

Design and Implementation of a Detailed Hybrid Security Model for Smart Web Portals Using Honeyword Protection, Keystroke Dynamics Verification, and Fuzzy Logic Decision Systems

SIDDHESH VAIDYA

siddhesh.18605@sakec.ac.in

Shah and Anchor Kutchhi Engineering College

BHAVESH PATEL

Shah and Anchor Kutchhi Engineering College

Research Article

Keywords: Honeywords, Keystroke Dynamics, Fuzzy Logic, Multilayered Security, Behavioral Biometrics, Web-Based Applications, Secure Web Portals, User Authentication, Adaptive Decision Engine, Distributed Environment

Posted Date: March 13th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-9059022/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.
[Read Full License](#)

Additional Declarations: No competing interests reported.

Version of Record: A version of this preprint was published at Journal of Computer Virology and Hacking Techniques on June 19th, 2026. See the published version at <https://doi.org/10.1007/s11416-026-00632-1>.

Design and Implementation of a Detailed Hybrid Security Model for Smart Web Portals Using Honeyword Protection, Keystroke Dynamics Verification, and Fuzzy Logic Decision Systems

Abstract

The high demand of various web-based applications is rapidly increasing with the recent ongoing advancements in digital technology. The key element of the success of these applications depends on effective user-friendly data access. This has fueled rapidly evolving research on the development of secure smart Web portals. Over the past years various ideas and authentication techniques have been proposed using fuzzy logic, biometric authentications etc. In this paper, we present a novel and comprehensive security framework that establishes a robust multilayered defense by using three distinct but synergistic mechanisms, such as honeywords, keystroke dynamics, and fuzzy logic. The system's foundational deception layer utilizes honeywords, complementing this, keystroke dynamics introduces a continuous behavioral biometric authentication factor, the framework also integrates fuzzy logic that acts as an adaptive decision engine. The proposed system is deployed and tested in the distributed environment.

Keywords: Honeywords, Keystroke Dynamics, Fuzzy Logic, Multilayered Security, Behavioral Biometrics, Web-Based Applications, Secure Web Portals, User Authentication, Adaptive Decision Engine, Distributed Environment.

1 Introduction

Recent advanced developments in the computer infrastructure, such as improved processing power, larger and more reasonably priced storage capacity, and high speed networking, have played a vital role in the accelerated growth of digital technology. The cornerstone of information security in today's linked digital world is user authentication which is considered as very crucial. The Password based authentication technique is still the popular and demanding technique because of its better ease and use for user familiarity, even with major advances in safe hardware and cryptographic techniques. However, passwords are susceptible to attacks such as dictionary attacks, phishing, brute force attempts, and most importantly, database breaches where hashed passwords can be obtained and cracked offline.

Integrating several complimentary authentication methods to build a strong, multi-layered security model is one viable way. In this study, three separate components, such as Honeywords, Keystroke Dynamics, and Fuzzy Logic Membership Functions, are combined in this study to suggest such a system. When combined, these methods provide strong protection against password compromise and user impersonation by addressing various flaws in traditional systems.

Honeywords are fake passwords stored alongside the actual password in a database. These decoy (phony) credentials are designed to be indistinguishable from the legitimate one, making it inconvenient and difficult for an attacker to determine the actual password even if the password file is compromised. When a honeyword is used to attempt a login, a separate server can detect the unauthorized attempt and raise an alert, thereby adding a deception-based detection mechanism to password security. Table 1 shows the real password of the user and various honeywords included for it in a hashed way.

Table 1: Example of Honeywords and Real Password

Credential Type	Password Stored	Attackers View
Real Password (RP)	Sid@2025	pF8dM... (Hash 1)
Honeyword 1	Sid@2045	e4qA1... (Hash 2)
Honeyword 2	Sib@2025	jR7yM... (Hash 3)
Honeyword 3	Sim@2025	sA6vU... (Hash 4)

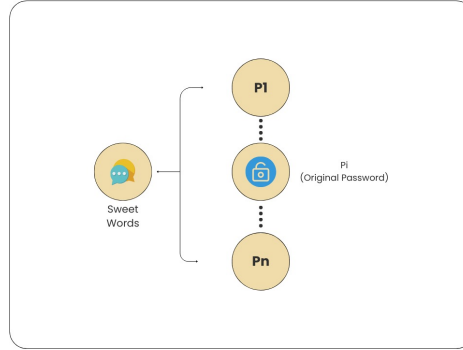


Fig. 1: Phony passwords with the original passwords

Although honeywords effectively tackle leakage of password and reuse of unauthorized password, they do not protect against attackers who gain access to a valid password through different social engineering or insider threats. This typical gap can be taken into consideration by Keystroke Dynamics, a behavioral biometric that validates (authenticates) users based on their typing rhythm (pace) and patterns. It is

quite challenging to duplicate each person’s distinct typing signature, which is influenced by motor abilities and muscle memory. Here, Fig. 1 shows the representation of honeywords, formed with the original and phony password, and Fig. 2 highlights the visualization of the key typing pattern when it is pressed and depressed.

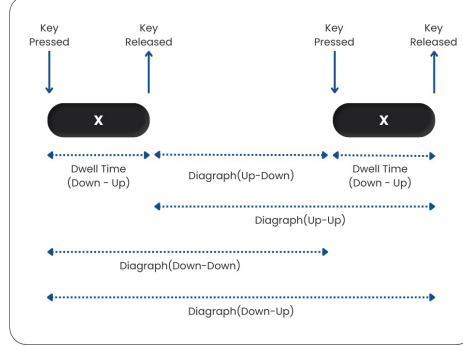


Fig. 2: Visualization of key patterns

The goal of this study is to create and assess a unique framework of authentication that combines all three types of methods to provide better defense against several attacks while preserving a smoother user experience. Our research has produced the following results based on the suggested system architecture and experimental evaluations:

- Creation of a secure password storage system with Honeywords that improves credential theft detection.
- Development of a biometric profiling system using keystroke dynamics to verify user identity behaviorally.
- Application of fuzzy logic to model and evaluate biometric data, increasing the robustness of the system under uncertain conditions.
- Implementation and evaluation of the proposed model using real user data to measure performance in terms of accuracy, false acceptance, and false rejection rates.

This paper is organized as follows: Section 2 surveys existing algorithms and approaches, Section 3 details the proposed system architecture, Section 4 analyzes the experimental results, and Section 5 concludes the study and discusses future research opportunities.

2 Review of Literature

Security is an integral component of modern web portals, including web, mobile, and cloud applications. In recent years, secure application development has attracted significant attention from researchers, leading to the proposal of numerous algorithms

and techniques. In this section we present a comprehensive survey and review of the research in honeywords, keystroke dynamics and fuzzy logic applications in security.

2.1 Secured Application Development Survey

In order to detect illegal access, Wang et al. [1] provides a significant security evaluation method, which stores phony passwords (honeywords) alongside real user passwords. The authors examines several number of attack methods that may divulge authentic passwords, including targeted attacks and numerous system intrusions. They suggest a brand-new criterion called “flatness” to gauge how well honeywords mislead attackers.

A behavioral biometric-based intrusion detection system that uses keyboard dynamics to determine the difference between authentic users and scammers is presented by Sridhar et al. [2]. The authors develop a model that can recognize unusual typing patterns in real time by examining parameters including dwell time, latency, and typing speed.

Keeping a goal to find out intrusions more accurately , Sridhar et al. [3] integrated fuzzy logic into keystroke dynamics. Traditional difficult threshold systems frequently encounter trouble with typing variations of the user that occur naturally. This fuzzy logic based method, on the other hand, provides a more comprehensive and more clear and clean examination of user behavior by permitting various degrees of membership.

Tural et al. [4] examines that how distinct digital signatures ingrained in individual typing practices can be emerged through artificial intelligence. To determine and classify keyboard (keystroke) patterns, the authors use Machine Learning (ML) methods such as decision trees and neural networks.

Biloti and group [5] investigates how keystroke timing and sequence patterns on touch which are also called as handheld devices may be described as stochastic processes and used to identify specific individuals. Their method seeks to identify users as distinct behavioral biometrics by capturing the temporal structure of touch typing activity through the training of Hidden Markov Models (HMMs) on these kinds of dynamic typing patterns. It increases the accuracy of user identification by taking use of individual differences in keystroke patterns.

Wahab et al. [6] investigates the security of distinguished account recovery techniques, a crucial weakness in contemporary authentication systems. The authors suggest using keystroke (keyboard) dynamics to upgrade the recovery process in order to behaviorally confirm user identity. During the recovery process, their system is made to record typing data and compare as well as contrast it with biometric profiles that have been preserved. Even in cases where static credentials are hacked/hijacked, the results of legitimate user testing show that this additional layer greatly lower down the risk of account hijacking. Keystroke (Keyboard) dynamics is presented in

the study as an affordable and easy to use addition to multi-factor authentication.

Juels and Rivest [7] introduced the seminal concept of “honeypasswords,” which are fake passwords stored alongside real ones to detect password file breaches. The idea is that if an attacker uses a honeypassword, the system can trigger an alert without the user being compromised. The authors discuss generation strategies that maximize the indistinguishability or flatness of honeypasswords.

Chandok and associates [8] recommends a behavioral biometric system that uses Machine Learning to classify user according to keystroke (keyboard) attributes. The authors in this study implemented classifiers such as Support Vector Machines (SVM) and Random Forests (RF) to authenticate/validate users and gain comprehensive metrics including key hold time, inter key delay, and error rate. To ensure a designated assessment, their collection contains typing samples from both authorized (valid) and unauthorized (invalid) users. The high accuracy of a model and low equal error rates frames that behavioral biometrics can be utilized in the real time authentication systems.

To build a strong password security mechanism, Pagar et al. [9] suggested a hybrid security paradigm that mix up honeypasswords with HoneyEncryption. Honeypasswords misguide deceivers during entering password, and HoneyEncryption provides an additional layer of protection by delivering plausible but incorrect plaintexts when the incorrect key is used. The goal of combining all these methods is to successfully identify and deceive intruders. This strategy provides a thorough and better defense against dictionary and brute force attacks, according to these kind of comparative studies and simulated attacks. By using deceit and obfuscation at several levels, the research showcases defense-in-depth.

In order to manage noisy biometric inputs, Heryanto et al. [10] implements a fuzzy verifier into the Lightweight Authentication Key Agreement (LAKA) protocol. Erratic data input usually causes traditional authentication techniques to fail, especially in behavioral biometrics. The verifier can adapt to the little changes in keystroke (keyboard) pressure and time by incorporating fuzzy logic. A formal model, security analysis, and empirical validation utilizing biometric data sets are highlighted by the authors in this study. Their findings illustrates that the altered (manipulated) protocol preserves security while augmenting fault tolerance and user experience in authenticating different tasks.

Shrivastava [11] in his work, provides a thorough and deep analysis of keystroke dynamics as a part of behavioral biometric for continuous (contiguous) and safe user authentication. Key feature extraction techniques are examined, and classical, machine learning, and deep learning approaches such as SVM, Random Forest, XGBoost, HMM, and CNN based models are evaluated. The precision of the model and Equal Error Rate (EER) are also highlighted in this study. The paper highlights affordability of keystroke dynamics, non-intrusiveness, and scalability. It addresses

contemporary issues like behavioral unpredictability and also recommends future research avenues for reliable and multimodal authentication systems.

Arsh et al. [12] investigates a several keystroke-based authentication models, such as hybrid systems, machine learning classifiers, and statistical techniques. The accuracy, computational efficiency, and spoofing resistance of each individual methods are evaluated by the authors. Additionally, they investigated the effects of ambient conditions, input gadget type, and dataset size. According to the results, hybrid models that combine statistics and artificial intelligence components produce the greatest outcomes. Based on the system requirements, the article offers a deep guidance for choosing the best keystroke authentication model.

Chakraborty et al. [13] evaluated whether AI models more especially, GPT-3 could identify patterns that distinguish legitimate passwords from honeywords. The authors assess GPT-3’s categorization ability after feeding it substantial datasets of honeyword-password pairings. The findings demonstrate that sophisticated language models can minimize the systems flatness by taking use of minute linguistic or syntactic hints in generated honeywords. The study offers methods for creating more indistinguishable honeywords and expresses tremendous serious concerns about the future of deception based security in light of AI developments.

Buckley and his group Keystroke dynamics, or the timing patterns in people’s typing, can exploit more than just authentication symptoms. They can also be used to infer or predict aspects of a identification of user, such their name and the native language, based only on their typing behavior [14]. The authors highlights the security and various privacy concerns of behavioral biometrics by demonstrating that well known character sequences (bigrams) are written with unique and authenticated timing patterns, implying that part of an anonymous name of user’s can be inferred from how (rather than what) they type.

The honeyword password cracking detection system, which was primarily presented by Juels and Rivest and keeps phony decoy passwords alongside the actual one so that any use of a honeyword flags a possible breach, is examined in this work by Erguler [15]. Erguler also examines the overall efficacy of honeyword (decoys) generation techniques, points out best possible flaws in the original honeyword approach, talks about several issues like storage costs and the “flatness” of honeywords, and proposes a different approach that chooses honeywords from user passwords which is already in use to make them more realistic and minimize storage overhead.

Akshaya [16] focused on creating fake (phony) honeywords, or decoys, without regard to user password settings. The authors used character level patterns and password syntax rules to generate honeywords that are statistically similar to the valid ones. This approach improves the flatness while reducing associated risks with using password datasets that have been very much compromised. The method is well scalable and suitable for various cloud authentication environments where privacy

and scalability are essential.

Siahaan [17] is a novel side channel attack approach that allows spoofing keyboard dynamics based authentication by extracting various typing patterns from screen recorded films. The authors suggest using a computer vision system that tracks the movement of the text pointer to estimate inter key timing without directly interacting with the keyboard of victim's. By analyzing the delays of the cursor, the method imitates the typing rhythm or pattern of victim's. The study evaluates the feasibility of the attack's and shows a significant evasion rate (up to 64.3% in password mode) by simulating authentication bypass on websites like KeyTrac. This work is crucial and important because it shows that even in the absence of keyloggers or physical access, behavioral biometrics, like keystroke (keyboard) dynamics, are susceptible to passive observation attacks.

N. Chakraborty [18] mentions flatness, or the ability to generate fake passwords (honeywords) that are usually identical to the real ones, as a troublesome feature of honeyword based authentication systems. The authors critically assess existing approaches and highlight some flaws, especially in relation to Denial of Service attacks and targeted guessing attacks. The authors propose an innovative and novel questionnaire based technique to overcome these issues, which employs episodic memory based questions to generate incredibly convincing honeywords while preserving typo safety and usability. The method significantly reduces the likelihood of attacker identification while boosting resilience to Denial of Service situations. The comparison and contrast research confirms that the proposed system outperforms state of the art methods in many aspects, including storage overhead, flatness, and attack resilience.

This research paper by Bhatt and his co-author [19] provides a comprehensive overview of the specialized study on keystroke dynamics as a behavioral biometric for user identification, tracing and treating its development from early studies to more recent level developments. It discusses how typing rhythm patterns can be used to verify identity of the user, examines various approaches and strategies in the relative area, and offers most fruitful data for upcoming research on biometric authentication systems that use typing behavior for security.

Shadman and team [20] introduces a brief, dedicated and comprehensive keystroke dynamics bunch of data which aims at enhancing cybersecurity by preventing Advanced Persistent Threats (APTs). The dataset is very carefully and thoughtfully designed to capture detailed patterns of user typing and behavioral traits of users, including timing, rhythm, and keypress sequences activities, making it suitable for behavioral biometric analysis. The authors describe the detailed methodology for collection of data, the overall structure of the dataset, and the features extracted, highlighting its potential for developing robust authentication systems and intrusion detection mechanisms. By providing a well structured resource, this work enables researchers to analyze typing behavior for detecting different anomalies, identifying

malicious activity, and improving overall security. The paper emphasizes that leveraging such a dataset can significantly strengthen the defense mechanisms against trained, knowledgeable and sophisticated cyber attacks, which bridges the gap between theoretical research and practical applications in the domain of cybersecurity.

Samonto [21] and team focuses on upgrading current protection in the power systems by optimizing fuzzy logic controller design. It presents a deep comparative study on different membership functions like Triangular, Trapezoidal, and Gaussian Membership Functions for their effectiveness in triggering intelligent relays. By modeling the fault scenarios in three phase system and analyzing transient responses, the researchers conclude that Gaussian membership functions as inputs and trapezoidal/-triangular as outputs yield the most reliable and accurate protection mechanism, achieving a 99% trip performance rate. The study also discusses future enhancements, such as the use of fractional fuzzy inference systems, indicating relevance of the methods for smart grid and microgrid environments.

A generalized methodology to improve the fuzzy transform (F-transform) by adaptively improving the location and support of radial membership functions is proposed by Cammarasana [22] and his team. Accuracy is limited by traditional methods of F-transform, reliance on uniformly sampled function centers, particularly in complicated 2 Dimensional domains. Using adaptive fuzzy and weighted sampling techniques, the authors present an energy minimization based optimization that produces more accurate and keen signal reconstruction with fewer membership functions. This method can be applied to high-dimensional and irregular domains since it greatly increases computational efficiency and reconstruction quality. The research lays the groundwork for more extensive uses in fuzzy signal analysis and picture processing.

Koh [23] proposes a behavioral biometric identification system that leverages the unique typing patterns of users for authentication. The system captures keystroke dynamics, including timing intervals and typing rhythm, and uses these as distinguishing features for individual identification. To efficiently classify users based on these features, the authors employ the Artificial Bee Colony (ABC) optimization algorithm, a swarm-intelligence technique known for its robust search and optimization capabilities. The paper details the system architecture, feature extraction process, and implementation of the ABC algorithm, demonstrating its effectiveness in accurately identifying multiple users from their typing behavior. By integrating keystroke dynamics with the ABC algorithm, this work highlights a novel approach for enhancing biometric identification systems, offering both improved accuracy and adaptability in real-world applications where traditional authentication methods may be insufficient.

Roy [24] presents a broad and systematic review of keystroke dynamics based models for authentication, identification, and prediction, analyzing studies published between the year January 2017 and March 2022. From 9,348 initial records, 127 high quality studies were selected, covering six distinct model types across desktop and

smartphone platforms. The review highlights keystroke dynamics as a behavioral biometric for continuous user verification, discusses challenges such as intra class variation and device heterogeneity, and summarizes state of the art features, classifiers, and research gaps to guide future work.

In the context of IoT contexts, Mukherjee with his team [25] examines the security and resilience of honeyword based authentication methods. The authors examine a number of flaws in current honeyword systems, concentrating on how hackers could make use of these flaws to differentiate real (legitimate) passwords from fake ones. The work identifies most possible dangers of using conventional honeyword techniques in resource constrained and diverse IoT devices through methodical cryptanalysis. The study also highlights the need for more advanced procedures that preserve both security and usability in contemporary connected devices, offering a number of insights and suggestions for strengthening the robustness of IoT authentication techniques.

Huang [26] and team evaluates session based and sliding window based test sample formation strategies, with sliding windows for 30 minutes which ranges from 01 to 30 minutes. Their best performing configuration using a single minute sliding window achieved a 01% False Acceptance Rate (FAR) and 11.5% False Rejection Rate (FRR). The authors simulated shorter insider attacks by injecting impostor keystrokes and demonstrate the high sensitivity of the system in detecting such threats. They utilize the Gunetti & Picardi algorithm as the baseline for similarity comparison, incorporating both “A” and “R” distance measures for n-graph analysis. The study concludes that free text keystroke dynamics that serves as a robust, continuous, and unobtrusive authentication method, especially effective in detecting transient insider threats.

To strengthen heavy security against multiple system intersecting attacks, Akshima [27] suggests improved honeyword generation techniques. The authors present evolving, enhanced and tricky password, user profile, and append secret models to produce near flat honeyword distributions in order to overcome the shortcomings of current techniques like the Paired Distance Protocol. Stronger overall password database breach detection, enhanced resistance to Denial of Service assaults, and greater typo safety utilizing Levenshtein distance are all demonstrated by experimental results on compromised password datasets.

The usefulness of keystroke dynamics as a behavioral based biometric for user authentication is examined by Nova [28]. It offers a thorough and deep comparison of several feature extraction methods and evaluates how well they work when paired with various machine learning models to identify users based on typing of their patterns. The study provides important insights for creating effective and dependable keystroke based authentication systems by highlighting the characteristics and models that can achieve the maximum level of accuracy and robustness. The report effectively provides positive direction for researchers and practitioners on refining behavioral biometric systems for improved security and usability by methodically comparing different techniques.

Xi [29] evaluates how well and quickly different machine learning algorithms work when applied to keyboard dynamics for authentication of user. By gathering and extracting all the keystroke characteristics that define typing behavior, the authors systematically assess the effectiveness of various learning models in differentiating authentic users from different imposters based on these behavioral patterns. The study provides useful insights for improving security in biometric verification by evaluating the efficacy and accuracy of several algorithms on keyboard data to observe which methods are more suited for creating dependable, behavior based authentication systems.

Khokhar [30] examines how the performance and accuracy of fuzzy inference system can be upgraded by including more output membership functions. The authors shows that a richer and larger set of output membership functions enables the system to more accurately capture subtle variations in the inferred results, leading to finer decision making responses compared to conventional configurations with fewer output functions by methodically increasing the granularity of output linguistic categories and assessing the resulting fuzzy model. This work contributes to the design of more accurate and reliable fuzzy logic systems for complex decision making and control applications.

Vaidya [31] and his team in the deep study presents a triad based framework of authentication that enhances cybersecurity by integrating honeywords, keystroke dynamics, and fuzzy logic membership functions. Honeywords help detect password compromise, keystroke dynamics provide behavioral biometric verification based on typing patterns, and fuzzy logic effectively manages uncertainty in user behavior. The combined approach strengthens access control and improves resistance to password attacks and impersonation, making it suitable for secure authentication in modern Industry 4.0 environments.

2.2 Various approaches employed in secure smart systems

Table 2 and Table 3 shows the various approaches employed in secured smart systems, which also compares and contrast existing systems based on several criteria which includes system design, anomaly detection, and mitigation strategies.

Table 2: Comparison of existing secured systems (Part 1 of 2)

Research Reference	System Design	Anomaly Detection	Mitigation	Response	Exp. Results	Firmware	Network Domain	Security Tech.
Security Analysis of Honeywords (2018) [1]	✓	✓			✓		✓	
Intrusion Detection using Keystroke Dynamics (2013) [2]	✓	✓			✓			
Keystroke Dynamics with Fuzzy Logic MF (2015) [3]	✓	✓	✓		✓			
AI and Keystroke Dynamics (2024) [4]		✓			✓			
HMM-based Touch Keystroke Dynamics (2023) [5]		✓			✓			
Account Recovery Protection via Keystrokes (2021) [6]		✓		✓			✓	
Honeywords Framework (2013) [7]	✓							
ML-based Keystroke Authentication (2022) [8]		✓			✓			
Honeyword and Honey Encryption (2017) [9]			✓					
LAKA Protocol with Honeywords (2023) [10]			✓					✓
Exploring Keystroke Dynamics (2025) [11]		✓			✓			
Multiple Keystroke Auth. Approaches (2024) [12]		✓			✓			
GPT-3 Distinguishing Honeywords (2023) [13]		✓					✓	
CLICKA Framework (2022) [14]				✓				
Honeyword Detection Remarks (2014) [15]	✓							
Flat Honeyword Generation (2017) [16]	✓							

Table 3: Comparison of existing secured systems (Part 2 of 2)

Research Reference	System Design	Anomaly Detection	Mitigation	Response	Exp. Results	Firmware	Network Domain	Security Tech.
Spoofing Keystroke Dynamics (2022) [17]		✓						
Questionnaire-based Honeywords (2017) [18]	✓							✓
Keystroke Dynamics Survey (2013) [19]	✓	✓						
APT Keystroke Dataset (2024) [20]		✓			✓			
Fuzzy-based Intelligent Relaying (2021) [21]	✓		✓		✓			
Adaptive MF and F-Transform (2024) [22]			✓					✓
Keystroke ID using ABC Algorithm (2019) [23]		✓			✓			
SLR on Keystroke Models (2022) [24]	✓							
Cryptanalysis of IoT Honeywords (2022) [25]		✓					✓	
Free-text Keystroke Evaluation (2017) [26]		✓			✓			
Secure Honeyword Generation (2019) [27]	✓		✓					
Comparative ML Study on Keystrokes (2022) [28]		✓			✓			
Evaluating Learning Algorithms (2023) [29]		✓			✓			
Higher Output MF Precision (2025) [30]	✓		✓					✓
Triad: Honeywords–Keystrokes–Fuzzy (2025) [31]	✓	✓	✓	✓	✓		✓	✓

3 System Workflow and Detailed Architecture

We propose a secure smart web portal that incorporates the honeyword technique, keystroke dynamics, and fuzzy logic. The system emphasizes authentication, pattern matching, and decision-making to provide stronger protection for data access.

The multi-factor authentication system integrates three key components: honeywords for deception-based security, keystroke dynamics for behavioral biometrics, and fuzzy logic for handling uncertainty in user behavior patterns with the help of tested PI Membership Function, where we will also showcase the efficiency in terms of FAR and FRR. Table 4 shows comparison on FAR and FRR.

Table 4: Comparison on False Acceptance Rate and False Rejection Rate

Attribute	False Acceptance Rate (FAR)	False Rejection Rate (FRR)
Definition	Unauthorized user accepted	Authorized user rejected
Error Type	Type I	Type II
Impact	Security Breach	User frustration
Desired Outcome	Should be low as possible	Should be low as possible
Example	Stranger unlocks your device	Owner denied access to the own device

3.1 Honeyword Generation

The system generates honeywords using a combination of techniques to ensure maximum flatness and indistinguishability from real passwords. The honeyword generation process follows below given steps:

- Examining existing password patterns
- Creating comparable phony passwords
- Implementation of typo-safety mechanisms
- Storage actual passwords in the authentication database

3.2 Keystroke Dynamics Analysis

The keystroke (keyboard) dynamics component captures and examines the following key timing features:

- Dwell time: Duration between key presses and releases
- Flight time: Time between consecutive key events
- Pressure patterns and Typing rhythm
- Variations in inter-key delays

A client side script records the ‘keydown’ and ‘keyup’ timestamps for each and every character in the password field for individual users log in attempt. A feature vector comprising the following statistical metrics is created from the below given raw data:

- **Dwell Time:** Dwell Time is also called as Key Hold Time which is the amount of time a key remains pressed during typing.

For a given key k , the dwell time is defined as:

$$\text{DwellTime}(k) = t_{\text{release}}(k) - t_{\text{press}}(k) \quad (1)$$

Where,

- $t_{\text{press}}(k)$ denotes the timestamp when the key is pressed (KeyDown)
- $t_{\text{release}}(k)$ denotes the timestamp when the key is released (KeyUp)

- **Mean Dwell Time:** The average duration a key is held down. For a given key k , the mean dwell time is defined as:

$$\overline{\text{DwellTime}}(k) = \frac{1}{N} \sum_{i=1}^N (t_{\text{release},i}(k) - t_{\text{press},i}(k)) \quad (2)$$

Where,

- N denotes the total number of occurrences of key k
- $t_{\text{press},i}(k)$ denotes the timestamp when the key is pressed in the i -th instance (KeyDown)
- $t_{\text{release},i}(k)$ denotes the timestamp when the key is released in the i -th instance (KeyUp)

- **Standard Deviation of Dwell Time:** Measures the consistency of key presses. For a given key k , the standard deviation of dwell time is defined as:

$$\sigma_{\text{DwellTime}}(k) = \sqrt{\frac{1}{N} \sum_{i=1}^N (\text{DwellTime}_i(k) - \overline{\text{DwellTime}}(k))^2} \quad (3)$$

Where,

- N denotes the total number of occurrences of key k
- $\text{DwellTime}_i(k) = t_{\text{release},i}(k) - t_{\text{press},i}(k)$
- $\overline{\text{DwellTime}}(k)$ denotes the mean dwell time of key k

- **Average Dwell Time:** the Average Dwell Time is essentially the same as the Mean Dwell Time. For a given key k , the standard deviation of dwell time is defined as: For a given key k , the average dwell time is defined as

$$\overline{\text{DwellTime}}(k) = \frac{1}{N} \sum_{i=1}^N (t_{\text{release},i}(k) - t_{\text{press},i}(k)) \quad (4)$$

Where,

- N denotes the total number of occurrences of key k
- $t_{\text{press},i}(k)$ denotes the timestamp when the key is pressed in the i -th instance (KeyDown)
- $t_{\text{release},i}(k)$ denotes the timestamp when the key is released in the i -th instance (KeyUp)

- **Total Time:** The Total Time (also called session time or typing duration) is the sum of the durations of all keystrokes (or sometimes the difference between the first key press and the last key release). For a sequence of M keystrokes, the total typing time is defined as:

$$\text{TotalTime} = t_{\text{release},M} - t_{\text{press},1} \quad (5)$$

Where,

- $t_{\text{press},1}$ is the timestamp of the first key press
- $t_{\text{release},M}$ is the timestamp of the last key release
- M is the total number of keystrokes in the sequence

- **Average Total Time:** The Average Total Time can be computed as the mean of total times across multiple typing sessions or repetitions: For S typing sessions, the average total time is defined as:

$$\overline{\text{TotalTime}} = \frac{1}{S} \sum_{j=1}^S \text{TotalTime}_j \quad (6)$$

Where,

- S denotes the total number of typing sessions
- TotalTime_j is the total typing time in the j -th session.

- **Flight Time:** Flight Time (also called key latency) measures the time between releasing one key and pressing the next key. For a sequence of keystrokes k_1 and k_2 , the flight time is defined as:

$$\text{FlightTime}(k_1, k_2) = t_{\text{press}}(k_2) - t_{\text{release}}(k_1) \quad (7)$$

Where,

- $t_{\text{release}}(k_1)$ is the timestamp when key k_1 is released
- $t_{\text{press}}(k_2)$ is the timestamp when key k_2 is pressed.

- **Average Flight Time:** The Average Flight Time is the mean of all flight times across multiple key pairs or sessions. For N consecutive key pairs, the average flight

time is defined as:

$$\overline{\text{FlightTime}} = \frac{1}{N} \sum_{i=1}^N \text{FlightTime}_i \quad (8)$$

Where,

- N is the total number of consecutive key pairs
- $\text{FlightTime}_i = t_{\text{press}}(k_{i+1}) - t_{\text{release}}(k_i)$

- **Sum of Square of Differences - Dwell Time:** For a given key k , the sum of squares of differences of dwell time is defined as:

$$SSD_{\text{DwellTime}}(k) = \sum_{i=1}^N (\text{DwellTime}_i(k) - \overline{\text{DwellTime}}(k))^2 \quad (9)$$

Where,

- N denotes the total number of occurrences of key k
- $\text{DwellTime}_i(k) = t_{\text{release},i}(k) - t_{\text{press},i}(k)$
- $\overline{\text{DwellTime}}(k)$ is the mean dwell time of key k

- **Sum of Square of Differences - Flight Time:** For N consecutive key pairs, the sum of squares of differences of flight time is defined as:

$$SSD_{\text{FlightTime}} = \sum_{i=1}^N (\text{FlightTime}_i - \overline{\text{FlightTime}})^2 \quad (10)$$

Where,

- N is the total number of consecutive key pairs
- $\text{FlightTime}_i = t_{\text{press}}(k_{i+1}) - t_{\text{release}}(k_i)$
- $\overline{\text{FlightTime}}$ is the mean flight time

- **Sum of Square of Differences - Total Time:** For S typing sessions, the sum of squares of differences of total time is defined as:

$$SSD_{\text{TotalTime}} = \sum_{j=1}^S (\text{TotalTime}_j - \overline{\text{TotalTime}})^2 \quad (11)$$

Where,

- S denotes the total number of typing sessions
- TotalTime_j is the total typing time in the j -th session
- $\overline{\text{TotalTime}}$ is the mean total time across all sessions

This feature vector serves as an input to the Fuzzy Inference System.

3.3 Fuzzy Logic Integration

The membership function of a fuzzy set is a generalization of the indicator function in classical sets. In fuzzy logic, it represents the degree of truth as an extension of valuation. Degrees of truth are often confused with probabilities, although they are conceptually distinct, because fuzzy truth represents membership in vaguely defined sets, not likelihood of some event or condition. Membership functions were introduced in the year 1965. The inherent variability in human typing behavior is handled by several fuzzy logic membership functions. Some of them are listed below:

- Bell Membership Function
- Gaussian Membership Function
- Singleton Membership Function
- Triangular Membership Function
- Sigmoid Membership Function
- Left-Right Membership Function
- Trapezoidal Membership Function
- Triangular Membership Function
- PI Membership Function

Detailed Architecture

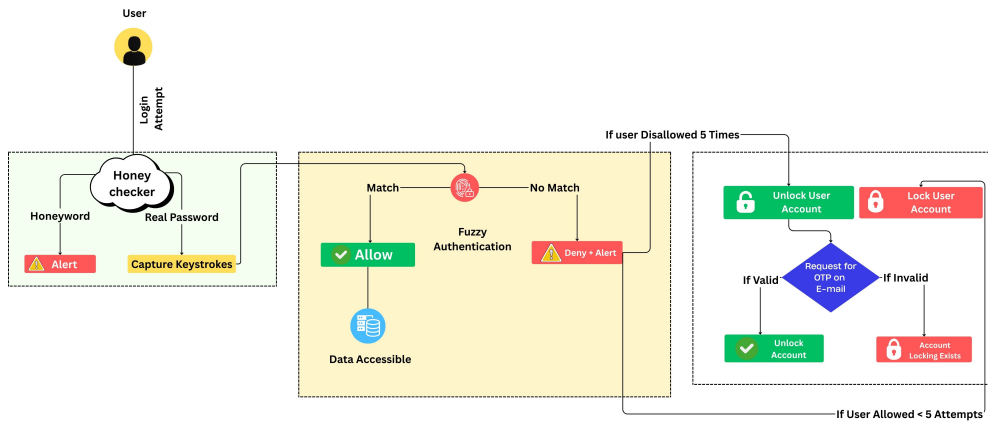


Fig. 3: The figure represents detailed architecture of the system

Fig. 3 illustrates a multi-layered authentication and security workflow that combines honeywords, keystroke dynamics and fuzzy authentication, and also an account lock/unlock process. Here's a clear step-by-step explanation of each section.

System Workflow Description

1. User Login Attempt

The user submits a password during login. This input is passed into the first security layer: the **Honey Checker**.

2. Honey Checker (Honeyword System)

The honey checker verifies whether the submitted password is:

A. A Honeyword (fake password)

Honeywords are decoy passwords stored alongside the real one.

Result:

- Generate an alert.
- The system may also capture keystrokes or other metadata for investigation.

B. The Real Password

If the entered password matches the real password entry:

- System proceeds to capture keystrokes (likely for behavioral analysis).
- User is passed to the next step: **Fuzzy Authentication**.

3. Fuzzy Authentication (Behavior + Credentials)

This layer uses typing rhythm to determine whether the login attempt behaves like the real (legitimate) user.

A. Match Found

- Behavioral profile matches expected patterns stored in the database.
- Access is granted, user can access data.

B. No Match

- Typing pattern mismatch, possible impersonation.
- Access is refused and an alert/prompt is generated.
- A counter tracks the number of times the user is denied the access.

4. Account Locking / Unlocking Mechanism

If any user amongst the valid or invalid has been disallowed 5 times, the system proceeds to the account protection module.

A. If User is Disallowed 5 Times

The account may enter to a locked state. The system sends a request for OTP (One-Time-Password) verification via email.

If the OTP is Valid:

- The system unlocks the user’s account.

If the OTP is Invalid:

- Account remains locked.
- “Account Locking Exists” is triggered.

B. If User Allowed > 5 Attempts

This refers to monitoring abnormal success patterns—e.g., too many successful attempts in unusual patterns—possibly triggering additional checks or locks as part of an anti-bot or anti-brute force safeguard.

The current investigation makes use of the PI membership function, as it was identified as the best suitable membership function in the previous literature, where the PI defines the degree of membership of an input to a fuzzy set using a curve shaped like the Greek letter π :

3.4 Pi-Shaped Membership Function (PIMF)

The Pi-shaped Membership Function (PIMF) is a commonly used fuzzy membership function that combines an S-shaped and a Z-shaped curve to model gradual transitions in fuzzy sets. It is defined using four parameters:

$$P = [a \ b \ c \ d] = [1 \ 4 \ 5 \ 10]$$

where:

- $a = 1$ is the lower bound where the membership value starts increasing from zero.
- $b = 4$ is the point where the membership function reaches its maximum value.
- $c = 5$ is the point where the membership value begins to decrease.
- $d = 10$ is the upper bound where the membership value returns to zero.

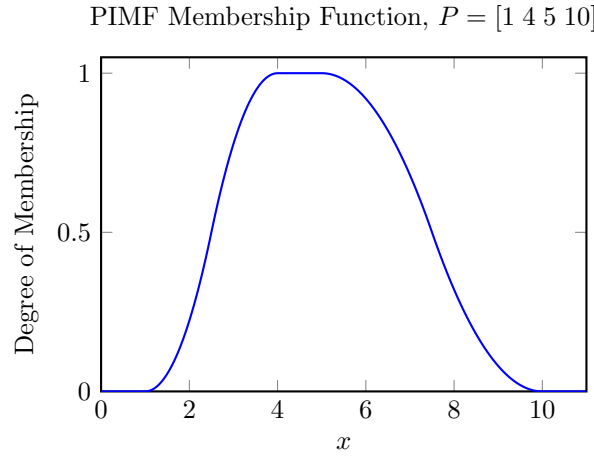


Fig. 4: Pi-shaped Membership Function (PIMF)

The membership function $\mu(x)$ is defined in parts as follows:

There are five different regions in the PI Membership Functions. There is no membership in the fuzzy set in the primary region, as indicated by the membership value of 0. The membership value in the second section rises steadily along a quadratic curve until it equals unity. The third area is a plateau where full membership is shown by the membership value staying at one. The membership value smoothly declines along a quadratic curve in the fourth region. Ultimately, the membership value returns to zero in the final region.

This smooth and continuous behavior makes the PI Membership Function suitable for modeling linguistic variables such as *medium*, *normal*, or *average* in systems based on fuzzy logic. Fig. 4 shows clear representation of PI Membership Function.

$$g(t; \alpha, \beta, \gamma, \delta) = \begin{cases} 0, & t \leq \alpha \\ 2 \left(\frac{t - \alpha}{\beta - \alpha} \right)^2, & \alpha \leq t \leq \frac{\alpha + \beta}{2} \\ 1 - 2 \left(\frac{t - \beta}{\beta - \alpha} \right)^2, & \frac{\alpha + \beta}{2} \leq t \leq \beta \\ 1, & \beta \leq t \leq \gamma \\ 1 - 2 \left(\frac{t - \gamma}{\delta - \gamma} \right)^2, & \gamma \leq t \leq \frac{\gamma + \delta}{2} \\ 2 \left(\frac{t - \delta}{\delta - \gamma} \right)^2, & \frac{\gamma + \delta}{2} \leq t \leq \delta \\ 0, & t \geq \delta \end{cases} \quad (12)$$

Where,

t = Current time

α = Average time – (2 * Standard Deviation)

β = Average time – Standard Deviation

γ = Average time + Standard Deviation

δ = Average time + (2 * Standard Deviation)

The function represents a Pi-shaped fuzzy membership function (PIMF) that describes the degree of membership of a timing feature based on statistical parameters is represented by the function $g(t; \alpha, \beta, \gamma, \delta)$. Behavioral biometric systems frequently employ this function to record natural fluctuations around an average value.

The current observed time is indicated by the variable t . The form and spread of the membership function are determined by the parameters α , β , γ , and δ , which are obtained from the statistical characteristics of the timing data.

- α = Average time – 2σ defines the lower boundary beyond which the membership value becomes zero.
- β = Average time – σ indicates the point at which the membership function reaches its maximum value.

- $\gamma = \text{Average time} + \sigma$ represents the point where the membership value begins to decrease.
- $\delta = \text{Average time} + 2\sigma$ defines the upper boundary beyond which the membership value again becomes zero.

The membership function is defined in a piecewise manner across seven regions. When the current time t is less than or equal to α or greater than or equal to δ , the membership value is zero, indicating no belongingness to the fuzzy set.

For values of t between α and β , the membership value increases smoothly following a quadratic curve, reaching unity at β . Between β and γ , the membership value remains constant at one, representing full membership. For values of t between γ and δ , the membership value decreases smoothly following a quadratic curve, eventually returning to zero.

By using quadratic functions, sudden changes in the membership values are prevented and seamless transitions between regions are ensured. This Pi-shaped structure is appropriate for modeling timing based features in fuzzy inference systems because it efficiently captures natural variability around the average time and offers robustness against many small deviations. Table 5 summarizes the behavior of PI Membership function.

Table 5: Summary of PI Membership Function

Aspect	Description
Shape	Smooth, symmetric, bell-shaped
Best suited for	Medium or average fuzzy values
Accuracy	High
Complexity	Moderate to High
Common applications	Control systems, AI, decision making

3.5 Applications of PI Membership Function

- **Fuzzy Control Systems:** Used in temperature control, speed control, and industrial automation.
- **Decision-Making Systems:** Applied in multi-criteria decision analysis involving moderate values.
- **Pattern Recognition:** Helps classify data with overlapping features.
- **Medical Diagnosis Systems:** Models intermediate conditions such as moderate blood pressure or normal heart rate.
- **Image Processing:** Used in edge detection, contrast enhancement, and noise reduction.
- **Biometric Authentication Systems:** Applied in FAR/FRR analysis to represent medium confidence levels.
- **Artificial Intelligence and Expert Systems:** Supports approximate reasoning and human-like decision modeling.

3.6 Algorithm

Algorithm 1: Character Initialization, JSON Processing, and HTTP Request Handling

```
1: Input: HTTP request parameters (URL, method, headers, body, timeout),  
   character sets, JSON string or Java object  
2: Output: Character maps/sets, JSON result, HTTP response  
3: procedure STEP 1: INITIALIZE CHARACTER SETS  
4:   Define strings:  
5:    $L \leftarrow$  "abcdefghijklmnopqrstuvwxyz"  
6:    $U \leftarrow$  "ABCDEFGHIJKLMNOPQRSTUVWXYZ"  
7:    $D \leftarrow$  "0123456789"  
8:    $S \leftarrow$  "!@#$%^&*()_+ "  
9:   for each character  $c$  in  $L$  do  
10:    Map index  $\rightarrow c$ ; insert into lowercase set  
11:   end for  
12:   for each character  $c$  in  $U$  do  
13:    Map index  $\rightarrow c$ ; insert into uppercase set  
14:   end for  
15:   for each character  $c$  in  $D$  do  
16:    Map index  $\rightarrow c$ ; insert into digit set  
17:   end for  
18:   for each character  $c$  in  $S$  do  
19:    Map index  $\rightarrow c$ ; insert into special set  
20:   end for  
21: end procedure  
22: procedure STEP 2: INITIALIZE JSON PROCESSOR  
23:   Load ObjectMapper instance  
24:   Register JavaTimeModule for date/time parsing  
25:   Provide:  
26:     readJson(json, type)  $\rightarrow$  Deserialized object  
27:     writeJson(object)  $\rightarrow$  JSON string  
28: end procedure  
29: procedure STEP 3: PROCESS HTTP REQUEST  
30:   Extract: URI, Method, Headers, Body, Timeout  
31:   Configure WebClient buffer size  
32:   if Method is POST then  
33:     Send POST request with JSON body  
34:   else if Method is GET then  
35:     Send GET request  
36:   else if Method is PUT then  
37:     Send PUT request with JSON body  
38:   else if Method is PATCH then  
39:     Send PATCH request  
40:   else if Method is DELETE then  
41:     Send DELETE request
```

```

42:   else if Method is HEAD then
43:       Send HEAD request
44:   else if Method is OPTIONS then
45:       Send OPTIONS request
46:   end if
47:   Apply headers and timeout
48:   Receive HTTP response as string
49: end procedure
50: procedure STEP 4: RETURN OUTPUTS
51:   Return initialized character mappings
52:   Return JSON serialization/deserialization results
53:   Return HTTP response
54: end procedure

```

The four main stages of the suggested algorithm are as follows:

- (i) Initialization of the Character Sets
- (ii) JSON processor configuration
- (iii) HTTP request implementation and execution
- (iv) Consolidated output generation

Every step is made to provide consistent connectivity with external services, structured serialization, and dependable data management. Below mentioned is a detailed explanation of each step.

1. Initialization of Character Sets

The method creates the four basic character groups such as: lowercase letters, uppercase letters, numerals, and special symbols that are needed for downstream processing activities in the primary step. After being defined as a string literal, each group is then converted into two complementary data structures which are as follows:

- (a) **Indexed Character Maps** — In the processes like controlled randomization or deterministic encoding, each and every character is coupled to a deterministic positional index that renders fast retrieval.
- (b) **Character Sets** – Allowing the constant time membership evaluation, which is mandatory for the tasks involving validation (such as input sanitization, token verification, and password assessment).

By ensuring that frequently used lexical categories are easily and quickly available in structured format, this step minimizes duplicate computation and offers a consistent representation for character handling procedures. The initialization stage is still computationally light because the size and content of these character lists are fixed.

2. JSON Processor Initialization

A customized JSON processing module is configured in the second step. This module supplements the capability offered by the ObjectMapper library in the context of Java-based systems. To effectively understand and serialize contemporaneous date and time based data formats, the processor is upskilled with extensions (such as the JavaTimeModule).

- (a) **readJson (json, type)** – Deserializes a JSON string into a strongly typed object.
- (b) **writeJson (object)** – Converts a Java object into its JSON representation.

3. HTTP Request Processing

The creation, setting up, and carrying out of HTTP requests are all part of the third step. Key request parameters, including URI, HTTP method, headers, body payload, and timeout value, are received by the procedure, which then sets up the underlying web client with the appropriate buffer allocations. The user-provided request method is mapped to the appropriate HTTP action using a method-selection mechanism. The JSON body created in Stage 2 is utilized as the request content for methods that support payloads (POST, PUT, PATCH). This stage's additional duties include:

- (a) **Applying HTTP headers**, such as authentication tokens or content type specifications.
- (b) **Configuring request timeouts** - It ensures resilience against network delays or non-responsive endpoints.
- (c) **Executing the request** – through a non-blocking web-client implementation.
- (d) **Capturing the response** in string form for subsequent parsing or analysis.

4. Output Consolidation

In the final stage, the algorithm clubs all the outputs generated throughout its execution. These include:

- (a) the initialized character maps and sets.
- (b) the results of JSON serialization or deserialization.
- (c) the raw HTTP response that the external service returned.

All things considered, the algorithm offers an organized and expandable framework that combines three crucial functions such as lexical initialization, JSON data handling, and HTTP communication into a single, coherent process. In the distributed and service based architectures, the distinct division of responsibilities among its four stages guarantees strong interaction, promotes reusability, and enhances maintainability.

4 Experimental Results & Discussions

Standard metrics will be brought in use for comparing and contrasting the performance of the system against several threat models. The prime evaluation is showcased in terms of False Acceptance Rate and False Rejection Rate which is as follows and gears up the researchers study approach:

- **False Acceptance Rate (FAR):** FAR measures the probability that an unauthorized user is incorrectly accepted by the system.

$$\text{FAR} = \frac{\text{Number of False Acceptances}}{\text{Total Number of Impostor Attempts}} \quad (13)$$

- **False Rejection Rate (FRR):** False Rejection Rate (FRR) is the probability that it incorrectly rejects an authorized user.

$$\text{FRR} = \frac{\text{Number of False Rejections}}{\text{Total Number of Genuine Attempts}} \quad (14)$$

Preliminary results demonstrate the efficacy of the integrated approach. The system was tested on a huge dataset which considered various password, keystroke as well as fuzzy logic aspects. As per the study, results for few users are shown due to the unavailability of space, which shows the fruitful initial evaluation results.

Table 6: Number of times the invalid user getting accepted during the dataset collection

Users/Attempts	25	50	75	100
Siddhesh Vaidya	0	1	0	2
Rohit Patil	1	2	0	1
Piyush Yawalkar	1	1	1	1
Siddhesh Sakec	0	1	1	1

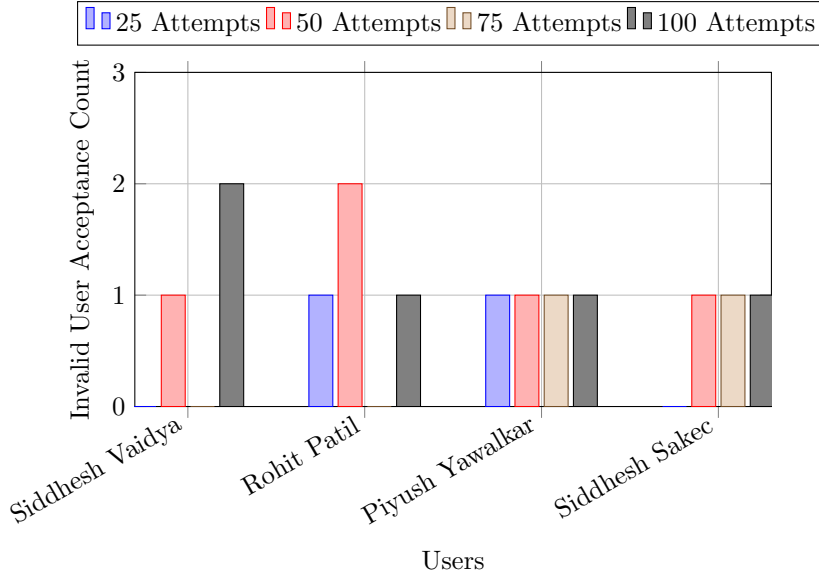


Fig. 5: Bar Chart shows Invalid User Acceptance Across Attempts

Table 6 summarizes the frequency with which invalid users were erroneously authenticated by the system during the current dataset collection phase. The evaluation is conducted across varying numbers of authentication attempts, specifically 25, 50, 75, and 100 trials, to analyze the robustness of the proposed authentication mechanism

under different testing conditions. Each table entry represents the total number of false acceptance events recorded for an individual invalid user corresponding to a particular number of attempts.

A lower false acceptance count indicates superior system performance, as it reflects a stronger ability of the authentication model to distinguish legitimate users from unauthorized ones. Here, the results demonstrates that the system sustains with a relatively low false acceptance rate across most users, suggesting effective resistance against unauthorized access. However, few observable variations across users and attempt counts indicate that false acceptance behavior is influenced by factors such as behavioral similarity between valid and invalid users, intra-user variability, and the size of the sample set used for evaluation.

Additionally, as the number of authentication attempts increases, certain users exhibit a marginal rise in false acceptances, highlighting the cumulative effect of repeated trials on system vulnerability. This kind of trend emphasizes the importance of evaluating authentication systems over extended usage scenarios rather than relying solely on limited test samples. Fig. 5 provides a bar chart visualization of these results, which offers a clearer comparative view of false acceptance patterns across users and their attempt levels, thereby it facilitates better interpretation of the system’s security performance.

Table 7: Number of times the actual user getting rejetced during the current dataset collection

Users/Attempts	25	50	75	100
Siddhesh Vaidya	1	1	1	4
Rohit Patil	2	1	1	3
Piyush Yawalkar	1	3	2	3
Siddhesh Sakec	1	1	2	2

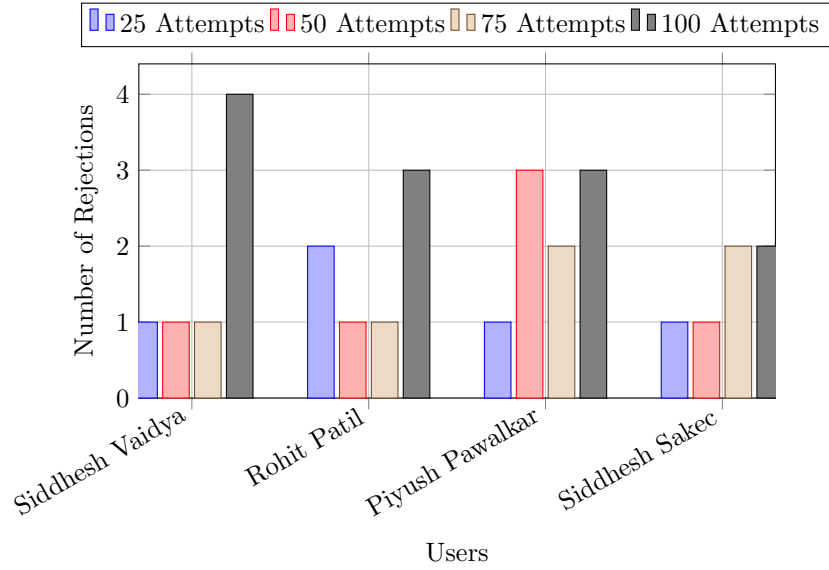


Fig. 6: Bar Chart Showing User-wise Rejection Attempts

Table 7 shows the number of times a legitimate (actual) user was incorrectly rejected by the system during the current dataset collection. The results are reported for different numbers of authentication attempts (25, 50, 75, and 100). Each value represents the count of false rejections experienced by a specific user for the given attempt set. Higher rejection counts indicate reduced usability, as genuine users are denied access. The typing variation across the users and increasing attempts reflects natural behavioral variability and its impact on False Rejection Rate (FRR) of the system. Fig. 6 shows the bar chart representation for better understanding.

Table 8: FAR and FRR Analysis for Different Users Across Multiple Attempts

Users	25 Attempts		50 Attempts		75 Attempts		100 Attempts	
	FAR	FRR	FAR	FRR	FAR	FRR	FAR	FRR
Siddhesh Vaidya	0.00	0.04	0.02	0.02	0.00	0.01	0.02	0.04
Rohit Patil	0.04	0.08	0.04	0.02	0.00	0.01	0.01	0.03
Piyush Yawalkar	0.04	0.04	0.02	0.06	0.01	0.03	0.01	0.03
Siddhesh Sakec	0.00	0.04	0.02	0.02	0.01	0.03	0.01	0.02

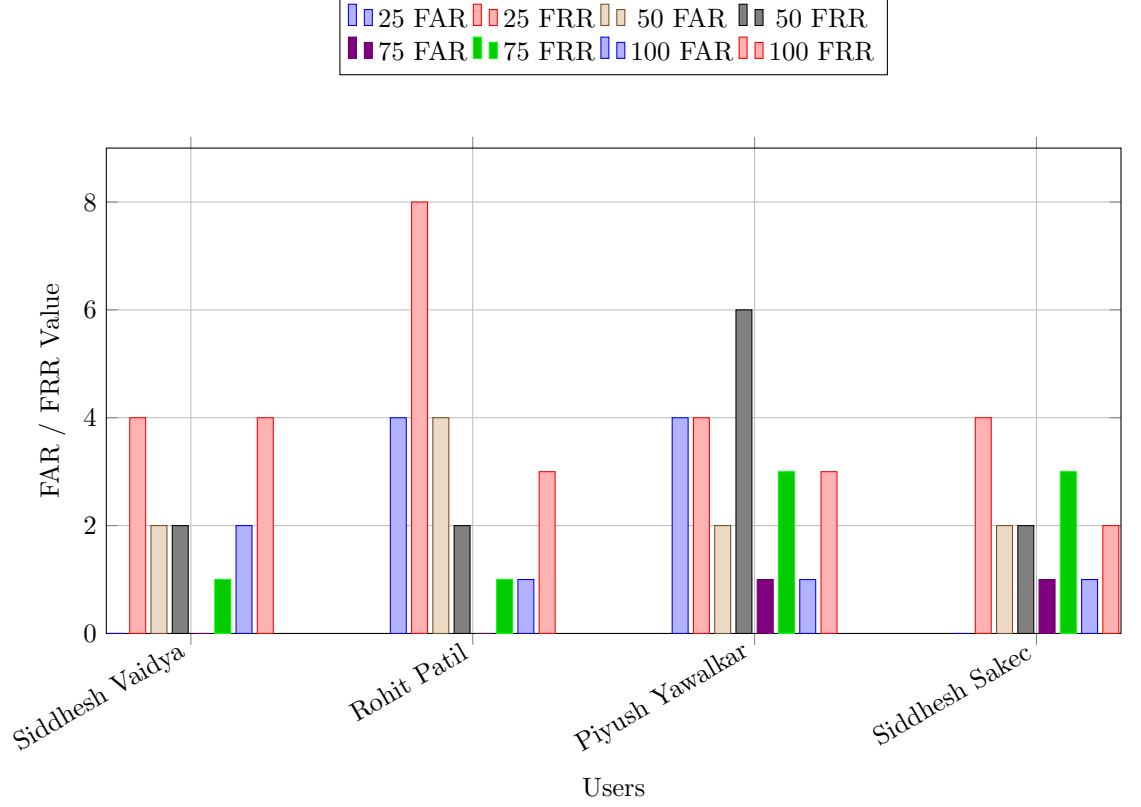


Fig. 7: Bar Chart shows FAR and FRR for Various Users Throughout Attempts

The False Acceptance Rate (FAR) and False Rejection Rate (FRR) for multiple users over different numbers of authentication/validation tries, such as 25, 50, 75, and 100, are displayed in Table 8. While FRR indicates the possibility that a genuine user will be accidentally denied access to the system, FAR quantifies the probability that an unauthorized (phony) user will be mistakenly granted access. Fig. 7 shows the bar chart representation for better understanding.

According to the findings observed in the study, FAR constantly stays low for every person enrolled in the system and throughout multiple attempts, which indicates effective and efficient protection mechanism against unwanted access. Due to such typical fluctuations in the user behavior during authentication, FRR is typically slightly greater than FAR, but also the motive would be to reduce FRR as well. Both FAR and FRR tend to decline or stabilize as the number of trials increases, indicating improved reliability as more data is taken into account. Overall, the findings show that the suggested authentication solution strikes a fair balance between security and usability.

5 Conclusion and Future Work

Honeywords, keystroke dynamics, fuzzy logic, HMM, SVM, and LAKA are the authentication techniques that we have thoroughly examined and reviewed in this paper. Building on these investigations, we provide a suitable authentication framework that uses fuzzy logic for decision making, keystroke dynamics for pattern matching, and the honeyword technique for authentication confirmation. The preliminary experimental findings are quite promising, showing notable security enhancements while maintaining usability for authorized users. Furthermore, the adaptability of the system guarantees long term resilience against impending, changing dangers.

A best possible technique for multi factor authentication is the combination of three security layers, such as fuzzy logic, keystroke dynamics, and honeywords. The future study may focus on following areas:

- Optimization of fuzzy membership functions
- Implementation of adaptive learning mechanisms
- Integration with mobile and IoT devices
- Large-scale deployment and evaluation

References

- [1] D. Wang, H. Cheng, P. Wang, J. Yan, and X. Huang, “A Security Analysis of Honeywords,” *Network and Distributed Systems Security (NDSS)*, vol. 16, no. 5, pp. 782–795, 2018. <http://dx.doi.org/10.14722/ndss.2018.23142>
- [2] M. Sridhar, T. Abraham, J. Rebello, and A. D’Souza, “Intrusion detection using keystroke dynamics,” in *Proc. 3rd Int. Conf. Trends in Inf., Telecommun. and Comput. (TITC)*, 2013, pp. 137–144. https://doi.org/10.1007/978-1-4614-3363-7_16
- [3] M. Sridhar, S. Vaidya, and P. Yawalkar, “Intrusion detection using keystroke dynamics & fuzzy logic membership functions,” in *Proc. Int. Conf. Technol. for Sustainable Develop. (ICTSD)*, 2015, pp. 1–10. <https://doi.org/10.1109/ICTSD.2015.7095873>
- [4] B. Tural, Z. Örpek, and S. Özmen, “Artificial Intelligence and Keystroke Dynamics: The Mysterious World of Personal Signatures,” in *Proc. Int. Congr. Hum.-Comput. Interact., Optim. Robotic Appl. (HORA)*, 2024, pp. 1–6. <https://doi.org/10.1109/HORA61326.2024.10550804>
- [5] U. Biloti, C. Bisogni, A. Castiglione, M. Nappi, and C. Pero, “User Identification through Hidden Markov Model-based Touch Keystroke Dynamics,” *Proc. 5th Int. Conf. on Bio-engineering for Smart Technologies (BioSMART)*, pp. 1–4, 2023. <https://doi.org/10.1109/BioSMART58455.2023.10162120>

- [6] A. Wahab, D. Hou, S. Schuckers, and A. Barbir, “Utilizing keystroke dynamics as additional security measure to protect account recovery mechanism,” in *Proc. 7th Int. Conf. Inf. Syst. Security and Privacy (ICISSP)*, 2021, pp. 33–42. <https://www.scitepress.org/Papers/2021/101912/101912.pdf>
- [7] A. Juels and R. L. Rivest, “Honeywords: making password-cracking detectable,” in *Proc. 20th ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, 2013, pp. 145–160. <https://doi.org/10.1145/2508859.2516671>
- [8] R. Chandok, V. Bhoir, and S. Chinnaswamy, “Behavioural biometric authentication using keystroke features with machine learning,” in *Proc. 4th Int. Conf. Innov. Trends in Inf. Technol. (ICITIIT)*, 2022, pp. 1–6. <https://doi.org/10.1109/INDICON56171.2022.10040168>
- [9] V. R. Pagar and R. G. Pise, “Strengthening password security through honeyword and Honeyencryption technique,” in *Proc. Int. Conf. Trends in Electron. and Informatics (ICEI)*, 2017, pp. 827–831. <https://doi.org/10.1109/ICOEL2017.8300819>
- [10] A. Heryanto, D. Stiawan, et al., “Modification of LAKA Protocol Using Fuzzy Verifier and Honeywords,” in *Proc. Int. Conf. on Computing, Information and Communication Systems (ICoCICs)*, 2023. <https://doi.org/10.1109/ICoCICs58778.2023.10277337>
- [11] A. Shrivastava and S. Bollavarapu, “Exploring Keystroke Dynamics: Enhancing Authentication Through Typing Patterns,” *Proc. International Conference on Computer Science and Communication Engineering (ICCSCE 2025)*, pp. 3127–3140, 2025. https://doi.org/10.2991/978-94-6463-858-5_261
- [12] A. Arsh, N. Kar, S. Das, and S. Deb, “Multiple approaches towards authentication using keystroke dynamics,” *Procedia Comput. Sci.*, vol. 235, pp. 2609–2618, 2024. <https://doi.org/10.1016/j.procs.2024.04.246>
- [13] N. Chakraborty, S. Wang et al., “The tables have turned: GPT-3 distinguishing passwords from honeywords,” in *Proc. Int. Conf. Inf. Syst. Security (ICISS)*, 2023, pp. 317–338. <https://doi.org/10.1109/CNS59707.2023.10288643>
- [14] O. Buckley, D. Hodges, J. Windle, and S. Earl, “CLICKA: Collecting and leveraging identity cues with keystroke dynamics,” *Computers & Security*, vol. 120, pp. 102780, 2022. <https://doi.org/10.1016/j.cose.2022.102780>
- [15] I. Erguler, “Some Remarks on Honeyword Based Password-Cracking Detection,” *IACR Cryptology ePrint Archive*, vol. 2014, p. 323, 2014. <https://eprint.iacr.org/2014/323.pdf>
- [16] K. Akshaya and S. Dhanabal, “Achieving flatness from non-realistic honeywords,” in *Proc. Int. Conf. Innov. in Inf., Embedded and Commun. Syst. (ICIIECS)*,

- 2017, pp. 1–3. <https://doi.org/10.1109/ICIECS.2017.8276120>
- [17] C. R. P. Siahhaan and A. Chowanda, “Spoofing keystroke dynamics authentication through synthetic typing pattern extracted from screen-recorded video,” *J. Big Data*, vol. 9, no. 111, 2022. <https://doi.org/10.1186/s40537-022-00662-8>
 - [18] N. Chakraborty, S. Singh, and S. Mondal, “On Designing A Questionnaire Based Legacy-UI Honeyword Generation Approach For Achieving Flatness,” *CoRR*, vol. abs/1708.01295, 2017. <https://doi.org/10.48550/arXiv.1708.01295>
 - [19] S. Bhatt and T. Santhanam, “Keystroke dynamics for biometric authentication — A survey,” *Proc. Int. Conference on Pattern Recognition, Informatics and Mobile Engineering (PRIME)*, 2013, pp. 17–23. <https://doi.org/10.1109/ICPRIME.2013.6496441>
 - [20] X. Wang, R. Shadman, D. Hou, F. Hussain, and S. Schuckers, “A Novel Keystroke Dataset for Preventing Advanced Persistent Threats,” in *Proc. 13th Int. Conf. on Pattern Recognition Applications and Methods (ICPRAM)*, 2024, pp. 894–901. <https://doi.org/10.5220/0012558000003654>
 - [21] S. Samonto, S. Kar, S. Pal et al., “Fuzzy based algorithm for stage-I of cascaded intelligent relaying,” *Int. J. Intell. Sci.*, vol. 11, no. 1, pp. 1–18, 2021. <https://doi.org/10.4236/ijis.2021.111003>
 - [22] S. Cammarasana and G. Patanè, “Adaptive membership functions and F-Transform,” *IEEE Trans. Fuzzy Syst.*, vol. 32, no. 5, pp. 2786–2796, 2024. <https://doi.org/10.1109/TFUZZ.2024.3360633>
 - [23] P. M. Koh and W. K. Lai, “Keystroke Dynamics Identification System using ABC Algorithm,” in *Proc. IEEE Int. Conf. on Automatic Control and Intelligent Systems (I2CACIS)*, 2019, pp. 108–113. <https://doi.org/10.1109/I2CACIS.2019.8825073>
 - [24] S. Roy, J. Pradhan et al., “A Systematic Literature Review on Latest Keystroke Dynamics Based Models,” *IEEE Access*, vol. 10, pp. 92192–92236, 2022. <https://doi.org/10.1109/ACCESS.2022.3197756>
 - [25] N. Chakraborty, M. Mukherjee, J. Li, M. Shojafar, and Y. Pan, “Cryptanalysis of a Honeyword System in the IoT Platform,” *IEEE Internet of Things Journal*, vol. 9, no. 4, pp. 2614–2626, 2022. <https://doi.org/10.1109/JIOT.2021.3080676>
 - [26] J. Huang, D. Hou, and S. Schuckers, “A practical evaluation of free-text keystroke dynamics,” *Proc. IEEE Int. Conf. Identity, Security Behav. Anal. (ISBA)*, 2017. <https://doi.org/10.1109/ISBA.2017.7947695>
 - [27] A. Akshima, D. Chang, A. Goel, S. Mishra, and S. K. Sanadhya, “Generation of Secure and Reliable Honeywords, Preventing False Detection,” *IEEE Trans.*

on *Dependable and Secure Comp.*, vol. 16, pp. 757–769, 2019. <https://doi.org/10.1109/TDSC.2018.2824323>

- [28] K. Nova, “Analyzing Keystroke Dynamics for User Authentication: A Comparative Study of Feature Extractions and Machine Learning Models,” *Sage Science Review of Applied Machine Learning*, vol. 5, no. 2, pp. 67–80, 2022. <https://journals.sagescience.org/index.php/ssraml/article/view/72>
- [29] T. Xi, I. Kuzminykh, B. Ghita, and T. Bakhshi, “Evaluating Learning Algorithms for Keystroke Based User Authentication,” in *Proc. IEEE Int. Black Sea Conf. on Communications and Networking (BlackSeaCom)*, 2023, pp. 283–288. <https://doi.org/10.1109/blackseacom58138.2023.10299695>
- [30] S. U. D. Khokhar, A. Nadeem, A. A. Rizvi, and M. Y. Noor, “Applying a higher number of output membership functions to enhance the precision of a fuzzy system,” *IEEE Trans. Emerging Topics in Computational Intelligence*, vol. 9, no. 1, pp. 394–405, 2025. <https://doi.org/10.1109/TETCI.2024.3425309>
- [31] S. A. Vaidya, and B. V. Patel, “A Triad for Enhanced Cybersecurity using Honeywords, Keystroke Dynamics, and Fuzzy Logic Membership Functions,” *Proc. International Conference on Emerging Trends in Intelligent Technologies for Trustworthy Systems (ICETI4T)*, 2025. <https://doi.org/10.1109/ICETI4T63625.2025.11132189>

Supplementary information. Not applicable

Conflict of Interest

There is no conflict of interest with respect to the publication of the paper.

Research Funding

No research funding was provided.