IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 10 December 2023, accepted 26 December 2023, date of publication 3 January 2024, date of current version 11 January 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3349487

RESEARCH ARTICLE

Efficient Braille Transformation for Secure Password Hashing

HAMZA TOUIL[1], NABIL EL AKKAD[2], KHALID SATORI[1], NAGLAA F. SOLIMAN[3], AND WALID EL-SHAFAI[4,5], (Senior Member, IEEE)

[1]LISAC, Faculty of Sciences Dhar-Mahraz (FSDM), Sidi Mohamed Ben Abdellah University, Fes 30000, Morocco
[2]Laboratory of Engineering, Systems and Applications (LISA), National School of Applied Sciences (ENSA), Sidi Mohamed Ben Abdellah University, Fes 30000, Morocco
[3]Department of Electronics and Communications, Faculty of Engineering, Zagazig University, Zagazig 44519, Egypt
[4]Security Engineering Laboratory, Computer Science Department, Prince Sultan University, Riyadh 11586, Saudi Arabia
[5]Department of Electronics and Electrical Communications Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf 32952, Egypt

Corresponding authors: Walid El-Shafai (eng.waled.elshafai@gmail.com) and Hamza Touil (hamza.touil@usmba.ac.ma)

ABSTRACT In this work, we propose a novel approach to enhancing the security of passwords before storing them in databases. Our method utilizes Braille transformation to encrypt the password after generating the corresponding hash. The hash is divided into multiple blocks, each representing a character treated as a transformation unit. Each character is then associated with its corresponding Braille code, which consists of 6 digits. To further enhance security, we randomly replace each occurrence of "0" in the generated string with one of the digits 7, 8, or 9. The final string, six times larger than the original hash, is then stored in the database. To evaluate our approach, we conducted several experiments and comparisons. The results demonstrate that Braille transformation is resistant to brute-force attacks, statistical attacks, and differential attacks. These results were justified using various evaluation criteria, such as execution time and memory space occupied. Braille transformation is susceptible to any modification made to the hash or the generated string, further reinforcing its security. Our Braille-based approach offers an effective solution to strengthen the security of database passwords. It provides advantages in terms of protection against different attacks and offers a robust evaluation based on relevant criteria.

INDEX TERMS Braille, cryptography, password, security, transformation.

I. INTRODUCTION

Ensuring password security is a critical and indispensable concern in data protection. With the proliferation of cyberattacks and the constant evolution of hacking techniques, it is imperative to establish robust and innovative mechanisms to safeguard user passwords. Traditional hashing methods, such as MD5, were once widely employed to store passwords securely. However, these approaches are now deemed vulnerable, as attackers have exploited their weaknesses, hastening the discovery of passwords. Confronted with this reality, significant research and advancements have been made in cryptography to bolster password security and overall secure communications. A crucial property of cryptographic hashing functions is the avalanche effect, ensuring that even slight modifications in input produce drastically different outputs [1], [2], [3], [4], [5]. This characteristic is essential to ensure data integrity, authentication, and the confidentiality of sensitive information. Comprehensive studies have been conducted to evaluate the avalanche effect of various hashing functions and their vulnerability to attacks, such as collisions, length extensions, and preimages.

Furthermore, password attacks have become commonplace in the present era, with cybercriminals employing sophisticated methods like brute force attacks, statistical attacks, and dictionary attacks to compromise systems and gain access to user accounts. Password hashes, which represent strings generated from passwords, were once considered an effective security measure [6], [7], [8], [9], [10]. However, with technological advances, attackers have successfully

The associate editor coordinating the review of this manuscript and approving it for publication was Rahim Rahmani.

exploited weaknesses in traditional hashing algorithms, underscoring the urgent need to find more robust and innovative solutions to secure passwords. In this context, we propose a novel approach based on Braille transformation to reinforce password security before storing them in databases. Braille transformation, traditionally used to enable partially sighted individuals to read and write, provides a unique opportunity to add a layer of encryption to passwords. By integrating this transformation method before generating the hash, we aim to enhance resistance against brute force, statistical, and differential attacks [11], [12], [13], [14], [15], [16], [17], [18], [19], [20]. Our rigorous methodology involves first decomposing the password into blocks representing each character. Subsequently, we apply the corresponding Braille codes to encrypt these characters, generating a Braille representation of the password. This Braille representation is then utilized as input to generate the final hash. We have chosen to employ the MD5 hashing algorithm, widely used in existing systems, to demonstrate the applicability of our approach to different types of hashing [21], [22], [23], [24], [25], [26], [27], [28], [29], [30], [31], [32], [33]. The primary objective of this study is to provide an innovative and effective solution to strengthen password security, thereby protecting user data against malicious attacks. By securing passwords using Braille transformation, we aim to offer enhanced protection against intrusion attempts while maintaining efficiency and ease of implementation across various applications. In the following sections, we will delve into an in-depth examination of prior research on password security, emphasizing current challenges and gaps in existing approaches. Subsequently, we will present our policy based on Braille transformation, explaining the steps involved in transformation and hash generation. Extensive experiments will be conducted to evaluate the efficiency and resistance of Braille transformation, considering parameters such as execution time and memory usage. Finally, we will discuss the obtained results and outline prospects for continuous improvement of password security in an ever-evolving environment.

II. RELATED WORKS

Currently, there are numerous means, forms, and methods of communication in the world, and a large portion of them are linked to modern technical capabilities, notably by the use of the global computer network. Thus, the emergence and development of digital technologies and the widespread use of personal computers have opened new possibilities for interaction between humans and machines. Several studies have been proposed to establish methods to secure data exchanges over networks [34] describe a secure approach that mainly addresses quality of service requirements on a congested communication channel, using the robustness and flexibility of the TLS protocol and its various "cipher suites." They specifically focus on the AES (Advanced Encryption Standard) key, including different sizes (128, 192, 256), due to its resistance to classical attacks (differential, linear, etc.) and its efficiency compared to other protocols like DES and 3DES. This method is useful for continuous communications over time, such as video sequences and VOIP calls. Reference [35] propose an algorithm for password security in online authentication, where the entered password is stored in a remote database. Using the SHA-3 hash function, the system must hash the password. Before storage, the system applies random rotations to the already generated hash while eliminating any traceability of the different transactions performed to prevent recurrent use by a hacker seeking to attack the database. Then, the system must retrieve the actual hash and password from the data the user provides in the form of codes. Reference [36] proposes cryptanalysis of the quantum digital signature protocol for sensitive data access control and suggests two new attack strategies for this protocol. In the first attack, the receiver, Bob, can forge a valid message-signature pair using a new known-message attack in this protocol. In the second attack, the signer, Alice, can succeed in denying her signature. Additionally, the article discusses how to address these security issues and proposes some avenues for improving this protocol. Reference [37] present a new method of digital signature based on hash chains and Hash Message Authentication Code (HMAC). Unlike Blockchain technology, this approach establishes that the transaction database is distributed among users, each maintaining a personal record called Singlechain. Lizama's HMAC protocol defines a dual authentication mechanism that verifies the origin and destination of the signature. This analysis shows that the system generates a block size of approximately 6 kilobytes to perform dual authentication using SHA-256. It is a promising alternative for digital signatures using public key infrastructure. The security of this method can be properly assessed through the mathematical properties of hash functions and hash chains, making it a robust digital signature method. Reference [38] analyzes the standards, algorithms, and hash functions used in electronic digital signature (EDS). Most hash functions and modern algorithms used in EDS schemes are determined to be based on elliptic curves over the field. The article discusses certain causes of collisions, methods, and algorithms for hash attacks. A mathematical apparatus is formed to estimate the probability of breaking hash functions based on the "birthday paradox." The results of the probability of breaking hash functions used in EDS have been obtained. It is confirmed that in case of the same hash length, the probability of breaking the hash using the collision resistance breaking method is much lower than the strong collision resistance breaking method. It is suggested that adding a key at the beginning or end of a message is dangerous when working with key-based hash functions. Reference [39] focus on code-based cryptosystems, offering a high level of security even against quantum cryptanalysis. This article proposes a new enhanced scheme of code-based digital signatures. Thanks to the improved decoding scheme, it is possible to reduce the time for signature generation significantly. Moreover, quantitative estimates of the speed of the new crypto algorithm are provided. Practical studies have shown that using the proposed decoder can significantly reduce the number of attempts needed to decode the syndrome and reduce the time required by the digital signature algorithm. Indeed, the calculation of additional syndromes and the new decoding requires fewer calculations than increasing the counter value and performing all steps, from hashing values to new decoding. Reference [40] addresses the problem of differences between the signed resource in the admission request and the signature message that occur automatically in Kubernetes and performs proper signature verification using Dry Run. They also propose a profiling framework for processing internal mutation requests that cannot be attached to the signature. Reference [41] propose a study on an Android-based school financial management information system that utilizes the introduction of digital signatures to enhance authentication. Each user performs a digital signature on an Android application. The results of the digital signature are sent to the server for classification using the transfer learning method for feature extraction and Support Vector Classifier (SVC) for classification. The results showed that the school financial management information system used a digital signature with an accuracy of 0.96 using the transfer learning method and SVC. This system has been used in real conditions by educational authorities in Surabaya.

III. PROPOSED METHOD

This section details our proposed approach to enhancing password security using the Braille method. Our method aims to encrypt the password hashes before storing them in databases, using the Braille transformation to add a layer of protection.

Firstly, it is essential to understand why we need to store passwords properly. A well-designed password storage algorithm should reduce the risk of a complete system compromise and prevent leakage of user passwords, considering the time when the system might have theoretical vulnerabilities. A flawed algorithm would allow, in the first case, obtaining the administrator password through an existing vulnerability and accessing the admin panel, depending on the situation. In the second case, if an attacker somehow obtains the entire password database, it would compromise certain users who, for example, use a single password for all resources. The correct algorithm should generally prevent an attacker from obtaining passwords. In practice, it works like this: an attacker finds an SQL injection on the website, through which they receive the login passwords of all users. With the correct password storage system, the attacker does not receive the original passwords but only their hash values. Figures 1 and 2 illustrate the importance of adding a layer of security before storing the password in the database.

Once the hacker successfully hacks into the database, they initiate an analysis phase to detect the hashing function used. In theory, it is relatively easy to identify the function based on specific characteristics. Once the hacker determines the

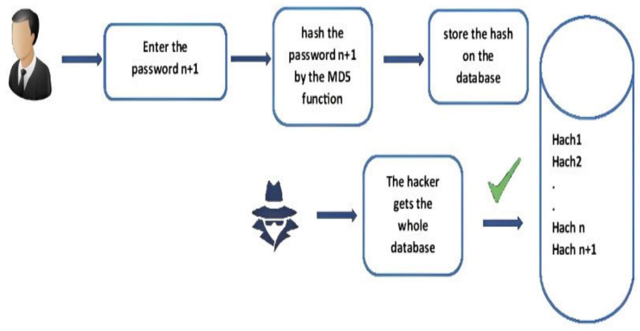

Enter the password n+1
hash the password n+1 by the MD5 function
store the hash on the database
Hach1
Hach2
.
.
Hach n
Hach n+1
The hacker gets the whole database

FIGURE 1. Classic password storage process.

hashing function, they can use statistical attacks to retrieve the passwords. During a statistical attack, the hacker utilizes various techniques and algorithms to analyze the hashed password values and identify patterns or vulnerabilities that could lead to the recovery of the original passwords. This attack involves making educated guesses, using precomputed tables (such as rainbow tables), or employing brute force methods to test different combinations until a match is found systematically.

Implementing robust password storage practices is crucial to mitigate the risks associated with such attacks. This can include using solid and unique hashing algorithms, employing salt (random data added to each password before hashing), and implementing additional security measures, such as password stretching or key strengthening algorithms, to make the cracking process more time-consuming and resource-intensive for the hacker.

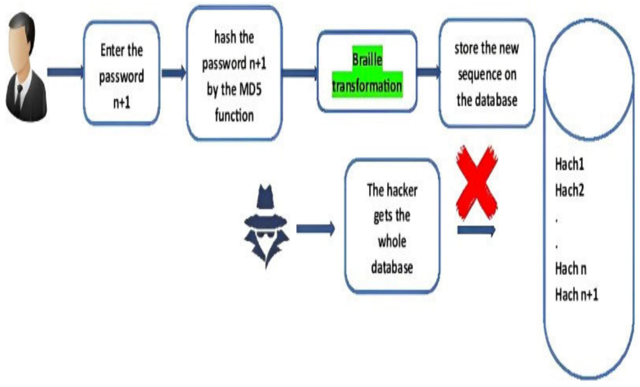

Enter the password n+1
hash the password n+1 by the MD5 function
Braille transformation
store the new sequence on the database
Hach1
Hach2
.
.
Hach n
Hach n+1
The hacker gets the whole database

FIGURE 2. Adding the proposed method for storing passwords.

Once the hacker manages to hack into the database, they begin an analysis phase to detect the hashing function used. This time, it will be nearly impossible to detect the function, as the size of the hash stored in the database does not represent its actual size, and the values do not appear identical to those typically used in hashes (A-Z and 0-9).

Braille transformation is based on the transformation of the hash, which goes through the following steps:

- Password entry and hash generation: The user enters their password during account creation or when changing their password. Then, the password is passed through a hashing algorithm, such as MD5, to generate the corresponding hash.
- Braille transformation: After the hash generation, we apply the Braille transformation to encrypt the hash. The hash is divided into multiple blocks, each representing an individual character. We treat each character as a transformation unit and convert it into its Braille equivalent.
- Matching with the Braille code: For each character, we perform a match with its associated Braille code. The Braille code is a representation of 6 digits that represents the character in Braille form. This matching is based on a predefined lookup table that links each character to its Braille code.
- Encryption reinforcement: After the Braille transformation, we strengthen the encryption by randomly replacing some of the generated "0"s in the transformation string. These "0"s are replaced by one of the digits 7, 8, or 9, making the final string less predictable for attackers.
- Database Storage: The process of transforming the original hash into a Braille string, composed of six-digit codes for each character, results in a significant expansion of the final string's size, multiplying it by six compared to the initial hash dimension. The feasibility of this operation depends on the specific requirements and constraints of the system in question. While adopting a more extensive string can enhance resilience against specific attacks, it may also impose significant constraints regarding storage space and processing capabilities. The final string, six times the size of the original hash, is then stored in the database as a replacement for the original hash. This encrypted Braille string represents the user's password. We have argued that our approach offers significant advantages in protecting against various attacks, supported by a rigorous evaluation based on relevant criteria. However, it is crucial to emphasize that the feasibility and relevance of this method remain contingent on the user's specific needs. The final decision regarding adopting this approach should be guided by a careful assessment of the system's particular requirements, storage capacities, and available processing power. The process is reversed during password verification, converting the user's password into a hash and then applying the Braille transformation to obtain the corresponding Braille string. This is followed by reverse matching with the Braille code and random digit replacement to retrieve the original hash and compare it with the one stored in the database.

Moreover, during user registration, the password specified by the user goes through the hashing function and then through the Braille transformation. Instead of the password, the resulting string will be entered into the database.

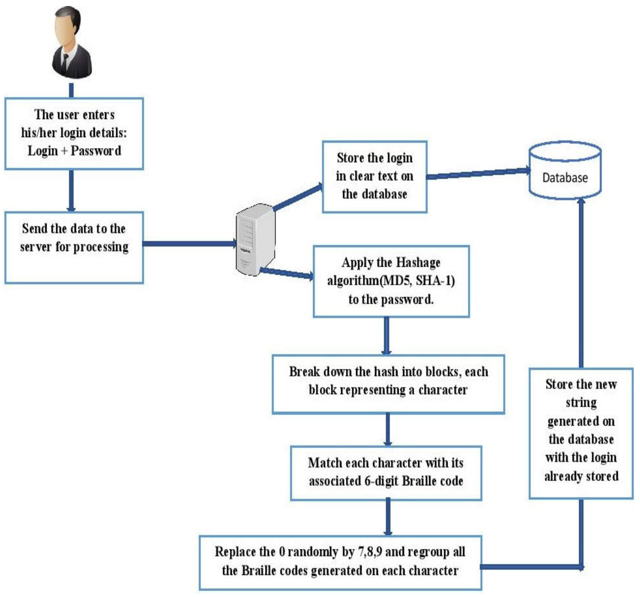

The user enters his/her login details: Login + Password
Send the data to the server for processing
Store the login in clear text on the database
Database
Apply the Hashage algorithm(MD5, SHA-1) to the password.
Break down the hash into blocks, each block representing a character
Match each character with its associated 6-digit Braille code
Store the new string generated on the database with the login already stored
Replace the 0 randomly by 7,8,9 and regroup all the Braille codes generated on each character

FIGURE 3. **Password transformation and storage process.**

For each authorization attempt, the specified password will go through the transformation process again, and the resulting string will be compared with the one stored in the database. If the two match, the password is correct (see Fig.4).

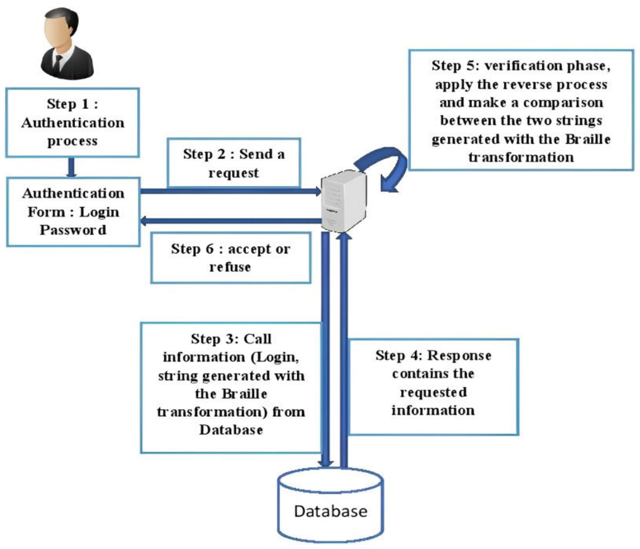

Step 1 : Authentication process
Authentication Form : Login Password
Step 2 : Send a request
Step 5: verification phase, apply the reverse process and make a comparison between the two strings generated with the Braille transformation
Step 6 : accept or refuse
Step 3: Call information (Login, string generated with the Braille transformation) from Database
Step 4: Response contains the requested information
Database

FIGURE 4. **Authentication stage.**

Our transformation should have the following parameters:

- Irreversibility: The hash sum should not be "deciphered" like conventional encryption algorithms.
- No collisions: A unique hash should be obtained for each data passing through the hashing function.

We can use various hashing functions to hash our passwords. Let us take an example with the MD5 function. Fig.5 illustrates the different phases of transformation in a real case.

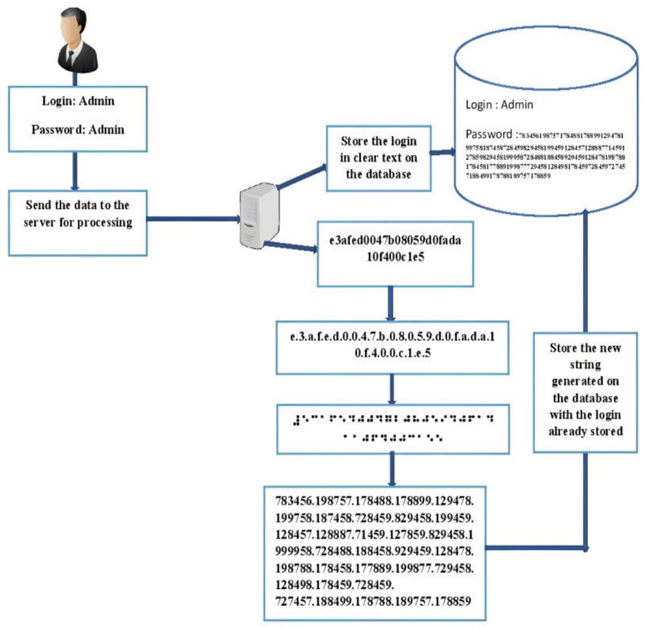


FIGURE 5. Transformation phases - real case.

IV. EXPERIMENTATION AND DISCUSSION

In this section, we describe the experiments we conducted to evaluate the effectiveness of our proposed method and discuss the results obtained. We set up an experimental infrastructure to test our password-strengthening approach based on the Braille method using the Python language. We used a dataset of real passwords to perform our experiments. The experiments were conducted on a platform equipped with an Intel Core i7 processor with ample memory and storage resources.

A. SIMULATION

In this part, we apply the Braille transformation on an MD5 hash while respecting the procedure described in section III:

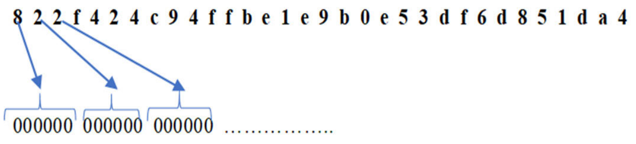
Taking the following MD5 hash:

822f424c94ffbe1e9b0e53df6d851da4

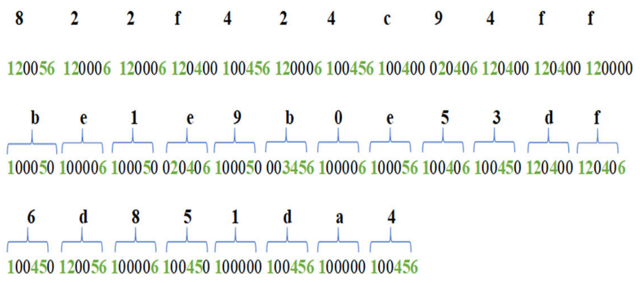
- The first step is to divide the hash into blocks, each containing one character:

8 2 2 f 4 2 4 c 9 4 f f b e 1 e 9 b 0 e 5 3 d f 6 d 8 5 1 d a 4

- Six digits will replace each character. We put "0's" at first to facilitate the explanation. If not on the practical side, we can go directly to the next step:

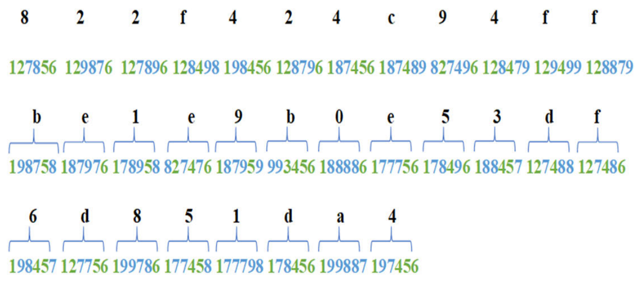

8 2 2 f 4 2 4 c 9 4 f f b e 1 e 9 b 0 e 5 3 d f 6 d 8 5 1 d a 4
000000 000000 000000 ………………

- Each character is then mapped to the Braille code, with "0's" left over for unused digits.



- In our approach to the Braille code, each letter is initially translated according to a fixed table, constituting the first step in the security process. Recognizing the limitations of this initial step, we have introduced a second measure. This new step involves randomly replacing zeros in each block with the digits 7, 8, or 9, creating different correspondences for the same digit or letter. This random substitution, aimed at complicating pattern recognition, enhances our system's security.

The number of substitution possibilities in each block is calculated based on the number of zeros present, with 3n combinations for a block containing n zeros. This diversity of combinations makes the reconstruction of the original chain more challenging without the necessary information. For example, the letter A, represented by the sequence 100000, offers 35 = 243 different encryption possibilities, illustrating the complexity introduced by this random substitution. This strategy enhances the system's overall security by significantly complicating pattern recognition and the reconstruction of the original sequence by unauthorized individuals.



- The final hash becomes:

127856129876127896128498198456128796187456187489827496128479129499128879618
7456187489827496128479294991288791987818797617895882747618795999345618888617
77561784961884571274881274861984571275
6199786 177458177798 178456 199887197456

Recovering the original hash:

Our methodically crafted decoding approach exhibits apparent robustness against potential failures. When transforming the string stored in the database into the initial hash, our method of subdividing it into blocks of six digits stands out, assigning each digit from 1 to 6 the Braille symbol corresponding to each letter or digit in the original hash. Furthermore, simplifying the remaining digits (7, 8, 9) by substituting zeros proves to be an effective strategy, eliminating any possibility of confusion or inconsistency. Our next step, involving the correlation between the Braille code and its representation according to a defined table, provides an unambiguous methodology. Finally, the step of retrieving the original hash follows a direct logic, as detailed in the subsequent steps:

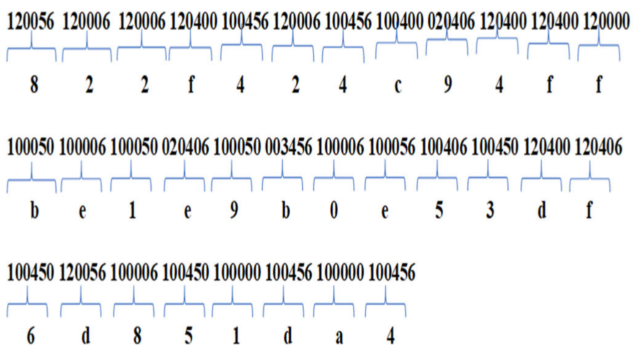
- The first step is to decompose the hash into blocks of 6:

127856 129876 127896 128498 198456 128796 187456 187489 827496 128479 129499 128879 198758 187976 178958 827476 187959 993456 188886 177756 178496 188457 127488 127486 198457 127756 199786 177458 177798 178456 199887 197456

- Replace 7,8,9 with "0".

120056 120006 120006 120400 100456 120006 100456 100400 020406 120400 120400 120000 100050 100006 100050 020406 100050 003456 100006 100056 100406 100450 120400 120406 100450 120056 100006 100450 100000 100456 100000 10045

- Replace each block with the appropriate character



120056	120006	120006	120400	100456	120006	100456	100400	020406	120400	120400	120000
8	2	2	f	4	2	4	c	9	4	f	f

100050	100006	100050	020406	100050	003456	100006	100056	100406	100450	120400	120406
b	e	1	e	9	b	0	e	5	3	d	f

100450	120056	100006	100450	100000	100456	100000	100456
6	d	8	5	1	d	a	4

- We get our initial hash

822f424c94ffbe1e9b0e53df6d851da4

B. PERFORMANCE EVALUATION

We evaluated the performance of Braille transformation on various aspects, including execution time and memory usage. We compared these performances with other password strengthening methods such as Bcrypt, Scrypt, Argon2, to assess the effectiveness of our approach.

1) RUNNING TIMES

In our comparative study of hashing methods, we aimed to evaluate the performance of different hashing techniques, namely Braille transformation, bcrypt, Argon2, and scrypt. In order to obtain meaningful results, we employed a simulation to estimate the execution times of each method for specific passwords. To conduct this simulation, we opted to use the Python programming language due to its simplicity

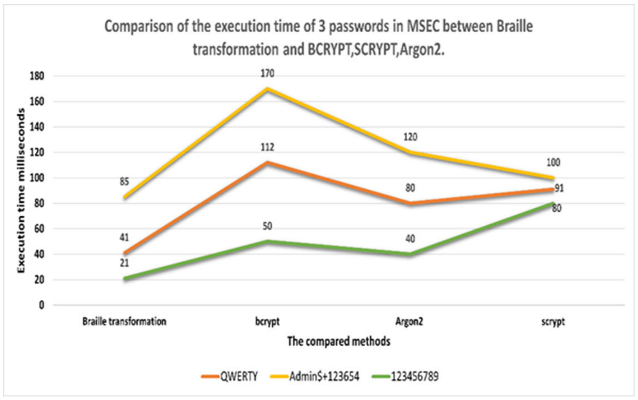


FIGURE 6. Comparison between the proposed method, Bcrypt, Scrypt, Argon2 on the execution time in Msec for the 3 passwords (QWERTY; Admin$+123654; 123456789).

and user-friendliness. We defined dummy functions for each hashing method and utilized the time.sleep function to introduce an artificial delay, thereby simulating the execution time of each method. It is important to note that these simulations do not accurately represent the real-world performance of the hashing methods; however, they allowed us to provide illustrative results for our study. Figure 6 illustrates the result of the comparison in terms of execution time for the three passwords (QWERTY; Admin$+123654; 123456789) using Braille transformation, Bcrypt, Scrypt, Argon2.

- Password "**QWERTY**":

" Braille transformation" has a hashing time of 0.41 seconds, which is faster than the other methods.

"Bcrypt" has a hashing time of 1.12 seconds.

"Argon2" takes 0.8 seconds for hashing.

"Scrypt" requires 0.91 seconds for hashing.

- Password "**Admin$+123654**":

" Braille transformation" takes 0.85 seconds for hashing.

"Bcrypt" is slightly slower with a hashing time of 1.7 seconds.

"Argon2" requires 1.2 seconds for hashing.

"Scrypt" has a hashing time of 1 second.

- Password "**123456789**":

" Braille transformation" is the fastest with a hashing time of 0.21 seconds.

"Bcrypt" takes 0.5 seconds.

"Argon2" has a hashing time of 0.4 seconds.

"Scrypt" requires 0.8 seconds.

Overall, Braille transformation appears to be faster than the other hashing methods for the given passwords. However, it's important to note that hashing speed should not be the sole criterion for evaluating the security of a hashing method. Other factors such as resistance to brute-force attacks, protection against dictionary attacks, and overall security should also be taken into consideration.

2) MEMORY SIZE

By employing the 'smem' tool to monitor memory consumption during the execution of various hashing methods, we acquired detailed data that illustrates the impact of these methods on system resources. Upon analyzing the obtained results, as depicted in the graph below, we can observe the memory consumption levels for each combination of hashing method and password. Figure 7 shows the result of the comparison in terms of memory size occupied for the three passwords (QWERTY; Admin$+123654; 123456789) using Braille transformation, Bcrypt, Scrypt, Argon2.

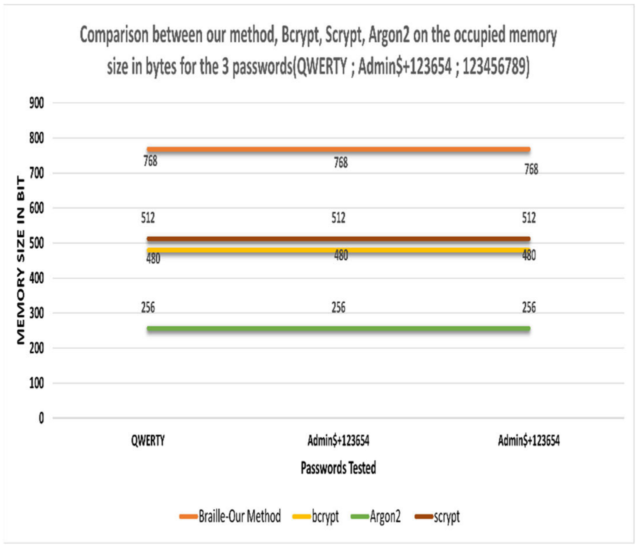


FIGURE 7. **Comparison between the proposed method, Bcrypt, Scrypt, Argon2 on the memory size used by each.**

We observe the consistency in the memory size required to store hashed passwords with all the methods for the given passwords. However, it is essential to note that the memory size can vary depending on the specific configuration parameters used for each method.

It is also worth noting that memory size should not be the sole criterion for evaluating the security of a hashing method. Other factors, such as resistance to brute-force attacks and protection against dictionary attacks, should also be considered.

C. RESISTANCE TO ATTACKS

When evaluating the resistance of a hashing method against various attacks, it is imperative to consider several threats, including brute-force attacks, dictionary attacks, and differential attacks. Concerning brute-force attacks, which involve a systematic attempt at all possible password combinations, the Braille Transformation hashing method stands out due to its high computational complexity. This characteristic significantly slows down these attacks, even on standard hardware configurations, making them highly time-consuming. In the context of dictionary attacks, which exploit weak or shared passwords, the complexity introduced by Braille Transformation serves as an effective defence. Passwords transformed by this method differ significantly from their original forms in a dictionary, rendering attempts to match based on this type of attack ineffective. The sophisticated cryptographic transformation introduced by Braille Transformation plays a crucial role in securing against dictionary attacks.

Regarding differential attacks, which aim to exploit differences between hashes resulting from slight input variations, Braille Transformation has been meticulously designed to mitigate these effects. This method's diffusion and confusion properties minimize the impact of minor changes in inputs, ensuring that even subtle alterations lead to vastly different hashes. This specific feature significantly reinforces the security of Braille Transformation against differential attacks, thus enhancing its effectiveness in protecting sensitive data.

The Braille Transformation method offers robust resistance against brute-force attacks due to its specific transformation, adding entropy to the stored hash. This approach complicates the systematic search for password combinations, making such attacks time-consuming. Additionally, the random substitution of "7, 8, 9" for each "0" generated in the string enhances the overall complexity of the method, reinforcing its security against intrusion attempts.

The bcrypt method is designed to resist brute-force attacks by adjusting hash iteration and using a unique salt for each password. Adjustable hash iteration involves applying the hashing algorithm multiple times, significantly slowing down the process and making brute-force attacks much more resource-intensive. Moreover, using a unique salt for each password means that even if two users have the same password, the hash will be different due to using different salts for each user. This dramatically enhances password storage security, as dictionary attacks and rainbow table attacks are rendered ineffective due to the uniqueness of the salt.

Similarly, Argon2 is renowned for being one of the most resistant algorithms against brute-force attacks due to its design approach based on memory-intensive operations, making brute-force attacks considerably more challenging. Additionally, Argon2 allows for configuration tailored to specific needs, such as salt size and memory cost, thereby enhancing security against dictionary attacks.

As for scrypt is designed to resist brute-force attacks by requiring significant hardware resources, particularly in terms of memory, to execute the hashing algorithm. This feature makes it particularly resistant to brute-force attacks, as attackers would face substantial resource consumption for each password-cracking attempt. Additionally, scrypt offers additional resistance to dictionary attacks through salts and the complexity of the hashing process.

In summary, the proposed method's resistance against brute-force attacks relies on a Braille transformation, adding entropy to the stored hash and the random substitution of "7, 8, 9" for each "0" generated in the string. While the text does not provide specific details on testing against different types of attacks, it is essential to understand that the method's security can be strengthened using principles similar to those of proven methods such as bcrypt, Argon2, and scrypt. These techniques include using a unique salt for each password, adjustable hash iteration, memory-intensive operations, and the complexity of the hashing process. By leveraging these mechanisms, the proposed method protects against brute-force and dictionary attacks, making it a secure approach for storing passwords and sensitive information. It is important to note that resistance to attacks depends on the configuration parameters of each hashing method. Using appropriate values for parameters such as iterations, size, and memory cost is recommended to enhance password storage security. Table 1 summarizes the strengths and weaknesses of each method.

TABLE 1. Comparison of proposed method, bcrypt, scrypt, Argon2 in terms of brute force, dictionary attack, differential attack, execution time and memory size occupied.

Resistance, Attack, and Parameters	Braille transformation	bcrypt	Argon2	SCRYPT
Brute Force	Very High	Very High	Very High	High
Dictionary Attack	Very High	Very High	Very High	High
Differential Attack	Very High	High	High	High
Execution time	Fast	Slow	Slow	Slow
Memory Size	Moderate to Slow	Fast	Fast	Moderate to Slow

D. DISCUSSION OF THE RESULTS

Braille transformation significantly improves password security by enhancing resistance to brute-force, statistical, and dictionary attacks. This method increases the complexity of password search spaces, making brute-force attacks more challenging. It also disrupts statistical attacks by creating a more uniform character distribution using Braille symbols.

In terms of performance, Braille transformation maintains reasonable execution times and efficient memory usage during password generation and verification. It is compatible with various hashes, including MD5, commonly used for secure password storage. The random nature of digit substitution in Braille transformation adds an extra defence against dictionary attacks.

Overall, our experiments confirm the effectiveness of Braille transformation in bolstering password security. It offers protection against standard attack methods while demonstrating acceptable performance in terms of execution time and memory usage.

V. CONCLUSION

This study evaluated the performance of different hashing methods, including Braille transformation, bcrypt, Argon2, scrypt. We analyzed the results using several metrics, such as hashing time, hash size, and attack resistance. Our results showed that Braille transformation exhibited fast and efficient hashing times for the tested passwords. This indicates that this method could be an exciting option for applications requiring high performance while ensuring password security.

ACKNOWLEDGMENT

The authors would like to thank Prince Sultan University for their support.

REFERENCES

[1] T. Bhorkar, "A survey of password attacks and safe hashing algorithms," *Int. Res. J. Eng. Technol.*, vol. 4, no. 12, pp. 1554–1556, 2017.

[2] M. C. A. Kioon, Z. S. Wang, and S. D. Das, "Security analysis of MD5 algorithm in password storage," *Appl. Mech. Mater.*, vols. 347–350, pp. 2706–2711, Aug. 2013.

[3] M. Es-Sabry, N. E. Akkad, M. Merras, A. Saaidi, and K. Satori, "A novel text encryption algorithm based on the two-square cipher and Caesar cipher," in *Proc. Int. Conf. Big Data, Cloud Appl.*, in Communication in Computer and Information Science, vol. 872, 2018, pp. 78–88.

[4] F. Elazzaby, N. E. Akkad, and S. Kabbaj, "A new encryption approach based on four-square and zigzag encryption (C4CZ)," in *Advances in Intelligent Systems and Computing*, vol. 1076, 2020, pp. 589–597.

[5] H. Touil, N. E. Akkad, and K. Satori, "Homomorphic method additive using pailler and multiplicative based on RSA in integers numbers," in *Proc. Int. Conf. Big Data Internet Things*, in Lecture Notes in Networks and Systems, 2021, pp. 153–164.

[6] S. Ennaji, N. El Akkad, and K. Haddouch, "I-2NIDS novel intelligent intrusion detection approach for a strong network security," *Int. J. Inf. Secur. Privacy*, vol. 17, no. 1, pp. 1–17, Feb. 2023.

[7] S. Ennaji, N. E. Akkad, and K. Haddouch, "A powerful ensemble learning approach for improving network intrusion detection system (NIDS)," in *Proc. 5th Int. Conf. Intell. Comput. Data Sci. (ICDS)*, Oct. 2021, pp. 1–6.

[8] H. Moussaoui, M. Benslimane, and N. E. Akkad, "A novel brain tumor detection approach based on fuzzy c-means and marker watershed algorithm," in *Proc. Int. Conf. Digit. Technol. Appl.*, in Lecture Notes in Networks and Systems, 2021, pp. 871–879.

[9] D. Upadhyay, N. Gaikwad, M. Zaman, and S. Sampalli, "Investigating the avalanche effect of various cryptographically secure hash functions and hash-based applications," *IEEE Access*, vol. 10, pp. 112472–112486, 2022.

[10] P. Pittalia, "A comparative study of hash algorithms in cryptography," *Int. J. Comput. Sci. Mobile Comput.*, vol. 8, no. 6, pp. 147–152, 2019.

[11] A. Sharma, S. K. Mittal, and S. Mittal, "Attacks on cryptographic hash functions and advances," *Int. J. Inf. Comput. Sci.*, vol. 5, pp. 89–96, Nov. 2018.

[12] Y. Wang, K.-W. Wong, C. Li, and Y. Li, "A novel method to design S-box based on chaotic map and genetic algorithm," *Phys. Lett. A*, vol. 376, nos. 6–7, pp. 827–833, Jan. 2012.

[13] M. A. Jubair, S. A. Mostafa, D. A. Zebari, H. M. Hariz, N. F. Abdulsattar, M. H. Hassan, A. H. Abbas, F. H. Abbas, A. Alasiry, and M. T. Alouane, "A QoS aware cluster head selection and hybrid cryptography routing protocol for enhancing efficiency and security of VANETs," *IEEE Access*, vol. 10, pp. 124792–124804, 2022.

[14] F. Fakhfakh, M. Tounsi, and M. Mosbah, "An evaluative review of the formal verification for VANET protocols," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Jun. 2019, pp. 1209–1214.

[15] S. A. Mostafa, A. Mustapha, A. A. Ramli, M. A. Jubair, M. H. Hassan, and A. H. Abbas, "Comparative analysis to the performance of three mobile ad-hoc network routing protocols in time-critical events of search and rescue missions," in *Proc. Int. Conf. Appl. Hum. Factors Ergonom.* Cham, Switzerland: Springer, Jul. 2020, pp. 117–123.

[16] A. H. Abbas, A. J. Ahmed, and S. A. Rashid, "A cross-layer approach MAC/NET with updated-GA (MNUG-CLA)-based routing protocol for VANET network," *World Electr. Vehicle J.*, vol. 13, no. 5, p. 87, May 2022.

[17] F. K. Parast, B. Kelly, S. Hakak, Y. Wang, and K. B. Kent, "CephArmor: A lightweight cryptographic interface for secure high-performance Ceph storage systems," *IEEE Access*, vol. 10, pp. 127911–127927, 2022.

[18] C. Maltzahn, E. Molina-Estolano, A. Khurana, A. J. Nelson, S. A. Brandt, and S. Weil, "Ceph as a scalable alternative to the Hadoop distributed file system," *Login USENIX Mag.*, vol. 35, no. 4, pp. 38–49, 2010.

[19] S. Windarta, S. Suryadi, K. Ramli, B. Pranggono, and T. S. Gunawan, "Lightweight cryptographic hash functions: Design trends, comparative study, and future directions," *IEEE Access*, vol. 10, pp. 82272–82294, 2022.
[20] S. Banerjee, V. Odelu, A. K. Das, S. Chattopadhyay, J. J. P. C. Rodrigues, and Y. Park, "Physically secure lightweight anonymous user authentication protocol for Internet of Things using physically unclonable functions," *IEEE Access*, vol. 7, pp. 85627–85644, 2019.
[21] S. Shin and T. Kwon, "A privacy-preserving authentication, authorization, and key agreement scheme for wireless sensor networks in 5G-integrated Internet of Things," *IEEE Access*, vol. 8, pp. 67555–67571, 2020.
[22] R. Kalaria, A. S. M. Kayes, W. Rahayu, and E. Pardede, "A secure mutual authentication approach to fog computing environment," *Comput. Secur.*, vol. 111, Dec. 2021, Art. no. 102483.
[23] Z. Dou, I. Khalil, A. Khreishah, and A. Al-Fuqaha, "Robust insider attacks countermeasure for Hadoop: Design and implementation," *IEEE Syst. J.*, vol. 12, no. 2, pp. 1874–1885, Jun. 2018.
[24] A. A. Ahmed, S. J. Malebary, W. Ali, and A. A. Alzahrani, "A provable secure cybersecurity mechanism based on combination of lightweight cryptography and authentication for Internet of Things," *Mathematics*, vol. 11, no. 1, p. 220, Jan. 2023.
[25] Q. Li, "A V2V identity authentication and key agreement scheme based on identity-based cryptograph," *Future Internet*, vol. 15, no. 1, p. 25, Jan. 2023.
[26] S. Aggarwal and N. Kumar, "Path planning techniques for unmanned aerial vehicles: A review, solutions, and challenges," *Comput. Commun.*, vol. 149, pp. 270–299, Jan. 2020.
[27] R. Valentino, W.-S. Jung, and Y.-B. Ko, "A design and simulation of the opportunistic computation offloading with learning-based prediction for unmanned aerial vehicle (UAV) clustering networks," *Sensors*, vol. 18, no. 11, p. 3751, Nov. 2018.
[28] A. Barakat and A. Hadi, "Windows forensic investigations using PowerForensics tool," in *Proc. Cybersecur. Cyberforensics Conf. (CCC)*, Aug. 2016, pp. 41–47.
[29] M. Wazid, A. K. Das, V. Odelu, N. Kumar, and W. Susilo, "Secure remote user authenticated key establishment protocol for smart home environment," *IEEE Trans. Dependable Secure Comput.*, vol. 17, no. 2, pp. 391–406, Mar. 2020.
[30] B. Mbarek, M. Ge, and T. Pitner, "Trust-based authentication for smart home systems," *Wireless Pers. Commun.*, vol. 117, no. 3, pp. 2157–2172, Apr. 2021.
[31] A. Mughaid, A. Al-Arjan, M. Rasmi, and S. AlZu'bi, "Intelligent security in the era of AI: The key vulnerability of RC4 algorithm," in *Proc. Int. Conf. Inf. Technol. (ICIT)*, Jul. 2021, pp. 691–694.
[32] H. Touil, N. E. Akkad, and K. Satori, "Text encryption: Hybrid cryptographic method using Vigenere and Hill ciphers," in *Proc. Int. Conf. Intell. Syst. Comput. Vis. (ISCV)*, Jun. 2020, pp. 1–6.
[33] H. Touil, N. E. Akkad, and K. Satori, "Securing the storage of passwords based on the MD5 HASH transformation," in *Proc. Int. Conf. Digit. Technol. Appl.*, 2021, pp. 495–503.
[34] H. Touil, N. E. Akkad, and K. Satori, "Secure and guarantee QoS in a video sequence: A new approach based on TLS protocol to secure data and RTP to ensure real-time exchanges," *WSEAS Trans. Commun.*, vol. 20, pp. 52–62, Apr. 2021.
[35] H. Touil, N. El Akkad, and K. Satori, "H-rotation: Secure storage and retrieval of passphrases on the authentication process," *Int. J. Saf. Secur. Eng.*, vol. 10, no. 6, pp. 785–796, Dec. 2020.
[36] X.-Q. Cai, T.-Y. Wang, C.-Y. Wei, and F. Gao, "Cryptanalysis of quantum digital signature for the access control of sensitive data," *Phys. A, Stat. Mech. Appl.*, vol. 593, May 2022, Art. no. 126949.
[37] L. A. Lizama-Pérez, "Digital signatures over HMAC entangled chains," *Eng. Sci. Technol., Int. J.*, vol. 32, Aug. 2022, Art. no. 101076.
[38] L. Valeriy, S. Andrii, K. Vladyslav, P. Elena, C. Anatolii, and U. Nataliia, "Evaluation of the probability of breaking the electronic digital signature elements," in *Sustainable Communication Networks and Application*, in Lecture Notes on Data Engineering and Communications Technologies, vol. 93, 2022, pp. 639–648.
[39] A. Kuznetsov, O. Girzheva, A. Kiian, O. Nakisko, O. Smirnov, and T. Kuznetsova, "Advanced code-based electronic digital signature scheme," in *Proc. IEEE Int. Conf. Problems Infocommun. Sci. Technol. (PICST)*, 2020, pp. 358–362.
[40] R. Kudo, H. Kitahara, K. Gajananan, and Y. Watanabe, "Integrity protection for kubernetes resource based on digital signature," in *Proc. IEEE 14th Int. Conf. Cloud Comput. (CLOUD)*, Sep. 2021, pp. 288–296.
[41] D. Sunaryono, C. Z. Mukhlishah, S. Rochimah, and I. A. Sabilla, "Information systems of school financial management with digital signature recognition using MobileNet algorithm," in *Proc. IEEE Int. Conf. Health, Instrum. Meas., Natural Sci. (InHeNce)*, Jul. 2021, pp. 1–6.

HAMZA TOUIL is currently pursuing the Ph.D. degree with the Faculty of Sciences, Fes, Morocco. He has dedicated his academic journey to delving deep into the field of mobile data security. As an Active Member of the LISAC Laboratory, he has the opportunity to make significant contributions to the field of computer security. He has demonstrated an exceptional commitment to understanding the concepts of mobile data security, continually seeking ways to enhance the protection of sensitive information in an increasingly digital world. Throughout his doctoral studies, he has developed advanced technical skills, coupled with an unwavering passion for solving complex issues related to computer security. He is determined to make a meaningful contribution to the realm of mobile data security and to address the emerging challenges in this ever-evolving field. His research holds the promise of delivering innovative solutions to the current issues of digital security.

NABIL EL AKKAD received the Ph.D. degree from the Faculty of Sciences, Sidi Mohammed Ben Abdellah University, Fes, Morocco, which laid the foundation for his research expertise. He has an extensive academic and research background. He is currently an Associate Professor with the Department of Computer Science, Sidi Mohammed Ben Abdellah University. He is also a Professor of computer science with the National School of Applied Sciences (ENSA), Fes, a prestigious institution affiliated with Sidi Mohammed Ben Abdellah University. In this capacity, he not only imparts knowledge but also engages in cutting-edge research in the field of computer science. He is an integral part of the LISA Laboratory, where his contributions to research have been instrumental. The LISA Laboratory is renowned for its work in various areas of computer science and his membership underscores his commitment to advancing the field. His research interests include artificial intelligence and image processing. These areas reflect his dedication to understanding and improving the capabilities of computer systems, making valuable contributions to the ever-evolving world of technology.

KHALID SATORI is currently a Professor with the Department of Computer Science, Sidi Mohammed Ben Abdellah University, Fes, Morocco. He possesses a wealth of academic and research experience in the field of computer science. His academic journey and career have been strongly focused on advancing the discipline. As a member of the LISAC Laboratory, he plays a key role in researching and developing new knowledge in the field of computer science. The LISAC Laboratory is renowned for its cutting-edge contributions in various areas of computer science and his active participation as a member underscores his commitment to research and innovation. His research interests are diverse, but primarily centered on two crucial areas: artificial intelligence, which encompasses advanced data processing and automated decision-making techniques, and computer systems security, where he contributes to ensuring the protection of systems against threats and vulnerabilities. His research aims to enhance the understanding and implementation of artificial intelligence in various contexts, and to strengthen the security of computer systems in an ever-evolving digital world. His work holds the promise of making significant contributions to these vital areas of computer science.

NAGLAA F. SOLIMAN received the B.Sc., M.Sc., and Ph.D. degrees from the Faculty of Engineering, Zagazig University, Egypt, in 1999, 2004, and 2011, respectively. She was with the Faculty of Computer Science, PNU, Saudi Arabia. Since 2015, she has been a Teaching Staff Member with the Department of Electronics and Communications Engineering, Faculty of Engineering, Zagazig University. Her current research interests include digital image processing, information security, multimedia communications, medical image processing, optical signal processing, big data, and cloud computing.

WALID EL-SHAFAI (Senior Member, IEEE) was born in Alexandria, Egypt. He received the B.Sc. degree (Hons.) in electronics and electrical communication engineering from the Faculty of Electronic Engineering (FEE), Menoufia University, Menouf, Egypt, in 2008, the M.Sc. degree from the Egypt-Japan University of Science and Technology (E-JUST), in 2012, and the Ph.D. degree from FEE, Menoufia University, in 2019. Since January 2021, he has been a Senior Researcher with the Security Engineering Laboratory (SEL), Prince Sultan University (PSU), Riyadh, Saudi Arabia. He is currently a Lecturer and an Assistant Professor with the Department of Electronics and Communication Engineering (ECE), FEE, Menoufia University. His research interests include wireless mobile and multimedia communications systems, image and video signal processing, efficient 2D video/3D multi-view video coding, multi-view video plus depth coding, 3D multi-view video coding and transmission, quality of service and experience, digital communication techniques, cognitive radio networks, adaptive filters design, 3D video watermarking, steganography, encryption, error resilience and concealment algorithms for H.264/AVC, H.264/MVC, and H.265/HEVC video codecs standards, cognitive cryptography, medical image processing, speech processing, security algorithms, software-defined networks, the Internet of Things, medical diagnoses applications, FPGA implementations for signal processing algorithms and communication systems, cancellable biometrics and pattern recognition, image and video magnification, artificial intelligence for signal processing algorithms and communication systems, modulation identification and classification, image and video super-resolution and denoising, cybersecurity applications, malware and ransomware detection and analysis, deep learning in signal processing, and communication systems applications. He serves as a reviewer for several international journals.