Enhancing Zero Trust with Continuous Authentication

A Study on Behavioral Biometrics Using Keystroke Dynamics

Author: Saivarshitha Martha

Master's Degree Project in
Information Security
Term 2025
Supervisor: Yuhong Li
Department of Computer
and Systems Sciences

Stockholm
University

Synopsis

Introduction: This thesis explores the integration of continuous authentication mechanisms with zero trust architecture, using keystroke dynamics as a behavioral biometric. Keystroke dynamics, the analysis of unique typing patterns, provides a non-intrusive, passively monitoring solution for continuous identity verification.

Research Question: How can continuous authentication using keystroke dynamics be implemented with machine learning to support zero trust security principles?

Methodology: This study employs an experimental research strategy, utilizing secondary data collection methods and applying a quantitative approach for analysis, where machine learning models are trained and evaluated on structured keystroke features.

Results: The research demonstrated that advanced deep learning models, particularly the CNN-LSTM architecture, significantly outperformed traditional machine learning approaches in continuous authentication using free-text keystroke dynamics. The system achieved high accuracy (up to 98.92 %) and aligned well with zero trust principles, confirming its potential for secure, real-time, and device-independent user verification.

Discussion: This study demonstrates that CNN-LSTM models can provide accurate, real-time continuous authentication using free-text keystroke dynamics, effectively aligning with zero trust security principles. However, scalability remains challenging due to the complexity of creating individual user models and limitations in dataset size. Future work should investigate on adaptive models and multi-modal biometrics to enhance generalizability and robustness.

Abstract

The core principle of zero trust architecture is based on **"never trust, always verify"**. Zero trust models emphasize the need for continuous monitoring and real-time validation. Traditional static authentication methods often fail because, stolen login credentials or an unattended session can be exploited by attackers. To address this issue, continuous authentication verifies user identity consistently throughout a session that supports zero trust principle.

One innovative approach is to use a user's typing patterns (Keystroke Dynamics) as a behavioral biometric authentication factor. This method identifies users passively, without disrupting their session. This study aims to develop a free-text continuous authentication system using machine learning that adapts to individual typing behaviors while supporting the zero trust principle.

The research employs an experimental strategy with secondary data collection and quantitative analysis to evaluate the performance of different machine learning models. The study utilizes Random Forest (RF), Decision Tree (DT), Support Vector Machine (SVM), Convultional Neural Network (CNN), and a hybrid model combining CNN with Long Short-Term Memory (LSTM), as machine learning algorithms to implement continuous authentication effectively.

Among these CNN-LSTM achieved accuracy of 98.92 % with low false acceptance and false rejection rates. Training across each individual helped to capture unique typing patterns. The findings demonstrate the feasibility of real-time, device-independent, and user-specific continuous authentication aligned with zero trust principles. While promising, challenges such as model scalability and user behavior variability highlight areas for future research, including adaptive modeling and multimodal biometric integration.

Keywords: zero trust architecture, continuous authentication, keystroke dynamics, machine learning.

Acknowledgements

I would like to express my gratitude to Yuhong Li, my supervisor at Stockholm University, for her guidance and insightful feedback throughout the development of this Master's thesis. Her expertise in cybersecurity played a crucial role in shaping the direction and quality of work.

I also extend my sincere thanks to my colleagues at the Department of Computer and Systems Sciences (DSV) who engaged with my work during seminars and peer reviews. Their thoughtful questions, critical reflections, and constructive input challenged me to refine ideas and improve the quality of this work.

To my family and friends, thank you for your patience, encouragement, and constant support during this demanding period. Your belief in me kept grounded and focused. Your prayers, kindness, and understanding meant more than words can say.

Abbreviations

Abbreviation	Description
AI	Artificial Intelligence
CA	Continuous Authentication
CNN	Convultional Neural Network
DT	Decision Tree
FAR	False Acceptance Rate
FRR	False Rejection Rate
LSTM	Long Short-Term Memory
ML	Machine Learning
NIST	National Institute of Standards and Technology
RF	Random Forest
SVM	Support Vector Machine
ZTA	Zero Trust Architecture

List of Figures

List of Tables

Contents

1

Introduction

The rapid growth of the digital landscape has brought about unprecedented challenges to information security technology. Organizations today are confronted with increasingly sophisticated and ongoing cyberattacks. Traditional security models that depend heavily on fixed perimeters and trusted internal boundaries are no longer effective in addressing these evolving threats. In light of this, zero trust Architecture (ZTA) has emerged as a revolutionary approach to cybersecurity. ZTA is founded on the principle that no entity whether it is internal or external, should not be automatically trusted. Instead, every access request must undergo continuous verification. This approach prioritizes strict access controls, dynamic authentication, and ongoing verification to reduce risks [1].

Artificial Intelligence (AI) plays a pivotal role in strengthening ZTA by automating threat detection and response. AI leverages machine learning, deep learning, and natural language processing to analyze vast datasets and detect anomalies that may signal potential breaches. The integration of AI within ZTA enables real-time, context-aware security measures. AI enhances zero trust by automating identity verification, detecting behavioral deviations, and adapting security policies based on changing threat landscapes. Machine learning (ML) offers various classification algorithms capable of distinguishing legitimate user activities from unauthorized events, thus providing a foundation for developing user models essential for Continuous Authentication (CA). As mobile devices and Bring Your Own Device (BYOD) policies introduce additional risks to organizational security, threat analysis and threat intelligence increasingly depend on machine learning methodologies to efficiently address these challenges.

Traditional authentication systems typically rely on initial login credentials (e.g., username and password, or multi-factor authentication) that remain active throughout a session. However, this approach has limitations; for example, if a user leaves their device unattended, a malicious actor could exploit the active session unless additional security measures are implemented [2].

To mitigate aforementioned risks, Continuous Authentication (CA) in combination with machine learning models can be used to learn the normal behavior of users and devices, flagging deviations as potential threats. This continuous behavioral analysis reinforces the core principle of ongoing verification of ZTA, transforming it from a reactive to a proactive security model [3].

Continuous Authentication (CA) persistently verifies user identity throughout a session, either actively or passively. Active authentication requires explicit user input, such as entering passwords or biometric verification, whereas passive authentication unobtrusively monitors user behavior in the background. CA enhances usability and security by automatically securing systems upon detecting anomalies. Techniques employed in CA include biometric authentication systems and context-aware methods, often combined into multimodal authentication systems to improve reliability and security [4].

Biometric authentication systems are categorized into physiological and behavioral types. Physiological biometrics, including fingerprint, facial, iris, or voice recognition, rely on stable physical traits but face challenges such as spoofing vulnerability, user discomfort, hardware costs, and privacy concerns. Consequently, interest in behavioral biometrics has increased, utilizing user behavior patterns such as swiping gestures, mouse movements, and typing rhythms for authentication. Among these methods, keystroke dynamics, or typing dynamics, has emerged as a promising passwordless authentication technique [4].

Keystroke dynamics involves analyzing unique typing behaviors, including timing data such as key press durations and intervals between key presses. Replicating an individual's typing pattern with precision is inherently challenging, making keystroke dynamics a robust biometric method. A significant advantage of keystroke-based authentication is its hardware-independent implementation, facilitating seamless and continuous authentication without user disruption. Beyond authentication, keystroke dynamics can serve as an effective intrusion detection mechanism [5].

The study aims to design and evaluate a continuous authentication system utilizing various machine learning models to analyze user behavior and detect anomalies based on keystroke dynamics data.

The structure of this study is as follows: **Section 2** provides extended background information and reviews recent literature; **Section 3** outlines the research methodology; **Section 5** discusses experimental results; and **Section 6** concludes the study and identifies future research directions.

1.1 Research Problem

Most existing research studies focuses on one-time or static authentication methods, verifying users only during login with fixed-text inputs in controlled settings/ environments. For example, the study by [5] primarily evaluates static authentication using key stroke and emphasizes the necessity for larger datasets and cross-device testing in realistic conditions.

This static approach is insufficient for zero trust security frameworks, which rely on continuous verification throughout the entire user session based on the core principle of "never trust, always verify." Although mobile-focused studies like [2] aim to achieve continuous authentication, they typically depend on fixed datasets, such as the CMU keystroke dataset [6], which do not represent real-world, dynamic typing behaviors adequately. Moreover, these studies recommend exploring deep learning methods to enhance model robustness and adaptability to diverse user conditions.

Additionally, free-text input, which closely mirrors real-world typing scenarios, is underused despite its suitability for context-aware continuous authentication. The study by [7] highlights this gap and advocates developing machine learning models capable of continuous authentication that adapt to variability in user behavior, devices, and context. Similarly, [8] emphasizes the need for individual-specific machine learning models instead of generalized models to reduce false positives and better capture personal typing behaviors across different contexts and devices.

Thus, despite existing foundational research, there remains a significant gap in creating a real-time continuous authentication system that effectively leverages free-text keystroke dynamics. Addressing this gap would contribute meaningfully to the fields of cybersecurity, Human-Computer Interaction (HCI), and Identity and Access Management (IAM), ultimately fostering a more secure and user-friendly digital environment. The aim of this study is to address this research void, offering valuable insights for both academics and practitioners in shaping the future landscape of authentication methods and technologies, fostering a more secure and user-friendly digital environment.

1.2 Research Question

How can continuous authentication using keystroke dynamics be implemented with machine learning to support zero trust principles ?

To address this primary research question, the following sub-questions were formulated:

Sub-RQ1: How can continuous authentication using keystroke dynamics can be implemented using machine learning ?

Sub-RQ2: What is the performance of different machine learning algorithms in detecting users based on keystroke dynamics ?

Sub-RQ3: To what extent can continuous authentication using keystroke dynamics support or enhance zero trust security principles ?

This research aims to design and evaluate a real-time continuous authentication system that leverages keystroke dynamics as a behavioral biometric, aligned with zero trust security principles. By applying machine learning models to analyze users' natural typing patterns, the study seeks to enable ongoing identity verification beyond the initial login phase. The objective is to strengthen access control by ensuring that only legitimate users retain session access, thereby reducing dependence on static, one-time authentication methods and mitigating the risk of unauthorized access in dynamic, high-risk environments.

1.3 Delimitation

One of the key delimitations of this study is the variability in user typing behavior, which can be influenced by factors such as emotional state, fatigue, stress, or environmental distractions. Since keystroke dynamics are behaviorally driven, the typing pattern of an individual may not remain consistent across all sessions. This fluctuation can affect the accuracy of the continuous authentication system, potentially resulting in false rejections or reduced reliability under certain conditions.

Another important delimitation is the system's reliance on historical data for profile creation. For new users who do not yet have an established keystroke profile, the model cannot perform accurate authentication until sufficient typing data has been collected and analyzed. This initial data acquisition phase introduces a delay in real-time deployment and limits the system's ability to immediately authenticate first time users.

1.4 Use of AI Tools

For this study Chat-GPT has been used for brainstroming the research topics and to gain in-deapth knowledge of the topic. Chat-gpt has been used in debugging the python code while training the machine learning models. Grammarly also been used in correcting the grammar mistakes to ensure a smooth flow of the report.

2

Extended Background

Continuous authentication using behavioral biometrics is a modern approach combining traditional machine learning and deep learning techniques.This chapter presents the theoretical aspects that enable the implementation of this approach.

2.1 Zero Trust Architecture

Zero trust is a modern security framework that works on the principle "**never trust**, **always verify**". National Institute of Standards and Technology (NIST), in its publication SP 800-207, defines ZTA as a security framework that eliminates implicit trust by continuously validating all access attempts, regardless of the user's location, device, or network. The model assumes that a breach has already occurred or is imminent, and therefore every access request must be explicitly verified [9]. NIST outlines the fundamental principles of zero trust as follows:

- **Assume breach:** This principle acknowledges that threats may already exist inside or outside the network perimeter. Reject the idea of a trusted internal network and treat every request, user, or device as potentially compromised.

- **Explicit verification of identity and context:** Access decisions must be based on strong identity verification and contextual data.

- **Enforcement of least privilege access:** Users, devices, and applications have the minimum level of access required to perform their tasks. This minimizes the potential for misuse or escalation of privileges.
- **Continuous monitoring and real time validation:** The system continuously monitors user behavior, session activity, and access patterns to detect anomalies or policy violations.
- **Micro segmentation:** The network is divided into smaller, isolated segments (e.g., by application, user group, or department), so that if one area is compromised, the attacker cannot move freely to other parts of the network.
- **Granular device and resource access Control:** Access decisions are based not only on who is requesting access, but also on what device is being used and which specific resource is being requested.
- **Data-centric security policies:** Security is built around protecting data directly, not just the infrastructure. Controls are applied based on data classification, sensitivity, and compliance requirements.
- **Policy enforcement at resource level:** Rather than relying solely on network level controls, zero trust enforces security at the point of resource access (e.g., application, API, database).
- **Security orchestration and automation:** To operate at scale and respond to threats in real time, zero trust architectures rely on automated threat detection, policy updates, and access control decisions.

2.2 Authentication

Authentication is a fundamental process in information system security. The need for reliable authentication methods is increasingly imperative due to the growing complexity of cyber security threats and attackers constantly seeking ways to bypass traditional identity control systems. The process includes identification (declaring an identity) and verification (confirming the accuracy of the declaration). For example, a user might declare their identity via a username (identification) and verify it using a password or biometric method (verification). [10]

Based on [11] authentication systems are broadly categorized into two main types: traditional and biometric. Each of these categories is further subdivided into two major subtypes, as illustrated in Figure 2.1.

- **Knowledge based (what you know):** is something the user knows, including passwords, patterns, or answers to security questions.
- **Object based (what you have):** is Something that user owns, including physical objects like cards or tokens.

Biometric based authentication techniques are further divided into physiological and behavioral.

- **Physiological based (What you are):** a method of verifying an individual's identity based on their inherent physical characteristics.
- **Behavioral based (What you do):** refers to the unique behavioral traits that can be used for human authentication.

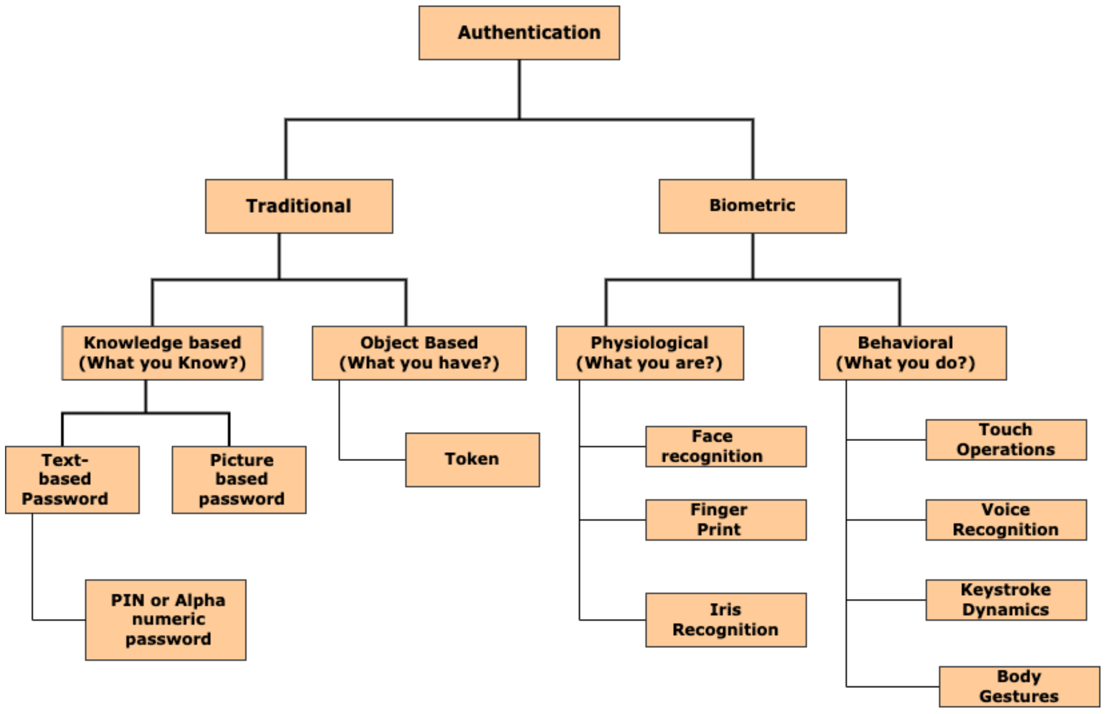


Figure 2.1: Types of Authentication [11]

Behavioral biometrics are further classified into four sub categories namely: Touch operations, Voice recognition, keystroke dynamics, and Body gestures [12].

- **Touch operation** involves analyzing how a user interacts with a touchscreen device. It includes features such as finger movement, touch pressure, swipe speed, and gesture patterns.

- **Voice recognition** model uses acoustic signals generated by the human body - such as breathing, dental clicks, or sounds produced during speech or eating - for user authentication. While this method offers a high degree of privacy and discretion, its effectiveness can be impacted by ambient noise and typically requires advanced signal processing techniques to ensure accuracy and reliability..

- **Body gesture** recognition focuses on identifying user's based on their physical movements, such as hand gestures, arm swings, or head movements. However, they can be obtrusive, as they usually require deliberate user actions.

- **Keystroke dynamics** refers to the analysis of a user's unique typing patterns on a keyboard or touchscreen. It captures timing features such as the duration a key is held (dwell time) and the time between key presses (flight time). These subtle variations in typing behavior can be used to verify or continuously authenticate a user's identity. It is unobtrusive and does not require additional hardware, making it suitable for real-time, continuous authentication systems.In this context, fixed-text refers to users typing a predefined string repeatedly, useful for static authentication, while free-text involves typing natural language passages, offering a more realistic basis for continuous authentication.

2.3 Continuous Authentication

Continuous authentication can be defined as authenticating continuously and passively monitoring users by means of recognizing user features and actions(i.e., physiological biometrics, behavioral biometrics, or context aware authentication modes).Continuous authentication has several advantages [13].

- **Increased security:** Continuous monitoring significantly increases the difficulty of system compromise.

- **Non-intrusive operation:** Users can verify their identity seamlessly without interrupting their ongoing activities.

- **Dynamic adaption:** Systems can adapt to changes in user behaviour, improving overall accuracy.

- **Easy integration:** Can be integrated into existing hardware without additional equipment costs.

- **Flexibility:** Combined with anomaly detection algorithms, continuous authentication can ensure a high level of security without affecting user experience. The use of such technologies opens new prospects for applications like personal data protection, secure access to critical infrastructures, and improved user experiences in digital environments.

2.4 Machine Learning

Machine learning is a subset of artificial intelligence that enables computers to learn from data and make decisions or predictions without explicit programming. The primary types of machine learning are discussed below:

- **Supervised learning:** This is the most common type of machine learning, where the model is trained on a labeled dataset. During training, the model learns a mapping between inputs (features) and outputs (labels). Once trained, the model can accurately predict outputs for new, unseen data. Examples include linear regression, logistic regression, decision trees, and support vector machines for classification problems.

- **Unsupervised learning:** In this approach, the model is trained on an unlabeled dataset and independently identifies patterns and relationships within the data. Common techniques include k-means clustering and principal component analysis (PCA).

- **Reinforcement learning:** This type of machine learning involves an agent that learns to make decisions by interacting with its environment. The agent receives rewards or penalties based on its actions and aims to maximize its cumulative reward over time. Reinforcement learning applications commonly include game playing, robotics, and resource management.

Deep learning, a specialized subfield of machine learning, uses algorithms based on multilayered artificial neural networks inspired by the structure and function of the human brain. Unlike traditional machine learning algorithms, deep learning methods are nonlinear, complex, and hierarchical, allowing them to learn effectively from vast amounts of data and produce highly accurate results. Typical deep learning applications include language translation, image recognition, and personalized medicine [14].

2.5 Related Work

Keystroke dynamics, a behavioral biometric modality, has increasingly been explored for its potential to offer continuous user authentication. Its inherent advantage lies in its unobtrusiveness and compatibility with existing devices, eliminating the need for additional sensors. This makes it highly suitable for real-world applications that require seamless identity assurance.

The study by [5] utilized the CMU benchmark dataset [6] to perform static authentication using keystroke dynamics as a biometric factor. The researchers evaluated multiple machine learning models, including random forest, support vector machine (SVM), k-nearest neighbors (K-NN), and neural networks. Among these, the random forest model achieved the highest accuracy of 93.31 %. However, a key limitation of this research lies in its one-time user assessment during login, lacking continuous monitoring of user identity throughout a session. The current study addresses this limitation by adopting a continuous authentication approach, enabling ongoing verification during the user session.

The study by [15] provides a foundation for integrating AI with zero trust. It recognizes the need for dynamic identity verification and behavioral monitoring in achieving zero trust objectives. While the study discusses different AI techniques, it does not delve deeply into continuous authentication models, especially those based on behavioral biometrics. The implementation challenges of AI based user verification are broadly addressed without practical frameworks. This current thesis work tries to address this gap by practically implementing a dynamic continuous authentication framework using a real-world dataset, thereby contributing to the development of an adaptive, zero trust aligned identity verification system.

Another study [2] focused on keystroke dynamics in mobile environments, using the HMOG dataset [16] to assess continuous authentication. The researchers explored seven machine learning algorithms, including ensemble methods such as the Random Forest Classifier (RFC), the Extra Trees Classifier (ETC) and the Gradient Boosting Classifier (GBC). Although the use of behavioral biometrics on mobile devices is a relevant and practical choice, given the integral role of smartphones in daily life, reliance on a fixed text data set have certain limitations. Fixed inputs do not fully capture the variability of natural unstructured typing behavior that occurs in real-world scenarios. Furthermore, this mobile-specific method lacks scalability, especially in the modern context, where users frequently operate on multiple devices, including desktops, tablets, and smartphones. The current study overcomes these challenges by using a free-text keystroke dataset, which better represents natural typing patterns and offers device independence, making the proposed approach more adaptable to diverse environments and zero trust frameworks.

The study [17] has taken a practical step toward real-world implementation by focusing on free-text keystroke dynamics. Their work emphasizes that fixed-text input lacks the variability found in actual user behavior, making free-text a more suitable candidate for authentication in natural settings. Although the study proposes using advanced machine learning to manage this increased complexity, it acknowledges the need for further development to effectively capture and interpret variability in typing patterns. Building upon this foundation, the current study develops and tests models specifically trained to adapt to these diverse behaviors, enabling more effective continuous verification.

The research [18] demonstrated the potential of deep learning to continuously authenticate users in real time using behavioral data, achieving reasonable accuracy with low false acceptance and rejection rates. In contrast to this, the present study has tried to integrate zero trust with continuous authentication and also worked on different machine learning models rather than solely focusing on one model.

similarly, [8] introduced an agent-based modeling approach to simulate free-text typing behavior for continuous authentication. Their research contributed valuable insights into behavioral variability and user simulation but did not implement a real-time working system nor did it directly assess machine learning model performance across multiple users with real-world typing data.In the future research section of that study, the authors emphasized the need for deploying personalized machine learning models for each user, a recommendation that has been implemented in the current work.

The survey paper [4] offers a broad overview of the continuous authentication landscape, including keystroke dynamics among other behavioral biometrics. They stress the importance of balancing usability with security, promoting non-intrusive methods that operate in the background without disrupting the user experience. However, the study primarily remains conceptual and lacks detailed empirical evaluation or model implementation. The present research addresses this by not only proposing a continuous keystroke-based solution but also validating it through experimental results using free text input data.

Lastly, the study [1] discuss how AI can enhance zero trust architecture by enabling dynamic identity verification, real-time behavioral monitoring, and adaptive access control. While the paper successfully articulates the synergy between AI and zero trust principles, it does not delve into practical implementations in specific to behavioral biometrics. The current study fills this void by leveraging AI-driven continuous authentication based on keystroke dynamics, directly applying these concepts to a security model that demands constant verification and real-time risk mitigation.

2.6 Research summary

Continuous authentication has emerged as a promising approach to user verification, particularly within the context of ZTA. While keystroke dynamics has been widely studied, most existing research focuses on static authentication — typically validating users only at the point of login. Although some studies have explored continuous authentication, they often rely on fixed-text input, which does not accurately reflect natural, real-world typing behavior. While fixed-text methods may be effective for static authentication, they are less suitable for continuous monitoring. Furthermore, many of these studies do not explicitly align their methodologies with the core principles of zero trust, which emphasize continuous verification and real-time security enforcement.

This study addresses these limitations by employing keystroke dynamics not only for initial login but also for ongoing, real-time continuous authentication. Unlike fixed-text approaches, this research utilizes free-text input, allowing users to type naturally and thereby better simulating real-world usage scenarios. The dataset used includes input collected from multiple devices, enhancing the system's adaptability across platforms.

Additionally, the study adopts the ZTA model, which prioritizes continuous verification, minimal implicit trust, and real-time identity assurance — principles that closely align with the goals of keystroke dynamics - based continuous authentication. To evaluate the effectiveness and performance of the proposed approach, various machine learning and deep learning models are applied to distinguish between legitimate and unauthorized users, with the goal of improving authentication accuracy and supporting secure, real-time access control.

3

Methodology

3.1 Research Strategy

This study adopts an **Experiment** as the research strategy, which is well-suited for evaluating causal relationships and system performance under controlled conditions. As defined by Denscombe [19], an experiment is an empirical investigation conducted in a controlled environment to examine the properties and interactions of specific variables.

In this context, the study evaluates various machine learning and deep learning models using a dataset of user keystroke dynamics. The independent variables are the keystroke timing features—such as dwell time and flight time—while the dependent variable is the classification outcome, indicating whether a user is authorised or unauthorised.

The experimental setup enables repeatable testing across different models using the same dataset, ensuring consistency and comparability. Although the system is deployed in a semi-controlled, realistic environment, it maintains the integrity of experimental design, thereby enhancing the validity of the findings.

Alternative Research Strategy: Case Study

An alternative research strategy considered for this study was the **case study approach**, which involves an in-depth examination of a single instance, such as an authentication system deployed in a specific bank or online examination platform. Case studies are particularly effective for exploring complex phenomena within their real-world context and are well-suited for naturalistic inquiry [19]. However, this approach was deemed unsuitable for the current study for several reasons:

- Emphasize in-depth rather than breadth, which limits the ability to generalize results.
- Do not facilitate repeated manipulation and comparative testing on different models.
- Need to be deployed in real-time, which would be logistically and ethically complicated in the context of this project.

Given these limitations, the experimental strategy was selected instead, as it enables systematic benchmarking of models under reproducible and controlled conditions.

3.2 Data Collection Method

This study employs a secondary data collection method, utilizing an openly available free-text keystroke dynamics dataset [20]. As highlighted by Denscombe [19], secondary data collection method offers several advantages, particularly when the dataset is reputable and aligns with the research objectives. In this work, the secondary data collection method was chosen for the following reasons:

- **Comprehensiveness and quality:** The dataset is well-labeled and extensive, enabling effective training and evaluation of machine learning models.
- **Ethical simplicity:** It avoids the complexities of real-time data logging and mitigates privacy concerns.
- **Reproducibility:** Using a publicly available dataset supports validation and replication of findings by other researchers.

Alternative Method: Primary Data Collection via Logging Tools

An alternative method approach would involve collecting primary data through python scripting or keystroke logging tools to capture real-time free-text input from users. While this method offers greater control over data variables and context, it was not selected for the following reasons:

- **Ethical and privacy concerns:** Real-time data collection requires extensive ethical approvals and user consent.
- **Limited scalability:** Gathering a sufficiently large and diverse sample within the available timeframe would be challenging.
- **Environmental variability:** Differences in user hardware and typing conditions could introduce inconsistencies in the data.

Therefore, secondary data collection was selected for its practicality, ethical simplicity, and direct relevance to the current study objectives.

3.2.1 Dataset

As outlined in Section 3.2, the KeyRecs keystroke dynamics dataset from Zenodo [20] was selected as the most appropriate data source for this study. The dataset offers a comprehensive set of features, including demographic information, and comprises both fixed-text and free-text components. It includes data from over 100 participants across various countries and was collected via an online platform, further enriched by the use of diverse devices as it resembles the realistic setting. Each participant contributed multiple sessions of keystroke events, which include timing features such as dwell time and flight time. The dataset is ethically sourced, anonymized, and pre-processed for research use. For the purposes of this study, the free-text keystroke data was prioritized, as it provides a more realistic basis for developing and evaluating continuous authentication (CA) systems. Refer to Figure 3.2.

3.2.2 Dataset Overview

Fixed-text component: The user is asked to type a specific, predefined text. In this dataset, participants typed the password "vpwjkeurkb" 200 times across two sessions. This repetition under controlled conditions helped in capturing consistent typing patterns and allowed for the analysis of behavioral stabilization.

Free-text component: The user types any arbitrary text. In this dataset, participants transcribed 10 simple literature passages across two sessions. While the text was predefined, it closely mimics natural typing behavior, making it well-suited for modeling continuous user authentication.

3.2.3 Keystroke Timing Metrics

Dwell Time: It is a time interval between pressing and releasing of same key. For example, if the "A" key is pressed at 2.000 seconds and released at 2.150 seconds, the dwell time is 0.150 seconds. Dwell time reflects individual typing characteristics and is a key feature in identity verification.
Flight Time: The time interval between two consecutive keystrokes can be categorized into four types: Down-to-Down (DD), Up-to-Up (UU), Up-to-Down (UD), and Down-to-Up (DU). These include measurements such as Key-Up to Key-Down, which captures the time between releasing one key and pressing the next, and Key-Down to Key-Down, which measures the time between pressing two successive keys. For example, if the "A" key is released at 2.150 seconds and the "B" key is pressed at 2.300 seconds, the resulting flight time is 0.150 seconds. This flight time reflects the rhythm and fluidity of a user's typing behavior, offering an additional layer of biometric distinctiveness. Figure 3.1 illustrates these timing metrics, with a detailed visualization of both flight time and dwell time.

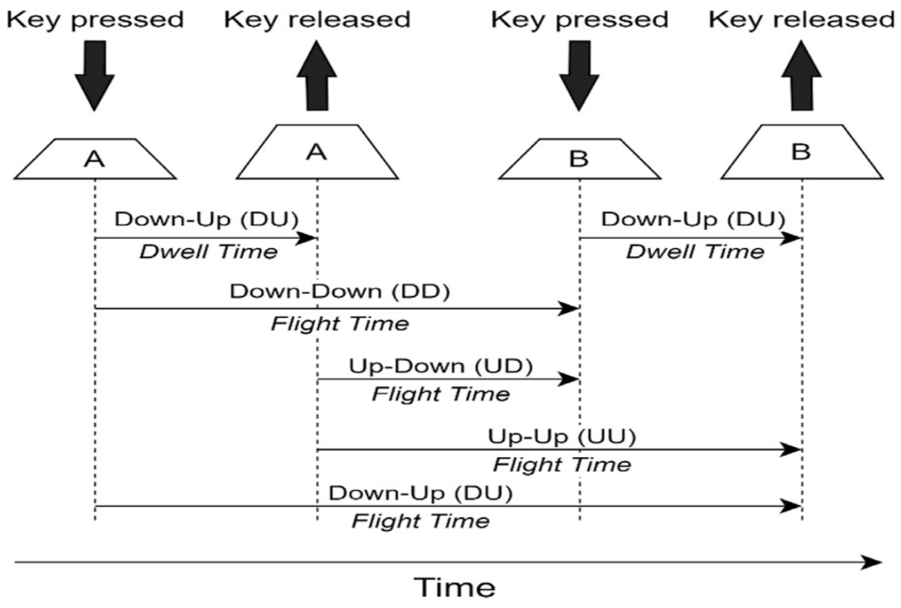
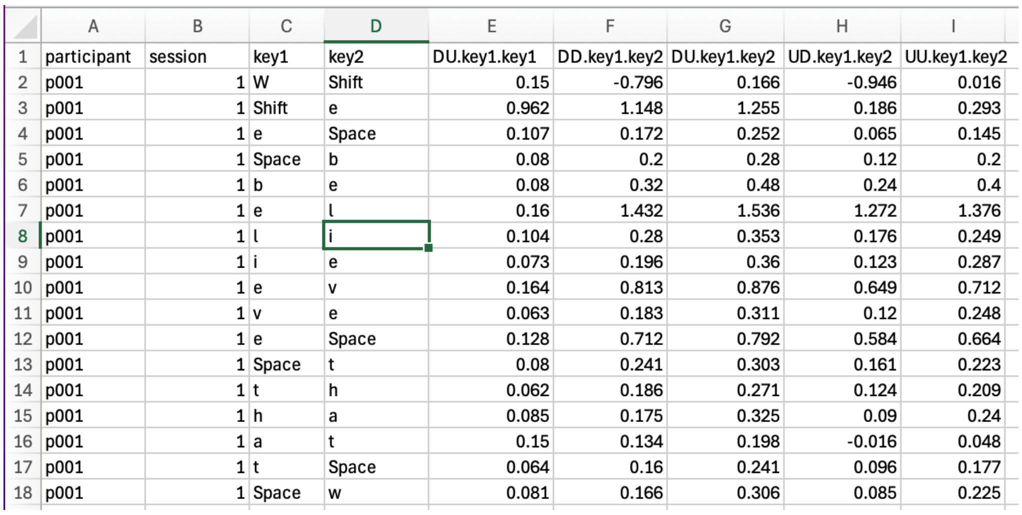


Figure 3.1: Time Based Features of Keystroke Dynamics

The free-text dataset include columns such as ‘participant‘, ‘session‘, ‘key1‘, ‘key2‘, ‘DU.key1.key1‘, ‘DD.key1.key2‘, ‘DU.key1.key2‘, ‘UD.key1.key2‘, and ‘UU.key1.key2‘. These represent various timing features essential for behavioral biometrics. Figure 3.2 presents the sample columns in the selected KeyRecs dataset [20].



	A	B	C	D	E	F	G	H	I
1	participant	session	key1	key2	DU.key1.key1	DD.key1.key2	DU.key1.key2	UD.key1.key2	UU.key1.key2
2	p001	1	W	Shift	0.15	-0.796	0.166	-0.946	0.016
3	p001	1	Shift	e	0.962	1.148	1.255	0.186	0.293
4	p001	1	e	Space	0.107	0.172	0.252	0.065	0.145
5	p001	1	Space	b	0.08	0.2	0.28	0.12	0.2
6	p001	1	b	e	0.08	0.32	0.48	0.24	0.4
7	p001	1	e	l	0.16	1.432	1.536	1.272	1.376
8	p001	1	l	i	0.104	0.28	0.353	0.176	0.249
9	p001	1	i	e	0.073	0.196	0.36	0.123	0.287
10	p001	1	e	v	0.164	0.813	0.876	0.649	0.712
11	p001	1	v	e	0.063	0.183	0.311	0.12	0.248
12	p001	1	e	Space	0.128	0.712	0.792	0.584	0.664
13	p001	1	Space	t	0.08	0.241	0.303	0.161	0.223
14	p001	1	t	h	0.062	0.186	0.271	0.124	0.209
15	p001	1	h	a	0.085	0.175	0.325	0.09	0.24
16	p001	1	a	t	0.15	0.134	0.198	-0.016	0.048
17	p001	1	t	Space	0.064	0.16	0.241	0.096	0.177
18	p001	1	Space	w	0.081	0.166	0.306	0.085	0.225

Figure 3.2: Columns in a Dataset

3.3 Data Analysis Method

The collected secondary data was analyzed using a quantitative approach, with machine learning models being trained and evaluated on a keystroke typing pattern of users in the dataset. The study by [19] identifies quantitative analysis as suitable method for research involving numeric data and classification phenomenon. Evaluation metrics such as accuracy, precision, and F1-score were used to assess the model performance. This quantitative analysis method aligns perfectly with the nature of the research question, which helps to determine the most effective model for continuous authentication.

Alternative Method: Comparative Case Analysis

An alternative data analysis method could be a comparative case analysis based on contextual factors like device type, user demographics, or usage scenarios. While this approach can provide rich insights, it is more qualitative and subjective, making it less appropriate for the performance benchmarking that this study prioritizes. Furthermore:

- It shifts the focus from technical model performance to contextual interpretation.
- It lacks the numerical rigor needed to draw reliable conclusions on model accuracy or robustness.

Therefore, quantitative analysis method was retained as the most suitable approach for the current work of analyzing keystroke dynamics.

3.4 Machine Learning Models and Key Variables for Keystroke-Based User Verification

The models selected for this comparative evaluation include traditional machine learning algorithms such as Random Forest (RF), Support Vector Machine (SVM), and Decision Tree (DT), as well as advanced deep learning models including Convolutional Neural Networks (CNN), Long Short-Term Memory (LSTM) networks, and a hybrid CNN-LSTM model.

These models were chosen because they are well-established for solving classification problems and offer complementary strengths when applied to behavioral biometrics like keystroke dynamics. Traditional models such as DT, RF, and SVM are reliable for interpreting structured statistical features and provide strong baselines with explainable outcomes. Meanwhile, deep learning models like CNN and CNN-LSTM are capable of automatically learning complex spatial and temporal patterns from raw sequences, making them well suited for real-time continuous authentication. Together, these models offer a comprehensive evaluation approach that balances accuracy, adaptability, and real-time performance in the context of user verification.

Decision Tree: Decision Tree models provide insights into feature importance and decision logic, making them ideal for analyzing which statistical keystroke features contribute most to authentication decisions. DT splits independent variables at decision nodes to predict the dependent variable. The key variables are:

- **max_depth**: Determines how many feature splits are allowed.
- **min_samples_split**: Avoids reacting to noise in the features.
- **criterion**: Chooses how to measure the quality of feature-based splits.

Random forest: The Random Forest model aggregates the results of multiple decision trees to improve classification robustness and reduce overfitting. In this model independent variables (keystroke features) are used in multiple decision tress and the final prediction (dependent variable) is based on majority voting. Also RF controls how deeply and repeatedly the model reads patterns from input keystroke features to decide if a user is genuine. The key variables are:

- **n_estimators**: More trees help generalize better from feature patterns.
- **max_depth**: Limits how deeply features are split, preventing overfitting.
- **min_samples_split / min_samples_leaf**: Controls how sensitive the model is to small differences in feature values.

Support Vector Machine(SVM): SVM is known for maximizing the margin between classes and can generalize well on unseen samples when data is limited or well-separated. SVM maps the feature space (independent variables) into higher dimension to find the best seperating hyperplane for classification. variables control the shape of decision boundary that separates genuine from impostor typing patterns. The key variables are:

- **kernel**: Changes how the features are interpreted or transformed.
- **C**: Controls tolerance for misclassification; higher values indicate stricter margins.
- **gamma**: Adjusts how much influence a single feature point has.

Convultional Neural Network(CNN): CNN applies filters (kernels) over the keystroke feature windows to learn spatial feature patterns. The key variables control how deeply and abstractly the model extracts relationships among input timing features. The key variables are:

- **filters**, **kernel_size**: Define what kind of patterns the model can detect.
- **dropout**: Controls overfitting by limiting reliance on specific features.
- **epochs**, **batch_size**: Affect how thoroughly the model learns from the features.

CNN-LSTM: This hybrid model enables the system to continuously monitor typing behavior, adaptively distinguishing between users even with minor behavioral shifts. Particularly useful in real-time systems, where temporal continuity and accuracy are crucial. It first extracts spatial patterns using CNN, then processes temporal structure with LSTM to model typing dynamics. This model blends feature-space precision with time-dependent behavior,and variables define how it does so. The key variables are:

- **conv_filters**, **lstm_units**: Determine how deeply the system captures spatial-temporal interactions.
- **sequence_length**: Controls the time scope of user behavior analysis.

One of the models discussed above will be selected as the best-performing based on a comparison of evaluation metrics after training each individual model on the chosen dataset. To ensure the reliability of the results, hyperparameter tuning was conducted over five iterations for each selected model. The key variables from the selected model will then be used in the Experiment.

3.5 Performance Evaluation Metrics

In line with the experimental strategy, this study aims to evaluate the performance of multiple machine learning and deep learning models by adjusting key input variables derived from keystroke dynamics. These features reflect the temporal structure of typing behavior and are critical in distinguishing authorized from unauthorized users. The basic evaluation metrics considered were accuracy, precision, F1-score, FAR, and FRR. These metrics provide deeper insights into the models' ability to correctly identify legitimate users while minimizing unauthorized access attempts. Below is the short description of each evaluation metric:

- **True Positive (TP)**: The model correctly predicts a positive class when the actual class is also positive.
- **True Negative (TN)**: The model correctly predicts a negative class when the actual class is negative.
- **False Positive (FP)**: The model incorrectly predicts a positive class when the actual class is negative.

- **False Negative (FN)**: The model incorrectly predicts a negative class when the actual class is positive.

- **Accuracy** is the overall correctness of the model.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision** measures the accuracy of positive predictions.

$$\text{Precision} = \frac{TP}{TP + FP}$$

- **F1 Score** is the harmonic mean of precision and recall.

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **False Acceptance Rate (FAR)** is the probability that an unauthorized user is incorrectly granted access to a system.

$$FAR = \frac{FP}{FP + TN}$$

- **False Rejection Rate (FRR)** is the probability that an authorized user is incorrectly denied access.

$$FRR = \frac{FN}{FN + TP}$$

- **FAR and FRR** are crucial metrics in evaluating authentication systems.

Accuracy provides a general measure of a model's correctness, it is not sufficient on its own for security-focused applications. Therefore, greater emphasis was placed on precision and recall. High precision reduces the likelihood of false acceptances (unauthorized access), while high recall ensures that legitimate users are not mistakenly denied access.

The F1-score, representing the harmonic mean of precision and recall, was also considered essential to evaluate the model's performance under varying conditions. In addition, False Acceptance Rate (FAR) and False Rejection Rate (FRR) were treated as critical indicators due to their direct impact on both security and user experience. A low FAR enhances the system's resistance to impersonation attempts, whereas a low FRR improves usability by minimizing disruptions for authorized users. For this reason, FAR and FRR are regarded as key metrics in the evaluation of authentication systems.

The model that achieved the best balance between FAR and FRR—while also maintaining strong accuracy and precision—was identified as the most suitable for real-time continuous authentication. Further discussions on model selection and performance demonstration are presented in Chapter 5 and Chapter 6.

3.6 Research Strategy Implementation

The core concept of the Continuous Authentication (CA) system is to capture each user's unique typing behavior during an initial enrollment phase. This behavioral profile is then used for ongoing verification after the initial static authentication. If the user's typing deviates significantly from the stored profile, the system flags it as a potential threat and terminates the session, aligning with zero trust security principles.

The research strategy adopted in this study is experimental in nature, aimed at systematically evaluating the performance of various machine learning models for continuous user authentication based on keystroke dynamics. This approach involves controlled experimentation, where input variables are manipulated to observe their effect on model performance.

The development and experimentation of machine learning and deep learning models were carried out using **Python**, with key libraries including **Pandas**, **NumPy**, **Scikit-learn**, **TensorFlow**, and **Keras**. All model development and testing were performed in the Visual Studio Code environment

In this context, the independent variables consist of keystroke timing features—such as dwell time, flight time, and digraph latency—extracted from the KeyRecs dataset. The dependent variable is the classification outcome, indicating whether a user is authenticated or not.

To ensure the robustness and reliability of the results, hyperparameter tuning was performed for each selected model. This process involved adjusting key parameters over five iterations to identify the optimal configuration for each algorithm. Models evaluated include traditional classifiers such as Decision Tree (DT), Random Forest (RF), and Support Vector Machine (SVM), as well as deep learning architectures like Convolutional Neural Networks (CNN) and a hybrid CNN-LSTM model.

The experimental strategy enabled a structured comparison of model performance using evaluation metrics such as accuracy, precision, F1-score, False Acceptance Rate (FAR), and False Rejection Rate (FRR). This methodology ensured that the selected model not only achieved high predictive accuracy but also maintained a balance between security and usability, aligning with the principles of real-time continuous authentication.

Further analysis and results are detailed in Chapter 5 and discussed in Chapter 6.

4

Experimental Framework

This chapter provides a comprehensive overview of continuous authentication within the Zero Trust Framework (ZTF), detailing the authentication process, model implementation, experimental setup and evaluation.

4.1 Continuous Authentication in Zero Trust Framework

Figure 4.1 presents the components of continuous authentication (CA) in a zero trust framework (ZTF). The entire process of authentication is managed by the CA Service, which constantly evaluates the user's behavior while they are accessing the system. This starts with the real-time collection of user behavior data, specifically keystroke dynamic features.

This real-time keystroke data is then fed into the CA Engine, which processes the input using machine learning algorithms. The engine compares this new behavior with user behavioral profiles, which are essentially pre-trained machine learning models built from the user's past typing patterns during enrollment. These profiles act as a reference to distinguish between authorized and unauthorized users.

Based on this comparison, the system produces a real-time classification result that labels the user's behavior as either authorized or unauthorized. The user or device refers to the individual interacting with the system—whose typing behavior continuously feeds into the authentication process. With every keystroke, the system captures behavioral features, enabling it to verify identity on an ongoing basis. This continuous loop ensures that access is maintained only by the authorized user throughout the session.

To enforce ZTA, the system integrates two critical components - Policy Decision Point (PDP) and Policy Enforcement Point (PEP). PDP receives the classification outcome from the CA engine and evaluates this outcome based on contextual policies (time of access, location, and device state) and decide whether the session should continue or not. PEP is a mechanism that acts on PDP decision. If PDP classifies behaviour as unauthorized, the PEP may lock the screen, terminate the session or reauthenticate the user. If the user is classified as authorized, the session continues uninterrupted.

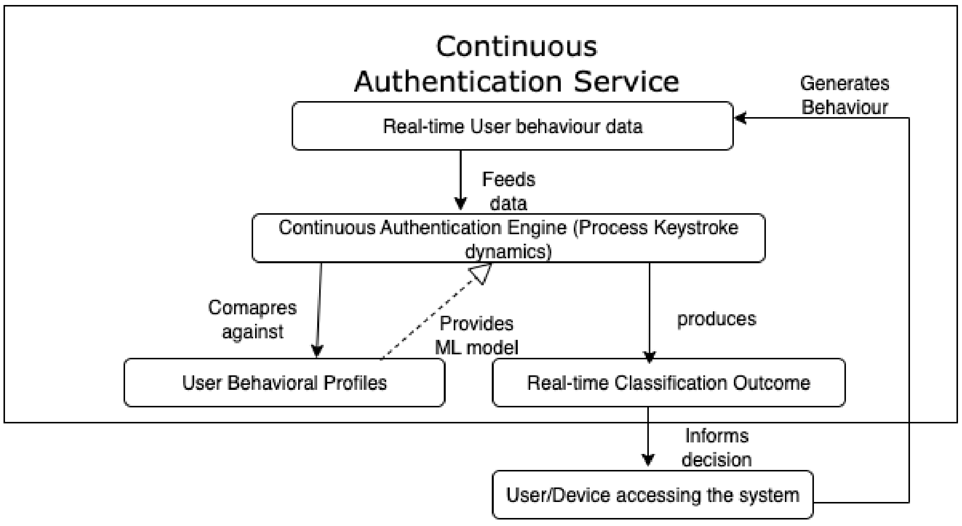

Continuous Authentication Service
Generates Behaviour
Real-time User behaviour data
Feeds data
Continuous Authentication Engine (Process Keystroke dynamics)
Comapres against
Provides ML model
produces
User Behavioral Profiles
Real-time Classification Outcome
Informs decision
User/Device accessing the system

Figure 4.1: Components of Continuous Authentication

4.1.1 User Enrollment and Continuous Authentication Process

This study proposes a two-phase model for enhancing user authentication security through AI-powered behavioral biometrics. The system is structured into two key phases: 1) User enrollment phase and 2) CA phase.

User Enrollment Phase:

As shown in Figure 4.2 during user enrollment proce, a user-specific behavioral profile is established. This begins with the user typing multiple sentences. The system captures and processes keystroke dynamics, particularly focusing on dwell time and flight time.

These features are essential in characterizing an individual's unique typing rhythm. Once data is collected, multiple machine learning models are trained on the extracted feature set to classify and recognize user-specific patterns. This trained model is then securely stored and used for subsequent real-time authentication. If a new user joins the organization, the whole process is repeated and machine learning model need to be trained again.

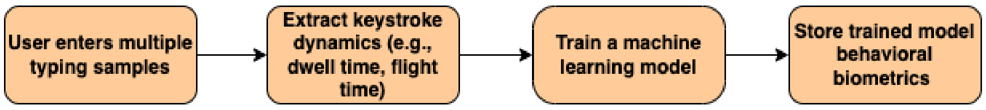

User enters multiple typing samples
Extract keystroke dynamics (e.g., dwell time, flight time)
Train a machine learning model
Store trained model behavioral biometrics

Figure 4.2: User Enrollment Process

Continuous Authentication Phase:

After successful enrollment, the operational phase begins, where CA is integrated into the user's login process. The user initiates access to the application using standard credentials, typically including a username and password. Depending on the organization's security policies, additional authentication layers, such as Multi-Factor Authentication (MFA) may be employed.

Upon successful static authentication, the system activates real-time keystroke monitoring. The AI-based authentication engine continuously compares the user's current typing behavior against the previously stored behavioral model. A mean similarity score is computed from the collected keystroke features that determines the confidence.

- If the confidence score exceeds a defined threshold, the behavior is deemed consistent with the enrolled profile, and the session continues.

- If the confidence score falls below the threshold, indicating a deviation from the usual pattern, the system classifies the activity as anomalous and blocks the session to prevent potential misuse.

This proposed working model depicted in Figure 4.3 can significantly reduce the risk of unauthorized access, even after the intial login and ensures adherence to the ZTA principle.

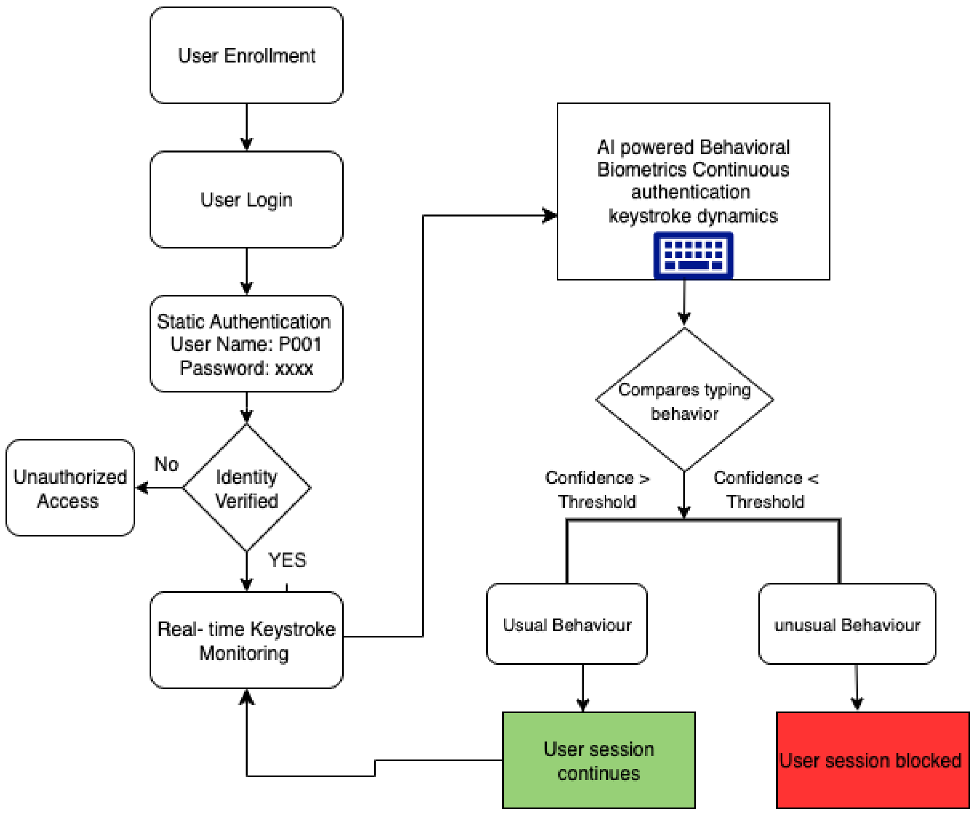


Figure 4.3: Working Model of Continuous Authentication

4.2 Implementation of Machine Learning - based Keystroke Dynamics for Continuous Authentication

For classifying users (authorized vs unauthorized), machine learning (ML) models were chosen as the core intelligence layer for identifying behavioral profiles specifically, typing patterns. From a system requirements perspective, the solution should:

- Continuously verify the legitimacy of users throughout their active session.
- Detect subtle changes in behavior that might indicate impersonation.
- Adapt to individual typing styles to reduce false rejections.
- Low false acceptance and rejection rates.

To meet these needs, training ML models became essential. Unlike traditional rule-based systems that rely on fixed logic, ML models have the ability to learn complex relationships from raw behavioral data and make dynamic predictions. They are capable of detecting nuanced typing behaviors (such as dwell time and flight time) that are unique to each user. The advanced deep learning approach implemented a per-user authentication system, which encapsulates the CNN-LSTM architecture for each individual user. This system trained and saved a unique CNN-LSTM model for each participant, enabling highly personalized authentication.

After the collection of keystroke data the model has undergone development and training. Figure 4.4 shows the major aspects in the development of machine learning model.

- **Data Collection & Preparation:** Raw keystroke data is gathered from multiple users across different sessions, capturing features such as dwell time and flight time. This data is then cleaned and organized to ensure consistency and completeness. Refer to Section 3.2 of Chapter 3.
- **Feature Engineering & Preprocessing:** Involves extracting statistical features (e.g., mean, standard deviation, percentiles) from sliding windows of keystroke sequences. The features are normalized using scaling techniques and any missing values are appropriately handled. Refer to Section 4.2.1.
- **Model Selection & Configuration:** Suitable models for example RF for traditional machine learning or CNN-LSTM for deep learning are selected based on the experimental setup. The architecture and hyperparameters are defined according to the expected behavior of user typing patterns. Refer to Section 3.4 to read more about ML models.
- **Model Training & Validation:** Includes training the model using authorized and unauthorized data, validating it on unseen samples, and applying techniques such as early stopping and cross-validation to avoid overfitting.
- **Model Evaluation & Deployment:** The trained model is assessed using metrics like accuracy, precision, and F1-score with special attention to false acceptance and rejection rates. Once validated, the model is deployed into a real-time system where it monitors live typing behavior for continuous user authentication. Refer to Chapter 5.

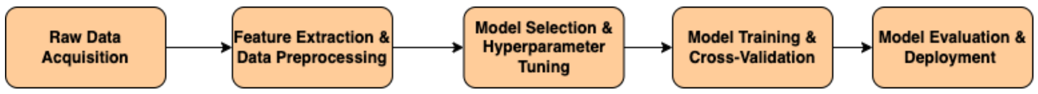

Raw Data Acquisition
Feature Extraction & Data Preprocessing
Model Selection & Hyperparameter Tuning
Model Training & Cross-Validation
Model Evaluation & Deployment

Figure 4.4: Process of Training a Machine Learning Model

As this study utilizes a secondary dataset, no primary data collection or preparation was conducted. Instead, a brief overview of the selected dataset is provided in Section 3.2 of Chapter 3. The models discussed in Section 3.4 are used for training.

4.2.1 Data Preprocessing

To ensure consistent and reliable input for both traditional machine learning and deep learning algorithms, a series of structured data preprocessing steps were applied to the raw keystroke dynamics data. Initially, the dataset containing keystroke timing features such as key hold time (DU.key1.key1) and various flight times (DD.key1.key2, DU.key1.key2, UD.key1.key2, UU.key1.key2)—was loaded from CSV files. All relevant features were explicitly converted to numeric values to address inconsistent or non-numeric entries. Missing or invalid values were either dropped or replaced using column-wise mean substitution, depending on the model requirements.

Following cleaning, features were extracted, and the data was segmented into required sliding window size to capture temporal behavior. A sliding window approach was used to segment the continuous typing stream into overlapping windows of fixed size. For each window, statistical descriptors—such as mean, standard deviation, median, and various percentiles—were computed to summarize user behavior over short intervals. This process generated compact feature representations that are well-suited for classification tasks. For deep learning models like CNN and CNN-LSTM, the raw window sequences were preserved to maintain the sequential nature of the input. All data was standardized using z-score normalization (Standard Scaler), ensuring a mean of zero and unit variance across features, which is critical for efficient training of both traditional classifiers and neural networks. Finally, the dataset was split into training and testing subsets using 80/20 ratio, allowing fair and balanced model evaluation across all implementations.

4.2.2 Hyperparameter Tuning

The development and experimentation of machine learning and deep learning models were carried out using python, with key libraries including pandas, NumPy, scikit-learn, TensorFlow, and Keras. All model development and testing were performed in the Visual Studio Code environment

To evaluate the effectiveness of various traditional and deep learning models for continuous user authentication using free-text keystroke dynamics, a series of controlled experiments were conducted. Each model was tested under five different configurations by varying critical hyperparameters along with the sliding window size used during preprocessing. This ensured a balanced exploration of both temporal resolution (through window size) and model complexity. Each model experiment setup has been clearly depicted in Appendix A

For the DT experiments (DT-Exp1 to DT-Exp5), the **max_depth** values were tuned from shallow to deeper trees (depths of 6 to 12), combined with both gini and entropy criteria. This allowed assessment of overfitting vs. generalization capacity in varying input windows.

For the RF classifier, five experiments (RF-Exp1 to RF-Exp5) were executed by varying the number of estimators (100–200), **max_depth** (None, 15, 10, 8), and **min_samples_split (2–5)** across window sizes ranging from 10 to 30. Similarly, the SVM model (SVM-Exp1 to SVM-Exp5) was tested using multiple kernel functions (rbf, linear, poly, sigmoid) and regularization parameters (C = 0.5 to 10.0), with corresponding adjustments in the gamma parameter.

For deep learning experiment, the CNN model (CNN-Exp1 to CNN-Exp5) was tested using combinations of 1 to 3 convolutional layers with filter sizes between 16 and 64, kernel sizes of (3, 3) and (5, 5), and dropout rates ranging from 0.1 to 0.5 to prevent overfitting. For sequence modeling, the CNN-LSTM hybrid model (CNNLSTM-Exp1 to CNNLSTM-Exp5) was evaluated by varying convolution filter size (16–64), LSTM memory units (32–128), dropout (0.1–0.5), and training epochs (10–25) across multiple window lengths.

By running these configurations systematically, the study aimed to uncover optimal design choices for each model class while ensuring consistency in feature engineering and evaluation metrics across all experiments.

4.2.3 Model Training

For RF, DT, and SVM the training begins with fitting the model to a labeled dataset where features (such as keystroke timing metrics) are mapped to class labels (e.g., authorized or unauthorized). These models are trained using traditional machine learning algorithms, where the tuned hyperparameters like the number of trees in RF, the maximum depth in DT, or the kernel type in SVM are fixed. Cross-validation (typically k-fold, e.g., 5-fold or 10-fold) is used to split the data into k subsets. The model is trained on k-1 subsets and validated on the remaining one. This process is repeated k times so that each subset is used once as a validation set.

For CNN and CNN-LSTM, which are deep learning models, the process is more iterative and computationally intensive. Training involves feeding batches of feature sequences (like sliding windows of keystroke timing vectors) into the network and using backpropagation with optimizers (e.g., Adam) to minimize a loss function (like binary cross-entropy). CNN captures spatial patterns in timing features, while CNN-LSTM captures both spatial and temporal dependencies. Cross-validation is usually done using either k-fold or stratified sampling, but due to training cost, sometimes a train-validation split is preferred instead. During training, techniques such as early stopping, dropout, and learning rate scheduling are used to prevent overfitting and ensure stable convergence.

4.2.4 Evaluation

In this section, focus is on evaluating how well various machine learning and deep learning models perform when applied to the task of continuous keystroke based user authentication. We use standard evaluation metrics such as Accuracy, Precision, F1-Score, False Acceptance Rate (FAR), and False Rejection Rate (FRR) to measure how effectively each model distinguishes between genuine users and impostors. These metrics help us understand not just how often the models get it right, but also how reliable and usable they are in a real-time authentication system.The best tradeoff model is choosen and deployed into the real-time application.

5

Results

5.1 Experimental Results

This chapter presents the outcomes of the experimental evaluation, highlighting the performance of various machine learning and deep learning models applied to continuous keystroke-based authentication. The results are analyzed using key metrics such as accuracy, precision, F1-score, False Acceptance Rate (FAR), and False Rejection Rate (FRR) to determine the most effective model configuration.

5.1.1 Decision Tree Experiment Results

Table 5.1 presents the experimental results of five Decision Tree configurations evaluated for continuous keystroke authentication, highlighting the relationship between key hyperparameters and model performance. The experiments varied in window size, maximum tree depth, minimum samples required to split a node, and the splitting criterion (Gini or Entropy). Among the tested configurations, **DT-Exp4** with a window size of 25, a maximum depth of 10, minimum sample split of 5, and using the entropy criterion achieved the best overall balance, recording an accuracy of 68.62 %, precision of 68.05 %, and the highest F1-score of 0.685. Additionally, DT-Exp4 maintained a favorable FAR of 32.97 % and a relatively low FRR of 29.79 %, making it well-suited for minimizing both unauthorized access and user inconvenience in CA. Comparatively, DT-Exp2 also performed well, with a slightly lower F1-score but a better FAR. DT-Exp1 and DT-Exp5 showed lower performance across all metrics, indicating that smaller window sizes and shallower trees may not capture enough behavioral information.

Experiment ID	**Accuracy**	**Precision**	**F1-Score**	**FAR**	**FRR**
DT-Exp1	0.6333	0.6242	0.635	0.4034	0.3297
DT-Exp2	0.6741	0.6760	0.675	0.3203	0.3313
DT-Exp3	0.6559	0.6565	0.660	0.3422	0.3457
DT-Exp4	0.6862	0.6805	0.685	0.3297	0.2979
DT-Exp5	0.6460	0.6320	0.645	0.4068	0.3014

Table 5.1: Decision Tree Experimental Results

5.1.2 Random Forest Experiment Results

The Random Forest experiments summarized in Table 5.2 investigate the impact of varying key hyperparameters, including window size, number of estimators, maximum tree depth, and minimum samples required to split—on the performance of continuous keystroke authentication. Among the five configurations, RF-Exp1 and RF-Exp5 demonstrate the highest performance, each achieving an accuracy of approximately 74.3 % and an identical F1-score of 0.745. Notably, **RF-Exp1** slightly outperforms RF-Exp5 in terms of both lower False Acceptance Rate (FAR = 24.6 %) and a marginally better False Rejection Rate (FRR = 26.7 %), making it the most reliable and balanced model for real-time deployment. RF-Exp1's setup—using a window size of 10, 100 estimators, no restriction on tree depth, and a minimal split size of 2—appears well-suited for capturing fine-grained behavioral patterns while maintaining model generalizability.

Experiment ID	**Accuracy**	**Precision**	**F1-Score**	**FAR**	**FRR**
RF-Exp1	0.7433	0.7466	0.745	0.246	0.267
RF-Exp2	0.7342	0.7375	0.733	0.255	0.276
RF-Exp3	0.7160	0.7181	0.715	0.275	0.293
RF-Exp4	0.7033	0.7009	0.700	0.298	0.295
RF-Exp5	0.7442	0.7454	0.745	0.250	0.262

Table 5.2: Random Forest Experimental Results

5.1.3 SVM Experiment Results

Table 5.3 summarizes the experimental results of five Support Vector Machine configurations applied to continuous keystroke authentication, with variations in window size, kernel type, regularization parameter C, and kernel coefficient gamma. Among these, **SVM-Exp2**, which utilized a radial basis function (RBF) kernel with a window size of 20, C=0.5, and gamma set to "auto", delivered the most balanced and reliable performance. It achieved the highest accuracy (66.36 %), a strong F1-score (0.68), and a reasonable balance between False Acceptance Rate (FAR = 38.64 %) and False Rejection Rate (FRR = 28.71 %). Although SVM-Exp3 reported a similar F1-score, it suffered from an unacceptably high FAR (84.20 %), making it unsuitable for real-world deployment. SVM-Exp2's relatively low FRR also suggests greater consistency in authenticating genuine users, which is crucial for usability in a CA system. These results indicate that careful tuning of the RBF kernel and regularization parameters can significantly enhance classification reliability, making SVM-Exp2 the most appropriate configuration for comparison and consideration in biometric-based CA.

Experiment ID	Accuracy	Precision	F1-Score	FAR	FRR
SVM-Exp1	0.5716	0.6236	0.47	0.2317	0.6216
SVM-Exp2	0.6636	0.6467	0.68	0.3864	0.2871
SVM-Exp3	0.5496	0.5250	0.68	0.8420	0.0495
SVM-Exp4	0.5684	0.5470	0.66	0.6843	0.1806
SVM-Exp5	0.4680	0.4662	0.47	0.5381	0.5257

Table 5.3: Support Vector Machine Experimental Results

5.1.4 CNN Experiment Results

The Convolutional Neural Network (CNN) experiments conducted for continuous keystroke authentication, detailed in Table 5.4, illustrate how architectural choices such as window size, number of convolutional layers, filter size, and dropout rates influence model performance. Among the five configurations, **CNN-Exp3** stands out as the most effective, achieving the highest accuracy (68.35 %) and F1-score (0.68), alongside a strong precision score of 69 %. With a window size of 30, two convolutional layers using 32 filters and a (3,3) kernel size, and a dropout of 0.2, CNN-Exp3 strikes an optimal balance between learning capacity and regularization. Its relatively moderate FAR (38.36 %) and the lowest FRR (24.88 %) among all configurations suggest that it effectively minimizes false rejections of genuine users—an essential feature for CA systems that must operate in real time without disrupting user experience. Although CNN-Exp5 exhibited a lower FAR, its substantially higher FRR compromises usability.

Experiment ID	Accuracy	Precision	F1-Score	FAR	FRR
CNN-Exp1	0.6419	0.6450	0.64	0.2962	0.4189
CNN-Exp2	0.6702	0.6700	0.67	0.3500	0.3099
CNN-Exp3	0.6835	0.6900	0.68	0.3836	0.2488
CNN-Exp4	0.6559	0.6550	0.66	0.3064	0.3802
CNN-Exp5	0.6479	0.6600	0.64	0.2143	0.4881

Table 5.4: CNN Experimental Results

5.1.5 CNN-LSTM Experiment Results

Table 5.5 presents the performance outcomes of five CNN-LSTM experiments applied to continuous keystroke authentication, with each configuration varying in terms of window size, convolutional filters, LSTM units, dropout rate, and training epochs. Among these, **CNNLSTM-Exp3** emerges as the most effective and well-balanced model, achieving the highest accuracy (98.92 %), precision (99.39 %), and F1-score (0.9891), while maintaining the lowest False Acceptance Rate (FAR = 0.0060) and a very low False Rejection Rate (FRR = 0.0157). This setup—using a window size of 30, 32 convolutional filters, 64 LSTM units, a modest dropout of 0.2, and 20 training epochs—provides an optimal balance between model complexity and generalization. Although CNNLSTM-Exp2 achieved slightly comparable performance, CNNLSTM-Exp3 offered marginally better accuracy and error control, making it the most suitable candidate for real-time CA.

Experiment ID	Accuracy	Precision	F1-Score	FAR	FRR
CNNLSTM-Exp1	0.9655	0.9767	0.9651	0.0227	0.0462
CNNLSTM-Exp2	0.9889	0.9913	0.9888	0.0087	0.0136
CNNLSTM-Exp3	0.9892	0.9939	0.9891	0.0060	0.0157
CNNLSTM-Exp4	0.9846	0.9910	0.9845	0.0089	0.0219
CNNLSTM-Exp5	0.9022	0.9520	0.8964	0.0426	0.1532

Table 5.5: CNN-LSTM Model Experimental Results

5.2 Comparitive Analysis

5.2.1 Accuracy

When it comes to accuracy, the ability of a model to correctly identify both authorized and unauthorized users, the CNN-LSTM model clearly outperforms all others. With near-perfect accuracy, it demonstrates a remarkable ability to learn and recognize the subtle nuances in keystroke dynamics. Among the traditional machine learning models, Random Forest achieves the highest accuracy, showing it's quite capable in distinguishing users based on typing behavior. Decision Tree and SVM follow closely, but their performance slightly dips, especially when faced with more complex or variable typing styles. On the deep learning side, CNN does reasonably well but doesn't quite reach the level of CNN-LSTM, reinforcing the importance of combining spatial and temporal modeling for behavioral biometrics.

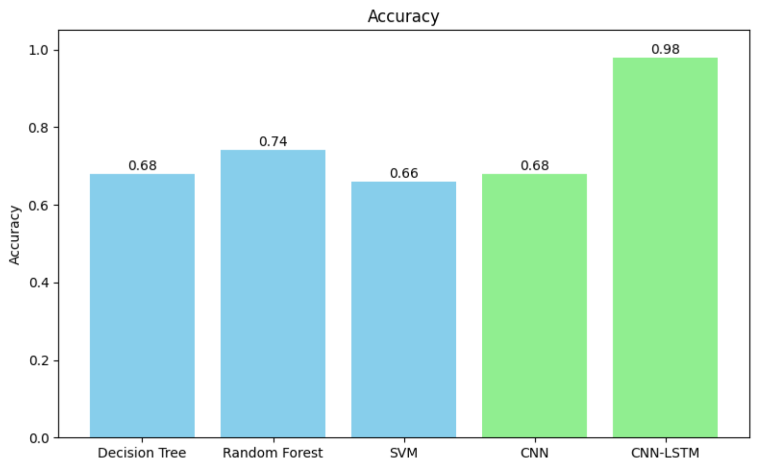


Figure 5.1: Accuracy comparision across models

5.2.2 Precision

Precision reflects how well the model avoids false positives—in this case, wrongly accepting unauthorized users as authorized users. Again, CNN-LSTM takes the lead with exceptional precision, meaning it rarely makes that critical error. This is important in real-world applications where unauthorized access must be tightly controlled. Random Forest also scores well on precision among machine learning models, reinforcing its robustness. On the other hand, SVM and Decision Tree trail slightly, showing they are more prone to confusion between legitimate and illegitimate behavior. The

CNN model outperforms SVM but is still less reliable than CNN-LSTM, indicating that capturing time dependencies (as LSTM does) adds significant value.

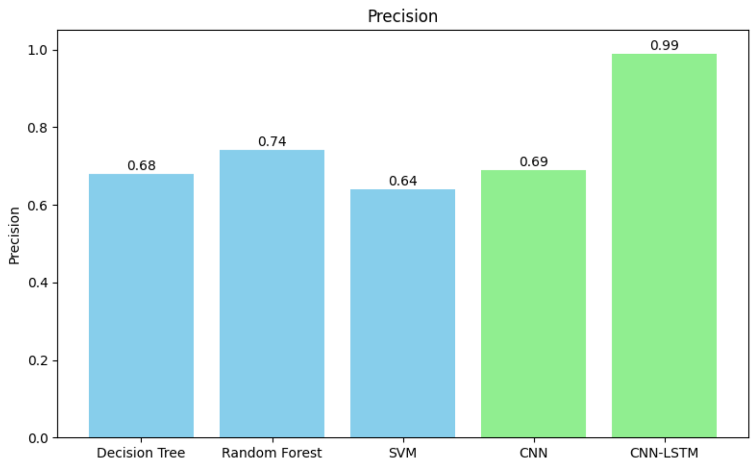


Figure 5.2: Precision comparision across models

5.2.3 F1-Score

The F1-Score provides a balance between precision and recall — essentially measuring how well the model handles both correct detections and avoiding errors. CNN-LSTM once again achieves the highest F1-Score, proving it's the most well-rounded model for continuous user authentication.

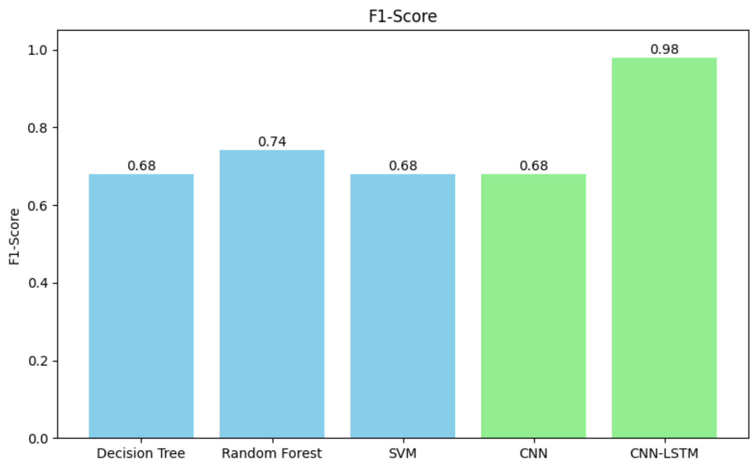


Figure 5.3: F1-Score comparision across models

Its ability to accurately detect users while minimizing false alarms or missed detections makes it especially suited for sensitive environments. Random Forest performs best among the traditional methods, maintaining a strong balance. DT, SVM, and even CNN show moderate F1-Scores, suggesting limitations in generalizing across variable user behaviors. The consistency of CNN-LSTM here is what sets it apart.

5.2.4 False Acceptance Rate

False Acceptance Rate (FAR) is a critical metric in security-sensitive applications—it tells us how often the system mistakenly grants access to an unauthorized user. In this category, CNN-LSTM excels, achieving an extremely low FAR $(0.006 \simeq 0.01)$. This means it's very effective in preventing unauthorized users from slipping through. Among machine learning models, Random Forest performs admirably with the lowest FAR of the group, making it the most trustworthy ML-based solution. However, SVM and CNN show noticeably higher FARs, suggesting they're more likely to allow an unauthorized user access. This highlights the CNN-LSTM model's superior security capabilities.

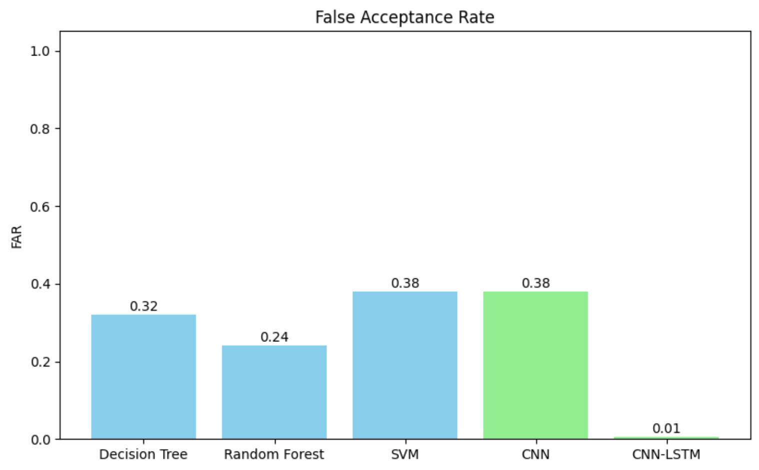


Figure 5.4: False Acceptance Rate across models

5.2.5 False Rejection Rate

False Rejection Rate (FRR) measures how often authorized users are mistakenly denied access. This is important from a usability perspective users shouldn't be repeatedly blocked from accessing their own accounts. CNN-LSTM once again leads with an incredibly low FRR ($0.0157 \simeq 0.02$), ensuring that real users are almost always allowed in without hassle. Random Forest shows the lowest FRR among traditional models, indicating that it handles behavioral variations reasonably well. DT, SVM, and CNN exhibit higher FRRs, meaning they may frustrate users by denying legitimate access more frequently. The balance achieved by CNN-LSTM between security (low FAR) and convenience (low FRR) is what makes it particularly powerful in real-time, CA systems.

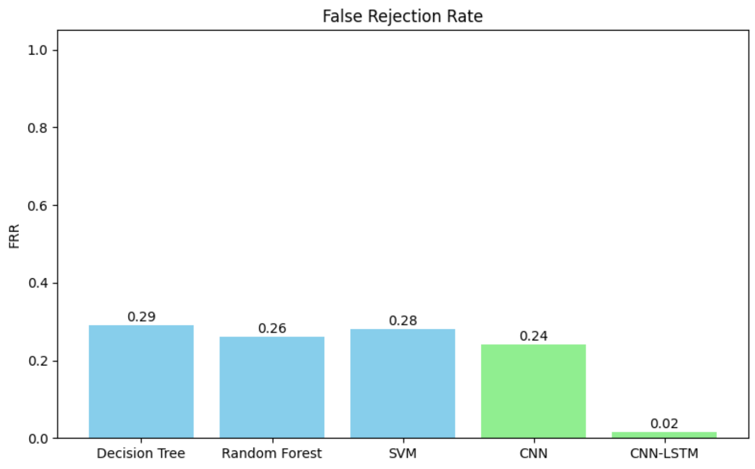


Figure 5.5: False Rjection Rate across models

5.3 Per User Training of CNN-LSTM

Following the analysis of various machine learning models, the CNN-LSTM architecture demonstrated the most promising accuracy. Building on the success of the CNN-LSTM Exp-3 configuration, this study adopted the same key parameters—such as a window size of 30 while shifting to a per-user training approach. Instead of using a single, generalized model for all users, separate models were trained for each individual to better capture their unique typing patterns. These models relied on keystroke timing features like DU, DD, UD, and UU latencies, structured into fixed-length sequences of 30. Each user model was trained with at least 100 samples, using two convolutional layers (with 64 and 32 filters of size 3) followed by LSTM layers (with 128 and 64 units). Dropout layers (0.2–0.3) were included to reduce overfitting. The training process leveraged the Adam optimizer, along with early stopping and learning rate scheduling, to ensure efficient and stable convergence.

A threshold of 0.5 has been defined i.e., if a user mean score greater than threshold then user is said to be authorized else the system will be detected as unauthorized.This configuration is chosen to balance temporal and spatial feature extraction, capturing both keystroke rhythm and typing flow. The per-user model strategy enhances adaptability to individual behaviors, allowing for improved detection accuracy, reduced false acceptance/rejection rates, and compliance with zero trust security principles, making it ideal for real-time CA scenarios. The performance analysis of first 20 users has been shown in Figure 5.6.

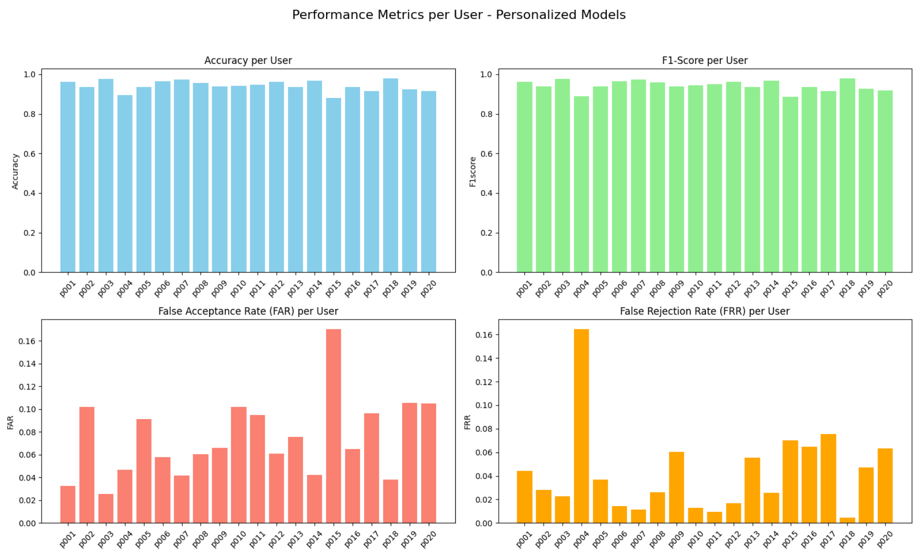


Figure 5.6: Per User Performance Metrics of CNN-LSTM Model

5.4 Demonstration

To demonstrate the real-time capability of continuous keystroke-based authentication system, a custom web interface was developed using the Dash framework in Python. The backend integrates a trained per-user CNN-LSTM model to verify typing behavior dynamically as users type into a text area.

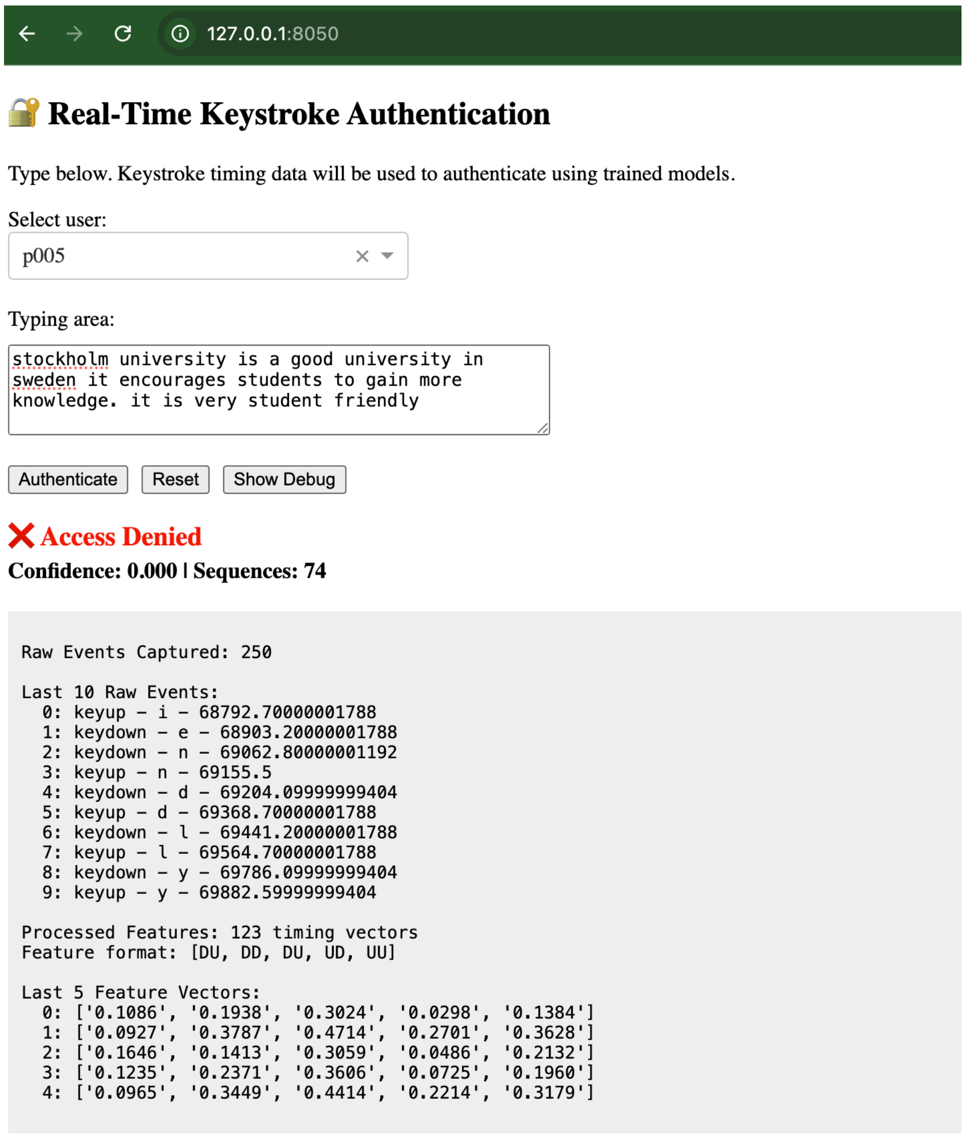
The interface captures raw keystroke events such as key down and key up timestamps using java script. Once the user selects their identity and types in the provided text box, the app feeds the captured sequences into their personalized machine learning model.

The following Figures 5.8, 5.7 illustrate the real-time visualization of the CA process. Imagine two users from dataset, p005 and p035 attempting to access a sensitive application. As part of ongoing monitoring, the CA engine actively observes their keystroke behavior in real time. When the "Authenticate" button is clicked, the system processes the captured keystroke events and generates an access decision. It also displays key information such as the total number of keystroke events recorded and the timings of last five feature vectors of each typed letter, which are critical in determining the authentication outcome.

In the Figure 5.8, the feature format for each digraph is as follows: DU.Key1.Key1, DD.Key1.Key2, DU.Key1.Key2, UD.Key1.Key2, UU.Key1.Key2. For user p005, the last five digraphs captured are ie, en, nd, dl, and ly. Similarly, for user p035, the last five digraphs are eu, ur, ro, op, and pe.

Confidence refers to the model's prediction score (often a probability between 0 and 1) indicating how strongly it believes the input matches the user's known typing behavior. A value closer to 1.0 means high confidence and the user is authorized, and closer to 0.0 means it's likely an unauthorized user. Sequences refers to the number of valid sliding windows or typing feature segments extracted from the raw keystroke data. For instance, if the model uses a window of 30 events to compute one feature vector, and the user types 100 events, it could produce 70+ sequences depending on overlap.

This setup effectively demonstrates the system's accuracy and reliability. For instance, if someone types as p035, but the typing style doesn't match the model trained on p035's actual behavior, the system correctly denies access, proving that it can distinguish between authorized and unauthorized user based on their unique typing patterns.



🔐 Real-Time Keystroke Authentication

Type below. Keystroke timing data will be used to authenticate using trained models.

Select user: p005

Typing area:

stockholm university is a good university in sweden it encourages students to gain more knowledge. it is very student friendly

❌ Access Denied

Confidence: 0.000 | Sequences: 74

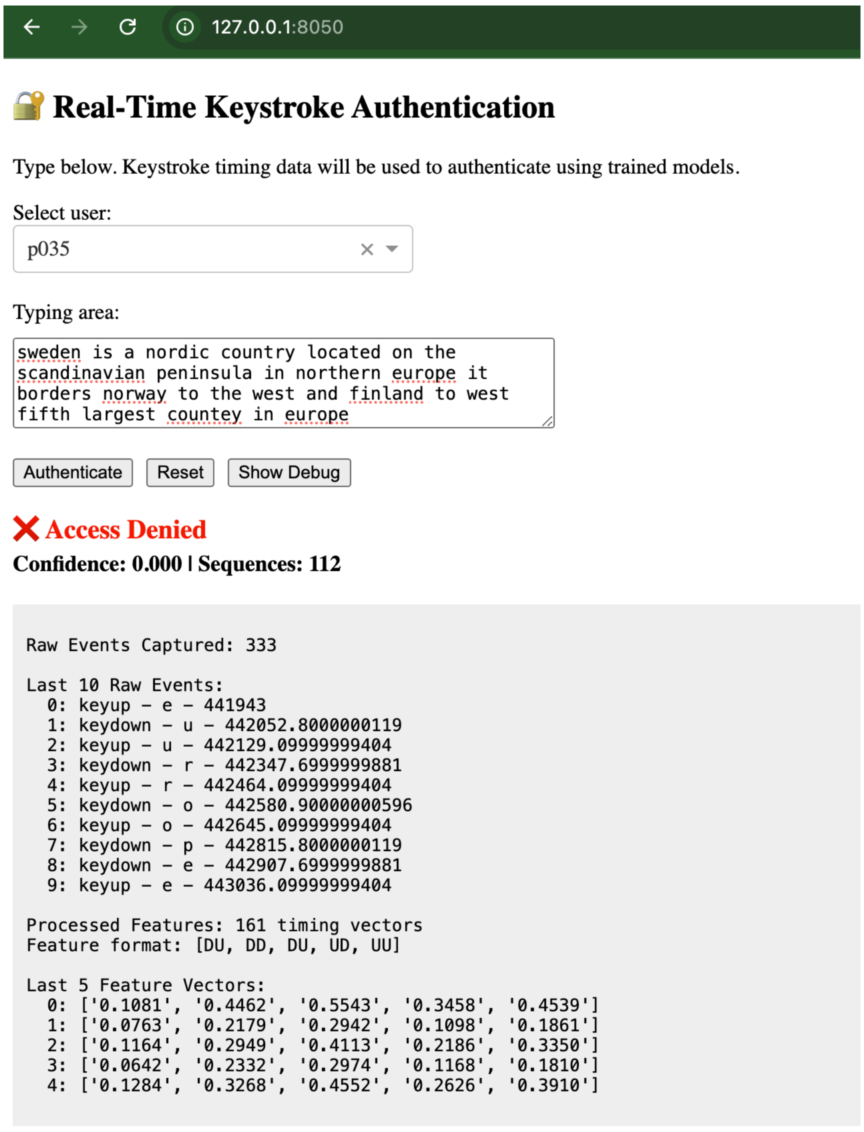
```
Raw Events Captured: 250

Last 10 Raw Events:
  0: keyup - i - 68792.70000001788
  1: keydown - e - 68903.20000001788
  2: keydown - n - 69062.80000001192
  3: keyup - n - 69155.5
  4: keydown - d - 69204.09999999404
  5: keyup - d - 69368.70000001788
  6: keydown - l - 69441.20000001788
  7: keyup - l - 69564.70000001788
  8: keydown - y - 69786.09999999404
  9: keyup - y - 69882.59999999404

Processed Features: 123 timing vectors
Feature format: [DU, DD, DU, UD, UU]

Last 5 Feature Vectors:
  0: ['0.1086', '0.1938', '0.3024', '0.0298', '0.1384']
  1: ['0.0927', '0.3787', '0.4714', '0.2701', '0.3628']
  2: ['0.1646', '0.1413', '0.3059', '0.0486', '0.2132']
  3: ['0.1235', '0.2371', '0.3606', '0.0725', '0.1960']
  4: ['0.0965', '0.3449', '0.4414', '0.2214', '0.3179']
```

Figure 5.7: Demonstration of Real-Time Continuous Authentication for User P005


127.0.0.1:8050

Real-Time Keystroke Authentication

Type below. Keystroke timing data will be used to authenticate using trained models.

Select user:
p035

Typing area:
sweden is a nordic country located on the scandinavian peninsula in northern europe it borders norway to the west and finland to west fifth largest countey in europe

❌ Access Denied

Confidence: 0.000 | Sequences: 112

```
Raw Events Captured: 333

Last 10 Raw Events:
  0: keyup - e - 441943
  1: keydown - u - 442052.8000000119
  2: keyup - u - 442129.09999999404
  3: keydown - r - 442347.6999999881
  4: keyup - r - 442464.09999999404
  5: keydown - o - 442580.90000000596
  6: keyup - o - 442645.09999999404
  7: keydown - p - 442815.8000000119
  8: keydown - e - 442907.6999999881
  9: keyup - e - 443036.09999999404

Processed Features: 161 timing vectors
Feature format: [DU, DD, DU, UD, UU]

Last 5 Feature Vectors:
  0: ['0.1081', '0.4462', '0.5543', '0.3458', '0.4539']
  1: ['0.0763', '0.2179', '0.2942', '0.1098', '0.1861']
  2: ['0.1164', '0.2949', '0.4113', '0.2186', '0.3350']
  3: ['0.0642', '0.2332', '0.2974', '0.1168', '0.1810']
  4: ['0.1284', '0.3268', '0.4552', '0.2626', '0.3910']
```

Figure 5.8: Demonstration of Real-Time Continuous Authentication for User P035

6

Discussion and Conclusion

6.1 Answer to the research question

This section provides an in-depth discussion of the findings presented in the Results chapter and addresses each of the research questions listed in Section 1.2 of Chapter 1. It highlights key observations, interprets model performance, and explores potential influencing factors that may have impacted the outcome, with a particular focus on the usability of continuous authentication within a Zero Trust framework.

Sub-RQ1: How can continuous authentication using keystroke dynamics be implemented using machine learning?

The proposed system demonstrates that machine learning (ML) models can be effectively utilized to implement continuous authentication by analyzing keystroke dynamics in real time. The architecture consists of two main phases i.e enrollment and continuous authentication. During enrollment, user-specific behavioral profiles are created using timing metrics such as dwell time and flight time. These profiles are later referenced for authentication during live user sessions.

In real-time operation, raw keystroke events are captured and processed into feature vectors which are then classified by trained machine learning models. If the typing pattern matches the stored profile, access continues; if not, the system can terminate or lock the session, enforcing zero trust principles. This modular integration of behavioral biometrics and machine learning provides a robust alternative to static credentials that obstructs the user access, particularly in environments where session-based identity verification is critical.

Sub-RQ2: What is the performance of different machine learning algorithms in detecting users based on keystroke dynamics?

The study evaluated a range of traditional and deep learning models: DT, RF, SVM, CNN, and CNN-LSTM hybrids. Results clearly indicated that CNN-LSTM outperformed all other models across performance evaluation metrics, such as accuracy, precision, F1-score, FAR, and FRR.

CNN-LSTM achieved 98.92 % accuracy, with a very low FAR (0.006) and FRR (0.0157), proving its strength in learning both spatial and temporal patterns in typing behavior.

Among traditional models, RF performed the best with 74.4 % accuracy and a relatively balanced FAR/ FRR. SVM and DT, while functional, exhibited higher false rates and less consistency across user inputs, making them less ideal for continuous authentication.

CNN models showed potential but lacked the sequential modeling capability that LSTM layers added, resulting in higher FAR/FRR than CNN-LSTM. This analysis validates the hypothesis that deep learning models with sequential learning capabilities (like LSTM) are better suited to handle behavioral biometrics such as keystroke dynamics.

Sub-RQ3: To what extent can continuous authentication using keystroke dynamics support or enhance zero trust security principles?

The framework proposed in this study aligns well with the zero trust paradigm of "never trust, always verify." Traditional login methods grant access for the duration of a session, assuming the user's identity remains valid which is a security threat. In contrast, the continuous authentication system developed here introduces session-long identity validation, ensuring that access is dynamically earned throughout user activity.

Integration of machine learning with zero trust policy components, Policy Decision Point (PDP) and Policy Enforcement Point (PEP) demonstrates how behavioral signals can drive real-time access decisions. Even though PDP/ PEP were not implemented in this prototype, their logical inclusion showcases how such a system could adaptively enforce policies based on evolving risk posture, contextual attributes (time, location, device state), and behavioral deviations.This strongly supports Zero trust principles by offering:

- Continuous risk evaluation based on user behavior.

- Minimized reliance on static credentials.
- Dynamic access control even after initial login.

6.2 Factors Influencing Model Performance

While the CNN-LSTM model showed excellent performance, there are some important factors that could influence the results and should be taken into account when considering real-world deployment.

6.2.1 User Typing Behavior

Some users in the per user training 5.6, such as p015, p010, p011, and p019 showed notably high FAR, primarily because their keystroke patterns were very simple and consistent. This kind of low-entropy behavior, while natural for some, makes it easier for others to mimic their typing and potentially bypass the system. Similarly, users like p004, p017, and p015 experienced higher FRR, which can happen when a personalized model overfits to a small amount of training data and struggles to generalize to slight variations in the user's typing. These findings highlight that user behavior plays a significant role in model performance. For users with predictable or limited variability in their typing patterns, it may be necessary to introduce adaptive thresholds or add an extra authentication factor to ensure both security and usability.

6.2.2 Window Size

How we slice and analyze the keystroke data also plays a big role. Using a sliding window approach helps the system look at typing over short periods. If the window is too small, the model may miss key patterns. If it's too large, the model might overlook quick and important changes. In this study, a window size of 30 worked especially well with CNN-LSTM, helping the model capture both immediate and long term typing behavior effectively.

6.2.3 Personalized Models Work Best

One of the biggest strengths of this study was training separate models for each user instead of using one general model for everyone. Since typing behavior is so personal, having individualized models significantly boosted accuracy and reduced errors. This user-specific approach proved to be highly effective for continuous authentication and aligns perfectly with the goals of zero trust policy.

6.2.4 Balancing Security and Usability

There's always a trade-off between keeping unauthorized users with false acceptance and annoying authorized users with false rejections. CNN-LSTM did a great job achieving that balance. However, some traditional models, like SVM struggled improving one metric often made the other worse. In real-world systems, finding that spot between usability and security is essential.

6.2.5 Real-World Typing

The dataset used in this study keyrecs was a solid choice, especially since it focused on free-text input that closely reflects how people type in the real world. Still, real-world typing can be affected by things like different keyboards, hand posture, user fatigue, or even stress. These variations aren't always captured in a dataset. So, while the system performed well in this setup, real-world deployments might show slightly different results depending on how and where people use the system.

6.3 Limitations

While the developed system demonstrates strong performance, few limitations must be acknowledged. Although the per-user training approach yields high accuracy, its generalizability to large scale enterprise scenarios may be limited due to the computational overhead and maintenance of multiple individualized models. Additionally, the sliding window technique, while effective, may not handle edge cases well—particularly in scenarios with very short or extremely rapid typing sessions—raising concerns about data imbalance. Furthermore, the system's validity during real-time deployment could be affected by external factors such as background processes, browser lag, or hardware-induced input delays, all of which may introduce noise and affect prediction accuracy.

The dataset is small where subjects are limited to only 100, this restricts the system's ability to generalize, potentially leading to overfitting and reduced robustness when deployed in real-world scenarios involving unseen users or device contexts. Furthermore, typing patterns can be influenced by external factors such as stress, emotional state, and behavioral fluctuations. These variations introduce potential inconsistencies in the biometric data, which could affect the accuracy and reliability of the authentication system under certain conditions.

6.4 Conclusion and Future work

Compared to static identity verification research approach, this study focuses on continuous authentication, offering a more dynamic and resilient security framework. Unlike prior work, this study experimented with a range of traditional and deep learning models, ultimately achieving higher accuracy through the use of hybrid ensemble techniques. Previous studies applied hybrid models on the CMU dataset for free-text keystroke dynamics and achieved an accuracy of 91.02 %. Similarly, other studies investigated different hybrid traditional learning models with free-text dataset, reporting accuracies between 73 to 81 %. However, these earlier studies focused purely on offline model evaluation and did not demonstrate real-time system integration and per user model training.

In contrast, this study integrates the trained per-user CNN-LSTM models into a real-time dash web interface, enabling interactive authentication directly through a browser. This study uniquely emphasizes continuous user verification as an essential feature of ZTA, where authentication must be continuous, adaptive, and context-aware. The originality of this research lies in the end-to-end implementation of user-specific CNN-LSTM models with live deployment capabilities a combination not previously presented in existing literature.

Moreover, the proposed method shows strong potential for real-world security-critical applications:

- In online examinations, it provides a robust mechanism to continuously verify the identity of candidates and detect impersonation attempts.
- In small-scale enterprise or high-risk settings, it aids in detecting insider threats or behavioral anomalies, enabling proactive security measures.
- Against automated bot attacks or script intrusions, keystroke dynamics provide a biometric signal that distinguishes human users from bots, offering a unique form of behavioral threat mitigation.

To enhance scalability, future work should involve training on larger datasets encompassing a broader range of users, making the system more practical for real-world deployment. Additionally, upcoming models should be designed to adapt to new users typing behaviors without requiring extensive retraining or enrollment, thereby reducing computational overhead and improving usability. Integrating keystroke dynamics with other behavioral biometrics such as mouse movement or gaze tracking could increase system robustness and reduce dependency on a single biometric trait.

Finally,deploying the system in live environments would enable evaluation of its transferability, long-term performance, and adaptability under real-world conditions. Also as mentioned in the high level system design Section 4.1, future enhancements could involve integrating the continuous authentication service with policy enforcement components in the zero trust framework, allowing for more dynamic and adaptive access control based on real-time user behavior.

6.5 Ethical and Societal Considerations

The dataset used in this study is the publicly available KeyRecs dataset, hosted on Zenodo. It was collected and shared by the original authors under an open data license and had already undergone ethical review and anonymization prior to its release. No personally identifiable information was included, and all users were identified using pseudonyms or participant IDs to maintain their anonymity. While keystroke dynamics is a behavioral biometric and generally more privacy-preserving than physical biometrics, it still carries potential risks if misused. To address this, the dataset was used strictly for academic, non-commercial purposes, with full acknowledgment of its source and adherence to the terms of use. No efforts were made to reverse engineer or deanonymize any subject. All machine learning models were trained and evaluated fairly, with precautions taken to avoid overfitting or misrepresenting results. The analysis focused entirely on methodological development, not individual profiling. These practices align with established ethical guidelines such as those outlined by [19] and standard university research protocols.

Bibliography

[1] S. Tiwari, W. Sarma, and A. Srivastava, "Integrating artificial intelligence with zero trust architecture: Enhancing adaptive security in modern cyber threat landscape," *INTERNATIONAL JOURNAL OF RESEARCH AND ANALYTICAL REVIEWS*, vol. 9, pp. 712–728, 2022.

[2] L. de-Marcos, J.-J. Martínez-Herráiz, J. Junquera-Sánchez, C. Cilleruelo, and C. Pages-Arevalo, "Comparing machine learning classifiers for continuous authentication on mobile devices by keystroke dynamics," *Electronics*, vol. 10, no. 14, p. 1622, 2021.

[3] E. G. Ogendi, "Leveraging advanced cybersecurity analytics to reinforce zero-trust architectures within adaptive security frameworks," *International Journal of Research Publication and Reviews*, vol. 6, no. 2, pp. 691–704, Feb. 2025, ISSN: 2582-7421.

[4] A. F. Baig and S. Eskeland, "Security, privacy, and usability in continuous authentication: A survey," *Sensors*, vol. 21, no. 17, p. 5967, 2021.

[5] E. Yılmaz and Ö. Can, "Keystroke biometric data for identity verification: Performance analysis of machine learning algorithms," *Computer Science*, no. IDAP-2023, pp. 143–150, 2023.

[6] K. S. Killourhy and R. A. Maxion, "Comparing anomaly-detection algorithms for keystroke dynamics," in *2009 IEEE/IFIP international conference on dependable systems & networks*, IEEE, 2009, pp. 125–134.

[7] A. Arsh, N. Kar, S. Das, and S. Deb, "Multiple approaches towards authentication using keystroke dynamics," *Procedia Computer Science*, vol. 235, pp. 2609–2618, 2024.

[8] R. Dillon *et al.*, "An agent-based modeling approach to free-text keyboard dynamics for continuous authentication," *arXiv:2505.05015*, 2025.

[9] V. Stafford, "Zero trust architecture," *NIST special publication*, vol. 800, no. 207, pp. 800–207, 2020.

[10] C. Bast and K.-H. Yeh, "Emerging authentication technologies for zero trust on the internet of things," *Symmetry*, vol. 16, no. 8, p. 993, 2024.

[11] R. Amin, T. Gaber, G. ElTaweel, and A. E. Hassanien, "Biometric and traditional mobile authentication techniques: Overviews and open issues," *Bio-inspiring cyber security and cloud services: trends and innovations*, pp. 423–446, 2014.

[12] Y. Liang, S. Samtani, B. Guo, and Z. Yu, "Behavioral biometrics for continuous authentication in the internet-of-things era: An artificial intelligence perspective," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 9128–9143, 2020.

[13] H. C. Volaka, G. Alptekin, O. E. Basar, M. Isbilen, and O. D. Incel, "Towards continuous authentication on mobile phones using deep learning models," *Procedia Computer Science*, vol. 155, pp. 177–184, 2019.

[14] DataCamp Editorial Team. "What is machine learning?" Accessed: 2024-05-18. (2024), [Online]. Available: `https://www.datacamp.com/blog/what-is-machine-learning`.

[15] D. Ajish, "The significance of artificial intelligence in zero trust technologies: A comprehensive review," *Journal of Electrical Systems and Information Technology*, vol. 11, no. 1, p. 30, 2024.

[16] Z. Sitová, J. Sedenka, Q. Yang, *et al.*, "Hmog: New behavioral biometric features for continuous authentication of smartphone users," *IEEE Transactions on Information Forensics and Security*, vol. 11, pp. 1–1, Jan. 2015. DOI: `10.1109/TIFS.2015.2506542`.

[17] J. Li, H.-C. Chang, and M. Stamp, "Free-text keystroke dynamics for user authentication," in *Artificial Intelligence for Cybersecurity*, Springer, 2022, pp. 357–380.

[18] L. Xiaofeng, Z. Shengfei, and Y. Shengwei, "Continuous authentication by free-text keystroke based on cnn plus rnn," *Procedia computer science*, vol. 147, pp. 314–318, 2019.

[19] M. Denscombe, *EBOOK: The good research guide: For small-scale social research projects*. McGraw-Hill Education (UK), 2017.

[20] T. Dias, J. Vitorino, E. Maia, O. Sousa, and I. Praça, "Keyrecs: A keystroke dynamics and typing pattern recognition dataset," *Data in Brief*, vol. 50, p. 109 509, 2023.

A

Appendix A

Below tabular format represents the hyperparameters used for each experiment across the various machine learning models that has been trained for continuous authentication.

Experiment ID	Window Size	max_depth	min_samples_split	Criterion
DT-Exp1	10	8	2	gini
DT-Exp2	20	12	4	entropy
DT-Exp3	30	None	2	gini
DT-Exp4	25	10	5	entropy
DT-Exp5	15	6	3	gini

Table A.1: Decision Tree Experimental Setup

Experiment ID	Window Size	n_estimators	max_depth	min_samples_split
RF-Exp1	10	100	None	2
RF-Exp2	20	150	15	2
RF-Exp3	30	200	10	4
RF-Exp4	20	100	8	5
RF-Exp5	25	120	None	3

Table A.2: Random Forest Experimental Setup

Experiment ID	**Window Size**	**Kernel**	**C**	**Gamma**
SVM-Exp1	10	rbf	1.0	Scale
SVM-Exp2	20	rbf	0.5	Auto
SVM-Exp3	30	poly	1.0	scale
SVM-Exp4	25	Linear	1.0	Auto
SVM-Exp5	20	sigmoid	0.8	scale

Table A.3: Support Vector Machine (SVM) Experimental Setup

Experiment ID	**Window Size**	**Conv Layers**	**Filters**	**Kernel Size**	**Dropout**
CNN-Exp1	10	2	32	(3,3)	0.3
CNN-Exp2	20	3	64	(5,5)	0.4
CNN-Exp3	30	2	32	(3,3)	0.2
CNN-Exp4	25	1	16	(3,3)	0.1
CNN-Exp5	15	3	64	(5,5)	0.5

Table A.4: Convolutional Neural Network (CNN) Experimental Setup

Experiment ID	**Window Size**	**Conv Filters**	**LSTM Units**	**Dropout**	**Epochs**
CNNLSTM-Exp1	10	32	64	0.3	10
CNNLSTM-Exp2	20	64	128	0.4	15
CNNLSTM-Exp3	30	32	64	0.2	20
CNNLSTM-Exp4	25	16	32	0.1	10
CNNLSTM-Exp5	15	64	128	0.5	25

Table A.5: CNN-LSTM Experimental Setup