuhanna, A. AlFaadhel, and A. Ara, “Enhanced System for Securing Password Manager Using Honey Encryption,” in 2022 Fifth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU), pp. 150–154, Mar. 2022.

[13] N. N. Khin and T. Win, “A novel hybrid encryption method based on honey encryption and advanced dna encoding scheme in key generation,” Journal of Computer and Communications, vol. 10, no. 9, pp. 22–36, 2022.

[14] T. Nirmalraj and J. Jebathangam, “A Password Secure Mechanism using Reformation-based Honey Encryption and Decryption,” in 2022 International Conference on Inventive Computation Technologies (ICICT), pp. 214–220, July 2022. ISSN: 2767-7788.

REFERENCES

[15] S.-F. Tan, K.-M. C. Lo, Y.-B. Leau, G.-C. Chung, and F. Ahmedy, "Securing mHealth Applications with Grid-Based Honey Encryption," in 2021 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET), pp. 1–5, Sept. 2021.

[16] J. Jaeger, T. Ristenpart, and Q. Tang, "Honey Encryption Beyond Message Recovery Security," in Advances in Cryptology – EUROCRYPT 2016 (M. Fischlin and J.-S. Coron, eds.), Lecture Notes in Computer Science, (Berlin, Heidelberg), pp. 758–788, Springer, 2016.

[17] H.-C. Chen, H. Wijayanto, C.-H. Chang, F.-Y. Leu, and K. Yim, "Secure Mobile Instant Messaging key exchanging protocol with One-Time-Pad substitution transposition cryptosystem," in 2016 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), pp. 980–

984, Apr. 2016.

[18] S. Sahu, "PROVIDING INFORMATION SECURITY USING HONEY ENCRYPTION," Advances in Mathematics: Scientific Journal, vol. 9, pp. 8249–8258, Sept. 2020.

[19] A. E. Omolara, A. Jantan, O. I. Abiodun, and H. E. Poston, "A Novel Approach for the Adaptation of Honey Encryption to Support Natural Language Message," Hong Kong, 2018.

[20] K. S. M. Moe and T. Win, "Enhanced Honey Encryption Algorithm for Increasing Message Space against Brute Force Attack," in 2018 15th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), pp. 86–89, July 2018.

[21] J. Burgess, "Honey encryption review," Apr. 2017. [Online; accessed 19-Feb-2023].

[22] R. Sahu and M. S. Ansari, "Securing Messages from Brute Force Attack by Combined Approach of Honey Encryption and Blowfish," 2017.

[23] P. B. Shamini, E. Dhivya, S. Jayasree, and M. P. Lakshmi, "Detection and avoidance of attacker using honey words in purchase portal," in 2017 Third International Conference on Science Technology Engineering & Management (ICONSTEM), pp. 260–263, Mar. 2017.

[24] K. Vivek Raj, H. Ankitha, N. G. Ankitha, and L. S. Kanthi Hegde, "Honey Encryption based Hybrid Cryptographic Algorithm:A Fusion Ensuring Enhanced Security," in 2020 5th International Conference on Communication and Electronics Systems (ICCES), pp. 490–494, June 2020.

[25] A. Juels and T. Ristenpart, "Honey encryption: Encryption beyond the brute-force barrier," IEEE Security Privacy, vol. 12, no. 4, pp. 59–62, 2014.

[26] E. Jintcharadze and M. Iavich, "Hybrid Implementation of Twofish, AES, ElGamal and RSA Cryptosystems," in 2020 IEEE East-West Design & Test Symposium (EWDTS), pp. 1–5, Sept. 2020. ISSN: 2472-761X.

[27] V. Nguyen, "Honey Encryption," Sept. 2022. original-date: 2016-11-13T17:54:38Z.

[28] "Honey Encryption." https://medium.com/smucs/honey-encryptione56737af081c. [Online; accessed 19-Feb-2023].

THANK YOU

Somil Jain

Cristina Hava Muntean

Rohit Verma