Experimental Analysis with different approaches for creating and managing a Cloud Based Secured and Encrypted Password Manager

Dr. P Naga Jyothi[1] Department of CSE, GITAM School of Technology, npothaba@gitam.edu, A.P.,INDIA.
Vachavaya Asish Raj[2] ,TCS System Engineer, vashish888@gmail.com, A.P.,INDIA.
Pentela Vinay Mohan[3], Graduate Software Engineer,Coforge ,vinaymohan768@gmail.com, A.P.,INDIA.
S Sanyasi Raju[4] ,Advanced Associate Software Engineer,ACCENTURE,,sagisunny25@gmail.com, A.P.,INDIA.
D G Venkatakrishnasrinidh[5], Jr.Software Engineer,Zensar Technologies,srinidh.deevi@gmail.com, A.P.,INDIA.

Abstract

Information security is the one of the leading domain for different types of networks, where confidentiality is essential. IS serve as the first line of defense for almost all of our electronic data, networks, servers, devices, accounts, websites, files, and other assets. As now a day's scenario has a plethora of passwords to remember and follow in some way. Because of the rapid and continuous increase in the number of online accounts that the average user must manage, memory has become a significant weakness of vanilla passwords, reducing their power and diversity by end-users. Password managers are introduced as a partial solution to help alleviate the problem. These software programs hope to use functional password features while reducing their memory load. However, password managers are not without flaws. This article provides an overview of how password management software applications work, with enhanced AES, MD5 with multithreaded PBKBF2 algorithms as well comparison of five password management programs are LastPass, Dashlane, Keeper, 1Password, and Keepass, discusses about the most recent recommendations for secure password management practice in terms of providing security. This article proposes better methods for storing passwords in the cloud and sees an advantage over traditional password manager implementation

Keywords-Vanilla passwords, security, cloud password manager

1. Introduction

Password Managers (PM) software systems attempt to reduce the problems associated with using a "vanilla" password in creating, storing, and managing passwords. In a safer and more usable manner. PMs differ in type based on their usage and performance. The Base functionality of PMs is varied in two classes as retrieval and generative PMs and these have been classified in various ways like text based, non-text, graphical, and digital object etc. With the current implementation of this cloud-based retrieval password manager, the idea is to secure the communication to servers and databases, which would open new doors to vulnerabilities, forcing new encryption methods and practices to be implemented [1].

2. Background Study

The author presents a method for the login process for online applications that ought to utilize the SHA512 encryption algorithm, split into many processes, which need investigation and system vulnerability check. The program configuration comprises of flowchart design and conceptual design of 512 bits to guarantee system security and data privacy. Penetration Testing SHA512 algorithm is better in terms of endurance and strength, Usage of MD5 hashing should be avoided (susceptible to collision attacks). SHA-512 hashing round should be optimized based on hardware resources to prevent computation halt. Some advantages of SHA 512 are no successful attack to date, and it can be used for any data type not only passwords and sensitive information but still, there are some disadvantages like while using hashing function it will be slower as a number of iterations increases, and it requires a table of eighty 64bit constants (a 640 bytes lookuptable) hence more memory space [1] [2].

Considering the aftereffects of exploration and conversation, it might be assumed that the login cycle in web-based applications requires refreshing the encryption technique used by the strategy for SHA 512 algorithm. To analyze and choose a gender at the attack complexity of the hashed password, salted password, key extending, and the ENP Database. Not powerless against pre-computation attacks query table and rainbow attack. The appropriation, audit, and repudiation of password management in a multiuser workspace to accomplish by keeping an access right and privilege level. The privilege level and a protection related to each given object decide if an ostensible access privilege for this object comparesto a powerful, perhaps more fragile access privilege or is denied[3].Furthermore, a subject that holds a key for an entry right that isn't the most good access right can share keys for more sensitive access opportunities at a tantamount honor level. Survey and revocation can be restricted to a subset of the by and the large, enormous number of passwords. Somewhere near two passwords for identical access right at various honor levels can be denied autonomously to one another. We have shown that a mystery expression study or repudiation influences are transitive and short-lived [4].

It's clearly a fact that public key techniques (e.g., exponentiations in a multiplicative gathering) are essential to make secret phrase frameworks secure against disconnected word reference assaults, while the commitment of public-key cryptosystems under a PKI (e.g., public-key encryption and electronic mark plans) isn't principal. It merits zeroing in that the ENP doesn't need extra components (e.g., salt) while suppressing query table attacks. Later on, other NDB generation algorithms will be observed and made familiar with the ENP to foster password security. Besides, various strategies, such as multifactor and challenge-response verification, will be brought into our password authentication system [5]. A key reduction system allows a subject that holds a key for a given item to circulate keys for more weak access right at lower honor levels. A subject that has a given item can overview or deny the passwords for this item by basically changing the security graph. The memory necessities to address an insurance outline are immaterial. The possibility of a security graph is introduced, imparting an association between the apparent access honor related to a secret word and the convincing access honor permitted by responsibility for key matching that secret key [6].

A front-end server keeps one portion of a password, and a back-end server, holding one more portion of the password, participates in validating a user and, in the interim, laying out a secret key with the user [5]. Rather than existing multi-server password systems, the system has an extraordinary potential for practical applications. Usage of 2 server models to only allow authorized and valid requests to the actual business logic servers and maintain a public server where all attacks can be detected and stopped if necessary. Public keys can be used to make encrypted API calls to public servers and private keys can be derived from those public keys to identify the authorized user [7].

Evasion of Public key Infrastructure (PKI). However, there are a couple of burdens of it like while forestalling a single point of vulnerability, it offers a system more chance to defend the attacks. Slowerthroughputs since keys have bigger word lengths. Inferable from the progression of the Internet, innumerable web-based administrations have emerged, in which password approval is the most extensively used authorization method, for it is available at an insignificant cost and easy to deploy. Consequently, password security by and large attracts unbelievable interest from the academia and industry [8].

3. Methodology and Design

The proposed work for some online PMs (Password managers) to release private data? As per David Silver, one of the current PMs, LastPass, has proactively spilled credentials to unapproved groups. There has been a known attack on other PMs. Software-based password management is another choice. The restriction of these PMs is that they must be utilized on one PC. This cloud-based password manager could be an add-ons of having the option to be utilized in different applications (games, documents, envelopes, and so forth) and on numerous PCs. The model passages in the authorization data table are ENPs and the results show that the ENP could oppose query table attacks and give more reinforced password surety under dictionary attacks. The system can be recommended under certain key points for obtaining the better results like applying strong encryption algorithms such as SHA512, AES, and PBKDF2 and usage of strong alphanumeric password. A random salt for PBKDF2 is generated on the first run and a distinct computer fingerprint is used as a pepper for PBKDF2, uses passphrase to encrypt and decrypt folders. The applicability can be done over mobile and web app without jeopardising security, and displays the user's overall security strength, be able to back up passwords in encrypted form with multiple cloud providers to be more fault tolerant. The more advantageous part is it is self-hosted as user can have complete control over your data and upto 2.5 million iterations of PBKDF2 can be carried.

3.1 System architecture

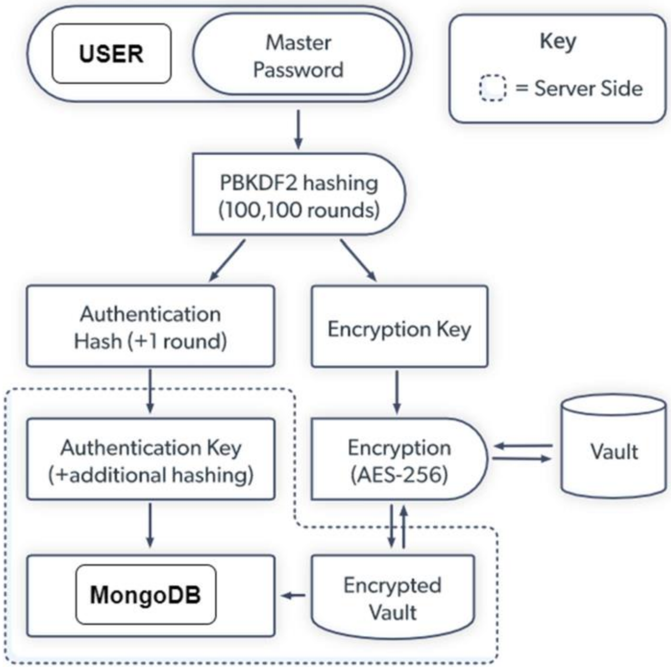

USER
Master Password
Key
= Server Side
PBKDF2 hashing (100,100 rounds)
Authentication Hash (+1 round)
Encryption Key
Authentication Key (+additional hashing)
Encryption (AES-256)
Vault
MongoDB
Encrypted Vault

Figure.1 System architecture

Figure 1 describes the journey of the password string provided by the user being encrypted and saved in an encrypted vault. The master password entered by the user is sent to the PBKDF2 module for hashing up to 100,000 rounds; this module is multi-threaded. Each thread shares a standard message queue to disseminate results at the end of the round. The result is replicated for two purposes, one for accessing the protected vault -Encrypted Key and the other for storing the master password to the user profile in our database - Authentication Key. One copy of PBKDF2 hashing is AES256 encrypted to provide additional phrase strength and stored in a vault. Later the user uses this key to access the encrypted vault. Another copy undergoes one more round of hashing to eliminate the risk of replication in terms of encrypted and authentication keys. The dotted line in the figure 1 denotes process happening inside database environment. Encryption Algorithms

3.2.1 Enhanced AES : To address the issue of over-calculation and overhead calculation, system tested and tweaked the Advanced Encryption Standard (AES) to reduce algorithm analysis and improve encryption performance. The essential objective of adjusting AES is to give less calculation and further develop data security. The altered AES algorithm has been optimized to give the quickest encryption speed conceivable.The duration of the block and the critical length in modified-AES are specified in accordance withthe AES.

The encryption process included 14 rounds of Inv-Shift rows, Inv-Permutation, and Addroundkey.As a result, in the proposed system, a modified AES algorithm with 14 similarities will be used to encrypt and monitor any data file (AES Protected System). For the encryption process, bit permutation is used, whereas the opposite bit permission is used for the encryption process for data files. The modified AES algorithm is structured as follows.

The size of the document (file) encrypted during unit encryption is the output of any encryption algorithm. As a result, a high encryption throughput rate indicates that the algorithm is fast and mustbe encrypted. The size of the document (file), which will be removed from encryption during the decryption unit time, is the excess encryption of any decryption algorithm. As a result, a high output value indicates that the algorithm is fast and ready to be encrypted. After uploading the data to the proposed system, the user can analyze it by clicking a button. The other encryption techniques' performance times for different file sizes are recorded. This analysis employs five encryption methods: DES, 3DES, AES, Two-fish, and the proposed optimized algorithm [9]. As a result, it has been demonstrated that the proposed Modified algorithm consumesless time on all files.

3.2.2 Enhanced MD5: The MD5 algorithm has various shortcomings, including weaknesses to different attacks, for example, a rainbow table, dictionary, birthday, etc. Many examinations have been directed to decide on the MD5 algorithm's defects [12] the writing audit centers around the different attacks on the MD5 algorithm. Many papers center on the different sorts of MD5 attacks used to break or capture the algorithm. Attack by Brute force is a strategy for comprehensive inquiry that can be applied to any data encryption or hashing function. A rainbow table assault is one in which the first message is found by utilizing atable that exists by registering the hashed yield for various sources of info using comparablecapacities. The rainbow table is speedier in light of the fact that the item is, at this point, put away forfaster calculation penetration. The plain text hashed result not entirely set in stone and put away inthe data set for fast and capable invading of the cryptographic calculation like MD5. A rainbow tableis an upgrade for the animal power procedure. Rainbow tables are a refined version of dictionary attacks in which the hashed result is data with a higher probability of being hijacked. Because the hashed production for the same original information is the same in all cases, the time required to compute the desired hashed outcome is greatly reduced. Thus, it is one of the MD5 algorithm's huge shortcomings in light of the fact that the output for similar input is similar for all the time.

Since the current MD5 algorithm has many imperfections and deficiencies, they can be alleviated by using a superior form that utilizes a hybrid algorithm with variable output length and critical value.The upgraded MD5 algorithm enhances the current MD5 algorithm by utilizing the key and fluctuating its size. Subsequently, the superior algorithm utilizes a crossover algorithm with irregular output length and key. Since the key is different for each client, the result changes for the different keys, and anyway asimilar key conveys a comparative result. The usage of the method for assisting foster the MD5 calculation clears out the risk of rainbow table, birthday, and word reference assaults in light of thefact that the outcome is different for various keys. Since the hashing function's length makes it powerless against extra episodes, the variable length will further develop the hashing function. The delineation portrays a superior form of the current MD5 algorithm.MD5's defects and deficiencies can be alleviated by utilizing a crossover algorithm of key-value pairs of fluctuating lengths. This MD5 algorithm creates 128-bit output data. Therefore, by changing the parameters, we can alleviate the inadequacies and shortcomings of the MD5 algorithm.

PHP, HTML5, CSS3, and JavaScript are the web technologies utilized in this process. To carry out the improved MD5 algorithm, a PHP server was utilized as a backend server. Front-end advancement was done utilizing web technologies like HTML5, JavaScript, and CSS3. JavaScript was utilized to execute the superior MD5 algorithm [10].

3.2.3 Optimized Multithreaded PBKDF2: To enhancing the PBKDF2-HMAC-SHA-2 Family and the PBKDF2-HMAC-LSH Family (LSH is a lightweight hash work created in South Korea in 2014) [11]. The improved strategies are named as HMAC and Hash work streamlining. The developed system applies to any leftover top of the line processors, as GPUs. While running PBKDF2, it exploits the accessible strings. To achieve a generally outrageous (the amount of lines) times execution improvement over a lone connection execution. A key with a raised level of security is normal in cryptographic application systems. Client passwords are used to get to the data in cloud servers, information encryption structures, and a couple of utilizations. The entropy of a client secret key, of course, is lower than that of a cryptographic key. Subsequently, the client secret key can't be used as the security frameworks critical. The Password-Based Key Derived Function (PBKDF) is customarily used to determine this issue.

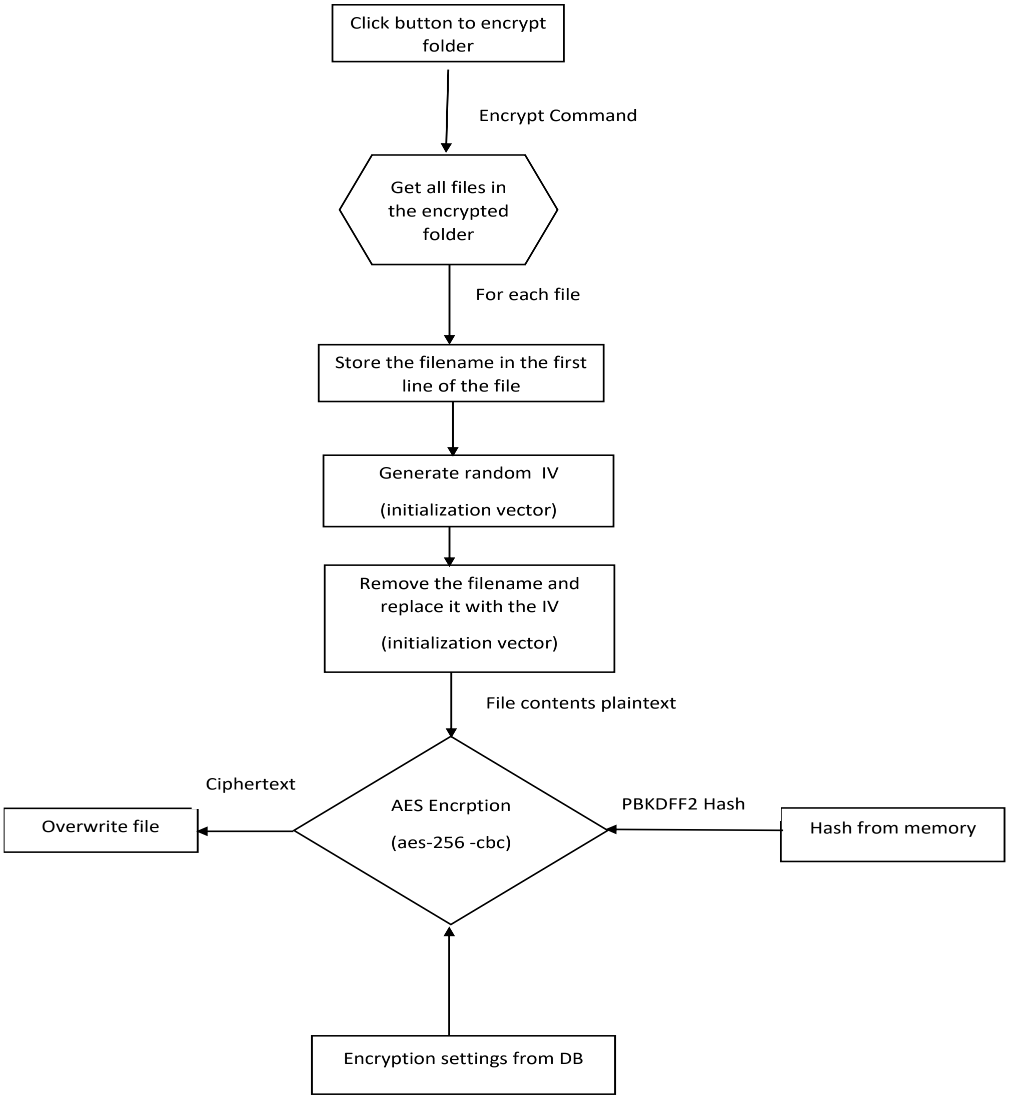


Figure 2: Flow chart of Encrypt Password Encryption Flow

The logic iterates through all the files in the folder on the user command to encrypt a folder. It performs the generation of the initialization vector (IV) from the binary data of the files. These values are stored in a hash map with filenames as key and IV as values. This process again happens in a multithreaded environment and asynchronously to make use of complete CPU cores and is extensible for large folder sizes. These IVs are then passed on to the AES256 encryption module for encryption, and during this process, hashes are checked to ensure data integrity is maintained before and after the encryption of the file. Once verified, the ciphertext is updated in the database illustrated in figure 2.

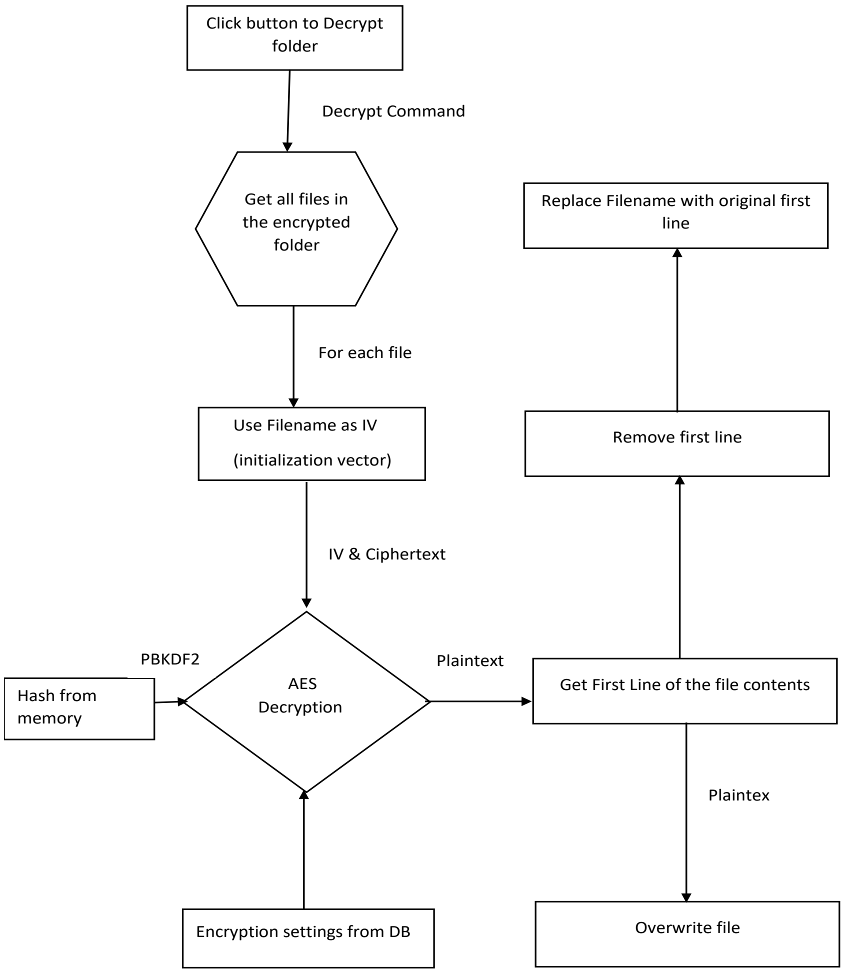
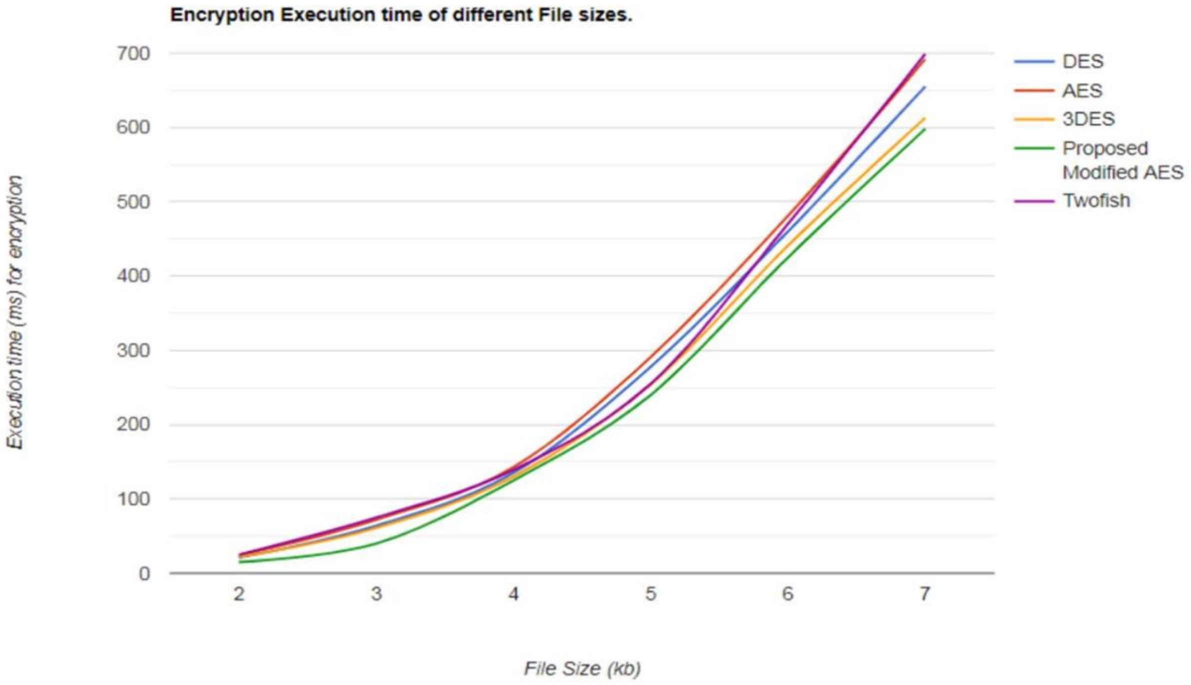

Click button to Decrypt folder
Decrypt Command
Get all files in the encrypted folder
For each file
Use Filename as IV
(initialization vector)
IV & Ciphertext
PBKDF2
Hash from memory
AES Decryption
Encryption settings from DB
Plaintext
Get First Line of the file contents
Plaintex
Overwrite file
Remove first line
Replace Filename with original first line

Figure 3: Flow chart of Decrypt Password Process

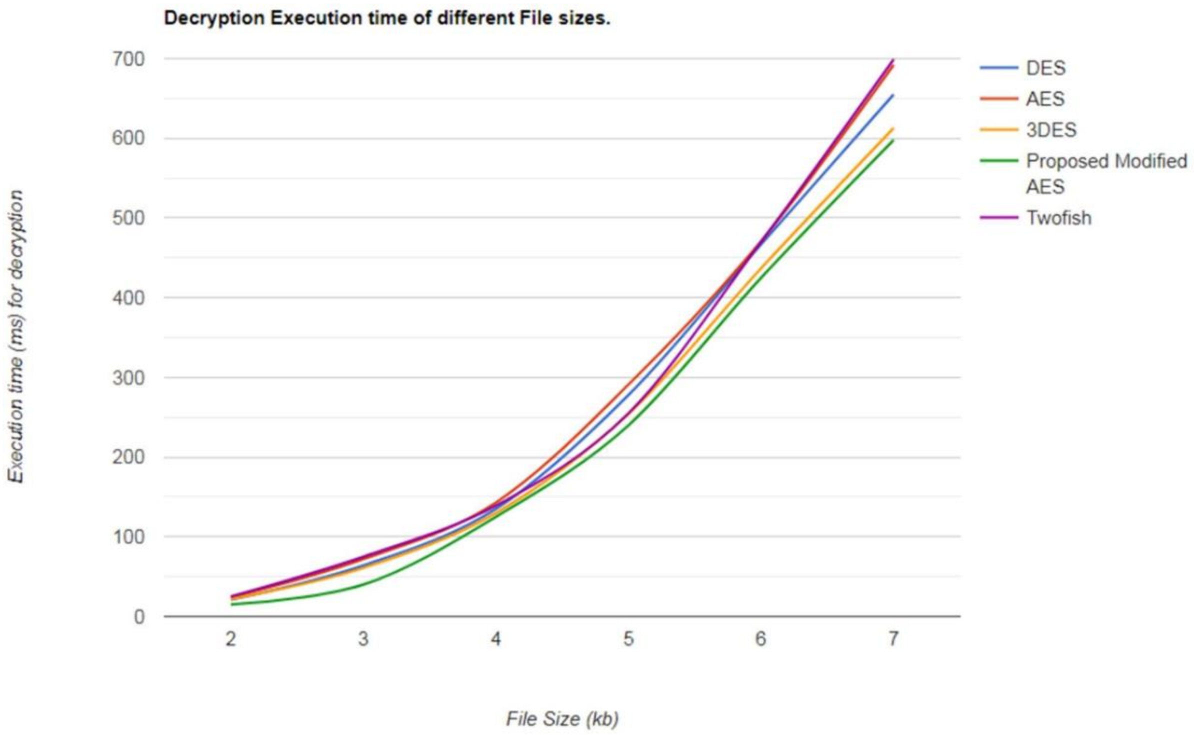
For the decryption process is shown in figure 3, the logic grabs the encrypted cipher text, which represents encrypted files, and passes it through the AES256 decryption module. This module is fed with file hashes to cross-check data integrity. This process occurs in a multithreaded environment, and processing happens parallel. The plain text obtained after decryption can't be represented as its original file just yet, due to the inclusion of IV in its binary data. This is quickly replaced as we store IV as a first-line parameter during encryption and exchange it with its filename. Once this conversion is done, we have the file in its original format and then represented to the user. Note that a file decrypted is not stored back in the database once encrypted, and all the decryption happens on run time and only when requested by the user.

4. Results and Discussions

1.1 Enhanced AES Algorithm



Results 1.Execution of encryption with variable file sizes, the proposed modified AES performs better compared to all other encryption methodology, at a point the file size and execution time of almost all encryption algorithm matches at 4kb. 86% of times from the above results, it is found that AES and Two fish perform the same with similar file sizes, the majority of variation is seen during the file size input of 4-6KB, but the proposed AES is seen to perform better in all variations compared to all the algorithms. The results were an average of 20 test cases for each encryption technique with the variable file size. The testing was automated programmatically, and the program gave all encryption techniques total resource usage and no bottlenecking environment to run at the full potential of CPU cores.



Results 2. We introduced the examination among the five calculations (5 block cipher and one stream cipher from symmetric encryption). Practically all the calculation's exhibition is something similar aside from altered AES for little record size. This is on the grounds that changed AES utilizes three rounds, with three keys utilized at each round. In this way, the encryption time is much contrasted with the others. Assuming we increment the record size, the presentation debases for AES, Two fish, and DES. Two fish, DES, and DES3, function admirably when packets are small. The cipher mode was CBC, key length 256 pieces, encoding mode = hexadecimal. The encryption characterized the cushioning plan to change over the contribution to a difference of 16 bytes, as this calculation generally involves this plan for encryption. Both encryption and unscrambling time follow a comparative example for Two fish contrasted with AES, yet there is an observable distinction when contrasted, and the proposed changed AES.

2. Conclusion and future scope

The attacker can peruse the password in clear text in the event that the PC is tainted with malware. For this situation, there are a few essential protections, for example, reinstalling the OS on the PC, which guarantees that the PC isn't contaminated. One more strategy for changing the encryption calculation from AES to RSA exists. Therefore, when the hardware device is connected to the PC or the telephone interfaces with the device, his public key is made public. At the point when a client endeavor to add another passage, the GUI application will initially scramble the secret phrase and send it to the equipment gadget, where it will be sent unmodified to the telephone. Whenever the user endeavors to get the secret word, the telephone sends the scrambled secret word to the hardware device, which unscrambles it and copies the console. At the point when passwords are added to the database utilizing this strategy, the PM keeps somebody from perusing the secret key in clear text. The limitation is that it requires additional power from the hardware device to figureout the private and public keys.

3. REFERENCES

1. Sumagita, Meiliana, et al. "Analysis of secure hash algorithm (SHA) 512 for encryption process on web-based application." International Journal of Cyber- Security and Digital Forensics (IJCSDF) 7.4 (2018): 373-381.
2. De Guzman, Froilan E., Bobby D. Gerardo, and Ruji P. Medina. "Implementation of enhanced secure hash algorithm towards a secured web portal." 2019 IEEE 4th International Conference on Computer and Communication Systems (ICCCS). IEEE, 2019.
3. Yang, Bian, et al. "Cloud password manager using privacy-preserved biometrics." 2014 IEEE International Conference on Cloud Engineering. IEEE, 2014.
4. Chu, Huiguang. Cloud Password Manager Using Privacy-preserved Biometrics. MSthesis. 2014.
5. Yang, Yanjiang, Robert H. Deng, and Feng Bao. "A practical password-based two- server authentication and key exchange system." IEEE Transactions on Dependableand Secure Computing 3.2 (2006): 105-114.
6. Lopriore, Lanfranco. "Password management: distribution, review and revocation." TheComputer Journal 58.10 (2015): 2557-2566.
7. Luo, Wenjian, et al. "Authentication by encrypted negative password." IEEETransactions on Information Forensics and Security 14.1 (2018): 114-128.
8. Burr, William E. "Selecting the advanced encryption standard." IEEE Security &Privacy 1.2 (2003): 43-52.).
9. Park, Jin Hyung, et al. "Security architecture for a secure database on android."IEEE Access 6 (2018):11482-11501.
10. Shanta, Jyoti Vashishtha. "Evaluating the performance of symmetric key algorithms: AES (advanced encryption standard) and DES (data encryption standard)." IJCEM International Journal of Computational Engineering & Management 15.4 (2012): 4349.
11. Kurniawan, Endang, and Imam Riadi. "Security level analysis of academic information systems based on standard ISO 27002: 2003 using SSE-CMM." arXiv preprint arXiv:1802.03613 (2018).
12. Riadi, Imam, and Eddy Irawan Aristianto. "An analysis of vulnerability web against attack unrestricted image file upload." Computer Engineering and Applications Journal 5.1 (2016): 19-28.