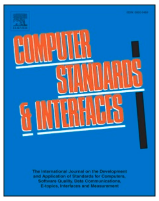
Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi



SecureMD5: A new stream cipher for secure file systems and encryption key generation with artificial intelligence

Isabel Herrera Montano [a,*], Juan Ramos Diaz [b], Sergio Molina-Cardín [b], Juan José Guerrero López [b], José Javier García Aranda [b], Isabel de la Torre Díez [a]

[a] Department of Signal Theory and Communications and Telematics Engineering University of Valladolid, Paseo de Belén, 15, 47011, Valladolid, Spain
[b] Department of Innovation, Nokia, Maria Tubau Street, 9, 28050, Madrid, Spain

A R T I C L E I N F O

Keywords:
Encryption algorithm
Encryption key generation
Artificial intelligence
Information security
Insider threat

A B S T R A C T

The insider threat to sensitive information posed by employees or partners of an organisation remains a major cybersecurity challenge. In this regard, the measures taken by organisations and companies to protect information are often insufficient. Primarily, due to the legitimate access and knowledge of security holes that these individuals possess.

This study proposes SecureMD5, an encryption algorithm designed specifically for secure file systems (SFS). The algorithm is based on custom one-way functions integrated into an encryption scheme that operates at the byte level. It uses 11 dynamic variables generated from contextual parameters such as file position, access time, random values, and user-specific keys. This approach ensures that SecureMD5 does not inherit the known vulnerabilities of MD5 as a standard cryptographic algorithm. Consequently, SecureMD5 is presented as an adaptive and robust solution that addresses the challenges posed by insider threats in SFS.

In parallel, a modular contextual key generation scheme is proposed, which can incorporate various challenges such as user identity, access time and device location. Biometric key generation based on Artificial Intelligence (AI) methods is evaluated independently from the validation of the encryption algorithm. In the evaluated biometric key generation scheme, the AI models MediaPipe Hand Landmark and LBPHFaceRecognizer from OpenCV have been used. These methods are part of a sub-key generation scheme based on contextual challenges. This scheme eliminates the need for key storage for dynamic and secure access to sensitive information.

SecureMD5 was validated by diffusion, confusion, entropy and performance analysis. It achieved 31 % higher entropy than comparable algorithms. Performance improved by 0.32 % compared to RC4. It also passed 87 % of NIST 800–22 tests, demonstrating its robustness against cryptographic vulnerabilities. In addition, SecureMD5 balances security and performance, with encryption times 25 % faster than a modified AES algorithm for 10 MB files. Biometric key generation methods were evaluated using metrics such as precision, accuracy, false acceptance rate and specificity, achieving satisfactory values above 80 % on all metrics. This work addresses critical gaps in information security, providing significant advances in protecting SFS against insider threats. The design and adaptability of SecureMD5 make it particularly suitable for sectors with strict security requirements, such as healthcare, finance, and corporate data management. Its ability to enable dynamic and secure access control addresses the real challenges posed by protecting confidential information from internal threats.

1. Introduction

Today, information security remains a major concern for organizations and companies handling sensitive information. Especially the insider threat is a handicap in the information security of any company [1–4]. In 2023, the online community "Cybersecurity Insiders" surveyed >326 cybersecurity professionals regarding insider threats. The report revealed that 74 % of the companies surveyed claimed a notable

* Corresponding author at: Department of Signal Theory and Communications and Telematics Engineering, University of Valladolid, Paseo de Belén, 15. 47011 Valladolid, Spain.

E-mail addresses: isabel.herrera.montano@uva.es (I. Herrera Montano), juan.ramos_diaz.ext@nokia.com (J. Ramos Diaz), sergio.molina_cardin.ext@nokia.com (S. Molina-Cardín), juan.guerrero_lopez.ext@nokia.com (J.J. Guerrero López), jose_javier.garcia_aranda@nokia.com (J.J. García Aranda), isator@tel.uva.es (I. de la Torre Díez).

https://doi.org/10.1016/j.csi.2025.104047
Received 11 September 2024; Received in revised form 10 July 2025; Accepted 11 July 2025
Available online 11 July 2025
0920-5489/© 2025 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY-NC-ND license (http://creativecommons.org/licenses/by-nc-nd/4.0/).

increase in the frequency of insider attacks, being vulnerable to them [5]. The new report [6] by Cybersecurity Insiders and Securonix presents the results of a survey of 467 cybersecurity professionals that reveals a 10 % increase in insider attacks over the past 5 years. The report [6] claims that there has been a 14 % increase in concern about malicious insiders over 2019. It further states that 90 % of professionals surveyed refer to the difficulty of detecting insider attacks as equal to or greater than detecting external attacks [6].

Traditional methods are not sufficient to circumvent the insider threat [7]. In this case, the authorized employee can access the information because the employee has the credentials or can share the credentials with a malicious user [8–11]. In this sense, the insider threat presents specific attacks different from the external threat. Preventing insider attacks is challenging due to access privileges. Employees often exploit security vulnerabilities within organizations [12–14]. Without forgetting the attacks known as chosen-plaintext attacks [15], chosen-ciphertext attacks [16], and known-plaintext attacks [17], that due to their characteristics there is a high probability that they are provoked by insiders.

The adoption of new methods of information loss prevention is critical in the field of insider threat. In the recent study [7], the special requirements of a valid encryption algorithm for Secure File Systems (SFS) against insider threats were presented. The main objective of this paper is to propose an encryption algorithm for the same purpose, which addresses the existing entropy constraint in Securecipher [7] and as an additional complement, this paper proposes AI techniques in the encryption key generation mechanism. The proposed scheme seeks to improve the security of confidential information and encryption keys. For this purpose, AI techniques are included in the key generation mechanism with the study of Python libraries: MediaPipe and OpenCV.

The main contributions of this paper are listed below:

1) Proposal of an encryption algorithm that addresses the security requirements of SFS and uses one-way functions of MD5. The functions are adapted to a scheme that operates with dynamic variables generated from contextual parameters of the environment.
2) Presentation of encryption key generation mechanisms that include AI to improve key security and facilitate their use.
3) Study of MediaPipe framework and OpenCV library from Python for the generation of biometric encryption keys.
4) Proposal of a new user recognition mechanism based on hand geometry.

The remainder of the paper is structured as follows: The next section describes the related work existing in the literature. Section 3 presents the system architecture, followed by the description of the experiments performed in this study and the validation of the proposed approaches in Section 4. Section 5 discusses the security and performance of the proposed encryption algorithm. Section 6 evaluates the strengths, limitations, and future work of the SecureMD5 algorithm, highlighting its applicability and potential areas for improvement. Finally, the main conclusions derived from this study are presented.

2. Related work

Numerous studies have been carried out in recent years to mitigate the risk of information loss [18–20]. Currently, the main source of data loss in organizations and companies is the human factor. This is due to the difficulty of protecting information from insider threats. Insiders have privileges to access information due to their status as workers and at the same time it is unfeasible to restrict the use of information because it affects the productivity of the worker and the company [11]. In this sense, our latest research work presents a cryptographic system integrated in a SFS that allows working with confidential information by applying the least intrusive security techniques possible [7,21].

Recently, other authors have also taken an interest in the subject, as is the case of the study presented by authors [22]. The study proposes a cryptosystem for the distribution of information in different data servers in a secure way against insider threats. In the research work of the authors [23–26] the concern for the secure transfer and storage of data in the cloud is highlighted, applying cryptographic approaches based on the encryption algorithm Advanced Encryption Standard (AES). Also, hybrid cryptography has been proposed in the study [27]. This is the combination of multiple conventional encryption algorithms to improve data storage security.

Since the emergence of AI, important changes have been observed in cybersecurity, mainly in lines such as cloud security, intrusion detection and IoT security [1,28]. One of the main weaknesses of an encryption algorithm is the security of its keys. This study and other research is leaning towards AI techniques for the generation of more secure keys [24]. The study [29] showed that the main AI algorithms used for biometric pattern recognition have been Convolutional Neural Network (CNN), Artificial Neural Network (ANN) and Support Vector Machine (SVM). Hand and face identification pattern recognition has been extensively investigated in this regard. The following is the state of the art of hand and face pattern recognition using AI.

2.1. *Hands recognition*

Previous studies have shown that it is possible to recognize gender and identify people based on the characteristics of their hands, using AI algorithms.

The author of the study [30] made available to the scientific community a dataset of 11 thousand images of hands in dorsal and palmar. This dataset was used in his study to train a CNN for gender classification and feature extraction. The features extracted with the CNN were passed to an SVM algorithm for biometric pattern identification, obtaining a hit rate higher than 95 % in all cases. The study [31] uses a total of 1640 hand images, 1200 of them obtained from common office scanner. The authors use the SVM algorithm for feature extraction and user identification, obtaining a hit rate higher than 95 % in both subsets of the data.

Hand imaging using office scanners has proven to be a good method for collecting hand images for identification pattern recognition [32]. Marek Klonowski et al. in their proposal [32] obtain a set of hand images using typical office scanners, the method is based on preprocessing the images for the calculation of 62 hand features. The calculated features (points and distances between them) allow to identify a user with 3 scanned images of his hand for an identification rate of 100 %.

The study [33] carried out at the University of Tehran in Iran, relies on feature extraction using reference points extracted with the MediaPipe framework of Python, ensuring a result of up to 98.7 %. In the study [34] the authors extracted hand features using OpenCV library for the development of a mobile biometric system, obtaining an error rate of 0.52 % which proved the feasibility of the proposed method. Other studies such as [35–38] apply the MediaPipe framework for sign language recognition and medical purposes due to its accuracy in hand and landmark detection.

2.2. *Face recognition*

Facial recognition is commonly applied for security and access control purposes because of its effectiveness in environments requiring identification [39,40].

The authors of [41] implement an encryption/decryption system based on keys, obtained from face recognition identification, using the Bio-Cryptographic technique. In the approach proposed by [41] the SVM algorithm is used for feature classification, obtaining a correct recognition accuracy of about 95 %. In [42], an encryption algorithm for user privacy protection is developed. The encryption algorithm has low complexity and is based on random unitary transforms and face recognition for key generation.

The authors of the study [43] perform a literature review in the period of 2018–2020 regarding face detection and recognition using the OpenCV library. The study reviews 20 studies using the OpenCV library for face detection and recognition and highlights its capabilities by comparing the results obtained in the reviewed studies, reaching an accuracy above 90 % in most of them. C. Pagano et al. in [44] use the OpenCV face and eye detection algorithm to extract regions of interest (ROI) from images captured from surveillance videos and Local Binary Patterns (LBP) algorithms for feature extraction from ROIs.

In the study [45], a face recognition framework based on LBP method is proposed for feature extraction. The authors highlight the LBP method for its low computational complexity and its ability for capturing more representative texture information with invariance to illumination changes. The face recognition approach proposed in this study also uses an LBP-based model from the OpenCV library for face detection and recognition.

3. System architecture

SecureMD5 is seamlessly integrated into a virtual file system (VFS). Thus, it ensures transparency for the user. While sensitive data never leaves the system in plain text. The VFS intercepts file access calls from user applications to the Real File System, dynamically performing encryption or decryption as required. This integration supports real-time contextual challenges, such as verifying user identity, access time, and device location, to generate secure keys for each operation.

Fig. 1 depicts the framework of the proposed system. It can be seen that once the application makes a call to a document, the VFS intercepts the calls as described in the operation table of [46]. Then, a series of challenges are executed to identify the user, place, date, time, and device of access to the information. Each challenge generates a subkey that together generates the key to the encryption algorithm. If the challenges identify the user, place, date, time, and device authorized to access the information, a correct key will be generated, otherwise the key will be incorrect. The modular approach to sub-key generation ensures that the encryption system can be flexibly adapted to the specific needs of the application. AI methods for biometric key generation have been independently evaluated in this study. These methods are not required for SecureMD5 to function as they are optional components within the broader framework of contextual challenges. Contextual challenges include parameters such as user identity, access time, and device location to form encryption sub-keys. These challenges will be selected based on the needs of the scenario in which it is implemented. This means that the set of contextual challenges used in each case can be customized based on the client's requirements.

To mitigate insider threat attacks and establish a level that allows the encryption algorithm to know which operation (encryption or decryption) to perform, the documents are marked. The marking mechanism involved in this study has been proposed and evaluated in the previous study [7]. The marking incorporates a random number specific to each document that is passed as a parameter to the encryption algorithm. The encryption algorithm then encrypts or decrypts as appropriate and the result is sent to the user application that made the request.

To evaluate the performance of SecureMD5 under comparable conditions, the Dokan VFS framework was used. This offering a practical implementation environment by addressing insider threats with robust security measures and context-based access controls [46,47].

3.1. Proposed encryption algorithm

SecureMD5 addresses the unique challenges of encryption in file systems vulnerable to insider threats. It was developed with the aim of mitigating the insider threat to confidential information. For this purpose, the requirements necessary for an algorithm to be valid against insider threats, as stated in the study [7], were considered. SecureMD5 is basically a hash function that reduces the sequence of parameters obtained to a single byte (B) and adds it to or subtracts it from the B of the position in question in the plain text, as described in the Eqs. (1) and 2.

$$cB = pB + f(\,p, frn, k) \tag{1}$$

$$dB = cB - f(p,\ frn, k) \tag{2}$$

Where, cB is the encrypted byte, pB is the plaintext byte, f is the hash function, k is the key generated by the context-based key generation mechanism, p is the position in the text of the plaintext byte being encrypted or decrypted, frn is the random number obtained from the mark and dB is the decrypted byte. Fig. 2 represents the SecureMD5 flowchart.

The following steps describe the encryption/decryption flow detailed in Algorithm 1 and Algorithm 2, supported by Eq. (1) and (2). In addition to addressing the key expansion and padding management of the algorithm.

First, the plaintext parameters, document markup, and key generation mechanism are obtained. Being p a vector of $8B$, frn a vector of $4B$ and k a vector of variable size and pB is $1B$ corresponding to the plaintext character to be encrypted. Secondly, the message to be passed to the hash function is constructed. The parameters $p,\ frn,\ k$ are converted to integers and concatenated, creating a list of M integers of variable size

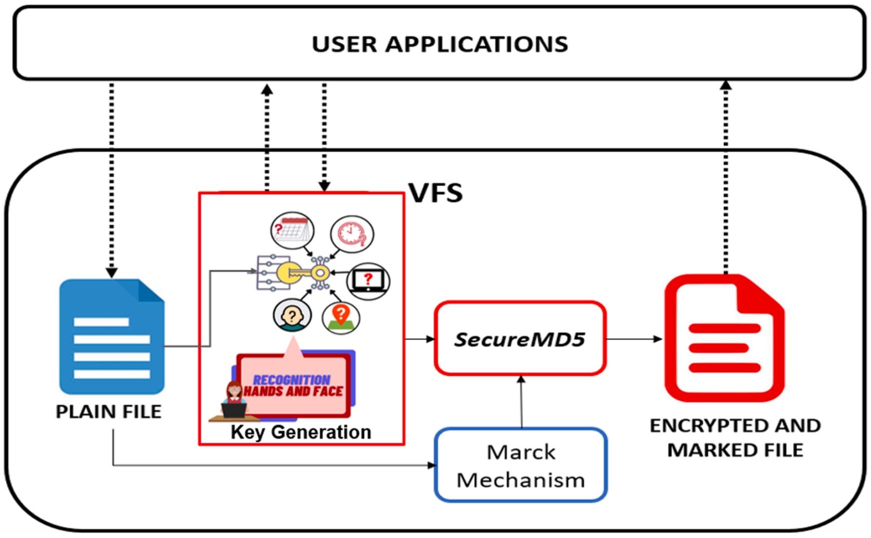


Fig. 1. System Architecture.

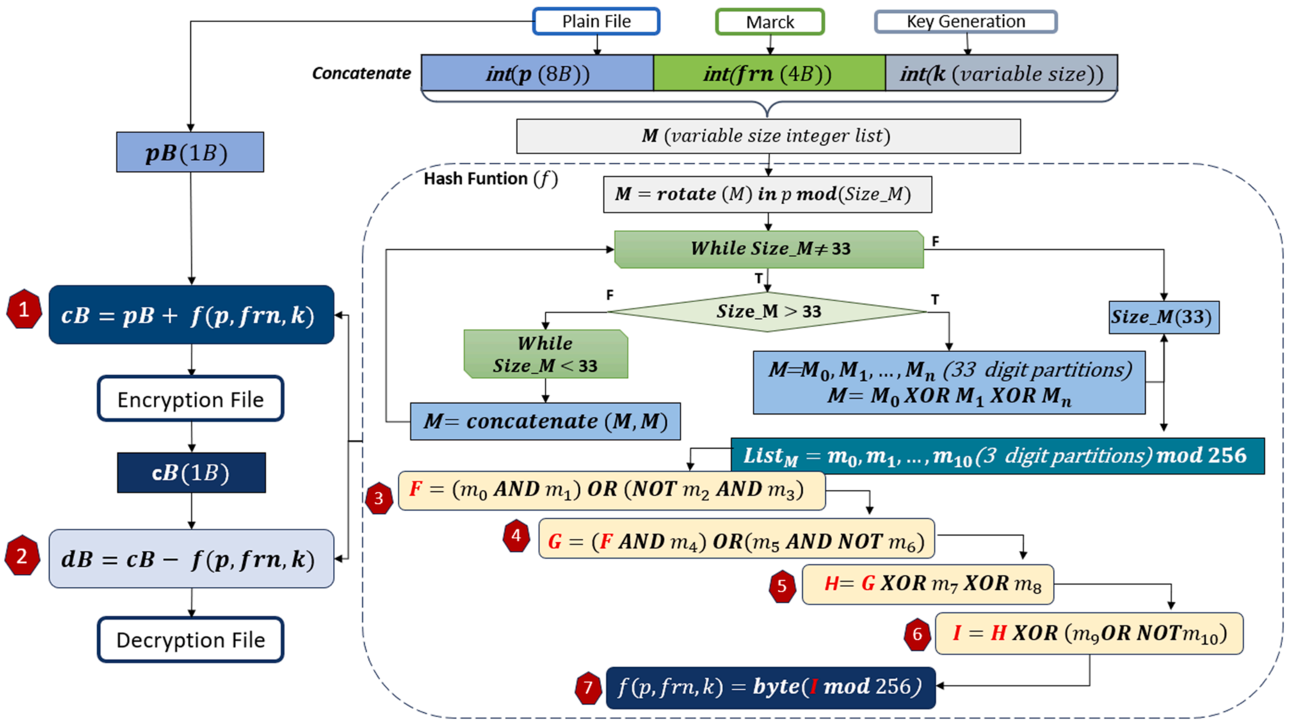

Plain File
Marck
Key Generation
Concatenate
int(p (8B))
int(frn (4B))
int(k (variable size))
pB(1B)
M (variable size integer list)
Hash Funtion (f)
M = rotate (M) in p mod(Size_M)
While Size_M≠33
Size_M > 33
While Size_M < 33
M= concatenate (M, M)
M=M0, M1, …, Mn (33 digit partitions) M= M0 XOR M1 XOR Mn
Size_M(33)
ListM = m0, m1, …, m10 (3 digit partitions) mod 256
1 cB = pB + f(p, frn, k)
Encryption File
cB(1B)
2 dB = cB − f(p, frn, k)
Decryption File
3 F = (m0 AND m1) OR (NOT m2 AND m3)
4 G = (F AND m4) OR(m5 AND NOT m6)
5 H= G XOR m7 XOR m8
6 I = H XOR (m9 OR NOT m10)
7 f(p, frn, k) = byte(I mod 256)
T
F

Fig. 2. Encryption algorithm flowchart. B= byte.

Algorithm 1
Encryption Algorithm.

```
Encryption Process
  1. Read $pB,\ p,\ frn,\ k$.
  2. Convert $p,\ frn,\ k$ to integers.
  3. Concatenate $p,\ frn,\ k$ to create the message in a list of integers ($M$) of size equal to the number of digits that make up the list, each position in the list corresponding to one digit.
  4. Rotate the list positions by p modulo list size ($Size_M$).
  5. Adjust the message to a size equal to 33 positions:
  While $Size_M \neq$ 33 do:
      If $Size_M$ >33:
        Create 33-position sublists ($M_0 - M_n$)
        XOR $(M0, M1, \ldots, Mn)$
      else:
        While $Size_M < 33$ do:
        Concatenate $(M, M)$
  6. Create list ($List_M$) of 11 integers ($m_0 - m_{10}$) of 3 digits, modulus 256. The numbers in the list are created for every 3 consecutive digits of the list M.
  7. Compute functions F, G, H, and I, as shown in equations 3; 4; 5 and 6 respectively.
  8. Compute the hash function as shown in Eq. (7).
  9. Compute the encrypted byte as shown in Eq. (1).
  End Process
```

Algorithm 2
Decryption Algorithm.

```
Decryption Process
  1. Read $pB,\ p,\ frn,\ k$.
  2. Convert $p,\ frn,\ k$ to integers.
  3. Concatenate $p,\ frn,\ k$ to create the message in a list of integers ($M$) of size equal to the number of digits that make up the list, each position in the list corresponding to one digit.
  4. Rotate the list positions by p modulo list size ($Size_M$).
  5. Adjust the message to a size equal to 33 positions:
  While $Size_M \neq$ 33 do:
      If $Size_M$ >33:
        Create 33-position sublists ($M_0 - M_n$)
        XOR $(M0, M1, \ldots, Mn)$
      else:
        While $Size_M < 33$ do:
        Concatenate $(M, M)$
  6. Create list ($List_M$) of 11 integers ($m_0 - m_{10}$) of 3 digits, modulus 256. The numbers in the list are created for every 3 consecutive digits of the list M.
  7. Compute functions F, G, H, and I, as shown in equations 3; 4; 5 and 6 respectively.
  8. Compute the hash function as shown in Eq. (7).
  9. Compute the encrypted byte as shown in Eq. (2).
  End Process
```

($Size_M$), where each position corresponds to a digit. Once we have the message, we pass it to the hash function and rotate the positions of the list in as many places as p module $Size_M$. Then, as long as $Size_M$ is different from 33, it is checked if it is greater or smaller. If $Size_M$ is <33, the list M is concatenated with itself until a message with >33 positions is obtained. When $Size_M$ is greater than 33, M is partitioned for each 33 positions and XOR between them. With the 33-position message, a list of 11 integers, each of three digits, modulo 256, is created.

Then, the positions of the list from m_0 to m_{10} are passed sequentially to the one-way functions of the MD5 hash algorithm as shown in equations 3; 4; 5 and 6.

$$F = (m_0\ AND\ m_1)\ OR\ (NOT\ m_2\ AND\ m_3) \tag{3}$$

$$G = (F\ AND\ m_4)\ OR\ (m_5\ AND\ NOT\ m_6) \tag{4}$$

$$H = (G\ XOR\ m_7\ XOR\ m_8) \tag{5}$$

$$I = (H\ XOR\ (m_9 OR\ NOT\ m_{10})) \tag{6}$$

Where, $F,\ G,\ H\ and\ I$ are one-way functions and m_0 to m_{10} are integers modulo 256 into which the message has been divided. The one-way functions used in SecureMD5 (F, G, H, I) are mathematical components inspired by the MD5 hash algorithm but applied in an entirely different structure and context. SecureMD5 does not use MD5's block hashing model or iterative compression. These functions are used once, on transformed contextual values, to generate a byte that alters the plaintext in a one-time, non-repeatable way (Eq. (7)).

$$f(p, frn, k) = byte(I\ mod\ 256) \tag{7}$$

Where, the byte corresponding to modulo 256 of the result obtained in function I is calculated (Eq. (6)). Finally, the byte is encrypted or decrypted by adding or subtracting the byte returned by the hash function, as shown in Eqs. (1) and 2. Algorithms 1 and 2 correspond to the encryption and decryption algorithms for each byte of text, respectively.

3.2. *Key generation with mediapipe and opencv*

In recent years, generation secure key has been the goal of our research [9,10,48]. Challenges are classified by goals or equivalence groups, such as the identification of the user, place, device, date, or time of access to sensitive information [7]. Specifically, in this paper we present the development of two challenges that correspond to the equivalences set of user identification, these are hand recognition and face recognition.

In this research we study the MediaPipe framework and the OpenCV library for the development of hand recognition and face recognition challenges in Python, respectively:

MediaPipe: is a framework available for different platforms such as Android, iOS, $C++$, JavaScript, and Python. This framework contains AI solutions for face detection and tracking of hands, iris, facial gestures, and body postures, among others. This paper uses the version available for Python, which provides solutions for hand detection and location of points of interest. The MediaPipe hands method contains the pre-trained AI model mp.solutions.hands allows detecting the palms of hands in an image or video. When palms are detected in the whole image, the handlandmarks model is applied for 21 landmarks localization by locating 21-coordinates 3D within the detected hand regions. It has the capability of direct prediction of the coordinates through regression algorithms [33]. In this study, these points are used to calculate distances that ensure biometric security. Captured images are deleted immediately after processing, preventing reuse or spoofing.

OpenCV: is an open-source library published under a Berkeley Software Distribution license, which allows image processing with AI. With the OpenCV library, it is possible to recognize faces using models: Eigenfaces, Fisherfaces, and Local Binary Patterns Histograms FaceRecognizer (LBPHFaceRecognizer) [43]. In this study, the LBPHFaceRecognizer model was used as a predictive model for face recognition. This model is based on comparing each pixel of an image with the surrounding pixels (neighbors). Taking each pixel as the center and assigning a threshold as follows: If the intensity of the central pixel is $\geq$ than its neighbor, a 1 is assigned and 0 otherwise. The algorithm is basically the concatenation of local histograms generated from each LBP region extracted from the image. Recognition is performed using a nearest neighbor classifier in the feature space calculated with Chi-square as the dissimilarity measure [49]. In this study, LBPHFaceRecognizer model from OpenCV was employed for face recognition. This method analyzes local binary patterns, processes data in real time, and deletes images after the training phase to enhance security against misuse.

The operation of the proposed challenges is described below. Both challenges require an initial data collection process for their correct operation. The initial process is only executed at the time of system implementation or if external conditions change, e.g. in the case of a change of authorized user, usual place of work or capture devices. To access confidential information, only the challenge is executed. In practice, only the data of the authorized user will be available and not of another (unauthorized) user who eventually wants to access the information. For this reason, both challenges have been developed and tested for one known user at a time.

3.2.1. *Recognition hand challenge*

The development of the hand recognition challenge proposed in this study is described in Fig. 3. In the initial process of this challenge, the user enters his or her name and a directory is created with the name entered by the user. Subsequently, the user is asked if has a scanner connected to the computer to capture images of the palm of his right hand. If the user answers yes, they are asked to save three images of their scanned hand in a predetermined location, these images are automatically deleted after obtaining their data. After storing the images, the distances between the points of interest extracted from the coordinates provided by MediaPipe's hand_landmarks.landmark function are calculated (see Fig. 4). Then, the averages of the distances of the three images are calculated and stored in a csv extension file. The name of the authorized user and his threshold ($\cup$) are also stored in the file. The threshold, is the result of calculating the maximum Root Mean Square Error (RMSE) between the mean of the distances of the three images and the distance of each of the images, as seen in Eq. (8).

$$RMSE(d, \mu) = \sqrt{\frac{\sum_{i=1}^{i=10} (d_i - \mu_i)^2}{10}} \tag{8}$$

Where d is the distance calculated on the image taken instantly and μ is the distances mean corresponding on the three images taken to create the database.

In the challenge execution phase, the user is asked if they have a scanner connected to the computer. If the user answers yes, they are asked to scan the right hand and save the image to a specified location. The image saved in that location will be automatically deleted once the challenge has been executed, to ensure that the user performs the operation every time it is necessary to execute it, thus reducing the risk of image plagiarism. The challenge execution process is described in Algorithm 3. The challenge will return a result ($rmse$), as shown in Eq. (9),

$$rmse = \begin{cases} 0,\ if\ RMSE = NULL \\ 1,\ if\ RMSE(d, \mu) \leq\ \cup \\ [2\ \leq rmse \leq 12],\ if\ RMSE(d, \mu) >\ \cup \end{cases} \tag{9}$$

Where $\cup$ is a threshold obtained from the maximum RMSE of the mean distances (μ) and the distances (d) of each image used to obtain μ. The result of $rmse$ will be 0 if the challenge cannot be executed for any

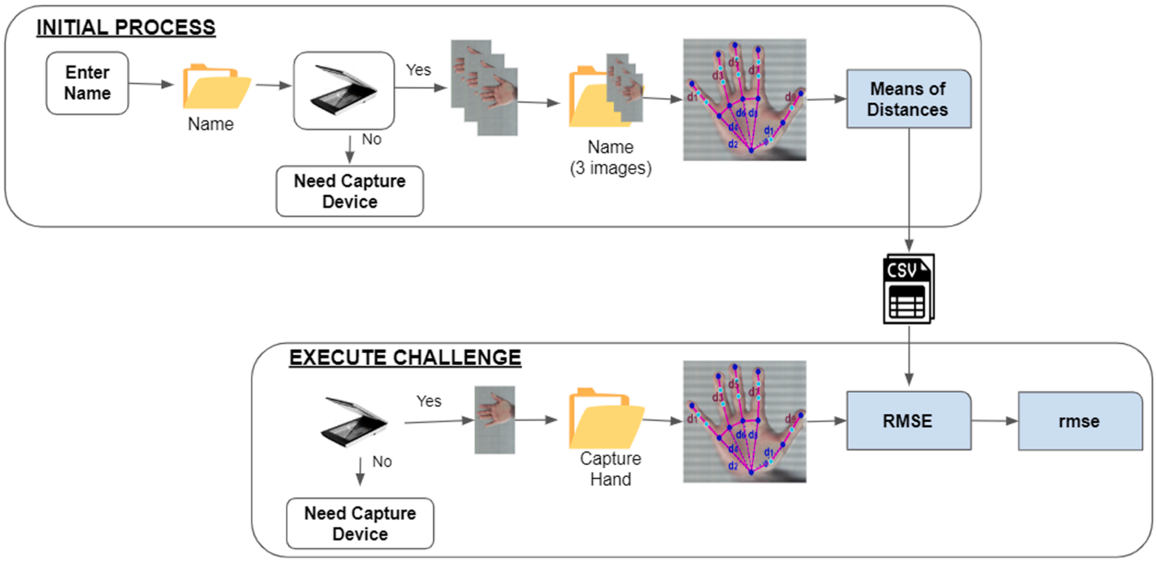


Fig. 3. Operating diagram of the hand recognition challenge.

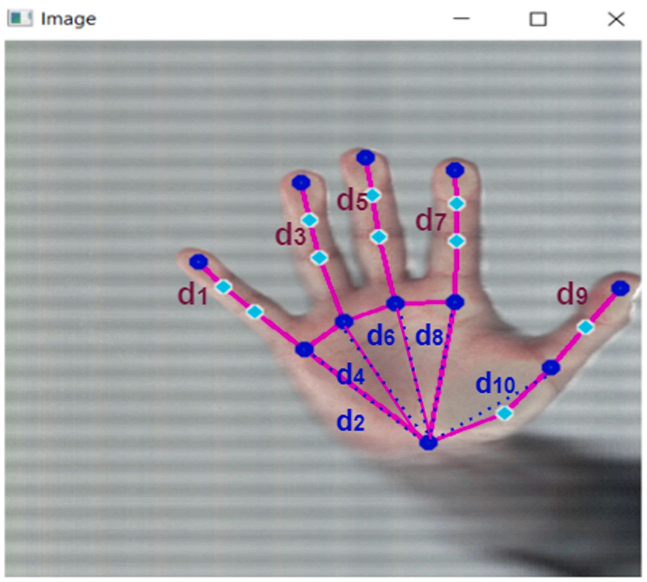


Fig. 4. Scanned hand image, points, and distances (d) of interest.

reason. If RMSE is less than ∪, the user is assumed to be an authorized user, returning a *rmse* equal to 1. If RMSE is greater than ∪, the user attempting to access is not the authorized user and the challenge will return a *rmse* equal to a value in the range between 2 and 12.

3.2.2. *Recognition face challenge*

The face recognition challenge requires an initial process of database creation and model training, as shown in Fig. 5. In the initial process, a directory is created with the name of the authorized user containing 300 images. The images are captured via a video stream taken by the computer webcam or stored in a path linked to the mobile phone via Bluetooth. The LBPHFaceRecognizer model is trained with the images stored in the previously created database directory. For classification, the name of the directory is taken as the label of the images. The model can be trained with one or several users, all trained users will be authorized users.

During the execution phase of the challenge, an image of the user who is going to access the information is captured and passed to the predictive model. The model returns a confidence level parameter, which will be within the range of 0 to 70 if it is one of the trained faces and the name of the recognized user. If the model does not know the identified face, instead of the name it will display the word "Unknown" and a confidence level outside this range.

The result of the challenge is described in Eq. (10),

$$resp = \begin{cases} 0, \ if \ res = NULL \\ 1, \ if \ res \le \ 70 \\ [2 \ \le resp \le 12], \ if \ res > \ 70 \end{cases} \tag{10}$$

Where *resp* is the response of the challenge and *res* is the value of the confidence level parameter obtained from the LBPHFaceRecognizer model. The challenge returns 0 if it could not be executed for some reason (e.g. lack of camera and linked mobile phone), 1 if the confidence value is in the range of 0–70 or a value within the range of 2 to 12 if the

Algorithm 3
Recognition Hand Execute Algorithm.

```
Recognition Hand Execute Process
  1. Check availability of the device required to run the challenge:
    if $device = True$ :
      1.1. The captured hand image is passed to the hand_landmarks.landmark function to locate the points on the image and obtain the coordinates.
        Coordinates = $hand\_landmarks.landmark\ (captured\_hand)$
      1.2. The distances (d) between the coordinates of the points of interest shown in Fig. 4 are calculated.
      1.3. $RMSE$ is calculated as described in Eq. (8).
      1.4.The result of the challenge ($rmse$) obtained from Eq. (9) is returned.
        Return $rmse$
  2. if $device = False$ :
        Return "Need capture device."
  End Process
```

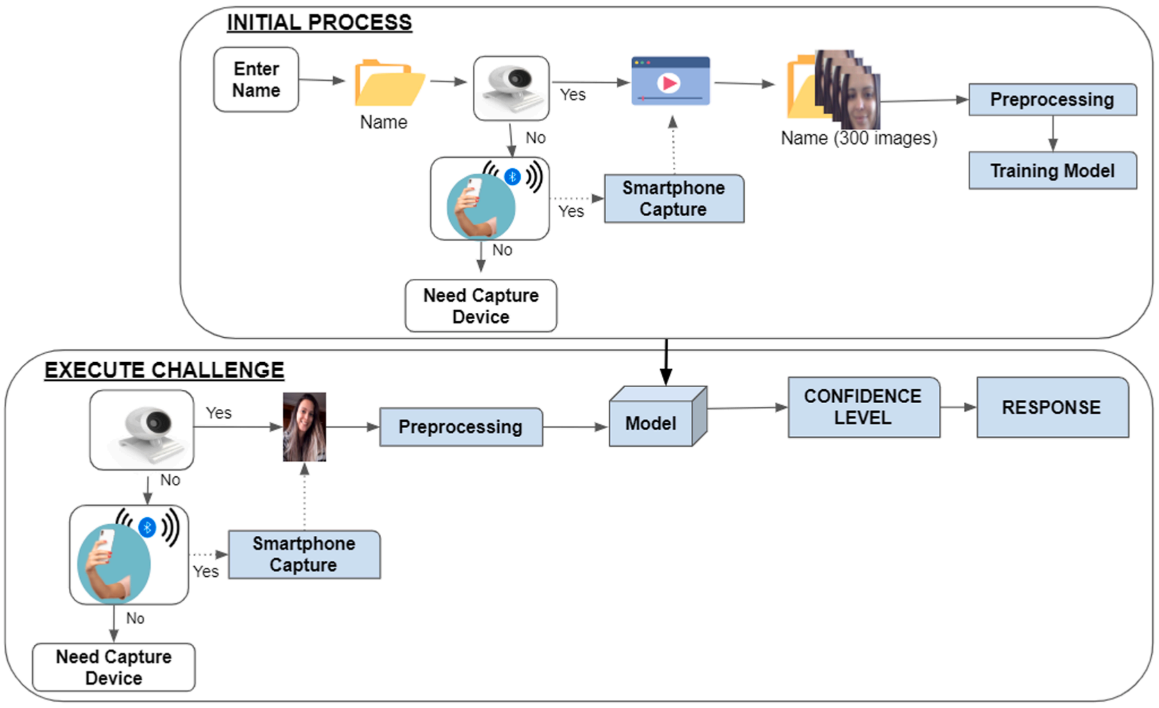


Fig. 5. Operating diagram of the facial recognition challenge.

confidence level is greater than 70. The challenge execution process is described in Algorithm 4.

3.2.3. *Key generation process with equivalent challenges*

An equivalence group is a set of challenges that result in the same subkey value and are therefore interchangeable. The equivalent challenges or those comprising equivalence groups, can be executed in sequential order, or independently. The mechanism used to establish the equivalence between the challenges, is the quantification of the results obtained from the execution of both challenges in 13 intervals (See Eq. (9) and 10). This quantification also allows generating subkeys with a wide cardinality that allows generating a long key without the need of executing as many challenges as the number of bits in the key. The cardinality of the challenge refers to the number of bits that each challenge contributes to the composite key. In this case, each proposed challenge provides 4 bits equivalent to 13 possible results.

Algorithm 5 describes the key generation process with equivalent challenges. If the challenges are in equivalent mode, when one fails due to lack of the necessary device for its execution, the next one is executed. As show in Fig. 6, the init() method will indicate whether the challenge has been successfully executed or if another equivalent challenge needs to be executed. In the execute() function (Algorithms 3 and 4), the challenge data is calculated and passed to "key generation", to generate the key that will be passed to the encryption algorithm. The generated subkeys and keys are never stored but are calculated at run time to ensure greater security.

4. Experiments and validation

This section describes the data used in the experiments and the validation of the recognition methods studied. Furthermore, the diffusion and confusion validation of the proposed encryption algorithm is described. In this study, the diffusion and confusion of the encryption algorithm is validated with the same methods used in the previous study [7] for this purpose. In this way, the result obtained by both algorithms is compared. The experiments were performed on a computer with the following features: Windows 10 operative system, Intel(R) Core (TM) i7–10750H CPU @ 2.60 GHz 2.59 GHz, and 16GB of memory (RAM).

4.1. *Dataset*

Small data sets were used to develop the proposed solutions. The detection models, both for hands and faces, are already pre-trained and this makes a large dataset for training unnecessary.

The hand recognition scheme is designed to obtain the data from three images of the authorized user for feature extraction. The dataset used in the validation of the hand challenge contains 40 images of the right hands as shown in Fig. 7. The images correspond to 5 female users in the age range of 30 to 45 years, obtained with a common office scanner. The requirements requested of each person for capturing the

Algorithm 4
Recognition Face Execute Algorithm.

```
Recognition Face Execute Process
  1. Check availability of the device required to run the challenge:
       if $device = True$:
       1.1. The captured face image is passed to the previously trained model.
          $res = LBPHFaceRecognizer\ (face\_capture)$
       1.2. The result of the challenge ($resp$) obtained from Eq. (10) is returned.
          Return $resp$
  2. if $device = False$ :
          Return “Need capture device.”
  End Process
```

Algorithm 5
Key Generation Algorithm.

```
Key Generation Process
  Check that the challenges are in equivalent mode.
  1. if $mode\_equivalent = True$ :
    1.1. init(). The availability of the devices is checked in each challenge:
    if Recognition Hand Execute Process () $\neq$ "Need capture device":
        $SubKey_1 =$ Recognition Hand Execute Process ()
      else if Recognition Face Execute Process () $\neq$ "Need capture device":
        $SubKey_1 =$ Recognition Face Execute Process ()
      else:
        $SubKey_1 =$ NULL
  2. if $mode\_equivalent = False$:
      2.1. The availability of the devices is checked in each challenge:
      If Recognition Hand Execute Process () $\neq$ "Need capture device":
        $SubKey_1 = Recognition\ Hand\ Execute\ Process()$
      else:
        $SubKey_1$=NULL
      If Recognition Face Execute Process () $\neq$ "Need capture device":
        $SubKey_2 = Recognition\ Face\ Execute\ Process()$
      else:
        $SubKey_2$=NULL
  3. The number of challenges executed (CC) is counted and non-NULL subkeys are checked to generate the key:
      for $i = 1$ to $i = CC$
        if $SubKey_i$=NULL
          $Key\_Generation$= concatenate ($Key\_Generation$, $SubKey_i$)
  4. Return $Key\_Generation$
  End Process
```

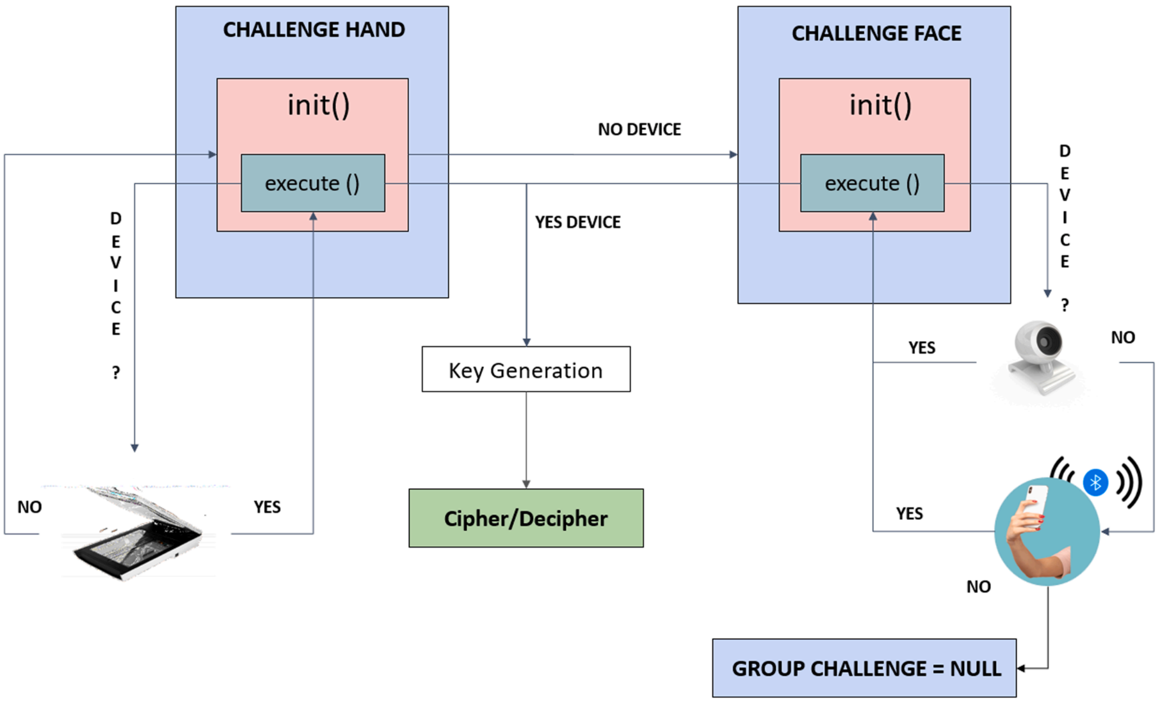


Fig. 6. Operation mode of the proposed challenges.

images were: 1) place the whole right hand and approximately 5 cm of the forearm corresponding to the wrist on the scanner glass, 2) extend the fingers and 3) place the hand in a comfortable position, such as the one that the user will adopt when identifying himself.

In the face recognition scheme, a streaming video of the authorized user's face is made, or the information is taken from a video stored in the directory where the captures made by the mobile phone are stored. Subsequently, the training file captures 300 images of the video to train the facial recognition model. A prerequisite for obtaining the results of this research is to perform the video for training the model in the workplace and with the usual lighting. Once the model is trained, the captures and the stored video are deleted if necessary. The tests performed with the facial recognition scheme involved 4 women and 1 man in an age range between 30 and 40 years.

4.2. Recognition methods validation

For the validation of the hand challenge, the characteristics of three images of the authorized user and the U are obtained for each of them in the initial process. The data obtained are taken by the challenge to calculate the RMSE between the distance means of the three images used in the initial process and the distances of 37 new images, corresponding to 4 unknown persons and the authorized user. The metrics used for the evaluation of the challenge were Precision (p), Accuracy (Acc), False

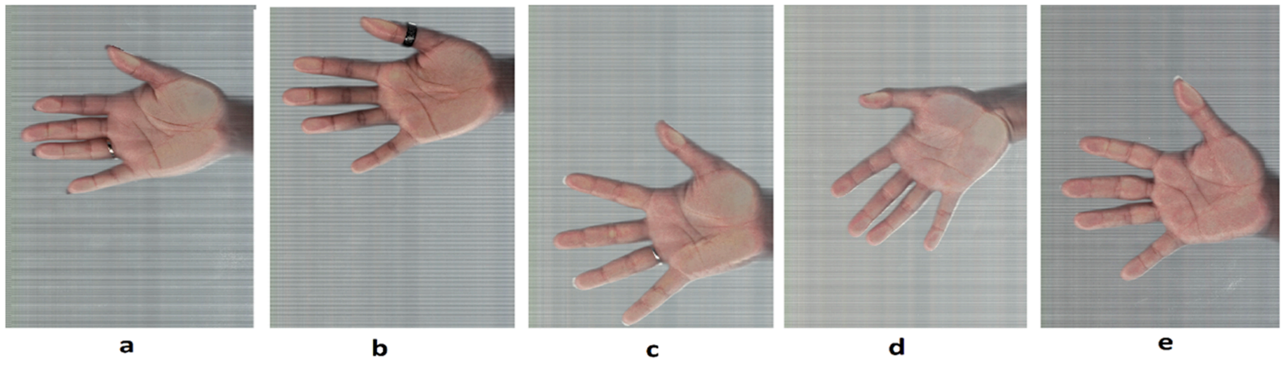


Fig. 7. Images of hands obtained by scanning.

Acceptance Rate (*FAR*) and *Specificity*, as shown in Eqs. (11)–(14), respectively. The *p* indicates the percentage of successes that the challenge has in classifying the true authorized user among all those classified as authorized users. The *Acc* indicates the accuracy of the challenge in classifying authorized and unauthorized users. The *FAR* indicates the rate of unauthorized users that the challenge classifies as authorized users, and the *Specificity* or true negative rate establishes the proportion of unauthorized users that are well detected. *Specificity* is of special importance for this study, because if the challenge has a high degree of *Specificity,* it will not allow an unauthorized user to access the information, this being the main objective of the group of equivalences of the proposed challenges.

$$p\ (\%) = \frac{VP}{VP + FP} * 100 \tag{11}$$

$$Acc\ (\%) = \frac{VP + VN}{(VP + FP + FN + VN)} * 100 \tag{12}$$

$$FAR\ (\%) = \frac{FP}{(VP + FP + FN + VN)} * 100 \tag{13}$$

$$Specificity\ (\%) = \frac{VN}{(VN\ +\ FP)} * 100 \tag{14}$$

Where, *VP* is the number of images that the challenge declares as authorized user and that are actually authorized user. *VN* is the number of images that the challenge classifies as unauthorized user and that are actually unauthorized users. *FP* is the number of images that the challenge declares as authorized user and that are actually not authorized users. *FN* is the number of images that the challenge detects as unauthorized users and that are actually authorized users.

Table 1 shows the results obtained from the confusion matrix with the tests performed on each user and the threshold to obtain the *rmse* value. It can be observed in the results mean values of $p = 80$ %, $Acc = 92$ %, $FAR = 3.6$ % and $Specificity = 95$ %. In this sense, it can be seen that the results of this method have been generally satisfactory. However, the precision in one of the users studied did not exceed 60 % because it has similar characteristics to other users. Therefore, the use of this identification method is recommended as a complement to other methods to guarantee security. In this study it is considered valid because the challenges are executed in groups to generate a key. That is, this method is used together with other identification methods that allow generating a secure key.

The hand recognition challenge had an execution time, including user interaction, of approximately 6.2 s. Image scanning time was not included, the test was done with the image stored in the capture directory prior to execution.

For the validation of the face recognition challenge, the model was trained 4 times with an authorized user each time. After training, the challenge was executed with the images of 5 unknown users and the authorized user. The metrics used to validate this approach were the same as those used to validate the hand recognition approach. The model had *p, Acc* and *Specificity* results of 100 %, with no false acceptances. The model allows to recognize in the same image an authorized user and an unknown user. When an authorized user and unknown users are recognized in the same image, the challenge returns a correct identification result. If the other challenges that are part of the key return correct results it is assumed that the other people in the image taken are part of the authorized user's work team. The execution time of the challenge including the interaction with the user and the capture of the image using the computer is approximately 7 s.

4.3. Diffusion validation

A good diffusion allows to protect the encryption algorithm from insider threat attacks [7]. To ensure the diffusion of the encryption algorithm, the *frn* parameter was introduced during the document marking process [7]. The evaluation of the SecureMD5 diffusion was performed with plain files filled with zeros (2; 8; 26 and 260 bytes), 100 keys (8 bytes) and 512 different values of *frn* (9 bits). To measure the diffusion and avalanche effect in SecureMD5, the program [50] described in Algorithm 5 of the study was used [7]. The test consists of calculating the MSE between expected distinct files and generated distinct files in the encrypted versions of small files. As many distinct files should be generated as the number of different *frn* values used for the test, in this case 512 for each key in each file.

Fig. 8 compares the MSE results for small files tested with SecureMD5 and Securecipher. Securecipher consistently achieves significantly lower MSE values, indicating superior diffusion across all file sizes. While SecureMD5 shows higher MSE values, reflecting lower diffusion for

Table 1
Results obtained from tests performed on the hand recognition challenge.

User	∪	VP	FP	VN	FN	***p***(%)	***Acc***(%)	FAR (%)	***Specificity***(%)
1	5.7	4	3	27	3	57	84	8	90
2	14.7	1	0	35	1	100	97	0	100
3	9.06	7	2	26	2	78	89	5	93
4	11.06	4	2	31	0	66	95	5	94
5	3.09	2	0	33	2	100	95	0	100

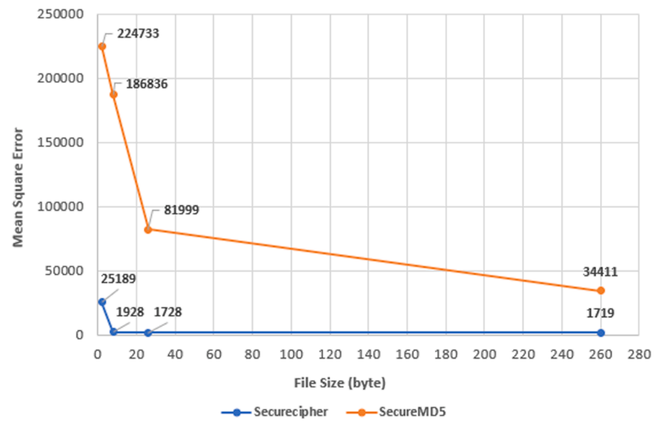


Fig. 8. Analysis and comparison of the diffusion of SecureMD5 and Securecipher algorithms in files of 2; 8; 16 and 260 bytes (B).

smaller files, it demonstrates an improvement as file sizes increase, though it remains less diffused compared to Securecipher.

4.4. Confusion validation

The NIST 800–22 randomization test was used to validate the confusion mechanism in SecureMD5. The results obtained, shown in Table 2, indicate that SecureMD5 achieved satisfactory outcomes in 87 % of the 15 tests performed, matching the performance of the Securecipher algorithm. This parity demonstrates that SecureMD5 meets the same randomization standards as a validated encryption algorithm under identical input conditions.

While a 13/15 pass rate might appear to suggest room for improvement, prior research [51] establishes that a true pseudo-random number generator has an 80 % probability of failing at least one of the 15 tests. Therefore, the 87 % success rate achieved by SecureMD5 exceeds this threshold and validates its capability to generate sufficiently random sequences for secure cryptographic operations.

These results confirm that SecureMD5 is well-suited for real-world applications requiring high randomization, such as protecting SFS from insider threats and resisting cryptographic attacks based on predictable patterns.

5. Security and performance analysis of securemd5

SecureMD5 is analyzed in this chapter by focusing on two key aspects: its security and performance. Building on the methodologies established in Securecipher [7], SecureMD5 is evaluated for its capabilities in mitigating insider threats through improved diffusion, confusion, and entropy, as well as its performance metrics such as encryption speed and resource consumption. The following sections provide a detailed analysis of these aspects.

5.1. Security analysis

This study directly builds on the methodologies established in Securecipher [7], leveraging its validated framework for diffusion and confusion evaluation. Securecipher provided critical insights into the requirements for encryption algorithms in SFS, particularly regarding insider threats. In this sense, the main attacks provoked by insiders are given by the knowledge of the plaintext, in this case the insider has knowledge of the plaintext and its ciphertext version. As well as the ability to choose plaintext, which allows the insider to reproduce versions of ciphertext as many times as desired. For this problem, the parameter *frn* has been introduced in the encryption algorithm, which makes the encrypted version of a file different each time it is encrypted. Although in the validation test of the diffusion, relatively high MSE

Table 2
NIST 800–22 test results applied to SecureMD5 and comparative with Securecipher. Own source.

TEST	n	M	STREAMS	Securecipher STATUS	SecureMD5 STATUS
Frequency (Monobit) Test	*1000*	–	*25*	✓	✓
Frequency (Block) Test	*2000*	*20*	*15*	✓	✓
Runs Test	*1000*	–	*15*	✓	✓
Longest Run of Ones in a Block Test	*2000*	–	*15*	✓	✓
Binary Matrix Rank Test	*128*	–	*15*	✓	✓
Discrete Fourier Transform (Spectral) Test	*2000*	–	*10*	✓	✓
Non-overlapping Template Matching Test	*2000*	–	*25*	✓	✓
Overlapping Template Matching Test	*20,000*	*10*	*500*	*101* ✓ *18* ×	*98* ✓ *21* ×
Maurer's Universal Statistical Test	*6500*	*9*	*25*	✓	✓
Linear Complexity Test	–	–	–	×	×
Serial Test	*2000*	*3*	*10*	✓	✓
Approximate Entropy Test	*1000,000*	–	*1000*	✓	✓
Cumulative Sums Test	*1000,000*	–	*1000*	✓	✓
Random Excursions Test	*2000*	*8*	*20*	✓	✓
Random Excursions Variant Test	*1000,000*	*500*	*25*	✓	✓

values were obtained for small files compared to the Securecipher algorithm, it was found that in large files it is significantly reduced. This leads to take measures in the SFS not to use this algorithm for the encryption of small files.

The high potential of the SecureMD5 algorithm has also been proven in terms of entropy (see Fig. 9). This aspect is very important in encryption algorithms, a good entropy makes the algorithm resistant to cryptanalysis attacks focused on the frequency with which words are repeated in the text. Fig. 9 presents the entropy comparison among plaintext, SecureMD5, Securecipher, and RC4. The test consists of counting the binary word frequencies of the plain file and the binary words generated in the encrypted versions with each algorithm. The Spanish version of "*Don Quijote*", 128 MB in size, and the SecureMD5, Securecipher and RC4 encryption algorithms were used as the plain file.

Fig. 9 shows that SecureMD5 has a much more normalized entropy than both the plaintext and the encrypted version generated by the Securecipher algorithm, being more like the entropy of the version generated by the conventional RC4 algorithm. The SecureMD5 encrypted version generated a vocabulary of 65,468 words, which exceeds the plaintext vocabulary by 64,388 words. In this context, vocabulary refers to the number of unique binary words generated in the ciphertext. A larger vocabulary indicates a more uniform distribution, which improves resistance against cryptographic attacks based on frequency

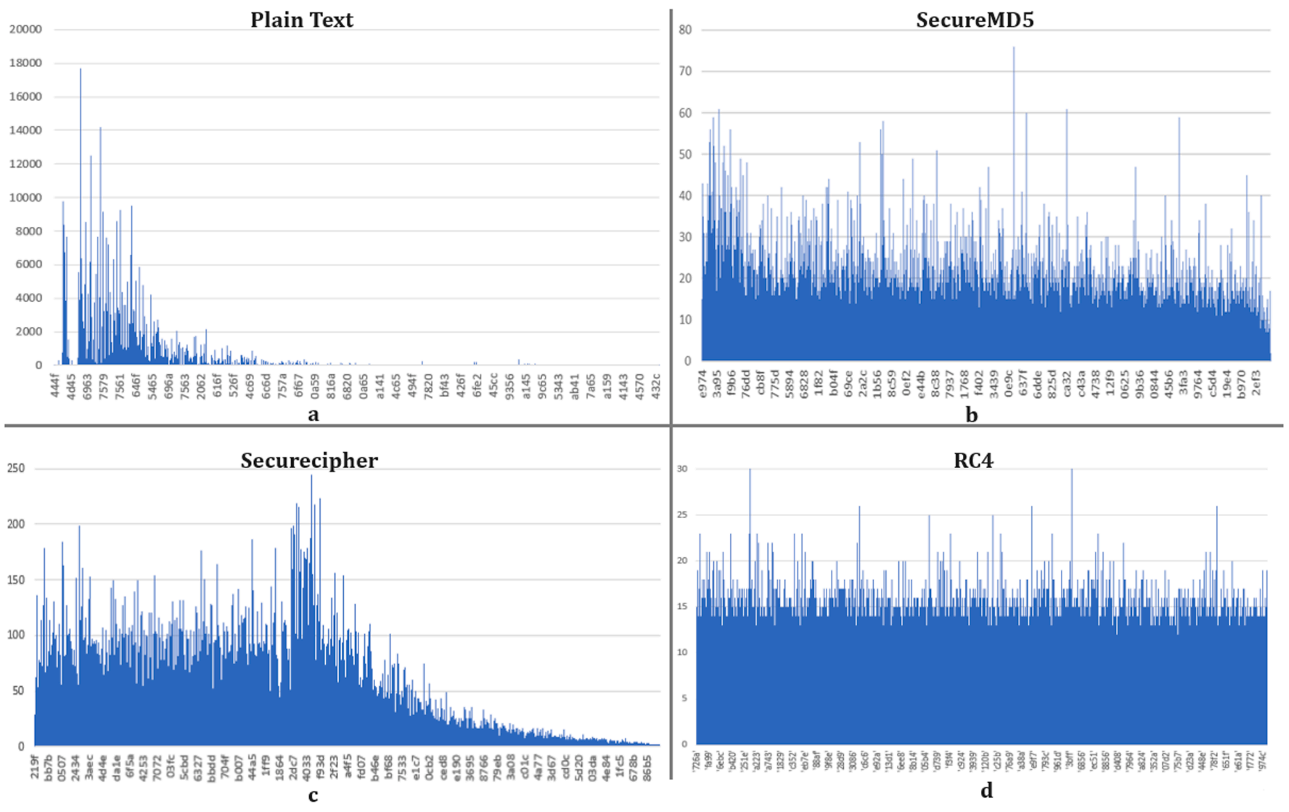


Fig. 9. Comparison of a) Entropy of plaintext. b) Entropy of ciphertext with SecureMD5. c) Entropy of ciphertext with Securecipher and d) Entropy of the ciphertext with RC4.

analysis.

The vocabulary generated by Securecipher is 45,152, a 31 % improvement in the vocabulary generated by SecureMD5 over Securecipher. This result highlights SecureMD5 ability to generate a more uniform and random distribution of binary words, significantly improving its resistance to cryptographic pattern analysis. While SecureMD5 improvement 0.32 % over RC4 in entropy may seem minor, it demonstrates that SecureMD5 matches and slightly exceeds the high randomness standards of RC4. In addition, SecureMD5 incorporates advanced features such as diffusion and insider threat resistance, offering a more complete cryptographic solution.

In insider threat scenarios, attackers often have access to both the plaintext and its corresponding ciphertext. To address this challenge, SecureMD5 incorporates a bitwise operation mechanism that introduces collisions. This means that two different elements (or characters) of the plaintext can produce identical corresponding encrypted output elements. Similarly, two identical elements can produce different encrypted output elements. This occurs due to the introduction of dynamic contextual elements into the encryption algorithm. These collisions allow breaking direct correlations between plaintext elements and their encrypted counterparts, making the ciphertext more resilient to known plaintext attacks.

Furthermore, SecureMD5 ensures that the same file, when encrypted multiple times, produces completely unique ciphertext outputs for each encryption instance. This is achieved by using dynamic contextual parameters that influence the encryption process. These parameters, such as user identity, access time, and file-specific attributes, alter the encryption state, ensuring high variability and unpredictability in the outcome.

This combination of collisions and contextual variability directly enhances the diffusion and confusion of the algorithm. Small changes in the plaintext or contextual parameters lead to significant and unpredictable variations in the ciphertext. This design not only complicates brute force and known-plaintext attacks, but also ensures that insider attackers cannot reliably use their knowledge of plaintext and ciphertext pairs to reverse engineer or predict future encryption outcomes. These features validate SecureMD5's suitability for protecting sensitive data in environments prone to insider threats.

While SecureMD5 does not include a formal proof of resistance to classical cryptanalytic models (e.g., IND-CPA or differential cryptanalysis), such models are not fully applicable to the specific adversarial context addressed in this work. In scenarios involving insider threats, the attacker may have access to plaintext and ciphertext but lacks the ability to control or reproduce the contextual parameters (p, frn, k) that govern the encryption process. These parameters are dynamically generated and non-reusable, breaking the assumptions required for structured input manipulation. This limitation of formal analysis in insider threat models has been previously discussed in the literature [7], where the Securecipher algorithm followed a similar design philosophy and evaluation methodology. Consequently, SecureMD5's security is assessed through its ability to disrupt inference patterns, ensure contextual entropy, and demonstrate non-deterministic encryption behavior, as validated by statistical tests and entropy-based diffusion metrics.

5.2. *Performance analysis*

To calculate the performance of SecureMD5, the encryption and decryption time of 0.001;1;10 and 128 megabyte (MB) files was measured. Fig. 10 illustrates the performance comparison between SecureMD5 and Securecipher across different file sizes (0.001, 1, 10, and 128 MB). Fig. 10 shows that SecureMD5 is more expensive as the file size increases. This is because SecureMD5 performs more operations than Securecipher, which makes SecureMD5 more secure in terms of confusion, but affects its performance. However, compared to the algorithm proposed in the study [25], SecureMD5 presents better performance. In

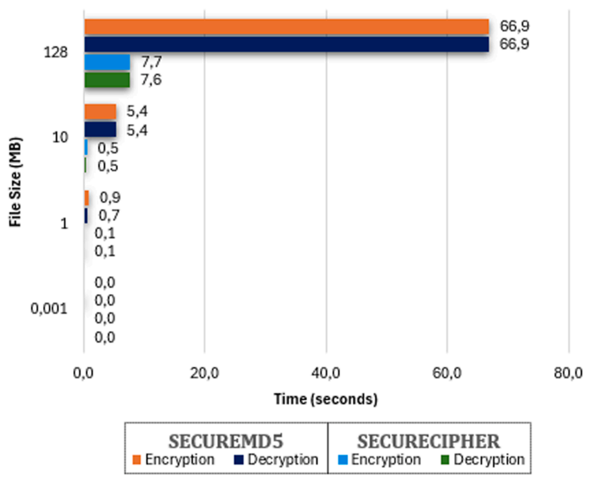


Fig. 10. Performance comparison of SecureMD5 and Securecipher for 0.001;1;10 and 128 MB files.

the study [25], a Modified AES algorithm is proposed, which takes 7.2 s and 8.2 s to encrypt and decrypt, respectively, an 8 MB file. If we consider that SecureMD5 takes 5.4 s for both operations for a 10 MB file, it is shown that SecureMD5 is at least 25 % faster. Additionally, the performance of SecureMD5 and Securecipher were compared in terms of resource consumption during encryption and decryption operations for a 128 MB file. Both algorithms operate bit by bit and had a maximum memory usage of 377 MB. However, SecureMD5 exhibited a higher CPU usage of 6.16 %, compared to the 1.69 % CPU usage of Securecipher. This difference is attributed to SecureMD5 enhanced confusion mechanisms. Involving more computational steps to achieve higher data diffusion, in contrast to Securecipher simpler approach.

6. Strengths, limitations and future work of securemd5 in SFS

SecureMD5 was developed as a robust encryption algorithm to address the specific challenges posed by insider threats in SFS. This section evaluates its strengths, limitations, and potential avenues for future improvement. The analysis highlights how SecureMD5 enhances entropy, confusion, and adaptability while acknowledging areas requiring optimization, particularly in resource usage and diffusion for small files. Table 3 provides a comparative overview of the performance metrics of SecureMD5 and Securecipher [7], the Modified AES algorithm proposed in [25], and RC4. Table 3 highlights key differences in entropy, encryption time, diffusion, and resource consumption, illustrating SecureMD5 strengths and trade-offs in addressing specific challenges of insider threats and SFS.

Table 3
Comparative Summary of SecureMD5, Securecipher, AES, and RC4.

Metric	SecureMD5	Securecipher [7]	Modified AES [25]	RC4
Entropy (Generated Vocabulary)	65,468 words	45,152 words	Not evaluated	Like SecureMD5
Encryption Time (10 MB)	5.4 s	0.5 s	7.2 s	Not evaluated
Diffusion (MSE, Small Files)	Lower diffusion (Higher MSE)	Higher diffusion (Lower MSE)	Not evaluated	Not evaluated
CPU Usage (128 MB)	6.16 %	1.69 %	Not evaluated	Not evaluated
Memory Usage (128 MB)	377 MB	377 MB	Not evaluated	Not evaluated

6.1. Strengths

SecureMD5 introduces key advancements for SFS. It improves entropy, generating 65,468 binary words in a 128 MB file, a 31 % increase over Securecipher which generated 45,152 words, as seen in Table 3. This ensures higher randomness and better resistance to frequency-based cryptanalysis. Furthermore, SecureMD5 demonstrates competitive performance for medium-sized files, with encryption times 25 % faster than the Modified AES algorithm for files of comparable size, as indicated in Table 3.

Table 4 provides a detailed comparison of SecureMD5 and MD5, highlighting SecureMD5′s advancements in critical cryptographic aspects. SecureMD5 leverages dynamic contextual parameters, including keys, positions, and random numbers, to improve unpredictability and generate different ciphertext outputs for identical plaintext files. Unlike MD5′s block-based processing, SecureMD5 works on a bit-by-bit basis. This allows precise access to specific file positions without decrypting the entire file. This feature is crucial for SFS.

Table 4 also illustrates SecureMD5′s superior resistance to brute force attacks, achieved through improved diffusion and confusion. Small changes in the plaintext or contextual parameters produce significant and unpredictable variations in the ciphertext, as validated by diffusion and confusion tests. In addition, SecureMD5 strategically handles collisions to alter correlations between plaintext and ciphertext. This increases ambiguity, in contrast to MD5′s vulnerability to collision attacks.

Finally, SecureMD5 demonstrates adaptability and robustness through its expanded set of processing variables (e.g., A, B, C, E, L, N). Designed specifically to counter insider threats, SecureMD5 offers a modern encryption framework that overcomes MD5′s limitations in SFS-critical scenarios.

6.2. Limitations

SecureMD5 presents limitations that impact its performance in specific contexts. One notable limitation is its suboptimal diffusion for small files, as evidenced by the higher MSE values compared to Securecipher, as shown in Table 3. This indicates lower diffusion efficiency, making SecureMD5 less suitable for applications involving small datasets, where Securecipher has a higher diffusion provides a distinct advantage. Additionally, SecureMD5 consumes significantly more CPU resources than Securecipher, with 6.16 % usage for a 128 MB file compared to 1.69 % for Securecipher, as reported in Table 3. This higher resource

Table 4
Comparative Analysis of MD5 and SecureMD5 Functional Enhancements and Adaptations for SFS. Own source.

Aspects Evaluated	MD5	SecureMD5
Base Function	Unidirectional functions F, G, H, I	Adapted unidirectional functions F, G, H, I
Variables Used	4 (F, G, H, I)	11 variables (A, B, C, E, L, N, etc.)
Operational Structure	Rounds with cross-dependencies	Independent operations without rounds
Use of Contextual Parameters	Not incorporated	Dynamic parameters (key, pos, frn)
Resistance to Brute Force	Limited by 128 bits	Improved through high diffusion and confusion
Collisions	Inherent vulnerability due to collisions attacks	Collisions introduced to enhance ambiguity and disrupt attacks
Capability to Access Specific Positions	Not applicable	Allows access to specific locations in any encrypted file instantly without decrypting the entire file
Text Processing	Operates on complete blocks	Bit-by-bit, adapted to encryption needs
Application to Insider Threat Scenarios	Not designed	Specifically designed for insider threats

consumption is mainly due to the enhanced confusion mechanisms incorporated in SecureMD5. Involving more computational steps to achieve high cryptographic strength. This contributes to its security but limits the applicability of the algorithm in resource-limited environments, such as IoT devices or mobile systems. Furthermore, while SecureMD5 demonstrates competitive performance for medium-sized files, outperforming the Modified AES algorithm with 25 % faster encryption times for 10 MB files, its encryption and decryption times increase significantly for larger datasets. This is due to its bit-by-bit operational structure, which, while beneficial for precision and security, introduces computational overhead that hinders scalability for extensive datasets. These limitations underscore the need for targeted optimizations to improve SecureMD5 diffusion for small files, reduce its resource consumption, and enhance its performance for large-scale data encryption.

6.3. Future work

Future work will focus on addressing its identified limitations and expanding its applicability to a broader range of environments. Key efforts will include optimizing diffusion mechanisms to improve performance for small files by reducing RMSE values and exploring the integration of diffusion strategies used in Securecipher. Additional research will aim to refine the computational structure algorithm to balance security and resource consumption, enabling its use in resource-constrained environments such as IoT devices and mobile systems. Performance evaluations will extend to diverse hardware setups, including devices with limited computational capabilities, to ensure adaptability across different operational scenarios. Furthermore, hybrid encryption approaches will be investigated, combining the strengths of SecureMD5 in entropy and confusion with Securecipher efficiency in diffusion. These advancements aim to position SecureMD5 as a versatile and scalable solution for SFS facing insider threats. Also, we proposed implementing the system in real scenarios will validate its practicality and effectiveness in diverse environments. As well as to continue investigating techniques and technologies to mitigate the risk of data leakage caused by insider threats.

7. Conclusions

This study proposes a mechanism based on a new encryption algorithm combined with a contextual key generation process to meet the requirements of internal threats. In this sense, SecureMD5 improves information security in SFS, especially against internal attacks. The contextual key generation scheme introduced incorporates the generation of biometric keys using AI. This includes the use of the MediaPipe framework for extracting hand geometry features and LBPHFaceRecognizer model from OpenCV for facial recognition of authorized users.

The hand recognition approach achieved an average Accuracy of 92 % and a Specificity of 95 %, while the facial recognition approach reached 100 % Accuracy, correctly identifying authorized users among unknown individuals. The performance evaluation demonstrated acceptable execution times of approximately 6.2 s for hand recognition and 7 s for facial recognition. These methods are recommended as complementary techniques to strengthen security when combined with other approaches. The proposed challenges, belonging to the same equivalence group, can generate subkeys either sequentially or independently, enhancing key flexibility and robustness.

Despite identifying a limitation in diffusion for small files (up to 260 bytes), SecureMD5 stands out for its high confusion, which reinforces its resistance to entropy-related cryptanalytic attacks. Compared to Securecipher, it achieved a 31 % improvement in vocabulary normalization and outperformed RC4 with 0.32 % more vocabulary. Its design, based on the use of MD5 unidirectional functions combined with contextual parameters and an optimized operational structure, ensures fast and efficient access to remote file positions, a critical feature for its integration into SFS. Furthermore, the incorporation of high diffusion and confusion mechanisms strengthens its resistance to brute force and cryptanalytic attacks. This allows establishing SecureMD5 as an effective solution for SFS.

This work addresses critical gaps in information security, significantly improving protection against insider threats. The proposed key generation mechanism ensures dynamic key generation without storage requirements and robust defense against spoofing attempts.

Original article statement

This manuscript is the authors' original work and has not been published or has it been submitted simultaneously elsewhere.

CRediT authorship contribution statement

Isabel Herrera Montano: Writing – original draft, Visualization, Software, Methodology, Investigation, Data curation. **Juan Ramos Diaz:** Software, Methodology, Investigation, Formal analysis, Conceptualization. **Sergio Molina-Cardín:** Writing – review & editing, Validation, Supervision, Methodology, Formal analysis. **Juan José Guerrero López:** Writing – review & editing, Validation, Software. **José Javier García Aranda:** Writing – review & editing, Validation, Supervision, Methodology, Conceptualization. **Isabel de la Torre Díez:** Writing – review & editing, Methodology, Investigation.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Isabel Herrera Montano reports financial support was provided by Santander Bank. If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This research has been partially supported by the Center for the Development of Industrial Technology (CDTI), Spanish Ministry of Science and Innovation under the project named “Secureworld: Technologies for Secure Digital Relationships in a Hyperconnected World”, IDI-20200518.

Additionally, we wants to thank the Spanish Ministry for Science, Innovation and Universities (MICINN), Agencia Estatal de Investigacion (AEI), as well as to the Fondo Europeo de Desarrollo Regional funds (FEDER, EU), under grant number PID2021–122210OB-I00, by MCIN/AEI/10.13039/501100011033 and "ERDF A way of making Europe", European Union, for partially funding this work in the framework of the project “Advanced Artificial Intelligence Techniques to Detect and Combat Unknown and Adversary Cybersecurity Attacks”, within Plan Estatal de Investigacion Cientifica, Tecnica y de Innovacion 2021–2023, Ministerio de Ciencia e Innovacion, Spain.

Isabel Herrera Montano has been funded through the UVa 2022 predoctoral contracts call, co-financed by Santander Bank.

Data availability

The data that has been used is confidential.

References

[1] N.N. Abbas, T. Ahmed, S.H.U. Shah, M. Omar, H.W Park, Investigating the applications of artificial intelligence in cyber security, Scientometrics 121 (2019) 1189–1211, https://doi.org/10.1007/s11192-019-03222-9.

[2] H. Alhindi, I. Traore, I. Woungang, Data loss prevention using document semantic signature, Lect. Notes Data Eng. Commun. Technol. 27 (2019) 75–99, https://doi.org/10.1007/978-3-030-11437-4_7.

[3] N. Liu, Cloud technology in the security management of enterprise document, in: 2011 S International Conference on Innovations in Bio-inspired Computing and Applications, IEEE, 2011, pp. 267–269, https://doi.org/10.1109/IBICA.2011.70.
[4] Z. Shokishalov, H. Wang, Applying eye tracking in information security, Procedia Comput Sci 150 (2019) 347–351, https://doi.org/10.1016/j.procs.2019.02.062.
[5] Holger Schulze. 2023 Insider threat report. 2023.
[6] Holger Schulze. New report reveals insider threat trends, challenges, and solutions. 2024.
[7] I. Herrera Montano, J. Ramos Diaz, J. Javier García Aranda, S. Molina-Cardín, J. José Guerrero López, I. de la Torre Díez, Securecipher: an instantaneous synchronization stream encryption system for insider threat data leakage protection, Expert Syst Appl (2024) 124470, https://doi.org/10.1016/j.eswa.2024.124470.
[8] A. Erola, I. Agrafiotis, M. Goldsmith, S. Creese, Insider-threat detection: lessons from deploying the CITD tool in three multinational organisations, J. Inf. Secur. Appl. 67 (2022) 103167, https://doi.org/10.1016/j.jisa.2022.103167.
[9] Garcia Aranda J.J.A. Ep 2 709 333 A1 European patent application, 2014.
[10] A. García, A. Garcia Moro, J.J. García, J. Roncero, V.A. Villagrá, H. Jalain, Context-based encryption applied to data leakage prevention solutions, in: Proceedings of the 14th International Joint Conference on e-Business and Telecommunications, SCITEPRESS - Science and Technology Publications, 2017, pp. 566–571, https://doi.org/10.5220/0006475205660571.
[11] I. Herrera Montano, J.J. García Aranda, J. Ramos Diaz, S. Molina Cardín, I. de la Torre Díez, J.J.P.C Rodrigues, Survey of techniques on data leakage protection and methods to address the insider threat, Clust. Comput 25 (2022) 4289–4302, https://doi.org/10.1007/s10586-022-03668-2.
[12] P. Chapman, Defending against insider threats with network security's eighth layer, Comput. Fraud Secur. 2021 (2021) 8–13, https://doi.org/10.1016/S1361-3723(21)00029-4.
[13] L. Li, W. He, L. Xu, I. Ash, M. Anwar, X. Yuan, Investigating the impact of cybersecurity policy awareness on employees' cybersecurity behavior, Int J Inf Manage 45 (2019) 13–24, https://doi.org/10.1016/j.ijinfomgt.2018.10.017.
[14] W. Li, W. Meng, L-F Kwok, I.P. HHS, Enhancing collaborative intrusion detection networks against insider attacks using supervised intrusion sensitivity-based trust management model, J. Netw. Comput. Appl. 77 (2017) 135–145, https://doi.org/10.1016/j.jnca.2016.09.014.
[15] Lee C.-.C., Lin T.-.H., Chang R.-.X. A secure dynamic ID based remote user authentication scheme for multi-server environment using smart cards. Expert Systems with applications 2011. https://doi.org/10.1016/j.eswa.2011.04.190.
[16] Mark Stamp RML. Applied cryptanalysis: breaking ciphers in the real world. 2007.
[17] S. Kremer, M.D. Ryan, Analysing the vulnerability of protocols to produce known-pair and chosen-text attacks. Electronic notes in theoretical, Comput. Sci. 128 (2005) 87–104, https://doi.org/10.1016/j.entcs.2004.11.043.
[18] S. Alneyadi, E. Sithirasenan, V. Muthukkumarasamy, A survey on data leakage prevention systems, J. Netw. Comput. Appl. 62 (2016) 137–152, https://doi.org/10.1016/j.jnca.2016.01.008.
[19] J. Leng, M. Zhou, J.L. Zhao, Y. Huang, Y. Bian, Blockchain security: a survey of techniques and research directions, IEEE Trans. Serv. Comput. (2021) 1, https://doi.org/10.1109/TSC.2020.3038641.
[20] S.R. Raj, A. Cherian, A. Abraham, A survey on data loss prevention techniques, Int. J. Sci. Res. 2 (2013) 2319–7064.
[21] I.H. Montano, La Torre de, I. Díez, J.J.G. Aranda, J.R. Diaz, S.M. Cardín, J.J. G. López, Secure file systems for the development of a data leak protection (DLP) tool against internal threats, in: 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), IEEE, 2022, pp. 1–7, https://doi.org/10.23919/CISTI54924.2022.9820170.
[22] S.S. Babu, K. Balasubadra, Revamping data access privacy preservation method against inside attacks in wireless sensor networks, Clust. Comput 22 (2019) 65–75, https://doi.org/10.1007/s10586-018-1706-1.
[23] S. Ahmad, S. Mehfuz, J. Beg, Cloud security framework and key management services collectively for implementing DLP and IRM, in: Materials Today: Proceedings, 2022, https://doi.org/10.1016/j.matpr.2022.03.420.
[24] S. Ahmad, S. Mehfuz, J. Beg, Hybrid cryptographic approach to enhance the mode of key management system in cloud environment, J Supercomput 79 (2023) 7377–7413, https://doi.org/10.1007/s11227-022-04964-9.
[25] A.E. Adeniyi, K.M. Abiodun, J.B. Awotunde, M. Olagunju, O.S. Ojo, N.P. Edet, Implementation of a block cipher algorithm for medical information security on cloud environment: using modified advanced encryption standard approach, Multimed Tools Appl 82 (2023) 20537–20551, https://doi.org/10.1007/s11042-023-14338-9.
[26] M.P.S. Bhondve, Efficiently encryption decryption schema for secure file Storage system in cloud computing, INTERANTIONAL J. OF SCI. RES. IN ENG. AND MANAG. 07 (2023), https://doi.org/10.55041/ijsrem25005.
[27] Ms.S. Suma, S.Failur Rahuman, A. Ghoushick, R. Hari Ganesh, File storage system using hybrid cryptography, Int. J. Adv. Res. Sci. Commun. Technol. (2023) 45–49, https://doi.org/10.48175/IJARSCT-8659.
[28] H. Wu, H. Han, X. Wang, S. Sun, Research on Artificial intelligence enhancing Internet of Things security: a survey, IEEE Access 8 (2020) 153826–153848, https://doi.org/10.1109/ACCESS.2020.3018170.
[29] Z. Zhang, H. Ning, F. Shi, F. Farha, Y. Xu, J. Xu, et al., Artificial intelligence in cyber security: research advances, challenges, and opportunities, Artif Intell Rev 55 (2022) 1029–1053, https://doi.org/10.1007/s10462-021-09976-0.
[30] M. Afifi, 11K hands: gender recognition and biometric identification using a large dataset of hand images, Multimed Tools Appl 78 (2019) 20835–20854, https://doi.org/10.1007/s11042-019-7424-8.
[31] S. Angadi, S. Hatture, Hand geometry based user identification using minimal edge connected hand image graph, IET Comput. Vis. 12 (2018) 744–752, https://doi.org/10.1049/iet-cvi.2017.0053.
[32] M. Klonowski, M. Plata, P. Syga, User authorization based on hand geometry without special equipment, Pattern Recognit 73 (2018) 189–201, https://doi.org/10.1016/j.patcog.2017.08.017.
[33] S. Ghanbari, Z.P. Ashtyani, M.T. Masouleh, User identification based on hand geometrical biometrics using Media-pipe. 2022 30th International Conference on Electrical Engineering (ICEE), IEEE (2022) 373–378, https://doi.org/10.1109/ICEE55646.2022.9827056.
[34] S. Barra, M. De Marsico, M. Nappi, F. Narducci, D. Riccio, A hand-based biometric system in visible light for mobile environments, Inf Sci (Ny) 479 (2019) 472–485, https://doi.org/10.1016/j.ins.2018.01.010.
[35] A. Halder, A. Tayade, Real-time vernacular sign language recognition using mediapipe and machine learning, J. Homepage: Www Ijrpr Com ISSN 2582 (2021) 7421.
[36] L-R Müller, J. Petersen, A. Yamlahi, P. Wise, T.J. Adler, A. Seitel, et al., Robust hand tracking for surgical telestration, Int J Comput Assist Radiol Surg 17 (2022) 1477–1486, https://doi.org/10.1007/s11548-022-02637-9.
[37] S. Shriram, B. Nagaraj, J. Jaya, S. Shankar, P. Ajay, Deep learning-based real-time AI virtual mouse system using computer vision to avoid COVID-19 spread, J Heal. Eng 2021 (2021) 1–8, https://doi.org/10.1155/2021/8133076.
[38] B. Subramanian, B. Olimov, S.M. Naik, S. Kim, K.-H. Park, J. Kim, An integrated mediapipe-optimized GRU model for Indian sign language recognition, Sci Rep 12 (2022) 11964, https://doi.org/10.1038/s41598-022-15998-7.
[39] Y. Kortli, M. Jridi, A. Al Falou, M. Atri, Face recognition systems: a survey, Sensors 20 (2020) 342, https://doi.org/10.3390/s20020342.
[40] Y. Zhang, X. Xiao, L-X Yang, Y. Xiang, S. Zhong, Secure and efficient outsourcing of PCA-based face recognition, IEEE Trans. Inf. Forensics Secur. 15 (2020) 1683–1695, https://doi.org/10.1109/TIFS.2019.2947872.
[41] A. Sardar, S. Umer, Implementation of face recognition system using BioCryptosystem as template protection scheme, J. Inf. Secur. Appl. 70 (2022) 103317, https://doi.org/10.1016/j.jisa.2022.103317.
[42] Y. Wang, T. Nakachi, Secure face recognition in edge and cloud networks: from the Ensemble learning perspective. ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal, in: Processing (ICASSP), IEEE, 2020, pp. 2393–2397, https://doi.org/10.1109/ICASSP40776.2020.9052992.
[43] R. TH Hasan, A. Bibo Sallow, Face detection and recognition using OpenCV, J. Soft Comput. Data Min. 2 (2021), https://doi.org/10.30880/jscdm.2021.02.02.008.
[44] C. Pagano, E. Granger, R. Sabourin, G.L. Marcialis, F. Roli, Adaptive ensembles for face recognition in changing video surveillance environments, Inf Sci (Ny) 286 (2014) 75–101, https://doi.org/10.1016/j.ins.2014.07.005.
[45] G.F. Plichoski, C. Chidambaram, R.S. Parpinelli, A face recognition framework based on a pool of techniques and differential evolution, Inf Sci (Ny) 543 (2021) 219–241, https://doi.org/10.1016/j.ins.2020.06.054.
[46] I.H. Montano, I. de La Torre Díez, J.J.G. Aranda, J.R. Diaz, S.M. Cardín, J.J. G. López, Secure file systems for the development of a data leak protection (DLP) tool against internal threats, in: 2022 17th Iberian Conference on Information Systems and Technologies (CISTI), IEEE, 2022, pp. 1–7, https://doi.org/10.23919/CISTI54924.2022.9820170.
[47] Dokan n.d. https://github.com/dokan-dev/dokany/wiki/Build.
[48] P. Holgado, A. García, J.J. García, J. Roncero, V.A. Villagrá, H. Jalain, Context-based encryption applied to data leakage prevention solutions, in: Proceedings of the 14th International Joint Conference on e-Business and Telecommunications, vol. 4, SCITEPRESS - Science and Technology Publications, 2017, pp. 566–571, https://doi.org/10.5220/0006475205660571.
[49] T. Ahonen, A. Hadid, M. Pietikäinen, Face recognition with local binary patterns, in: T. Pajdla, J. Matas (Eds.), Computer Vision - ECCV 2004, Springer Berlin Heidelberg, Berlin, Heidelberg, 2004, pp. 469–481.
[50] Ramos Diaz J. Securecipher difusion validation 2022.
[51] M. Sýs, Z. Říha, V. Matyáš, K. Márton, A. Suciu, On the interpretation of results from the NIST statistical test suite, Rom. J. Inf. Sci. Technol. 18 (2015) 18–32.