JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

HONEY ENCRYPTION ALGORITHMS TO PROTECT DATA

Dr. V Vasanthi[1], J Logeshwaran[2].

Assistant professor[1], Student[2].

PG & Research, Department of Computer ApplicationsS

Hindusthan College of Arts and Science

Coimbatore, India.

ABSTRACT: A novel encryption method called honey encryption ensures that messages decoded with false keys produce messages that appear to be real, protecting users from brute force attacks. In this work, we demonstrate our implementation of honey encryption and use practical real-world examples like credit cards and fundamental text messaging to illustrate its utility. We also add public-key encryption capability to the fundamental honey encryption technique. Finally, we go over the restrictions. We encountered additional requirements for bolstering our applications throughout implementation. Users typically choose well-known passwords that are simple to remember and decipher. In databases, passwords are frequently secured using cryptographic hash functions. When the passwords are weak, there are various hash-cracking tools that can quickly decrypt these hashes. Weak passwords are an issue for hashing, but they also compromise the security of Password-Based Encryption (PBE) schemes, where a password is used to encrypt the message. PBE is mostly utilized in password managers and is used to protect sensitive data. The Password Manager (PM) creates a small database of passwords and the accounts they belong to. This database is encrypted using a user-selected master Password, making the Master Password susceptible to brute force attacks. In this review work, we have examined Honey Encryption (HE), a novel encryption technique that offers resistance to brute force attacks by assuring that messages decoded with false keys produce messages that appear to be genuine.

CHAPTER 1

INTRODUCTION

1.1 PROJECT DEFINITION

In order to keep information secret, written or generated codes must be created using cryptography. Data may be communicated without being compromised by anyone decoding it back into a readable format thanks to cryptography, which transforms it into an unreadable format for unauthorized users. Cryptography is used in information security on numerous levels. Without a decryption key, the data cannot be read. Both while being stored and while being in transit, the information keeps its integrity. Additionally, cryptography supports non-repudiation. This implies that neither the information's originator nor its recipient can deny having done so. Cryptology is another name for cryptography.

Prior to this, all secure data was encrypted into cipher texts and could only be decrypted into plain text with a specific or pre-agreed cipher-key. When attackers attempted to breach protection levels using outdated encryption techniques, they would instead receive garbled words with no discernible meaning. That would give the hackers the impression that the key is incorrect, and they would continue using another cipherkey. This method is addressed by honey encryption (HE).Every attack that uses a bad cipherkey will result in a phoney plain-text or honey message, according to HE. Despite appearing real, some mails are false. In this manner, the attackers will have a large collection of phoney plain-texts that all appear to be real text. Therefore, even if the attacker has the real content, they still need to find it among a sea of fake texts.

We introduced honey encryption in response to the issue of inadequate passwords (HE). In the context of computer security, the term "honey" typically refers to a fake resource intended to seduce or fool an attacker. For instance, servers called honey pots are made to entice attackers for surveillance. He makes cipher text that, when decrypted with the wrong password or key, produces a message that appears to be genuine but is actually fake. Therefore, attackers are unable to determine when decryption has been accomplished.

1.2 THE VULNERABILITY OF WEAK PASSWORDS

Strong offline password-cracking assaults may easily hack weak passwords. A hash function, a one-way cryptographic operation, is frequently used to safeguard passwords in server databases. For instance, the hashed form of user Alice's password P is kept on file (P). It is standard practice to add a salt, a password-specific random number s, to the hash—that is, to save h(P, s)—to avoid hostile pre-computation of hashes across passwords. A user attempting to log in as Alice can confirm a password P by hashing it again and comparing it to the previously generated hash.

Digital data security is the subject of cryptography. It refers to the creation of systems that offer core information security services and are based on mathematical algorithms. Cryptanalysis is the term used to describe the art and science of decrypting cypher text. The cipher text is produced

by the cryptographic process and is used for storage or transmission. It entails researching cryptographic systems with the goal of overturning them. Cryptanalysis is also used to evaluate the security capabilities of new cryptographic algorithms as they are being developed. While cryptanalysis examines how to break cryptosystems, cryptography is concerned with the design of cryptosystems.

1.3 PASSWORD HASHING

One-way cryptographic operations known as hash algorithms transform any quantity of data into a fixed-length "fingerprint" that cannot be reversed back to the original material. Another characteristic of hash algorithms is that they completely differ from the original hash if the input changes, no matter how little. There are numerous ways to decode simple hashes and retrieve the original passwords. Therefore, simply hashing the passwords does not provide the level of protection that is required. The following are a few of the frequent methods used to decipher plain text password hashes:

- Dictionary and Brute force attacks
- Rainbow table
- Look up table

DICTIONARY AND BRUTE FORCE ATTACKS

These are the two most typical methods for discovering passwords. One of the simplest methods to break a hash is to guess the password; hash each guesses, and then check that each guess's hash matches the hash being cracked; if they match, the guess is the password.

RAINBOW TABLE

Using lookup tables, where the main idea is to pre-compute the hashes of passwords in a password dictionary and store them together with their associated passwords in the lookup table data structure, is a very successful technique for fast breaking multiple hashes of the same type.

LOOK UP TABLE

The time-memory trade-off methodology is implemented by the rainbow table, which is a speedy and efficient cryptanalysis method. They are comparable to lookup tables with the exception that, in order to reduce the size of the lookup tables, they sacrifice the speed of hash cracking. Rainbow tables are more efficient since more password hash solutions can be kept in the same amount of memory due to their reduced size.

1.4 USING A PASSWORD TO ENCRYPT

In addition to making hashing more difficult, weak passwords also make it more difficult for users to employ the Password-based Encryption (PBE) method to encrypt sensitive data. Hashing and PBE are both susceptible to guessing attacks. An encryption method called enc () and a related decryption function called dec

make up the PBE technology (). A password P is used to encrypt a message M as,

Cipher text C = enc P(M)

The message can be decrypted as, M = dec P(C)

When a decryption attempt is made with the wrong password P' P, dec P(C) generates an error message, making it obvious that the password was input incorrectly. As a result, the conventional method explicitly indicates an error notice when a password entry is wrong.

Because PBE cipher texts use authenticated encryption, attackers trying to decrypt it successfully are aware when they have done so. Even though the passwords are thought to be secure, brute-force assaults can still succeed if enough calculations are made. A valid-looking message output, but more significantly, an invalid-looking output as confirmation of an unsuccessful attempt, can be used to verify the decryption of a cipher text by brute force guessing of passwords.

CHAPTER 2

LITERATURE SURVEY

In [1] Honey Encryption and AES based Data Protection against Brute Force Attack by S V Manisekaran, K B Sarmila has been added to the proposed system

Here Wireless communication technologies are developing quickly, and this includes massive data handling and transfer between devices. Machine-to-machine interaction (M2M), which creates a lot of data, was developed as a result of this evolution. A platform for data movement, data processing, and a place for data to rest is provided by cloud computing. It faces many difficulties in making resources available, including data privacy**.** The conventional approach to preventing unauthorized access and maintaining the confidentiality of sensitive data is password-based encryption (PBE). The currently in use encryption methods are susceptible to numerous assaults. This work examines the performance of honey encryption in resistance to denial-of-service attacks and performs cryptanalysis of AES with honey encryption against brute force attack. Several directions for further research on data safety in cloud and restricted environments are provided in this paper's conclusion.

In [2] A Novel Password Secure Mechanism using Reformation based Optimized Honey Encryption and Decryption Technique by J. Jebathangam, Nirmalraj T has been added to the proposed system.

Here the awareness of the need to protect the many applications that collaborate with and serve Internet users has increased due to the exponential growth of online services. To be authorized, users must provide their credentials, such as their user name and secret code, to the servers. Since there have been numerous security lapses, it is important to protect this sensitive data from being used for illicit purposes. Systems must be protected against a variety of hazards. This article presents a fresh strategy for fending off brute force attacks. A solution is offered in which the user gets the keypad every time. The web server creates an encrypted password for the user's computer or device authentication after the keypad is established. Users must enter the updated one-time password (OTP) each time they access the website

in place of the original encrypted password, which will be used for authentication. This study uses optimum honey encryption (OH-E) and decryption, as well as reformation-based encryption and decryption, to protect passwords.

In [3] Hybrid Secure Cloud Storage data based on improved Encryption Scheme by B. Deepthi, R. Deepika, G. Ramani; Md Shabbeer has been added to the proposed system.

Here A tool for storing data is cloud computing. Security of data storage has emerged as a major issue. As the cloud provides services based on user demand, users can access, share, and transact with the data. How secure is the data given that it comes from several sources on the cloud? Data security concerns are escalating quickly as more data is transferred online. There are numerous encryption techniques available to protect sensitive data from unauthorized users. Data is protected using encryption and decryption techniques, and only authorized individuals are able to access the data. However, the Brute Force approach can occasionally find the buried data. A mix of AES and proxy re-encryption with Honey encryption is utilized in the suggested method to enhance data confidentiality and authentication issues. The solution enhances the security of data that has been outsourced. Only messages that appear plausible can be accessed by unauthorized parties using honey encryption and hybrid cryptography.

In [4] Enhanced System for Securing Password Manager Using Honey Encryption by Albatoul AlMuhanna; Afnan AlFaadhel; Anees Ara has been added to the proposed system.

Here Since password managers combine all user passwords in one location; brute force poses a threat to them. We describe an improved approach that uses honey encryption (HE) in the password manager to enable resilience against brute-force assaults in order to get over this limitation. The suggested system recognises and blocks any attempt by an intruder to gain access to any account. By comparing the passwords entered by the intruder with automatically created honey words that closely resemble the genuine passwords; the detector detects irregular attempts at using faulty passwords. Additionally, prevention functions by leading an attacker to a phony account with fictitious credentials in order to deceive the intruder. With execution duration of 0.36 seconds, the suggested system offers the password manager good security and integrity. Additionally, we compare honey with the OTP method to our three alternative situations for the suggested approach. Experiments demonstrate that our approach would take longer to implement the honey encryption with other security techniques, but would produce better security features than implementing the honey encryption alone.

In [5] A Password Secure Mechanism using Reformation-based Honey Encryption and Decryption by J. Jebathangam, T Nirmalraj has been added to the proposed system.

Here As technology continues to improve, various upgrades are being used on the internet to receive and analyse massive volumes of data. However, the processes involved in data storage and analysis call for some security measures. Password protection is the most often used technique for

achieving data security. The barrier to admission into the accessible environment is a password. Most of the data is kept on servers, which are widely used storage devices. A weak password pattern makes it more vulnerable to data breaches. User-generated simple passwords make it easier for hackers to crack complex passwords. The suggested framework contains a reliable password-based encryption and decryption system based on Honey. Additionally, a dynamic keypad modifying system built on reformation is used. Each time, new passwords will be created, and the letters on the keypad will have a different alignment. System access will not be permitted for weak passwords. Here, the user ID and password are both encrypted using the Honey encryption and decryption technique. The proposed framework takes into account text, alphanumeric characters, special characters, as well as capital and small letters, while generating passwords. The suggested approach guarantees the complexity of the password scheme. Every time a user enters the suggested secured system, the user ID will be verified. The password cannot be entered by unauthorised user IDs. In terms of security, the suggested strategy is effective when compared to cutting-edge techniques covered in current implementations.

In [6] Securing mHealth Applications with Grid-Based Honey Encryption by Fatimah Ahmedy, Gwo-Chin Chung, Ka-Man Chirs Lo, Soo-Fun Tan, Yu-Beng Lea has been added to the proposed system.

Here Applications and technology for mobile healthcare (mHealth) have promised to improve healthcare quality at a lower cost, especially in rural areas. However, the growing number of security breaches and patient data leaks raise concerns about the need for quick attention to mHealth applications' security dangers and privacy issues. While recent mobile health applications that rely on password-based authentication cannot withstand password guessing and cracking attacks, several countermeasures have recently been implemented to protect mobile health applications. These include One-Time Password (OTP), grid-based password, and biometric authentication. But brute force attacks, man-in-the-middle attacks, and persistent malware attacks can defeat these defences. In this research, grid-based honey encryption was proposed by fusing grid-based authentication and honey encryption. The suggested grid-based honey encryption can be used against shoulder surfing, smear, and replay attacks in addition to being more effective against password assaults than more current honey encryption. The suggested Grid-based Honey Encryption develops a UN distinct counterfeit patient's record that closely resembles the actual patients' records in light of each legitimate off-base guess password, as opposed to denying access as a current security defence technique in mobile healthcare applications.

In [7] Improved Honey Encryption to Protect the Internet users from Various Attacks by J. Jebathangam, Nirmalraj T has been added to the proposed system.

Here Data breaches and security problems have escalated over the past few months, which have raised concerns about the security of online

banking. Due to the fact that it is exclusively dependent on a password, the current authentication system for internet banking is not strong enough to withstand the recent wave of successful password guessing and cracking assaults. The authors of this research suggested Honey Encryption (HE), which adds an additional layer of security on top of the user authentication procedures currently in place, as a viable solution to this issue. The HE method rapidly generates bogus bank data that is impossible to detect when a malicious user guesses the password for a cloud account. It then links the attacker to an account that has a fake password. As a result, the attacker is unable to ascertain whether the password that was guess was a good one. This has directly led to a rise in the difficulty of password guessing and attack cracking. This research also examines several attack models and potential safeguards for password-based authentication systems.

In [8] HPAKE: Honey Password-authenticated Key Exchange for Fast and Safer Online Authentication by Kaitai Liang, Wenting Li, Ping Wang has been added to the proposed system.

Here one of the most used safe mechanisms for real-world web applications is password-only authentication. However, it is easily vulnerable to a real threat from both internal and external attackers, namely password leakage. The authentication server's password file could be compromised by an external attacker, and an insider could steal the credentials on purpose or unintentionally disclose them. There are currently two major methods to stop the leak: the honeyword technique for external attackers and enhanced password-authentication key exchange (aPAKE) against insiders. However, none of them can withstand both assaults. We offer the idea of honey PAKE (HPAKE) to close the security gap by enabling the authentication server to detect password leakage and achieve security that goes beyond what is typically possible with aPAKE. On top of the honeyword method, honey encryption, and OPAQUE, a standardised aPAKE, we design an HPAKE structure. We do a formal analysis of our design's security, achieving insider resistance and password breach detection. We put our design into practise and deploy it in the actual setting. According to the experimental results, our protocol only takes 71.27 milliseconds to run completely, with computation taking 20.67 milliseconds and communication taking 50.6 milliseconds. This demonstrates that our design is secure and useful for practical applications.

In [9] Advanced Honey Encryption: An Escape-less Trap for Intruders by Piyush has been added to the proposed system.

Here the applications of computer network communication technologies are expanding quickly every day. In parallel, it causes occurrences of user-experienced computer network abuse. Most of them have to deal with the potential of pressure during conversation as a result of such situations. Although there are several types of security technology accessible, cryptography is now thought to be more reliable. The majority of digital networks rely on cryptosystems, which guarantee the confidentiality and integrity of network communication even in the presence of third parties. No matter how hard businesses may work

to maintain their defences, it appears like there will always be some crafty hackers who manage to get around every barrier in their path. Because of this, attempts have been made to lay new foundations for the science of cryptography in the communications networks. One of these initiatives was Honey Encryption, which converts its defensive action into a deflecting action [3] by showing hackers a variety of phony keys that seem to be original. As people choose shorter passwords out of laziness and hackers' skills continue to advance, a more potent strategy is suggested in this study article. When employed with honey encryption, it will produce outstanding results. This study focuses on honey encryption, a suggested method for transforming a defensive response into a detection (locating the hacker) and deflection action by creating a certain kind of false key. Additionally, it will demonstrate how the victims and other users who have a similar risk of assault will be informed about the attack. This study paper's focus is on the fundamental ideas of honey encryption, the proposed concept for enhancing its potency and effectiveness, the implementation of this concept in practical settings, and how it functions in its whole.

In [10] An approach towards making Honey Encryption easily available by Chandraket Singh, S. Nagraj; Apoorv Singh, Samridhi Mishra has been added to the proposed system.

Here Juels and Ristenpart introduced honey encryption (HE) as a defence against brute force assaults in 2014. Honey encryption adds an extra degree of security to the user by sending phoney messages when the user enters the erroneous password. These fake messages all contain sweet phrases in the distribution transforming encoder (DTE). Security worries among consumers are growing as technology develops. In this study, we offer strategies for making honey encryption accessible and useful. In order to reduce the headache of programming from scratch, we have developed an Application programme interface (API) that would aid in proper enhancement of easily making honey encryption for the programmers. In order to reduce the headache of programming from scratch, we have developed an Application programme interface (API) that would aid in proper enhancement of easily making honey encryption for the programmers. We have suggested various ways to use honey encryption with less message space as well as ways to use it with messages in natural language. This paper outlines our strategy for making honey encryption easily accessible, making work on it simple, and accelerating honey encryption research.

CHAPTER 3

EXISTING SYSTEM

3.1 EXISTING SYSTEM

One of the most popular study areas in cloud computing is the secure storage and retrieval of documents. Despite the fact that numerous searchable encryption techniques have been developed, only a small number of them enable effective retrieval of documents that have been encrypted according to their properties. This study introduces the first hierarchical attribute-based encryption system for document collections. If

several documents share a common access structure, they can be encrypted together. Both the cypher text storage volume and the encryption/decryption processing time are reduced as compared to cypher text-policy attribute-based encryption (CP-ABE) systems. Then, using the TF-IDF model and the attributes of the documents, an index structure known as the attribute-based retrieval features (ARF) tree is created for the document collection. The ARF tree is developed using a depth-first search method to increase search efficiency, which can be further enhanced by parallel processing.

Data owners can distribute their protected data via cloud storage with authorised users while keeping the access control restrictions hidden thanks to CP-ABE (Ciphertext-Policy Attribute-Based Encryption). A strategy to restrict users from gaining access to a data owner's limited number of data objects that constitute a conflict of interest or whose combination is sensitive has not yet been researched. We examine the underlying relationships between these specific data items in this work, establish the idea of the sensitive data set constraint, and suggest a CP-ABE access control system with concealed characteristics to address the sensitive data set constraint.A flexible, partially hidden constraint strategy is used in this scheme. Due to the separation of duties principle, our plan separates the enforcement of the access control policy and the constraint policy into two separate entities to increase security. After the system has been set up, the data owner can partially change the sensitive data set constraint structure thanks to the hidden constraint policy.

3.2 DISADVANTAGES

- "AND" search limits the flexibility of assigning the attributes to the documents.
- The update strategy of the ARF tree is not proposed.

CHAPTER 4

PROPOSED SYSTEM

4.1 PROPOSED METHOD

Encryption is the process of turning plain text into a cypher text. Decryption is the process of transforming the encrypted text back into the original text; the entire procedure is referred to as cryptography. Algorithms come in two varieties. Both of them are symmetric and asymmetric. An example of an asymmetric algorithm is honey encryption. The papers are encrypted and decrypted using this algorithm. The information is encrypted with a public key. The original data as well as the publicly available data will be transformed into encrypted text. The receiver will receive this. The recipient will receive a letter with the private key for decryption. Once the right key is entered, decryption is feasible. The original document will be downloaded if the right key is used. If the wrong key is entered, a duplicate copy of the document will download. The search algorithm is the OR algorithm. Also used for searching is breadth first search.

Based on how encryption and decryption are implemented in the system, there are fundamentally two different types of cryptosystems:

- Symmetric Key Encryption
- Asymmetric Key Encryption

The link between the encryption and the decryption key is the essential distinction between various cryptosystems. Both keys are logically linked together in every cryptosystem. With a key unrelated to the encryption key, it is nearly hard to decrypt the encrypted text.

ADVANTAGES

- Honey encryption provides complete security for documents.
- "OR" algorithm is used to retrieve the documents based on any one of the attributes.
- Computational cost is effective.

CHAPTER 5

MODULES DISCRIBTION

5.1 MODULES

- **USER**
- **FILE SELECTION**
- **ENCRYPT**
- **DECRYPT**
- **RECEIVE**

5.2 MODULE DESCRIPTION

5.2.1 USER REGISTRATION & LOGIN

Users have to register with the necessary details. These details will be stored in the database. Users have to login in order to send and access files. User has to login with the correct details. If details match, the process will proceed to the next step.

5.2.2 SELECT FILE & ENCRYPT

User has to select the file that has to be encrypted. Then encrypt the file using honey encryption. The file along with the public key will be used for encryption. Upload the encrypted file in the database.

5.2.3 DOWNLOAD FILE & DECRYPT

User has to login for downloading the file. After download, user has to request for the decryption key to the file uploaded. Decryption key will be mailed to the requested user. Once the decryption key is received, user can decrypt the file. If correct key is entered, original file will be received. If incorrect key is entered, a file from the database will be received.

5.2.4 SEARCH AND RETRIEVE FILES

User can search for file using keywords. Single keyword or multi keywords can be used for searching the files. When a single keyword is entered, files related to the entered keyword will be retrieved. When multi keywords are entered, files related to the entered keywords will be retrieved.

CHAPTER 6

ALGORITHMS AND METHODS

6.1 HONEY ENCRYPTION SCHEMES

The implementation of the message space, which contains all possible values for passwords, is one of the two key components of this design. The second component is the Distribution-transforming Encoder, which uses the designated functions to encode or decode the message space. Using a given value for n, the probable values are mapped to a seed. The seeds are assigned based on the likelihood that the password will be used. The

seeds are given a higher likelihood than the unlikely/uncommon passwords, much like for more popular passwords.

The primary concept behind the pure honey encryption method is DTE (Distributed Transforming Encoding). Through DTE, honey encryption controls the plaintext space. Let p over the message L represent the probability distribution over the message space. The message L is encoded by the distribution transformer as a K-bit seed, S0, 1K, and is decoded using the inverse DTE method, decode (S) =L. The distribution of messages can be well modelled using DTE. DTE encryption and DTE decryption are both included in the HE's internal structure. The overall operation of the Honey Encryption is described by the two algorithms.

Honey Encryption Algorithm:

H □ Enc(X, L)

S □ $ encode (L)

R □ $ {0, 1} n

S‟□ H(R, X)

C □ S'⊕S

Honey Decryption Algorithm:

H □ Dec(X, (R, C))

S‟□ H(R, X)

S □ C ⊕S‟

L □ decode(S)

Return L

H stands for a cryptographic hash function, X for a key, L for a message, S for a seed, R for a random string, C for a cipher text, and $ for the possibility of using a certain number of uniform random bits with the honey encryption process. When the Honey Encryption is used to encrypt a plaintext message L, it first converts L to S and then converts S to an encrypted state using an appropriate symmetric encryption technique and a key X. These stages are explicitly described in the aforementioned techniques, and Honey encryption offers great message recovery security. An illustration of this operation would be the encryption of soft drink flavors. Three flavors, including apple, mango, and orange, are included in this example.

The example below can be used to explain honey encryption. Let's suppose Bob wants to encrypt his preferred soft drink flavour, L= Mango, and communicate it to Alice using a shared secret key, 0000. In order to map the message L into the space of the 2-bit strings 00, 01, 10, and 11, Bob creates a flavour soft drink DTE. The way it works is that via DTE, an encoded Apple will have the value 00 and an encoded Orange will have a randomly selected value of 10 or 11.Mango, the message Bob used to encrypt it, has the value 01. Bob chooses a random string R, calculates S=H(R, X), and assumes S=(R, 0000) =11 before computing C=1101=10 and sending it to Alice.

Using the key that Bob and Alice shared, which is key X=0000, Alice decrypts C. As a result, S=CS'=1011=01, S=H(R, 0000) =11, and the encode (01) = Mango, the message is successfully recovered by Alice. Let's say Eve, the

attacker, tries to crack it. Since he is unsure of the key being used, he makes the assumption that it is one such as 1432, H=(R, 1432) = 00, S=CS=10, and decode (10) = orange. This innovative encryption method thus deceives the attacker.

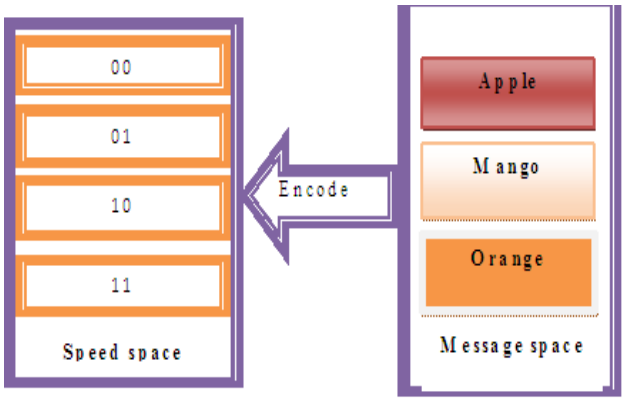

Fig DTE mapping

The message "apple" (with pm = 1/4), "mango" (with pm = 1/4), and "orange" (with pm = 1/2) all map to the numbers 00, 01, and 10, respectively; pm is a probability distribution over the message space.

CHAPTER 7

EXPREMENTAL ANALYSIS

We employ a finding from Berenrink, Friedetzky, Hu, and Martin [8] to assess more settings in general. This result builds on a method dubbed "majorization" that Azar, Broder, Karlin, and Upfal [1] had previously used for the balls-and-bins setting.

Distributions like pk and pd can be thought of as vectors over R with the proper dimension. Below, we use the assumption that the vector's parts are arranged in decreasing order, i.e., that pk(i) pk(j) for I j. Let pk, p′ k R a and let m be a number.

Distributions like pk and pd can be thought of as vectors over R with the proper dimension. Below, we use the assumption that the vector's parts are arranged in decreasing order, i.e., that pk(i) pk(j) for I j. Let pk, p′ k R a and let m be a number.

If Pa i=1

p ′ k I = Pa

i=1 pk[i] and

Pj i=1 p ′ k I

Pj i=1 pk[i] for every 1 j a,

then p ′ k majorizes pk,

Represented as p ′ k pk.

According to memorization, p ′ k is more "concentrated" than pk because each prefix of any length in p ′ k has a cumulative weight that is at least equal to the cumulative weight of a prefix of the same length in pk.

CHAPTER 8

FUTURE ENHANCEMENT

When A knows some background information about the target message, HE security is put in jeopardy. This severely limits the use of HE in circumstances like securing RSA or HTTPS private keys. The HE construction's assumption that the key and message distributions are independent is a second drawback. These two constraints can be developed and defined in the future.

CHAPTER 9

CONCLUSION

This project proposes a Honey Encryption system for protecting the public cloud file storage. The existing file protection relies on password-based encryption, which is vulnerable to password guessing attacks like brute-force, dictionary, or rainbow table assault. The suggested plan adds an extra layer of security to already-encrypted files. Instead of blocking their data access as would be the case with a standard file encryption technique, the extended HE algorithm creates a false file that is very identical to the actual file when the attacker tries to access the encrypted data with his guessing password. It is apparent that the suggested scheme's message space is pre-fixed, and that as the lengths of file names and their extensions rise, so do the complexity and size of inverse sampling tables.

CHAPTER 10

REFERENCES

[1] S V Manisekaran, K B Sarmila "Honey Encryption and AES based Data Protection Against Brute Force Attack" published in 2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC).

[2] J. Jebathangam, Nirmalraj T "A Novel Password Secure Mechanism using Reformation based Optimized Honey Encryption and Decryption Technique", 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS).

[3] B. Deepthi, R. Deepika, G. Ramani; ; Md Shabbeer, "Hybrid Secure Cloud Storage data based on improved Encryption Scheme" Published in: 2021 International Conference on Emerging Smart Computing and Informatics (ESCI).

[4] Albatoul AlMuhanna; Afnan AlFaadhel; Anees Ar "Enhanced System for Securing Password Manager Using Honey Encryption", Published in: 2022 Fifth International Conference of Women in Data Science at Prince Sultan University (WiDS PSU).

[5] J. Jebathangam, T Nirmalraj "A Password Secure Mechanism using Reformation-based Honey Encryption and Decryption", Published in: 2022 International Conference on Inventive Computation Technologies (ICICT).

[6] Fatimah Ahmedy, Gwo-Chin Chung, Ka-Man Chirs Lo, Soo-Fun Tan, Yu-Beng Lea, "Securing mHealth Applications with Grid-Based Honey Encryption", Published in: 2021 IEEE International Conference on Artificial Intelligence in Engineering and Technology (IICAIET).

[7] J. Jebathangam, Nirmalraj T "Improved Honey Encryption to Protect the Internet users from Various Attacks", Published in: 2022 3rd International Conference on Electronics and Sustainable Communication Systems (ICESC).

[8] Kaitai Liang, Wenting Li, Ping Wang, "HPAKE: Honey Password-authenticated Key Exchange for Fast and Safer Online Authentication", Published in: IEEE Transactions on Information Forensics and Security (Early Access)

[9] Piyush, "Advanced Honey Encryption: An Escape-less Trap for Intruders", Published in: 2018 4th International Conference on Computing Communication and Automation (ICCCA).

[10] Chandraket Singh, S. Nagraj; Apoorv Singh, Samridhi Mishra, "An approach towards making Honey Encryption easily available", Published in: 2020 2nd International Conference on Advances in Computing, Communication Control and Networking (ICACCCN).