International Journal of Computing and Digital Systems
2025, VOL. 17, NO.1, 1-12

http://dx.doi.org/10.12785/ijcds/1571032920

Protecting Cloud Service Providers Based on an Efficient Password-based Authentication System

Saja J. Mohammed [1]

[1]*Department of computer science , College of computer science and mathematics, University of Mosul, Mosul, Iraq*

Received 25 May 2024, Revised 6 December 2024, Accepted 9 December 2024

Abstract: Password-based authentication systems are one of the most popular authentication systems. They are an essential way to protect many online applications and accounts. Although it represents only a single factor of authentication, when constructed and implemented properly, it can represent a relatively high level of security. Cloud computing services also use this type of protection. Text passwords are used in the cloud environment to authenticate and authorize access to cloud services or preserve private and sensitive data kept in the cloud. For this reason, a robust password is needed, which must be safe from possible attacks. That requires the password to be as strong as possible; On the other hand, it will produce a problem of forgetting them. For these types of problems, this paper proposes a new password generator algorithm based on the 4D hyperchaotic system and a genetic algorithm. If the system receives the same input, the proposed algorithm will generate the same strong password. On the other hand, a genetic algorithm is also used to enhance the generated password if it loses one of the determined conditions to be considered a robust password. Then the generated password will be used to protect user data stored with any cloud provider. Testing results gave a 79.8 value of bit entropy for the generated passwords and 93% of the generated passwords were classified as "very strong passwords". Finally, when testing the system against the credential stuffing attack, the system shows mitigating the risk of this attack by a rate of 95%.

Keywords: Cloud computing security, Authentication system, Genetic algorithm, 4D hyperchaotic system, SHA-256, Password-based authentication systems.

1. Introduction

In the last era, cloud computing was and is still the most trending service, however, there are many concerns surrounding its usage. Account hijacking is a major risk to cloud data security. Two prevalent instances of extremely inadequate password security are the reuse of passwords and the use of weak passwords. Because a single stolen password may be used on several accounts, this problem makes phishing schemes and data breaches more harmful [1],[2],[3].

On the other side, the cloud computing ecosystem is at risk of attacks because of the distant locations of resources and virtualization technologies. Because every client in cloud computing has access to the same resource location, there is a risk to system security. Furthermore, there is a problem with integrity when it comes to transfer, storage, and retrieval. In addition, there isn't a single standard to guarantee data integrity. Because various vendors use distinct structures for data access and storage, clients are unable to quickly switch suppliers and are stuck with only one. The requirement for a consistent standard for encryption, decryption, and client control is the other issue raised. When sensitive data is sent and stored in a cloud environment without adequate encryption and security, there is an increased risk of attacks [4], [5].Therefore, organizations must put robust security measures in place to protect user data, such as encryption and access limitations, and promptly alert clients in the case of a breach. Online, sensitive data is protected by several methods, including frequent data backups, authentication, and encryption [4].
There are three various authentication mechanisms; the first is "password-based authentication". It is a method used for verifying the user's identity by demanding they offer a password. "Password-based authentication system" is one of the most widely used methods for the digital world authentication [6],[7].
A client can safely use password-based authentication to gain access to services like emails hosted by a service provider. To keep unapproved users from using their services, the client needs to give the service provider their login and password. If the username and password combination supplied by the client matches the username and password combination in the service provider's

E-mail address: sj_alkado@uomosul.edu.iq

database, the client is allowed access to the service he has requested. The primary benefit of password-based authentication is its ease of use and memorization [8], [9]. "Token-based authentication" verifies a user's identification before allowing them to access a server, network, or other protected system. During this validation procedure, a security token that the server provides must be used. The service's duties also include enabling user inquiries and security token verification. Smart cards, USB keys, mobile devices, and Radio Frequency Identification (RFID) cards are examples of electronic devices often used for identification and authentication. Every time a device is used, a new password is created, making it possible to use a security token to log in to a computer or virtual private network. To accomplish that, the user must input the password (which is generated by the device) into the equivalent prompt [10].

"Biometric-based authentication" is a security process based on an individual's distinct physical biometric characteristics. It is employed to regulate entry to both digital and physical resources [11] [12] [13].

The paper focused on text-based authentication systems, according to their popularity and easiness. It takes into consideration a strong text password, which can be regenerated, and then used to protect cloud service providers. Traditional text-based authentication involves entering login credentials directly as alphanumeric characters in the login box, using a text password and username. If compared to other authentication methods, text-based passwords are less costly and require less time to create [14].

A password is a string of alphanumeric characters and symbols that are used to verify a user's identity, provide access to a resource, or authenticate a user. Online privacy may be more easily compromised by attackers and criminals if bad password practices are used. On the other hand, using numerous passwords may be complex, susceptible to difficulties in remembering several passwords, and the temptation to reuse a single login credential across many accounts, among other concerns [15].

The user's ability to develop strong password habits, such as frequently changing passwords, avoiding using the same login information across different systems, and producing long, and complex passwords (combination of special characters, symbols, and numbers, is a major factor in how security level of the used passwords are[15].

Later, text password-based authentication becomes risky because users fail to recall text passwords' length and strength. Users are therefore likely to choose weak passwords to improve recall. Furthermore, using techniques like dictionary cracking, guessing, shoulder surfing, and other methods, hackers may easily get the passwords [14], [16].

Across all cloud risks and the problems of text password authentication systems, many works have been interested in cloud security, privacy, and authentication. They take into consideration the cloud security requirements; one of these requirements is authentication, which is the focus of this paper. The paper is focused on the most popular and easy-to-use "text-password authentication system," trying to cover the problem of an unmemorable, complex password. The paper provides a complex text password from a memorable code that is entered by the user. Then use the generated password to protect the user account of any cloud provider. The generated robust password is tested by pre-defined conditions. The generated algorithm is based on a hyper-chaotic system integrated with a genetic algorithm, which guarantees that the generated password is robust according to the predefined conditions. The robust password, finally, is used to investigate the cloud account.

The paper is structured as follows: After the general introduction and the fundamentals of the authentication which are written in Section 1, section 2 there is a Literature Review of the related works. The 4D hyperchaotic system was explained in Section 3, whereas Section 4 presented the principle of Genetic Algorithm (GA). Then the Secure Hash Algorithm (SHA) is explained in section 5. The proposed password generator was comprehensively shown in Section 6, and the results of the discussion were explored in Section 7. Last, the paper is finished with its conclusion in section 8.

2. Literature Review

Over the past few years, some text-based authentication techniques have been proposed for use in protecting cloud computing environments. This section emphasizes certain works that are interested in that goal.

In 2020, there was a paper [17] presented a unique multifactor safe mutual authentication technique based on hashing that was secure against replay, forgery, and MITM attacks. It incorporated mathematical hashing characteristics, certificates, nonce values, and traditional user IDs. We tested our suggested approach on the Microsoft Azure cloud, and the outcomes were assessed. The Scyther tool was used for the security analysis, while GNY belief logic was used for the formal analysis. The findings showed that robust, secure authentication could be provided by the suggested system.

In the same year[2], and with the aid of a password, biometrics, and a mobile device, the suggested system offered strong three-factor authentication, dependable data protection, and the ability to counteract current attacks. It also provided additional benefits over the previous strategy. This scheme offered the majority of increased security functions in addition to addressing security problems.

In 2021 [18], a hybrid graphical user authentication technique based on questions is proposed for portable cloud computing environments. The proposed approach has advantages over the recall and recognition-based approaches without maintaining any private data on cloud servers. Surveys and experimental studies were done to find out how satisfied users were with the proposed scheme's usability and performance. The study's findings demonstrate that the suggested approach is safe, user-

friendly, and impervious to possible password assaults like shoulder surfing and brute force password guessing.
In 2022,[19] there was a paper that proposed a cloud computing authentication system using a virtual smart card ID produced with clutter techniques which was used to prevent the fake cloud servers and also protected that cloud data from many hackers.
In 2024, there was a paper [20] that proposed an application-based multi-layer, Multi-Factor Authentication (MFA) software that improved cloud security measures. By using application-based multi-factor authentication, the solution offered a better method of securing cloud computing (MFA). Based on user profiles, the system ensured safe access by utilizing encryption mechanisms. These profiles were made up of legitimate usernames, passwords, and application-generated token numbers. Furthermore, the system was made more secure by combining location checks with the Time-based One-Time Password (TOTP) method (IETF RFC 6238), which strengthened the overall security measures. The system underwent a comprehensive testing process, with a test web application hosted on the cloud server receiving special attention. The outcome confirmed that all three of the integrated authentication elements in the application worked as intended.
Also, in 2024, [21] according to the user authority, a paper suggested a safe and dependable user authentication method that gave authorized people access to the user's PHR stored in the cloud. By combining a smart card and password, the suggested authentication technique enabled the owner and other authorized users to access the pertinent personal data by logging into the system. In that investigation, users' identities were authenticated, and hostile infiltration and theft were successfully prevented through the deployment of an authentication technique based on bilinear pairing.
As seen in the related works, the mix between Genetic algorithms and chaotic systems was not utilized in the previous works to generate complex passwords. So this paper uses these methods in an interleaved way to show the result of these combinations. in generating strong and complex passwords.

3. 4D Hyperchaotic System

A chaotic system is a deterministic system that can produce a large number of great pseudorandom sequences because it is very sensitive to changes in the starting value. This is congruent with the keystream needed for many securities applications according to the generated sequence's diffusion and scrambling[9],[22].
A state of chaotic and seemingly random behavior that results from deterministic equations is referred to as chaos in a dynamical system. While there could be several variables in a conventional chaotic system that behave in complicated and unpredictable ways, these variables are often coupled in a somewhat straightforward way. The quantity and complexity of variables involved in a system determine whether it is chaotic or hyperchaotic. A subset of chaotic systems with four or more variables is called hyperchaotic systems [23] [24].

A hyperchaotic system is defined as a chaotic system that contains two or more Lyapunov exponents. Additionally, the phase space that embeds the hyperchaotic attractor must have a minimum dimension greater than three. These suggest that, compared to chaos, hyper-chaos features more intricate, dynamic events. Because of its larger dimensions, more randomness, and unpredictability, hyper-chaos has more potential uses than chaos, including secure communications, nonlinear circuits, lasers, and other devices [25].
Mathematically, any chaotic system with more than one "positive Lyapunov exponent" and simultaneously richer and more extended dynamics in the phase plane provides the foundation for the mathematical characterization of a hyperchaotic system. When k chaotic systems are coupled, a hyperchaotic attractor with n positive Lyapunov exponents is obtained; the transition from chaos to hyper-chaos demonstrates that the attractor's dimension increases and the second Lyapunov exponent grows continuously. The hyper-chaos exhibit more complex dynamical behaviors than the normal chaotic systems. By including a straightforward state feedback controller, such as Chua's circuit, Chen system, or Lorenz, a hyperchaotic system may be produced both computationally and experimentally [26].
As explained above, hyper-chaotic systems have applications in a wide range of fields, including secure communication, encryption, and random number generation. They are of importance in these domains because of their potential to make it more difficult for an opponent to predict or comprehend the behavior of the system due to their increased complexity. Researchers study hyper-chaotic systems for their theoretical properties and practical applications in fields that benefit from complexity and unpredictability [27].
Multiple variables in hyper-chaotic systems lead to even more complicated and chaotic behavior, frequently with extra degrees of nonlinearity and complexity in their dynamics. Elevated positive Lyapunov exponents, a crucial sign of chaos, measure the system's susceptibility to initial conditions and characterize these systems. In a hyper-chaotic system, the presence of more than one positive Lyapunov exponent typically suggests that many variables are developing chaotically in different directions. Equation (1) explains the hyper-chaotic system that is utilized in this paper [27], [28]:
Equation 1

$$\begin{aligned} \dot{x} &= -x - 4y, \\ \dot{y} &= x + z^2 + aw, \\ \dot{z} &= 1 + x, \\ \dot{w} &= -by, \end{aligned}$$

in which $[x, y, z, w]^T R^4$ is a state vector and the two positive constant parameters are a and b. This system

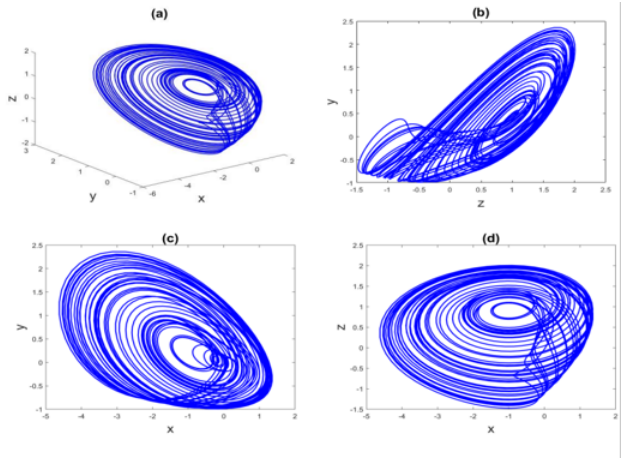

Figure 1. 4D hyperchaotic Sprott S system

exhibits chaotic hidden attractors as explained in Figure. 1.

4. Genetic Algorithm

In informatics and computational mathematics, the term " Genetic Algorithm" (GA) refers to the broad category of evolutionary algorithms. These algorithms, which focus on bio-inspired operators like selection, convergence, or mutations, are widely employed to produce excellent solutions to optimize and search problems. In recent years, metaheuristic algorithms have been used to address intricate real-world issues that emerge from many domains, including economics, politics, management, and engineering. The algorithms may be roughly categorized into two groups: single-solution algorithms and population-based metaheuristic algorithms [29][30].

GA is a well-known algorithm based on the biological phenomenon of evolution. The genetic algorithm follows the Darwinian concept of natural selection, favoring individuals most well-suited to their environment. The Genetic algorithm has three essential components: chromosomal representation, fitness selection, and operators that emulate biological processes. The operators inspired by biological processes include selection, mutation, and crossover. During selection, chromosomes are chosen based on their fitness value to undergo further processing. The crossover operator selects a random locus and modifies the sub-sequences across chromosomes to generate progeny. During the mutation process, some segments of the chromosomes undergo random flipping, influenced by probability [29].

The majority of genetic algorithms are probability-based criteria; however, because they incorporate historical data, which makes them sophisticated and complex, they also perform well against local random search, which employs random solutions and is unable to identify the best possible solutions[31]. The classical Genetic algorithm is explained in the following algorithm [32]. Where Figure 2 explains the steps of Genetic Algorithm [29].

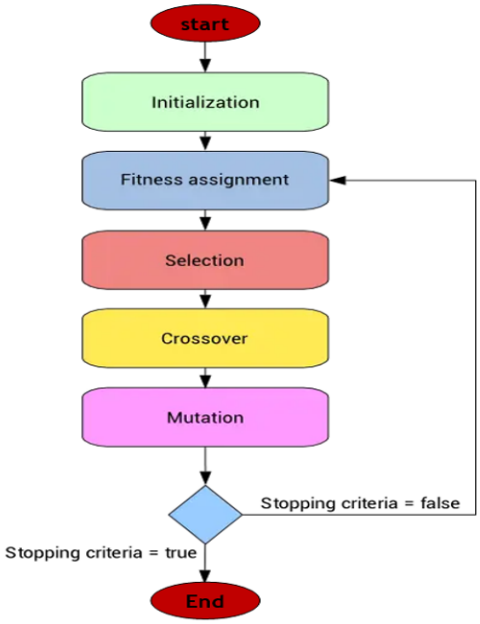

Figure 2. Algorithm of the classical genetic Algorithm

The steps of obtaining fitness value using GA can be explained as follows:

Input: Population p

Output: Fitness value

1. Randomly consider populations p.
2. Find out how fit the population is.
3. Continue doing steps 4 through 7 until convergence.
4. Select a parent at random from the population.
5. Creates a new population by use of crossover operation.
6. To perform mutation operation, introduce arbitrary chromosomes into a new population.
7. Determine the fitness of recently created populations.
8. Output the fitness value.

Time limits, fitness limits, generations, function tolerance, constraint tolerance, and other factors are some examples of GA ending criteria. Genetic algorithms have the following characteristics [33]:

a. The genetic algorithm is a stochastic optimization approach that doesn't require an excessive amount of mathematics to solve the optimization issue. The intrinsic properties of the problem do not require consideration throughout the search process due to its evolutionary nature. It may directly operate the structural item and has a broad variety of application scenarios and ranges, whether it is discrete or continuous, linear or nonlinear.

b. The genetic algorithm utilizes the objective function as the search function, relying solely on the fitness function to assess an individual without the need for complex derivations or additional information. It then performs genetic operations to facilitate information exchange among individuals within the population, thereby

minimizing reliance on problem-solving and enhancing flexibility.
c. The multi-point parallel search strategy adopted by the genetic algorithm prevents the search from converging to the local optimal solution since it is not confined to a single point. Fast real-time optimization is now possible because of the genetic algorithm's parallel settlement features, which also allow us to increase the algorithm's operating speed through large-scale parallel computing.
d. The parameters themselves are not operated upon by the genetic algorithm; rather, it codes the parameters. Rather than relying on certainty, its optimization techniques take into account the equivalent likelihood. Essentially, it is a comprehensive framework for resolving system optimization issues, which has much advanced beyond just an optimization method.
These days, Genetic algorithms are used in a lot of places. Examples include the Internet of Things, smart traffic signal systems, intelligent routing in MANET of smart devices, blockchain technology, cloud computing's load balancing and work scheduling, and engineering pedagogy [1itekatoch2021review.

5. Secure Hash Algorithm

The "Secure Hash Algorithm" (SHA) has been the most often utilized hash function in recent years. The fact that nearly all other commonly used hash functions had significant cryptanalytic flaws by 2005 meant that SHA was essentially the only standardized hash algorithm still in use. Since SHA's architecture closely resembles that of Message Digest 4 (MD4), it is based on the MD4 hash algorithm [34].
Preprocessing and hash computation are the two processes that make up each SHA algorithm. Preprocessing includes initializing data to be utilized in the hash calculation, padding a message, and parsing the padded message into m-bit blocks. From the padded message, the hash computation creates a message schedule. It then iteratively constructs a sequence of hash values using the schedule, functions, constants, and word operations. The message digest is calculated using the final hash value that is produced by the hash computation [35].
Nowadays, SHA-256 is the most often used SHA function because it offers a high level of protection given the capabilities of modern computers. A 256-bit hash value is computed by SHA-256 for a 512-bit input message. The hash value for a length message might need to be calculated by the genuine program. The message is split up into several 512-bit data blocks in these situations. Padding is added if the final block has less than 512 bits. Figure 3 displays the hash computation for a lengthy message The "SHA-256 algorithm" determines intermediate hash values one block of data at a time. For the hash computation of the next block, the hash value from the previous block serves as the initial hash value. The result of the final data block is considered to be the hash value of the whole message. Figure 4 shows an overview of SHA-256 operations [36]. Table I detects

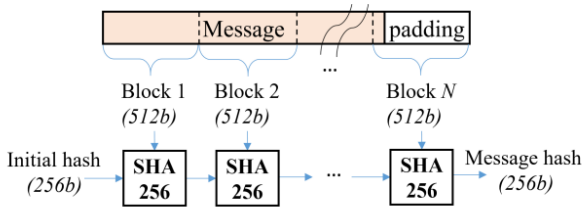

Figure 3. The hash computation for a lengthy message

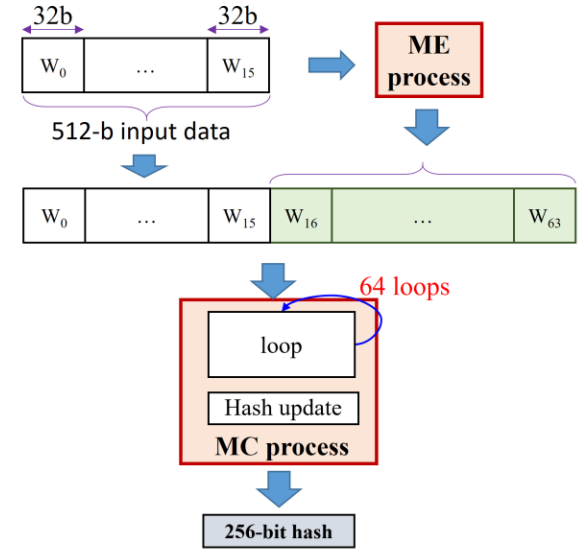

Figure 4. An overview of SHA-256 operation

the differences between SHA-256 and other traditional and new hashing methods.

6. The Proposed Password-based Authentication System

Using a "text password-based authentication system" to give access for a user to any account needs a long, and complex password, which mean a strong password. That is often what creates the forgetting problem. The password-forgetting problem may force the user to save it in a close place to remember it when needed, which exposes it to the risk of violation.
The proposed method tries to solve the problem of using text-based passwords in an authentication system, especially in cloud provider accounts. It generates strong and complex passwords from a rememberable code. The proposed system is divided into two major phases:

Phase A. Generating a strong password.
Phase B. Designing a text-based authenticated model to protect cloud accounts.
Passwords are generated using a 4D hyperchaotic system

TABLE I. The differences between SHA-256 and other methods

Algorithm	Time Complexity	Computational Complexity	Security	Usability
MD5	Fast (O(1))	Low (constant time)	Weak; vulnerable to collisions and pre-image attacks	Easy to implement; widely supported, but not recommended for security
SHA256	Fast (O(1))	Low (constant time)	Strong; resistant to pre-image and collision attacks, but faster than password hashing algorithms	Easy to implement; widely supported
SHA512	Fast (O(1))	Low (constant time)	Strong; similar security properties to SHA-256, but produces a larger hash	Easy to implement;widely supported
bcrypt	Moderate (O(n log n))	Moderate (depends on cost factor)	Strong; designed to be slow, making brute-force attacks more costly	Simple to use with libraries; widely supported
PBKDF2	Moderate (O(n))	Moderate (depends on iterations)	Strong; uses salt and configurable iterations, but can be faster than bcrypt and Argon2	Simple to use, but less flexible than Argon2
Argon2	Moderate (O(n))	High (configurable memory usage)	Very strong; memory-hard design makes it resistant to GPU and ASIC attacks	More complex to implement but gaining support

integrated with genetic algorithms. Each one of the previous principles plays a great role in the password generation stages. The chaotic system provides the required randomization that helps to create the initial passwords. It depends on the user input value to begin its job. By number. of steps, the password is created. Where the genetic algorithm is used to test and increase the robustness of the generated password, its stages are applied when the created password has some weak points.

It is worth noting that the restrictions imposed in the proposed algorithm to consider the password a strong one are:

- At least the password length must be 10 characters.
- At least 50% of it includes capital letters (A-Z), and small letters (a-z).
- The other ratio is divided between the number characters (0-9), the special characters (!"#$%&()*+,-./:;¡=¿?@[]^_—¯}).
- Do not contain respective similar characters.
- Every two consecutive alphabetical characters in the generated password must not be consecutive in the alphabet.

The proposed algorithm has one limitation: It requires at least two characters to generate a good result. When one character is entered into it, a weak password is generated. So, the simi-long input to the proposed algorithm generates a promisingly strong password.

A. *Phase A, Generating a strong password*

The following is an explanation of the steps in the suggested password-generating algorithm (phase A):

The proposed algorithm to generate robust passwords:

Input: user initial data (D), password length Output: Robust password used in cloud services provider.

1- Accept input data from the user (say UD), with password length (from 8 to 16). UD must be at least 3 characters.

2- Convert UD to ASCII code format (say UD asci).

3- Using UD ASCII, compute 4D hyperchaotic system initial value (X).

4- Generate chaos sequence using the 4D hyperchaotic machine, then generate an initial password (PWi) from the resulting sequence.

5- Using the specific robustness conditions to check the robustness of the PWi : if PWi reaches all condition exit with PWi,
else continue.

6- To begin with the genetic algorithm, PWi will be the first chromosome (Y1).

7- Changing the parameters of the 4D hyperchaotic system and generating a new password will be the second chromosome (Y2).

8- Begin with the GA operator by applying the following operations:

a) Crossover operation between Y1 and Y2 (using single-point crossover (char to char)).

b) Mutation operation.

9- If the result does not satisfy the conditions of a robust password:
Take the most robust chromosome between Y1 and Y2; consider it as Y1; go to 7.
Else exit with the robust password (PW robust).

B. *Phase B, designing a text-based authenticated model to protect cloud account*

To protect the cloud account from any unauthorized access, a proposed model was suggested based on a text password authentication system. The proposed model used the generated password in phase (A). Phase (B) has three options: sign up/registration, login/authentication, and recovering missing passwords phases. All phases need to input the e-mail address of the user. But the operation direction takes another path in each one of them.

1) *Sign up/ Registration/Identification Phase*

A new user always needs to register (or sign up) on any specific account. Normally, any registration needs a user name, user e-mail address, and password as inputs to the authentication system. So, the proposed registration phase can be detailed as follows:

1. The first step in completing the registration for a cloud account is to enter the user's name and email address.
2. The second step is to enter a robust password (PW). The robustness of the input PW is checked (according to the previously explained conditions); if it passes all conditions, go to 5; else, open a new window to generate robust PW

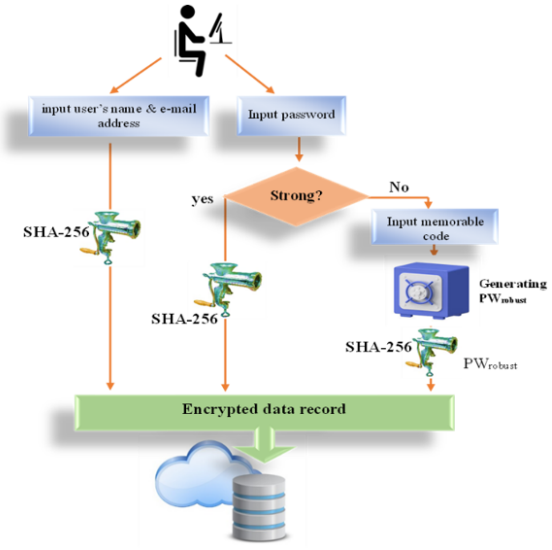

Figure 5. The proposed registration steps

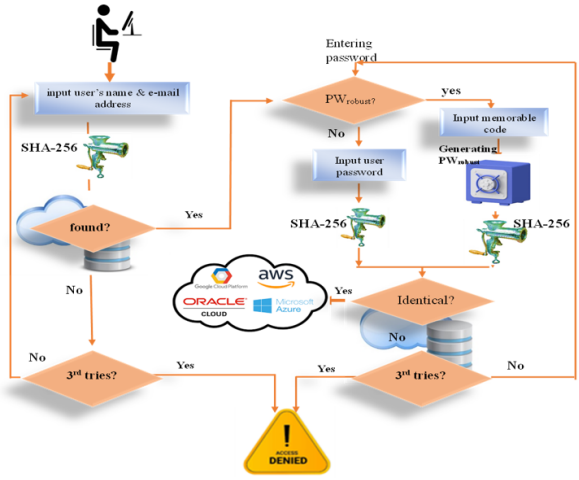

Figure 6. The proposed login steps

(PW robust).
3. Ask the user to input a memorable code. This operation is the beginning to generate the robust strong password (PW robust). The user can then use the output PW.
4. Enter the generated password (PW robust).
5. Hashing all input data (using SHA-256 hash algorithm).
6. Create an encrypted user's record (user name, e-mail, and PW), then save it in a cloud-secure database. Figure 5 shows the steps of the proposed registration (sign up) algorithm.

2) Login/Authentication phase

When a trusted user tries to sign in to his/her account in the cloud provider, the following steps are applied:
1. Receiving the user's name and encrypting them by SHA-256.
2. Search in the cloud password's database for the entire encrypted user name; if there is an identical encrypted one: continue login operations; else the user can try to input a new user name 3 times; if after that he/she fails, exit with an error message and deny access.
3. If the user is authorized, continue to input the user's email address and his/her password.
4. For input generated password (PW robust): open the dialog window to input the user memorable code.
5. Generate a strong password (PW robust) using user code; the generated password will be entered directly with user information.
6. After sending this information, they hashed and continued to compare it with the stored one, which was placed in the same row as the identical user's name.
7. If they are identical, the user is authorized, else give the user three times to reinput his information; if fail, deny the user access.
Figure 6 explains the login process in the proposed algorithm.

3) Reset the password

The action of this phase is to address the forgetting problem of the password code (not the generated password itself) or reset the stored password. The input code is required essentially to generate the same strong password each time the user tries to log into a cloud account using the proposed system. When the code is forgotten, a new code with a newly generated password must be used.
The steps for resetting the complex password are:
1. Asking the user to input the login (registered) e-mail.
2. Checking if the email is stored in the system DB, if it is not found, exit with an error message " unauthenticated user" else continue.
3. Sending verifying request messages to the input e-mail.
4. If verified correctly, ask the user to input a new complex password.
5. If not a complex password, begin to generate a new complex password using a new memorable code and hash it with SHA-265.
6. Exchanging the stored one with it. Figure 7 detects the illustrated steps for resetting the complex password.

C. Using the proposed algorithm with the existing cloud authentication systems

According to the variety of real cloud providers, the proposed authentication system can be used with real cloud systems, as explained in the following subsections:
a. Text password-based cloud authentication systems: the proposed authentication system can be used with this type of cloud authentication system as explained in the previous sections. Amazon Web Services (AWS) is an example of the most popular cloud provider that uses this type of authentication, a single-factor authentication system based on text password authentication.
b. Multi-factor authentication (MFA)-based cloud

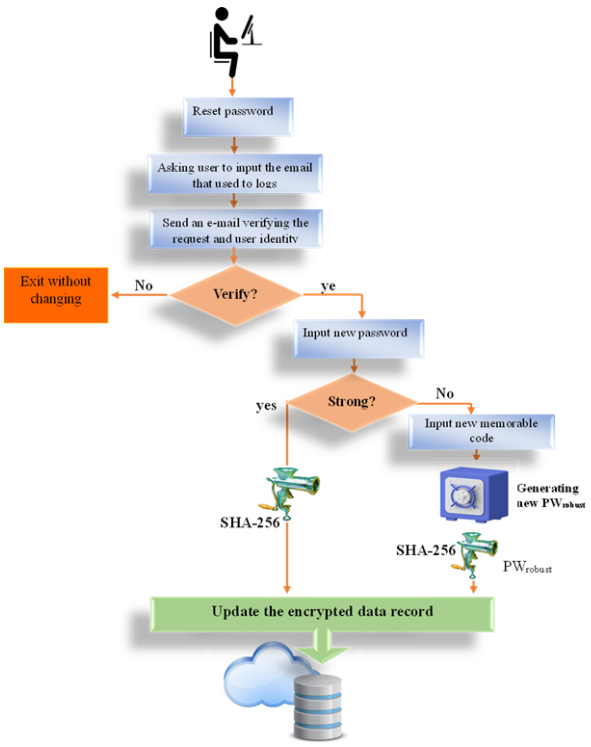

Figure 7. Resetting the complex password process

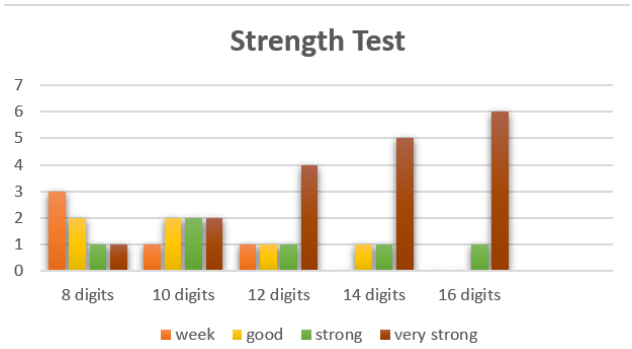

Figure 8. The results of testing various lengths of the generated passwords

authentication system: several cloud providers used this type of authentication system. They use multiple factors of authentication, but the main one is the password and the other may be face recognition such a cloud provider uses this kind of authentication system as Asure cloud provider. In this case, the proposed system is integrated with the face recognition stage to accomplish the authentication process at this type of cloud provider.

7. Results and Discussion

The usage of simple passwords is one of the primary vulnerabilities, hackers use wordlists that are easily accessible. Furthermore, passwords are extremely susceptible to dictionary assaults since they depend on words from dictionaries, phrases that are often used, or keyboard patterns. Words contained in dictionaries are not the only targets of dictionary assaults. Hackers have produced wordlists with specific combinations of popular passwords, passwords that have been compromised in the past, and even personally identifiable data like addresses, birthdays, and names. This implies that your password can still be weak if it's simple to link it to personal data, even if it doesn't contain any dictionary words.

Many practical experiments were applied based on the proposed algorithm. All these experiments were applied on a PC (CPU=Core i5, RAM= 16 G) with Windows 11 64-bit operating system and Visual C# 2022 programming language. The experiments tested the strength of the generated password, the time consumed, and the system stability against a credential-stuffing attack. Table 2. shows some examples of generated passwords of various lengths using the proposed algorithm.

As Table II shows, the generated passwords (explained in Table 1) underwent many tests to verify their robustness. Some of the most popular websites are used to test their robustness [37][38][39][40]. The results of testing explained in Figure 8 show that among various generated passwords with various lengths, 95% of the passwords of length 16 were classified as "very strong.". Only 5% of them were classified as weak passwords (those of length 8).

The elapsed time for generating passwords using the proposed algorithm is calculated. Table III explains the result of the calculation of the time required to generate the complex password, as well as the time to sign up and log-in process. The results show that the time is linearly proportional to the length of the password. The maximum time (2.998381) couldn't be noticed by the user. The proposed algorithm does not affect the length of user input compared with the required length of the generated password.

The crackability of the generated passwords was also tested using the "Password Checker Online" online service. [41]. This service depends on "the brute force attack" to check the crackability of the generated passwords using various machines. Table IV explains the results of some test samples of the passwords generated using 'Password Checker Online'. According to the results of this test, the proposed algorithm is not easily penetrable to brute force attacks.

As seen, The generated passwords remained resistant to brute force attacks that were directed at passwords from different types of devices with different speeds and capabilities, ranging from normal speed to super and super speed, as in medium-sized botnet machines. The results showed that the shortest time taken by a brute force attack on passwords generated according to the proposed method

TABLE II. Some examples of the generated passwords with various lengths

Case	User Input	Password length	Generated password
1	config	8	}fA:#4h&
2	Compute	9	Bs3XzV1
3	myname	10	k̇&4$d - W$?9
4	Net2024	11	(4p-$;¿Ã,S
5	Mobile	12	m)A,gB5$^9/\tilde{X}$
6	mobile	13	}fhA:4h&B50%
7	pass	14	9BS*8zg=X:af
8	Word	15	fZ@?4W¡f%zrHnuA$
9	Iris	16	BkS{j8;zg=j4]A2

TABLE III. The elapsed time (in Milliseconds)

Password length	Password generation average time	Average time for registration phase	Average time for login phase
8	0.0178267	0.145073	0.553841
9	0.0444572	0.197459	0.205744
10	0.1284238	0.338456	0.490146
11	0.1927973	0.382394	0.510439
12	0.2358594	0.468560	0.500964
13	0.5745209	0.737553	0.799523
14	0.7710640	0.948576	1.073437
15	0.8702945	1.699576	2.305233
16	1.0109572	2.677465	2.998381

TABLE IV. The estimated time of brute force attack using Password Checker Online using various machines

Password length	Standard Desktop PC	Fast Desktop PC	GPU	Fast GPU	Parallel GPUs	Medium size botnet
8	2 years	6 month	2 month	1 month	4 days	1 minute
9	2x103 years	46 years	18 years	9 years	11 month	2 hour
10	208x103 years	52x103 years	21x103 years	10x103 years	78 years	6 days
11	20 x106 years	5 x106 years	2 x106 years	277 x 103 years	78x103 years	20 years
12	2x109 year	459 x106 year	184x106 year	92 x106 year	9 x106 year	2x103 years
13	173x109 year	43x109 year	17x109 year	9x109 year	863x106 year	173x103 year

is one minute on a password of 8 characters long and using a medium-sized botnet device, while the attack takes a period of 173x103 years on a same device used to guess a password of 13 characters long. In contrast, an 8-character password takes a full year to discover with the same attack on a regular personal computer. Another 13-character password takes 173x109 years on the same device. These results confirm and prove the resistance of the generated passwords to a brute force attack in which the time factor is the main and final criterion for the success of this type of attack on passwords. The complex password significantly increases the resistance against dictionary attacks due to its increased length and the variety of characters. By incorporating uppercase letters, lowercase letters, numbers, and special symbols, the password creates a larger pool of possible combinations. This complexity makes it far more time-consuming for attackers to crack using precompiled lists of common passwords and phrases. Also, that makes them more resistant to dictionary attacks which exploits the propensity of users to choose common or weak passwords. The entropy of passwords is also calculated using the "password strength" online service [42]. The entropy measures the unpredictability of the generated passwords. The bigger the entropy value, the harder the password cracks. When a password's entropy falls between 28 and 35 bits, it is considered extremely weak. It is reasonable (i.e., reasonably secured for network and enterprise usage) if it falls between 36 and 59. A strong password is indicated by entropy values greater than sixty [43]. The proposed system-generated passwords in this checker have entropy values ranging from 39 (minimum) to 79.8 (highest). That proved the difficulty of cracking the generated passwords using the proposed algorithm, as explained in Figure 9.

The system was attacked by a "credential stuffing attack", this type of attack is quite similar to a brute force attack, but it makes use of information from earlier breaches to make password guessing easier[44]. The limitation imposed on the proposed system about the number of login tries (3 attempts in 5 minutes) mitigates this type of attack

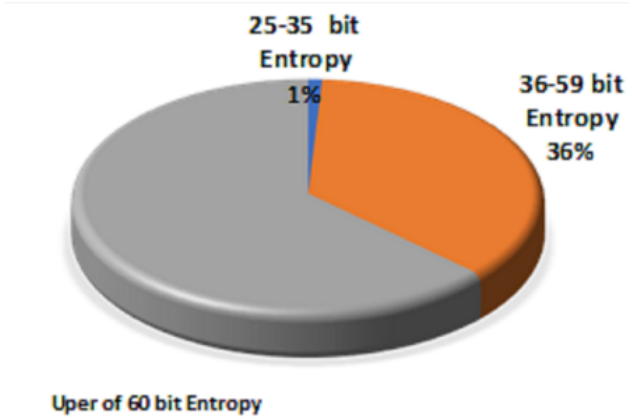

Figure 9. The resulted bit entropy

and defeats it. The proposed system proved its resistance to this type of attack. Table V shows several attempts to apply an unauthenticated login and the state of the system against them. After 30 attempts to apply unauthenticated login to the proposed system. The system gave 95% of the "blocked" state to mitigate the credential stuffing attempts.

8. Conclusion

In this paper, an authentication system for cloud provider account protection is proposed. The system was based on the text password, a robust and complex one only, to investigate user-authenticated access. The paper proposes an authentication system with a complex password generator based on the 4D hyperchaotic system and the genetic algorithm. The 4D hyperchaotic system generates the initial password using user input through ordered steps. If the generated password does not achieve all the conditions considered for a complex password, the genetic algorithm is used to enhance the resulting password. The proposed algorithm proved that the generated passwords are safe against brute-force attacks. That fact was concluded when the attack took a long time to discover the password. 63% of the generated passwords have a bit entropy greater than 60 (that means robust password), whereas 36% have acceptable strength. Also, the practice proved that elapsed time is linearly proportional to the length of the password and affected by user input. The maximum elapsed time does not exceed 0.19 milliseconds, which makes the algorithm acceptable to be used in everyday life. Also, the proposed system shows its stability against the credential stuffing attack by 95% of the blocked state against this type of attack.

9. Acknowledgment

The authors are very grateful to the University of Mosul/College of Computer Science and Mathematics for their provided facilities, which helped to improve the quality of this work.

References

[1] S. ACHAR, H. PATEL, and S. HUSSAIN, "Data security in cloud: A review," *Asian Journal of Advances in Research*, pp. 1099–1106, 2022.

[2] S. Nalajala, B. Moukthika, M. Kaivalya, K. Samyuktha, and N. Pratap, "Data security in cloud computing using three-factor authentication," in *International Conference on Communication, Computing and Electronics Systems: Proceedings of ICCCES 2019*. Springer, 2020, pp. 343–354.

[3] K. Ali and K. Alsaif, "Palm print features for personal authentication based on seven moments," *AL-Rafidain Journal of Computer Sciences and Mathematics*, vol. 14, no. 2, pp. 63–74, 2020.

[4] M. H. Barkadehi, M. Nilashi, O. Ibrahim, A. Z. Fardi, and S. Samad, "Authentication systems: A literature review and classification," *Telematics and Informatics*, vol. 35, no. 5, pp. 1491–1511, 2018.

[5] S. J. Mohammed and D. B. Taha, "From cloud computing security towards homomorphic encryption: A comprehensive review," *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, vol. 19, no. 4, pp. 1152–1161, 2021.

[6] S. Sherif, Y. H. Ghallab, and Y. Ismail, "Theoretical analysis for the fluctuation in the electric parameters of the electroporated cells before and during the electrofusion," *Medical & Biological Engineering & Computing*, vol. 60, no. 12, pp. 3585–3600, 2022.

[7] S. J. Mohammed and D. B. Taha, "Paillier cryptosystem enhancement for homomorphic encryption technique," *Multimedia Tools and Applications*, vol. 83, no. 8, pp. 22 567–22 579, 2024.

[8] M. Alajmi, I. Elashry, H. S. El-Sayed, and O. S. Faragallah, "A password-based authentication system based on the captcha ai problem," *IEEE Access*, vol. 8, pp. 153 914–153 928, 2020.

[9] S. J. Mohammed and D. B. Taha, "Privacy preserving algorithm using chao-scattering of partial homomorphic encryption," in *Journal of Physics: Conference Series*, vol. 1963, no. 1. IOP Publishing, 2021, p. 012154.

[10] Z. Xu, J. Xu, and L.-D. Kuang, "A token-based authentication and key agreement protocol for cloud computing," in *2021 IEEE 6th International Conference on Smart Cloud (SmartCloud)*. IEEE, 2021, pp. 38–43.

[11] P. Padma and S. Srinivasan, "A survey on biometric based authentication in cloud computing," in *2016 International Conference on Inventive Computation Technologies (ICICT)*, vol. 1. IEEE, 2016, pp. 1–5.

[12] Y. Lu and D. Zhao, "Providing impersonation resistance for biometric-based authentication scheme in mobile cloud computing service," *Computer Communications*, vol. 182, pp. 22–30, 2022.

[13] S. J. Mohammed, "Developing a hybrid pseudo-random numbers generator," in *International Conference on Forthcoming Networks and Sustainability in the AIoT Era*. Springer, 2024, pp. 276–286.

[14] P. Golar and R. Sharma, "An advanced knowledge based graphical authentication framework with guaranteed confidentiality and integrity," *International Journal on Recent and Innovation Trends in Computing and Communication*, vol. 11, pp. 720–730, 08 2023.

[15] A. Ezugwu, E. Ukwandu, C. Ugwu, M. Ezema, C. Olebara, J. Ndunagu, L. Ofusori, and U. Ome, "Password-based authenti-

TABLE V. Examples of system resisting against credential stuffing attack

credential stuffing attack	State	False Negative	False Positive
User1: }fA:#4h&	Blocked	No	No
User2: Bs3XzV1	Blocked	No	No
User3: k`&4$d - W$?9	Blocked	No	No
User4: (4p-$:¿Ã,S	Successful login	No	Yes
User5: m)A,gB5$^9/\tilde{X}$	Blocked	No	No
User6: fhA:4h&B50%	Blocked	No	No
User7: 9BS*8zg=X:af@	Blocked	No	No

cation and the experiences of end users," *Scientific African*, vol. 21, p. e01743, 2023.

[16] S. Kaur, G. Kaur, and M. Shabaz, "A secure two-factor authentication framework in cloud computing," *Security and Communication Networks*, vol. 2022, no. 1, p. 7540891, 2022.

[17] S. Lingamgunta, "Multi factor two-way hash-based authentication in cloud computing." *International Journal of Cloud Applications & Computing*, vol. 10, no. 2, 2020.

[18] K. H. Al-Shqeerat and K. I. Abuzanouneh, "A hybrid graphical user authentication scheme in mobile cloud computing environments," *International Journal of Communication Networks and Information Security*, vol. 13, no. 1, pp. 68–75, 2021.

[19] A. M. John, S. P. Mary, K. M. Prasad, M. Naveena, and N. Laveti, "Authentication for cloud computing system through smartcard," *International Journal of Cloud Computing*, vol. 11, no. 5-6, pp. 518–528, 2022.

[20] R. O. Okeke and S. O. Orimadike, "Enhanced cloud computing security using application-based multi-factor authentication (mfa) for communication systems," *European Journal of Electrical Engineering and Computer Science*, vol. 8, no. 2, pp. 1–8, 2024.

[21] C.-H. Liu, T.-L. Chen, C.-Y. Chang, and Z.-Y. Wu, "A reliable authentication scheme of personal health records in cloud computing," *Wireless Networks*, vol. 30, no. 5, pp. 3759–3769, 2024.

[22] Y. Hu, H. Wu, and L. Zhou, "A novel hyperchaotic 2d-sfcf with simple structure and its application in image encryption," *Entropy*, vol. 24, no. 9, p. 1266, 2022.

[23] S. John and S. Kumar, "6d hyperchaotic encryption model for ensuring security to 3d printed models and medical images," *Journal of Image and Graphics*, vol. 12, no. 2, pp. 117–126, 2024.

[24] M. T. Younis, "Applying the method of enhancing feedback control on a 4d hyperchaotic system," *AL-Rafidain Journal of Computer Sciences and Mathematics*, vol. 13, no. 1, pp. 13–21, 2019.

[25] N. Cui and J. Li, "A new 4d hyperchaotic system and its control," *Aims Math*, vol. 8, no. 1, pp. 905–923, 2023.

[26] R. D. Méndez-Ramírez, A. Arellano-Delgado, M. A. Murillo-Escobar, and C. Cruz-Hernàndez, "A new 4d hyperchaotic system and its analog and digital implementation," *Electronics*, vol. 10, no. 15, p. 1793, 2021.

[27] S. F. Al-Azzawi and M. A. Al-Hayali, "Coexisting of self-excited and hidden attractors in a new 4d hyperchaotic sprott-s system with a single equilibrium point," *Archives of Control Sciences*, vol. 32, 2022.

[28] M. A. Al-hayali and F. S. Al-Azzawi, "A 4d hyperchaotic sprott s system with multistability and hidden attractors," in *Journal of Physics: Conference Series*, vol. 1879, no. 3. IOP Publishing, 2021, p. 032031.

[29] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: past, present, and future," *Multimedia tools and applications*, vol. 80, pp. 8091–8126, 2021.

[30] I. Turki, "Using the hybrid ga-ant algorithm to find the optimal path in computer networks," *AL-Rafidain Journal of Computer Sciences and Mathematics*, vol. 16, no. 1, pp. 121–129, 2022.

[31] A. Lambora, K. Gupta, and K. Chopra, "Genetic algorithm-a literature review," in *2019 international conference on machine learning, big data, cloud and parallel computing (COMITCon)*. IEEE, 2019, pp. 380–384.

[32] T. Alam, S. Qamar, A. Dixit, and M. Benaida, "Genetic algorithm: Reviews, implementations, and applications," *arXiv preprint arXiv:2007.12673*, 2020.

[33] S. Han and L. Xiao, "An improved adaptive genetic algorithm," in *SHS web of conferences*, vol. 140. EDP Sciences, 2022, p. 01044.

[34] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed. USA: Prentice Hall Press, 2017.

[35] F. Pub, "Secure hash standard (shs)," *Fips pub*, vol. 180, no. 4, 2012.

[36] T. H. Tran, P. Hoai Luan, and Y. Nakashima, "A high-performance multimem sha-256 accelerator for society 5.0," *IEEE Access*, vol. PP, pp. 1–1, 03 2021.

[37] "How Secure Is My Password? — nordpass.com," https://nordpass.com/secure-password/.

[38] "Password Strength Meter — passwordmonster.com," https://www.passwordmonster.com/.

[39] "Password Tester — Test Your Password Strength — Bitwarden — bitwarden.com," https://bitwarden.com/password-strength.

[40] "LastPass - How secure is your password? — lastpass.com," https://lastpass.com/howsecure.php.

[41] O. D. T. Team, "Password Checker - Evaluate pass strength, dictionary attack — password-checker.online-domain-tools.com," http://password-checker.online-domain-tools.com/#google_vignette.

[42] "Passwords Strength — rumkin.com," https://rumkin.com/tools/password/.

[43] F. Glory, A. Aftab, O. Tremblay-Savard, and N. Mohammed, "Strong password generation based on user inputs," 10 2019, pp. 0416–0423.

[44] M. Ba, J. Bennett, M. Gallagher, and S. Bhunia, "A case study of credential stuffing attack: Canva data breach," 12 2021, pp. 735–740.