Wasit Journal for Pure Science

Journal Homepage: https://wjps.uowasit.edu.iq/index.php/wjps/index
e-ISSN: 2790-5241 p-ISSN: 2790-5233

Honeywords Generation Technique based on Meerkat Clan Algorithm and WordNet

Maher A. Ahmed[1], Omar Z. Akif[2]*

[1]Research Iraqi Commission for Computer, Informatics Institute for Post Graduate Studies
Baghdad, IRAQ
[2] Department of Computer, College of Education for Pure Science (Ibn al-Haitham), University of Baghdad
Baghdad, IRAQ

*Corresponding Author: Maher A. Ahmed

DOI: https://doi.org/ 10.31185/wjps.269
Received 09 October 2023; Accepted 04 December 2023; Available online 30 December 2023

ABSTRACT: The efficiency of the Honeywords approach has been proven to be a significant tool for boosting password security. The suggested system utilizes the Meerkat Clan Algorithm (MCA) in conjunction with WordNet to produce honeywords, thereby enhancing the level of password security. The technique of generating honeywords involves data sources from WordNet, which contributes to the improvement of authenticity and diversity in the honeywords. The method encompasses a series of consecutive stages, which include the tokenization of passwords, the formation of alphabet tokens using the Meerkat Clan Algorithm (MCA), the handling of digit tokens, the creation of unique character tokens, and the consolidation of honeywords. The optimization of the performance of the Meerkat Clan Algorithm (MCA) involves the careful selection of parameters. The experimental findings have exhibited noteworthy levels of precision and optimum efficacy, particularly in tasks such as proposing words with similar meanings, forecasting numerical values, and producing distinctive symbols. The attainment of this achievement is facilitated by a confluence of factors, encompassing the caliber of data, the judicious use of algorithms or models, and the ongoing process of iterative improvement to consistently enhance outcomes. In order to achieve the appropriate levels of accuracy and functionality, it is crucial to engage in the process of conducting experiments, thoroughly testing the system, and making necessary improvements. The empirical findings provide confirmation of the effectiveness of the MCA in producing a varied and protected collection of honeywords. This is especially evident in the case of alphabet tokens, which are distinguished by their autonomous creation and strong security characteristics. The analysis of correction rates, specifically in relation to the password "Lion1999*," demonstrates the aforementioned results. This study reveals an average accuracy of honeyword production up to 0.729847632111541. In the same manner, the accuracy of the password "house2000" is determined to be 0.761325846711256. Additionally, when considering a sample of 100 passwords, the mean accuracy of honeyword creation is calculated to be 0.7073897168887518. The findings collectively highlight the effectiveness of the MCA in generating honeywords that possess improved security characteristics.

Keywords: Honeywords system, Meerkat clan algorithm, Password security, and WordNet.

1. INTRODUCTION

The major aim of Honeywords system is to enhance user account security through the detection and prevention of unauthorized access attempts. Through adding honeywords to the password database, this is accomplished. The usage of honeywords significantly improves user account security by making it much harder for malicious parties to infer or breach the genuine password [1]. A useful technique to generate honeywords that is modeled after meerkat behavior is the Meerkat Clan Algorithm (MCA). During such process, possible solutions are divided into smaller factions, or "clans," each of which has a different approach to solving the optimization problem. Those clans interact, collaborate, and share a variety of solutions in an effort to find the most effective solution [2]. This process has proven effective in solving a variety of optimization problems. This proposed method's MCA attempts to solve a major challenge, which is lowering predictability and resolving flaws which are frequently associated with statistical analysis. The invention of honeywords,

which make it extremely difficult for attackers to discern them from real passwords, achieves the aforementioned goal [3]. The present study presents a new approach to create honeywords through using Wordnet and the Meerkat Clan Intelligence Algorithm, with the main goal being to strengthen password security. This research also emphasizes the drawbacks of earlier techniques for generating honeywords, such as their lack of variability, susceptibility to statistical analysis, and reliance on static honeywords. Since users frequently use weak passwords and reuse them for multiple accounts, it is clear that weak password limitations are frequently insufficient to successfully counter attacks like dictionary and brute-force attacks. This method was suggested as a possible approach to strengthen password database security against dictionary-based and brute-force attacks due to such factors [4]. The principal aim of this study is to develop an innovative algorithm which can produce honeywords that closely resemble genuine passwords, making them unidentifiable from potential attackers. To improve password security, the optimization regarding the similarity between honeywords and genuine passwords is used.

2. BACKGROUND THEORETICAL

This section will discuss the review of three research papers, and then introduce an overview of honeywords and the MCA.

2.1 Related works

This section offers a scholarly analysis of three studies which address the use of honeywords to improve password security (Win et al., 2018; Yasser et al., 2022; AlHamdani et al., 2023). Win et al. [5], a novel approach to honey encryption and the creation of honeywords have been introduced to improve the security of sensitive data. This research's approach creates honeywords which are resistant to dictionary attacks and other common password-based attacks by combining hash functions and encryption techniques. Empirical evidence supports the suggested strategy's greater efficacy and security over current approaches. Yasser et al. [6] presented a novel approach in their research, whereby they proposed a system for generating honeywords, which are intentionally misleading passwords aimed at identifying and deterring unauthorized attempts to compromise password security. The method being proposed employs the Harmony Search Algorithm, a widely recognized optimization technique that draws inspiration from the creative process of music composition. To enhance the intricacy of distinguishing genuine passwords from decoys, researchers employ a method that involves using a dataset of authentic passwords to generate honeywords that exhibit statistically comparable characteristics. The approach proposed by AlHamdani et al. [7] entails employing the discrete salp swarm algorithm to produce honeywords that closely resemble real passwords. The present methodology exhibits enhanced performance in terms of both effectiveness and efficiency when compared to prior methodologies. The authors' study makes a valuable contribution to the field of password security by presenting a novel strategy to address the dangers associated with password attacks.The assessment of several methods for creating honeywords reveals positive outcomes in relation to both security and usability metrics. This paper provides a significant addition to the domain of password security and cybersecurity by presenting a practical methodology to efficiently address the challenges related to the identification of password breaches. A comprehensive examination of prior research endeavors provides valuable insights into the advancements made in the field of honeyword generation. The aforementioned findings provide a significant contribution to enhancing password security and serve as a guiding framework for future research endeavors aimed at bolstering security measures.

2.2 Honeywords

Honeywords, sometimes referred to as honey passwords or decoy passwords, are artificially generated passwords that are strategically included in databases alongside authentic passwords with the intention of augmenting password security [8]. Pseudo-passwords are deliberately included in databases alongside genuine passwords with the aim of augmenting the efficacy of password security mechanisms. Honeywords are intentionally designed to hinder unauthorized access attempts and confuse malicious actors in their efforts to differentiate authentic passwords from fraudulent ones [9]. The primary aim of this research is to develop honeywords, which are deliberately crafted passwords that closely mimic authentic passwords, with the intention of enhancing the complexity for prospective adversaries. The utilization of honeywords offers several advantages. One of the key benefits is the capacity to promptly detect security problems by employing the honeychecker. The aforementioned approach expeditiously alerts information technology personnel in the event of a login attempt utilizing any of the honeywords [10]. Furthermore, it facilitates the identification of potential security vulnerabilities and enables timely interventions to mitigate such risks. Moreover, the use of honeywords leads to the prompt suspension of the relevant user account, while the incorporation of system segmentation offers an extra advantage [11]. The honeychecker functions independently of the central system and is responsible for the generation of honeywords. The division under consideration is tasked with the responsibility of preserving the

integrity of the honeychecker, guaranteeing its security against any potential system breach. According to reference [12], this subsequently results in a general improvement in security.

The honeywords system incorporates a comprehensive and diverse assortment of words and phrases obtained from WordNet, which may be employed to generate honeywords. WordNet is a lexical database that organizes words into groups of synonyms referred to as synsets [13]. WordNet is used to build a huge database of connected words and phrases, which may be leveraged in the creation of honeywords. The aforementioned databases provide a comprehensive compilation of commonly employed English vocabulary that may be employed for the generation of honeywords. This functionality facilitates the integration of linguistically genuine sentences into the honeywords.

2.3 Meerkat Clan Algorithm (MCA)

The intelligence system employed by the Meerkat Clan is derived from the social behavior observed in meerkats, a type of mongoose. The current approach replicates various attributes commonly observed in meerkat behavior, including task allocation, coordinated efforts, collective evaluation, adaptability, and learning abilities. The primary goal of this strategy is to enhance the efficiency of problem-solving by leveraging the effectiveness of collaborative endeavors. This concept is frequently utilized in optimization and problem-solving scenarios [14]. Meerkats demonstrate collective decision-making, allocation of tasks based on individual capabilities, communication through vocalizations and body language, as well as adaptability and the ability to learn from mistakes. The program aims to harness collective intelligence by simulating these behavioral characteristics in order to address complex issues.
The intelligence algorithm employed by the Meerkat Clan exhibits several significant features:

- The distribution of subtasks among various agents or components inside the algorithm has a resemblance to the task allocation observed in meerkats. The distribution of resources is determined by the qualifications and competency of each agent or component [15].
- The phenomenon of communal decision-making may be observed in meerkats, as they actively participate in the exchange of experiences and ideas in order to achieve a consensus. In a similar vein, the algorithm incorporates a mechanism for collaborative decision-making through the aggregation of inputs from many actors [16].
- There is no information in the user's text that has to be rewritten in an academic way. Meerkats have a notable level of flexibility and display a tendency to acquire novel knowledge. The method has the potential to integrate many components that enable agents to adjust their strategies based on inputs and utilize previous iterations to get improved results [17].

The application of the Meerkat Clan intelligence approach is frequently noticed in problem-solving and optimization scenarios when the collaborative intelligence of a group can result in improved and more efficient solutions. This method's primary goal is to use collaboration as a means of resolving complex problems through modeling the cooperative and coordinated conduct seen in meerkats [18].

Meerkat Clan Algorithm

Input: Clan size (n), foraging size (m), care size (c), worst foraging rate (Fr), worst care rate (Cr), neighbor solution (k).
Output: Best Solution.
Begin
  Generate random clan of solutions clan (n)
  Compute fitness for clan solutions
  $Sentry = best\ solution\ of\ the\ clan$
  Divide the clan into two groups (foraging & care).
  While not termination condition Do
    For $i = 1$ to m
      Generate k neighbors from foraging set
      Foraging (i) = best one from k neighbor
    End for
    Swap the worst for Fr solution in foraging group with best ones solution in care group;
    Drop the worst Cr solution from care group and generate ones solution randomly;
    Select the best one of foraging call it $best\_forg$
    If $best\_forg <= Sentry$ then
      $Sentry = best\_forg$
    End if
  End while
End

FIGURE 1. -The pseudo-code for the Meerkat Clan Algorithm [19]

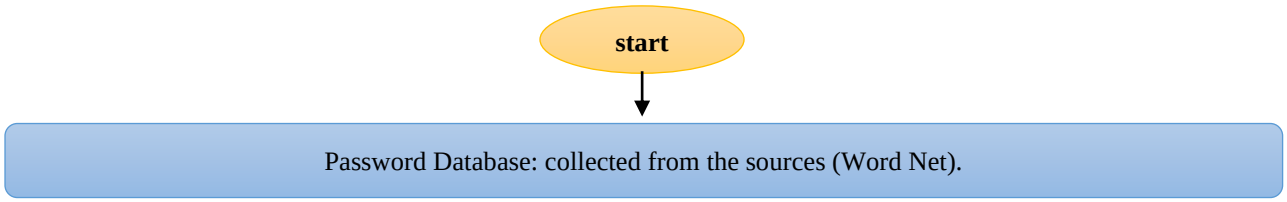
3. THE BLOCK DIAGRAM OF PROPOSED WORK

This section provides a thorough analysis regarding the method and design of the suggested method, which focuses on employing the MCA to generate honeywords. The recommended method entails a step-by-step process which starts with WordNet data acquisition for creating a password database. Next, for dividing passwords into unique tokens and ensure that username matches are preserved, the Tokenizer is put into practice. The Alphabet Generating component utilizes the MCA to produce a range of word possibilities by means of synonymic relationships. The Digits Generating module incorporates numeric features by either selecting random tokens from specified lists or utilizing a random digit generator. The Special Characters Generating component enables the incorporation of special characters while implementing an exclusion rule to avoid including characters that already exist in the username. During the final step known as Sweetwords, the honeywords created are combined with the original passwords utilizing the Meerkat Sweetwords Clan Algorithm. To evaluate the effectiveness of the proposed approach, it is crucial to measure the level of similarity attained between the honeywords and the genuine passwords. The evaluation is conducted using the Levenshtein distance metric. The Honeychecker system is designed to detect compromised user accounts and initiate a forced password-changing procedure upon the entry of a honeyword

Algorithm1: The proposed honeywords generation using the Meerkat Clan

Input: Password, Output: List of Honeywords

- **Step 1: Data Acquisition:** The first stage entails obtaining data from WordNet in order to compile a wide range of words and phrases for the future production of honeywords.
- **Step 2: Tokenization:** During this step, the system performs character categorization inside the sugarword, dividing them into separate tokens according to their character type (such as alphabets, numerals, and special characters). Tokens that correspond to the user's username are deliberately preserved for the purpose of generating honeywords.
- **Step 3: Alphabet Generation:** The system uses the MCA to find the best solutions by taking into account concepts such as closeness, quality, different responses, stability, and adaptability. These principles are applied to the alphabet tokens obtained from the database.
- **Step 4: Fitness Evaluation:** The fitness evaluation stage plays a crucial role in assessing the correlation between the authentic password and the honeywords created by the system during the alphabet generation phase. The purpose of this evaluation is to measure the degree of disparity between the honeywords and the genuine password. A reduced correlation value signifies an increased level of dissimilarity or "flatness" between the honeywords and the original password.
- **Step 5: Generation of Digits**: In this phase, the digits token is carefully examined to see if it aligns with elements in the list for the given year or falls inside a series of consecutive numbers that occur often (within a range of ±5 numbers).
- **Step 6: Special Characters Generation:** The system creates special character tokens in a random manner. A generator that also operates randomly processes these tokens. This procedure guarantees that the honeywords consist of a diverse range of unique and non-recurring special characters.
- **Step 7: Token Aggregation:** The process of honeywords generation involves aggregating all the tokens created in the previous phases and concatenating them to form honeywords.
- **Step 8: Honeywords Generator (Sweetwords):** The process involves the utilization of a honeywords generator, sometimes referred to as Sweetwords. This particular component combines honeywords with the original sugarword, applying the MCA. The phenomenon of sweetwords undergoing random permutation is seen, wherein the sugarword is introduced at a spot that is randomly chosen.
- **Step 9: Similarity assessment:** The process of similarity evaluation involves utilizing the Levenshtein distance metric to compare honeywords with actual passwords. The objective is to determine if the honeywords exhibit a high level of similarity to real passwords, which is shown by the distance falling inside a certain threshold.
- **Step 10: Honeychecker:** Following the generation of sweetwords, the system proceeds to communicate the user's index and the corresponding sugarword's index to the honeychecker module.
- **Step 11: Account Flagging:** In the event that the password supplied by the user corresponds to any of the sweetwords, the system instantly detects the penetration of the user's account and triggers a prompt for the user to swiftly change their password.



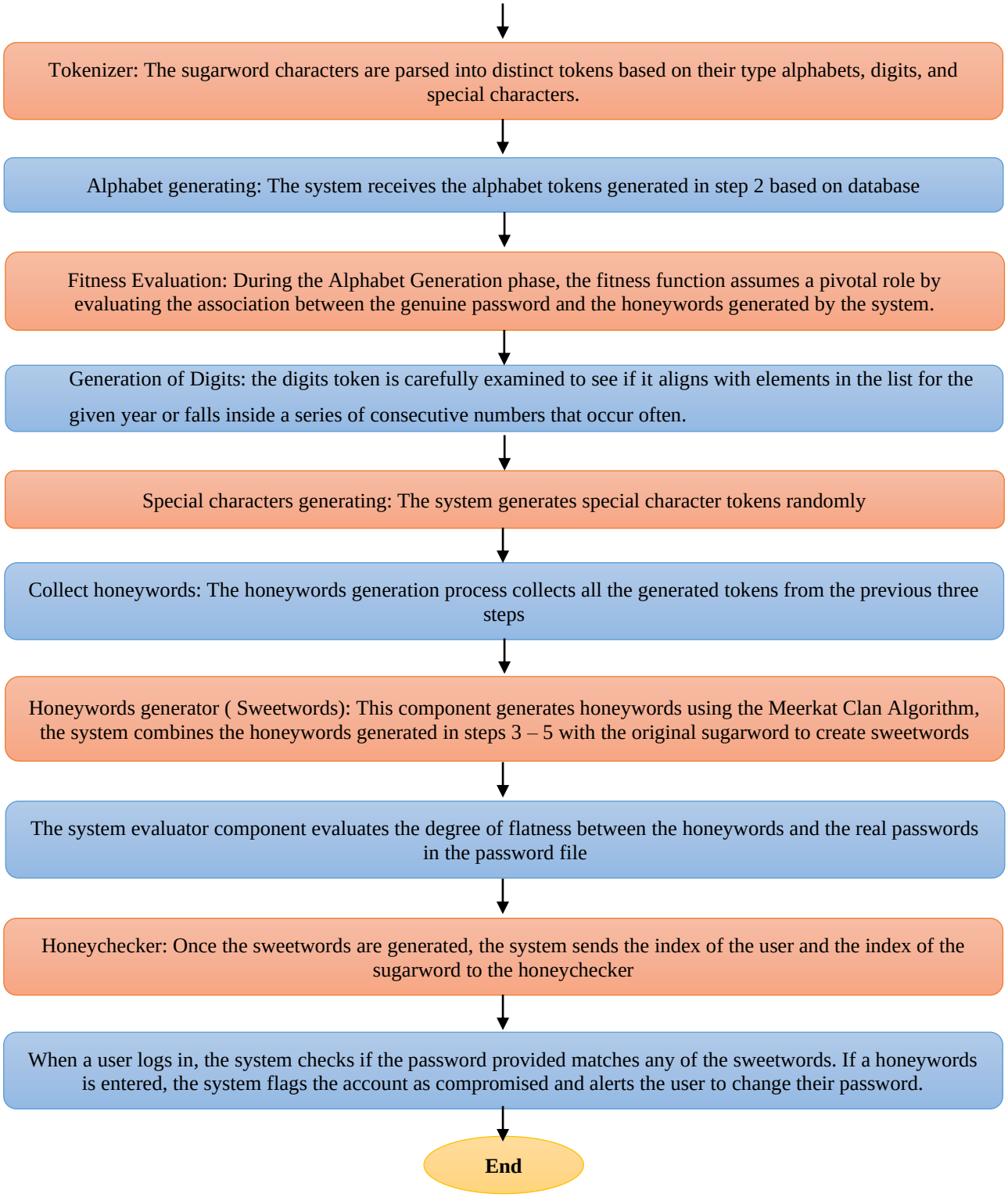


FIGURE 2. -The block diagram of the proposed system

4. EXPERIMENTAL RESULTS

The outcomes derived from the honeyword-generation system were classified into two primary divisions: an extensive examination of individual tokens and a comprehensive review of whole passwords. The system underwent a rigorous testing and assessment procedure to evaluate its efficacy in generating honeywords with robust security attributes. The findings of the study demonstrated that the system effectively generated a wide variety of alphabet tokens, hence enhancing the security of the honeywords. By employing this approach, the creation of very intricate honeywords was achieved, hence augmenting their resilience against unauthorized endeavors to deduce passwords.

Based on a comparative analysis, it is evident that the proposed system exhibits enhanced safety measures, increased complexity, and a reduced susceptibility to hacking compared to previous methodologies. To optimize the system's efficiency and provide secure password solutions, a meticulous process of parameter values selection was conducted. The experimental findings presented in this study offer support for the general effectiveness of the honeyword-generation process, demonstrating its superiority in comparison to earlier methodologies. The approach employed the utilization of WordNet, a lexical database, to identify synonyms for designated words, hence augmenting the diversity of honeywords generated. The integration of an extensive array of keywords, phrases, and applications sourced from WordNet enhanced the credibility of honeywords. The findings and further examination underscored the strategic significance and discernible impacts of this methodology. provide an illustrative example, a compilation of phrases like 'Jump', 'Project', 'Walk', and 'Clean' was provided. The results were systematically arranged and documented in Table 1, encompassing distinct columns for the initial terms and their corresponding equivalents.

Table 1. - The sample of synonyms when using WordNet

Index	Word	synonyms using WordNet
0	Jump	Spring, leaping, bound, rebound
1	Project	Research, survey, study, schooling
2	Walk	Pass, impel, progress, propel
3	Clean	Wash, scrub, scour, clean out

The tokenizer plays a vital role in the honeyword creation process since it decomposes a password into its fundamental constituents, encompassing alphabet tokens, digit tokens, and special character tokens. The approach systematically extracts and preserves distinct aspects of passwords, such as strings, numeric characters, and special characters, while excluding any other components. The final outcome of this process involves the creation of a meticulously arranged table, as seen in Table 2, which displays the initial passwords with their corresponding segments.

Table 2. - Tokenizer the password to String, Number, and Special Part

Index	Password	String part	Number part	Special part
0	descender2018$	descender	2018	$
1	SecurePwd	securePwd		
2	11223344		11223344	
3	truepass!	Truepass		!
4	ceramic2020	Ceramic	2020	
5	Ma19er82+	Maer	1982	+
6	Hero2000/	Hero	2000	/

During the alphabetic generation phase, the honeyword creation system utilizes the alphabet tokens acquired from the preceding Tokenizer step. The tokens are subjected to the MCA to produce candidate words. For instance, when presented with a collection of alphabet tokens, such as "a," "b," and "c," the MCA can generate phrases such as "cab," "bac," and "abc." Subsequently, the aforementioned phrases can be further elaborated by identifying synonymous expressions. As an illustration, the term "cab" can generate synonymous alternatives, including "taxi," "carriage," and "vehicle." Likewise, the term "bac" has the potential to yield synonymous alternatives like "reverse," "inverted," and "retrograde." Finally, the word "abc" has the potential to generate synonymous alternatives, such as "alphabet," "letter," and "character." The iterative characteristic of this methodology amplifies the degree of password security by broadening the spectrum of potential honeywords. Furthermore, the honeywords generation technique that has been devised demonstrates good management of digit tokens by verifying their presence against predetermined lists, such as years or number sequences. Upon detection of a digit token, the system employs a random selection from the corresponding list to generate honeywords, hence ensuring the mitigation of obviously discernible patterns. For instance, when the digit token is "1988," the system possesses the capacity to select either "1989" or "1987" from the provided array of years. This aids in augmenting the diversity of honeywords generated. The MCA alphabet generator facilitates diversity by utilizing an iterative methodology for generating and improving alternative solutions while adhering to principles such as closeness, quality, and the inclusion of varied answers. The process of producing honeywords also includes the generation of unique character tokens through the utilization of a randomly produced special character generator. An illustration of this concept is the use of tokens to represent alphabets and numerals, represented as "love" and "6789" respectively. In this context, it becomes feasible to include a special character token, such as "%", within the honeyword. This would provide a hypothetical honeyword, exemplified by the string

"love6789%". In the subsequent phase, referred to as "Collect Honeywords," the amalgamation of tokens representing alphabets, digits, and special characters yields a wide range of honeywords. These honeywords provide the foundation for creating sweetwords in the subsequent stage. The aforementioned intricate approach ultimately improves the level of password security, as shown in Table 3.

Table 3. - Show the Table of honeyword generator based on MCA

Password	Honeyword1	Honeyword2	Honeyword3	Honeyword4	Honeyword5
Sumara1999@	Mustafa1998)	Mustahsan1999=	Usama1997@	Asmat1996(	Safwat1997”
Adeeb1985(	Kaleema1984^	Adoration1986+	Adeel1985$	Kafeel1983.	Love1982:
Fareed1980$	Noorie1981:	Cooky1984!	Confection1982/	Comfit1981?	Rafeek1980<
Lour1965)	Lower1964”	Lower1963+	Nawel1966:	Low1967*	Robeel1967”
Zaroon1993!	Gohar1993*	Raghad1994@	Nilofar1995:	Gohar1994;	Nudar1992,
Mohsin1994!	Naila1995_	Nihal1994(	Liban1997$	Linah1995^	Ilhan1994$
Rabeea1988$	Sheema1984]	Confound1985*	Taabeer1986&	Faseeh1985!	Tehseen1987/
Marzuq1981!	Rizq1984*	Mohga1982;	Izma1985(	Qadim1980/	Zaighum1983&
Baheer1984/	Premier1984%	Nasreen1985}	Flower1986@	Ragheb1987!	Mahbeer1983(
Erina1993-	Mustaneer1994-	Sundas1993]	Yameen1992)	Mobeen1990/	Mussaret1991>

The methodology for producing honeywords, which utilizes the MCA, encompasses a comprehensive analysis of parameter values, empirical observations, and a comparative assessment with respect to prior approaches. Table 4 presents the meticulously selected parameter values that were then modified following an extensive series of experiments in order to enhance the performance of the MCA algorithm. The objective of this iterative process is to optimize the parameters of the algorithm with the aim of enhancing its performance and assessing its efficacy within the particular domain of honeywords generation.

Table 4. - The Proposed System Parameter Values

No	Parameter	Value
1	Population Size (n)	50
2	Sentry	1
3	Foraging Group Size (f)	n/2
4	Care Group Size (c)	(n/2) - 1
5	Generated Neighbor Tokens Size (Nt)	1
6	Max-Generation (Mg)	100
7	Number of Generated Alphabet Tokens (a)	1
8	Number of Generated Digits Tokens (d)	1
9	Number of Generated Special Characters	6

The system performs a sequence of assessments on the password, which include evaluating its length and character composition, spanning alphabetic, numeric, and special characters. Subsequently, these assessments are juxtaposed with pre-established minimal criteria, namely, a minimum of four alphabetic letters, four numerical digits, and at least one special character. If the function successfully passes all of these tests, it will return a value of "True"; otherwise, it will return "False," therefore assuring adherence to the specified password conditions. The proposed system uses Meerkat Clan Intelligence Algorithm (MCA) as the foundation for honeyword generation in an effort to overcome the inherent constraints observed in previous methods. The aforementioned methodology has resulted in positive results about both the aspect of security and the production of words that are semantically meaningful. The system effectively demonstrates the efficacy of the MCA in producing honeywords for different sorts of tokens, with a specific focus on alphabet tokens. This emphasis is in line with the pursuit of desirable characteristics, such as the ability to generate tokens independently, the production of diverse solutions, and the enhancement of honeyword security. The empirical results confirm the accomplishments of the system, and the suggestions propose a greater population size than the maximal generating capacity.

Table 5. - show the result of the proposed MCA

User name	Password
Maher	['dragon1993@', 'raonar1993[', 'Dragon1993}', 'dragon1993~', 'nadr1993/', 'nora1993{', 'rand1993{', 'rona1993/', 'nora1993$', 'rona1993{', 'flying_lizard1993.', 'estragon1993=', 'randa1993"', 'dinar1993%', 'nadir1993+', 'norah1993/', 'nudar1993,', 'nadra1993-', 'hodan1993|', 'gohar1993}', 'nigar1993{', 'norah1993(', 'Draco1993(', 'gohar1993;', 'nigar1993/', 'Dragon1993:', 'tarragon1993^', 'tarragon1993,', 'rohaan1993[', 'agharr1993#', 'natara1993{', 'romana1993\\', 'nilofar1993:', 'wordah1993&', 'tarragon 1993<', 'raghad1993@', 'roshan1993=', 'zaroon1993*', 'tarragon 1993(', "haroun1993"', 'noriza1993,']
Haroun	['drains2018+', 'dinar2018-', 'nadir2018@', 'nasir2018]', 'rasin2018:', 'nasri2018}', 'dinar2018;', 'nadir2018@', 'drain2018_', 'sidra2018;', 'nadhir2018!', 'nadira2018=', 'ridwan2018,', 'naadir2018,', 'nadr2018)', 'rand2018:', 'raid2018&', 'dina2018]', 'radi2018&', 'nida2018+', 'rida2018>', 'dani2018+', 'rani2018,', 'iskandar2018.', 'sikandar2018_', 'nadira2018-', 'naadir2018{', 'nargis2018{', 'narjis2018*', 'shadin2018,', 'sarina2018{', 'sabrin2018^', 'nasira2018%', 'siddra2018.', 'nadira2018}', 'danish2018+', 'sidrah2018{', 'nibras2018@', 'parinda2018+', 'nadirah2018_', 'nijad2018(']
Taysir	['tawdry2023%', 'uday2023;', 'ruwayd2023}', 'tharya2023*', 'suhayr2023:', "thayer2023"', 'taysir2023(', 'ishrat2023@', 'ward2023;', 'arsh2023[', 'ratty2023%', 'usayd2023*', 'hayud2023.', 'dunya2023:', 'ubayd2023]', 'udayl2023>', 'yasirah2023[', 'riyasat2023:', 'sarwath2023?', 'sitarah2023#', 'hurayth2023"', 'aryisha2023(', 'sheryar2023=', 'asriyah2023.', 'shurayh2023}', 'shuraym2023}', 'shehyar2023!', 'sariyah2023#', 'thoraya2023^', 'yathrib2023(', 'dua2023~', 'radwa2023+', 'riyad2023+', 'yawer2023`', 'warda2023.', 'yawar2023=', 'lubayd2023\\', 'burayd2023~', 'budayl2023}', 'junayd2023?', 'ghayda2023/']
Nawaz	['plugged1985_', 'blockade1985<', 'block1985<', 'puncher1985(', 'peg_away1985{', 'nuwayla1985|', 'nuh1985_', 'plug1985=', 'occlude1985|', 'nuwaylah1985`', 'gulab1985*', 'fireplug1985+', 'khuwaylah1985,', 'uwayam1985(', 'block_up1985(', 'abdel1985.', 'nuha1985~', 'husn1985:', 'abdulwaliy1985+', 'uwaysah1985]', 'waliyah1985:', 'block_off1985}', 'plug_away1985|', 'male_plug1985-', 'alya1985;', 'plug1985;', 'luay1985,', 'wala1985{', 'khaled1985`', 'mokbul1985<', 'robeel1985~', 'gulzar1985\\', 'ghulam1985?', "gulfam1985"', 'uhban1985`', 'nunah1985;', 'junah1985|', 'hudun1985+', 'husni1985"', 'lunah1985`', 'nuhaa1985+']

The assessment of the effectiveness of the proposed methodology centers on the correction rate, a metric employed to quantify the precision with which a given password or input aligns with the intended value. The calculation of the correction rate is accomplished by computing the normalized edit distance between the password and the specified phrase "Honeywords." The findings of this assessment are succinctly documented in Table 5, thereby providing significant insights into the efficacy of the proposed technique.

Table 6. - show the correction rate between password and honeywords

Id	password, password, correction rate	honeywords, password, correction rate
1	(Love1933*,Love1933*,1.0)	(kameel1933&,Love1933*,0.6565656565656566)
2	(screen1990$,screen1990$,1.0)	(sireen1990&,screen1990$,0.7272727272727273)
3	(lower1989@,lower1989@,1.0)	(rowel1989-,lower1989@,0.8)
4	(foot1994(,foot1994(,1.0)	(hoof1994!,foot1994(,0.6666666666666667)
5	(mustafa1982*,mustafa1982*,1.0)	(mastura1982^mustafa1982*,0.6666666666666667)

The study encompasses many crucial phases of implementation. The initial stage entails the establishment of a database and an associated table utilizing the SQLite architecture. Subsequently, the data obtained through the utilization of the MCA is meticulously entered into the aforementioned table. The study also encompasses the development of a user interface that incorporates a login form. The presented form has many input fields designed for the entry of usernames and passwords, beside buttons that facilitate the login process and grant access to the primary functionality, sometimes referred to as the "Main Honey". The employment of ipywidgets and IPython display tools facilitates the building of this user interface. Two distinct procedures have been devised to manage the verification of login credentials

and oversee the implementation of the "Main Honey" functionality. The arrangement of interface components within a VBox container is designed in a coherent manner to enhance their visual presentation. In order to get comprehensive functionality, it is important to incorporate supplementary functions and integrate a background image. Figure 1 presents graphic representations of the database table and the login form for the purpose of providing a visual reference.

FIGURE 3. - the interface of the proposed work

5. CONCLUSION

The present work suggests employing the MCA as a strategy to enhance password security by generating honeywords. This study highlights the significance of implementing a more robust and user-centric approach to address the inherent constraints of current honeyword generation techniques, namely their predictability and susceptibility to statistical analysis. The MCA is influenced by the behavioral patterns seen in meerkats. It introduces a unique methodology that involves partitioning potential solutions into various subgroups or clans, hence fostering cooperation and information exchange among these factions. This research paper introduces a thorough approach and framework for the production of honeywords. It encompasses the collection of information from WordNet, the segmentation of passwords into tokens, the generation of tokens for alphabets, digits, and special characters, and the incorporation of honeywords alongside original passwords through the utilization of the Meerkat Sweetwords Clan Algorithm. The honeychecker module identifies compromised accounts by doing a comparison between passwords provided by users and honeywords. The empirical testing findings demonstrate the efficacy of the proposed honeyword production technique. The results of this study show that the procedure can generate a wide variety of honeywords with robust resistance against password guessing attempts and high complexity. The proposed system exhibits notable attributes in relation to security, complexity, and resistance to attacks, as substantiated by a comparative analysis conducted with previous methodologies. In summary, this research study makes a valuable contribution to the domain of password security by advocating for the use of the MCA as a means of creating honeywords. The proposed technique provides a flexible and user-friendly mechanism for generating honeywords, which successfully misleads attackers attempting to discern genuine passwords. The empirical findings underscore the advantages of the proposed strategy as compared to prior techniques and validate its effectiveness. The implementation of honeywords, as devised by the MCA, offers organizations a significant enhancement in password security, the ability to detect unauthorized access attempts and a reduction in the possible risks associated with data breaches.

REFERENCES

[1]. Thite. M. V, Nighot. M., “Honeyword for security: A review,” International Journal, Vol.6, no.5, 2021.
[2]. Ali. S. M, Mahmood. N. T. and Yousif. S. A., “Meerkat Clan Algorithm for Solving N-Queen Problems,” Iraqi Journal of Science, pp.2082-2089, 2021.
[3]. Chang. D, Goel. A., Mishra. S and Sanadhya. S. K, “Generation of secure and reliable honeywords, preventing false detection,” IEEE Transactions on Dependable and Secure Computing, Vol.16, no.5, pp.757-769, 2018.
[4]. Alshaibi. A, Al-Ani. M, Al-Azzawi. A, Konev. A. and Shelupanov. A, “The comparison of cybersecurity datasets,” Data, Vo.l7, no.2, pp.22, 2022.
[5]. Thanda. Win and Myat Moe. Khin Su “Protecting private data using improved honey encryption and honeywords generation algorithm,” Diss. MERAL Portal, 2018.
[6]. A. Yasser. Yasser, T. Ahmed. Sadiq, and AlHamdani. Wasim “A Proposed Harmony Search Algorithm for Honeyword Generation,” Advances in Human Computer Interaction, 2022.
[7]. AlHamdani. Wasim, T. Ahmed. Sadiq and A. Yasser. Yasser, “Honeyword Generation Using a Proposed Discrete Salp Swarm Algorithm,” Baghdad Science Journal, vol 20, no.2, pp. 125-131, 2023.
[8]. Yu. Fangyi, Martin, and Migual. Vargas, “Targeted honeyword generation with language models,” arXiv preprint arXiv, pp.2208.06946, 2022.

[9]. A. Yasser. Yasser, T. Ahmed. Sadiq, and AlHamdani. Wasim, "A scrutiny of honeyword generation methods: Remarks on strengths and weaknesses points," Cybernetics and Information Technologies, Vol.22, no.2, pp.3-25, 2022.
[10]. Sailaja. C. V, Reddy. B. T., "Creating secure and dependable honey words to increase password security," Annals of the Romanian Society for Cell Biology, pp.19588-19594, 2021.
[11]. Kanta. A., Coisel. I, and Scanlon. M, "A survey exploring open source Intelligence for smarter password cracking" Forensic Science International: Digital Investigation, Vol.35, pp.301075, 2020.
[12]. Li. W, Zeng. J, "Leet usage and its effect on password security," IEEE Transactions on Information Forensics and Security, Vol.16, pp.2130-2143, 2021.
[13]. Priyatno. J., and Bijaksana. M. A, "Clustering synonym sets in english wordNet," In 2019 7th International Conference on Information and Communication Technology (ICoICT) pp. 1-4(2019). IEEE.
[14]. Cruz-Duarte, J. M. Ortiz-Bayliss, J. C. Amaya, I. and Pillay. N, "Global optimisation through hyper-heuristics: Unfolding population-based metaheuristics," Applied Sciences, Vol.11, no.12, pp.5620, 2021.
[15]. Saleh. H. A, Sattar. R. A, Saeed. E. M. H and Abdul-Zahra. D. S, "Hybrid features selection method using random forest and meerkat clan algorithm," TELKOMNIKA (Telecommunication Computing Electronics and Control, Vol20, no.5, pp.1046-1054, 2022.
[16]. Muhsen. A. R., Jumaa. G. G., AL Bakri. N. F. and Sadiq. A. T, "Feature Selection Strategy for Network Intrusion Detection System (NIDS) Using Meerkat Clan Algorithm," International Journal of Interactive Mobile Technologies, Vol.15, no.16, 2021.
[17]. Chakravarty. P, Cozzi. G, Ozgul. A. and Aminian. k, "A novel biomechanical approach for animal behaviour recognition using accelerometers," Methods in Ecology and Evolution, Vol.10, no.6, pp.802-814, 2019.
[18]. Mahmood. N, "Solving Capacitated Vehicle Routing Problem Using Meerkat Clan Algorithm," INTERNATIONAL ARAB JOURNAL OF INFORMATION TECHNOLOGY, Vol.19, no.4, pp.689-694, 2022.
[19]. Sadiq. A. T., Abdullah. H. S., and Ahmed. Z. O, "Solving flexible job shop scheduling problem using meerkat Clan algorithm," Iraqi Journal of Science, pp.754-761, 2018.