A Few-Shot Practical Behavioral Biometrics Model for Login Authentication in Web Applications

Jesús Solano, Lizzy Tengana, Alejandra Castelblanco, Esteban Rivera, Christian Lopez and Martín Ochoa
AppGate Inc.
first.last@appgate.com

Abstract—**Risk based authentication has been advocated as complement to traditional authentication mechanisms in order to raise the bar against attackers in possession of stolen credentials. Behavioral biometrics has received attention in the literature in the past decade, however the best results have been obtained in the so-called continuous setting and with enough training data, usually spanning several hours of user interaction. In this paper we explore the more challenging scenario of behavioral biometrics as an effective risk-based authentication technique using both mouse and keyboard information at login time (static authentication), assuming only between 3 and 7 login sessions per user for training. In a controlled but realistic experiment with 89 subjects we achieve a FRR of 10.73% and FAR of 23.34% for a model trained using only 5 login attempts, each performed in less than 30 seconds on average. We also evaluate our prototype with 2000 users from production data in the banking domain.**

I. Introduction

Password-based authentication in web-based services is widely popular, however it suffers from various security drawbacks [7], [39]. In many implementations, and for usability reasons, users can still choose relatively simple passwords that are prone to brute-forcing or guessing. Even when passwords are relatively secure, if not properly stored, they are prone to be lost in data breaches [27]. Moreover, domain-specific malware [31] often targets credentials of financial services.

Risk-based Authentication [19], [34] has been advocated to complement password-based authentication and raise the bar against attackers impersonating victims via traditional credentials. One technique that has been studied extensively in the literature and that has lately also found its way in commercial implementations is behavioral biometrics [22], [23], [37]. This technique has shown promise in various contexts, perhaps the most studied is the behavior of mouse movements [16] and keystrokes [25]. However, the best results in terms of False Acceptance Rate (FAR) and False Rejection Rate (FRR) have been obtained in the so-called *continuous* setting [1][8], which refers to interactions of users for long periods of time with a machine. Also, for those techniques to work, it is necessary to collect hours of interaction before building a model which captures the user's behavior.

Workshop on Measurements, Attacks, and Defenses for the Web (MADWeb) 2020
23 February 2020, San Diego, CA, USA
ISBN 1-891562-63-0
https://dx.doi.org/10.14722/madweb.2020.23011
www.ndss-symposium.org

In the context of web applications it is challenging to use some of the published results on continuous authentication for several reasons. First, the length of a session might be relatively small (i.e. a few seconds), for instance in domains such as banking, where users typically login to perform one or two transactions. Also, in such scenarios many users typically login rarely, so it might take months to gather sufficient sessions to train a model. So on the one hand, one would need many sessions in order to train a robust model and on the other hand, the model should be good enough to decide whether the session is legitimate or not for a very brief window of user interaction. The best published results in behavioral biometrics for logins achieve good accuracy (over 95% [21], [4]) but require at least 50 sessions for training, which is not acceptable in many practical scenarios.

Moreover, the best results are usually obtained by training one model per user, which might be difficult to scale to settings with millions of users. Furthermore, there are multiple practical questions that have been so far not studied deeply, such as how to best evaluate models against realistic attacks (i.e. an attacker typing exactly the same credentials as the victim) and how to design models that can operate on aggregated data (due to the sensitivity of passwords) among others.

In this paper we tackle thus the issue of building a robust and scalable static authentication solution for web applications using a few-shot training scheme that considers the historical behavioral pattern from the user to verify his/her identity. We discuss the design criteria of our solution and present the results of an in-depth evaluation in a controlled setting with 89 users. In order to evaluate our model in a realistic setting, we measure its precision in a production environment with over 14000 distinct sessions belonging to ca. 2000 distinct users that have more than 5 login sessions each.

Note that in this work, given the difficulty of the task (short interaction and few-shot learning), we consider *practical* an accuracy close to 80%, where one might privilege a low FRR over a low FAR depending on risk appetite and usability considerations. For instance a model that incorrectly classifies a legitimate login session as malicious once out of every 10 login attempts, while correctly classifying 3 out of 4 attacks, is of practical value since **a)** suspicious login sessions can be challenged by means of two-factor authentication, while keeping friction relatively small, **b)** several risk-based authentication factors can be taken into consideration to build an even more precise model, as exemplified in [30] and **c)** there is a trade-off in terms of risk acceptance when minimizing the number of sessions needed to build a risk assessment model, given that an unnoticed attack can happen before the model is trained and potentially poison the model trained if the attack is not properly identified by other means. In other words, although a more accurate model could be achieved by training with several login attempts, with that strategy there is a higher the risk that an attack can go potentially unnoticed.

Our contributions are in sum:

- A novel machine-learning technique for static authentication using between 3 and 7 login sessions for training that achieves a FAR between 23.51% and 22.67%, and a FRR between 18.16% and 7.64%, in an experiment with 89 users.
- A detailed evaluation design for static authentication.
- Evidence of scalability and practicality from a production implementation with 2000 users.

II. Background

Researchers have proposed several risk-based authentication strategies to improve security of web-based environments and other applications. One approach is the use of factors that describe user behavior as a complementary feature [15]. This approach is known in the literature as behavioral biometrics [2]. The main advantage of this approach is that whoever wants to impersonate a given user must not only possess the victim's credentials, but also imitate the victim's behavior during a given time window, demanding a much deeper knowledge to perform an attack.

There are two user authentication scenarios: *static* and *continuous*. *Static authentication (SA)* represents the process of user identification at a single point in time in a session, usually at the beginning (login process). During this stage, the user provides information (e.g. username and password) which will be stored and then compared with a signature profile stored in a database. Such comparison is performed, in order to verify that the user is who he claims to be, ultimately granting or denying access to the required service.

A distinctive characteristic of static authentication is that the tasks performed by users are fixed and limited, representing a case which would facilitate the construction of a behavioral user classification model. Nevertheless, the amount of information gathered during login time is usually too scarce to perform a deep behavioral analysis, so static authentication through behavioral biometrics still represents a challenge.

The constant verification of user identity is known as *continuous authentication (CA)*. During this process, the user has the freedom allowed by the application or system, leading in most cases to free text and free mouse movement characteristics. These conditions require the monitoring of user identity to be independent from the task performed. The hypothesis of CA is that the attacker managed to bypass the login security barrier. Therefore, CA is carried out repeatedly throughout the session, collecting users behavioral information and constantly verifying their identity.

Among the main challenges presented by behavioral biometrics is that of adaptability to changes in user patterns, which could be treated through comprehensive datasets, artificial intelligence tools and online learning models [32].

A. Features used for mouse and keyboard dynamics

Keystroke and mouse biometrics are proposed as a viable behavioral authentication solution, considering the widely accessible and non-intrusive nature of such interaction. Recent studies have tried to evaluate the feasibility of mouse, keyboard and other biometric information for static and continuous authentication in several real word applications, details can be found in section V. Consequently, better understanding of the feature extraction methods for mouse and keyboard dynamics has been gained in the field.

Features of mouse interaction relevant for classification models include: spatial information of the trajectory between mouse events described as curves [39] and statistical descriptors over raw mouse movements dynamics [2]. On the other hand, keystroke features are generally computed from the sequence of key events, such as key-down and key-up [6]. *Digraph* or *trigraph* latency models are good estimators of behavioral characteristics, where an *n-graph* is estimated with the latency between *n* keystroke events [25].

Fusion of multimodal biometric data can be performed at the feature, matching and decision levels [26]. In feature level fusion, calculated features are integrated early in the model and the same classification model integrates all biometric inputs [21]. Matching score fusion schemes use independent classification models for each biometric trait and then fuse the output scores to create a risk estimator [11]. Finally, the highest level of fusion integrates the decisions of multiple classifiers into a single classifier.

B. Few-Shot Learning

Machine learning algorithms require large amounts of data in order to succeed. As a result, most techniques usually lack the ability of learning from a low number of examples. To address this problem, Few-Shot learning is proposed by researchers for many applications in the field [36]. The fundamental idea is to rapidly generalize a task from a limited supervised experience. A natural solution to alleviate this scarcity of training samples is to augment synthetically the existing samples for each training class [38], [13]. In order to augment data, researchers have invoked invariant transformation in feature space [9], [35], [3]. Section III explains in detail our approach on few-shot learning using a Random Forest algorithm with data augmentation.

C. Attacker model

For the scope of our work, we assume an attacker has compromised the user's credentials (login/password) and then attempts to impersonate the victim using the stolen credentials. We assume an attacker does not have information about behavioral patterns of a victim and uses mouse and keyboard naturally or can use a script (i.e. Selenium). In our solution, we will assume a JavaScript monitor will collect mouse and keyboard events and compute aggregations on the client side.

We assume this monitor is not compromised by an attacker, and is protected using orthogonal countermeasures from the software protection domain. Completely disabling the monitor should trigger alarms, and we do not focus on those attacks as it is out of the scope of this paper.

III. Approach

The main goal of our work is to design a model which is capable of learning from unique human-computer interactions in login scenarios to verify user identity implicitly. As mentioned above, we consider in our work two different sources of information, namely mouse and keyboard events. From those events we build dynamic features for each user and feed a supervised machine learning model. With our approach we aim to answer fine-grained research questions that help in understanding the stability and robustness of our technique. The general research question of our work is:

***RQ:** Is it possible to design an accurate, fast and scalable behavioral biometrics model with a few number of training sessions in static authentication scenarios?*

Additionally, one of the main concerns in behavioral biometrics has to do with the question of whether the trained model has learned abstract behavioral patterns from user independently of what the user typed or it has learned something about the particular recorded interaction. To address this concern, we propose the following questions to be answered using our experiments' evaluation:

RQ-A: Is it possible to capture targeted attacks against victims by training with non-attack login attempts from other users typing different credentials to those of the victim?
RQ-B: What is the impact on the model performance if the user changes his/her password?
RQ-C: How does the number of logins n needed for training affect the model performance in terms of FRR and FAR for some small values of n?

Each one of these research questions is considered and answered in section IV.

A. Design considerations of the machine learning model

The general idea behind the analysis of user interaction with mouse and keyboard is that timing and movement directions can be used to build a user profile that complements traditional authentication systems. As we stated in section II, most systems require the user to interact many times and/or for long periods of time in order to learn those behavioral patterns. However, typically login interactions are very short. Our proposed solution aims to address this shortcoming by considering a few-shot machine learning model to detect a given user in static authentication environments.

As we are interested in static authentication, we define a session as the time window in which a user is performing the login process. In order to describe user behavior, we gathered mouse movements and keyboard strokes for all sessions. Raw mouse movements are represented as tuples of timestamp and Cartesian coordinate pairs. To analyze mouse data, we define five different sets of point-by-point based features. The first and second set of features are inspired by Ahmed et al. [2]. They proposed a split of the movements' space into eight different directions as shown in Figure 1. In order to capture this usage pattern, we calculate (1) the average speed in each of the eight movement directions as well as (2) the movement direction histogram which tell the percentage of movements performed by the user in each direction during the login session. The remaining three features are inspired by the work of Zheng et al. [39]. They proposed a set of angle-based metrics: direction, angle of curvature and curvature distance. The direction is related to the angle of direction between two consecutive points (3). The angle of curvature is the angle formed by the two straight lines connecting three consecutive points(see angle y in figure 2) (4). The curvature distance is the ratio between the length of the line $\overline{AB}$ to the perpendicular distance from point B to line $\overline{AB}$ (5). These sets of angle-based metrics are not based on a user's environment (i.e. screen size, resolution, brand of mouse, pointer sensitivity) and thus they are relatively independent of the user's platform [39]. From each session we thus extract a vector of mouse features consisting of 272 features.

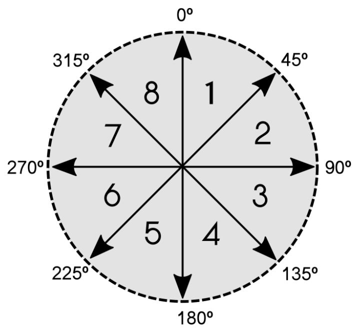


Fig. 1: Different mouse directions considered to build mouse dynamics [30].

On the other hand, to analyze keystrokes we define a set of timing features which are focused on building a unique typing rhythm for a specific user. Such features focus on the time when each key is pressed or released as the user types. Raw keyboard events are represented as tuples of timestamp, key and action (Press/Release). Since users type sensitive information at login time, the data is analyzed in an anonymized fashion. The keyboard was unified into one unique zone, which abstracts away from particular sets of possible keys typed. In our work we define four different

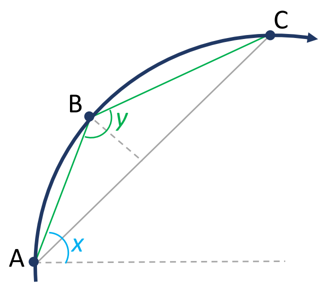


Fig. 2: Illustration of angle-based metrics. Adapted from [39].

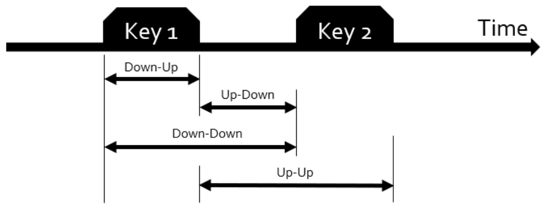


Fig. 3: Illustration of keyboard timing features.

timing metrics: Down-Up, Up-Up, Down-Down and Up-Down time. The (1) Down-Up keyboard feature is the duration of time from a released key until the next key release. The (2) Up-Up feature is the latency from one release until the next key is released. The (3) Down-Down feature is the latency between two key-pressed events. The (4) Up-Down feature is the duration of time from the previous key-release until the next key is pressed. Figure 3 illustrates timing metrics for keyboard events. Over these metrics we calculate mean, standard deviation and median for each session. From each session's raw keyboard data, we extract a vector of keyboard features consisting of 12 features.

We then combine both mouse and keyboard feature vectors into a single vector in a feature-level fusion fashion. We thus obtain a vector of 284 values per login attempt, that abstracts the behavioral biometric profile of the user.

As we mention in Section II common algorithms implement some of the previously discussed features, but require several authentication attempts (up to 50) to train a robust model. For this reason, we aimed to work on how to increase the difference between the features coming from a legitimate user compared to the attacker incoming features. As we are dealing with a sequence of authentication tries, where users perform very similar actions, it is natural to think that historical patterns are important and they can be exploited to improve the prediction accuracy of our classifier.

There are several ways to consider data from previous sessions in machine learning frameworks [10], most of them with prominent results. In our use case, the inclusion of user history is an effective way to increase the separation of user-attacker in the feature space. We propose to compare the associated feature vector of an incoming login with the mean behavior of the user in the last n logins in a feature-wise way. The history vector at session t is calculated as follows:

$$h_t(n) = \frac{\sum_{i=t-n}^{t-1} f_i}{n} \tag{1}$$

where h_t is the history vector, n is the number of sessions to consider from the past and f_i is the feature vector of each of the previous iterations. The comparison of a new incoming session behavior against the history is given by equation:

$$f_t = |f_t - h_t(n)| \tag{2}$$

where f_t is the incoming calculated feature vector and h_t is the mean of the last n features. The idea is that if a new incoming session is performed by the same subject, the absolute difference should be small. Otherwise, the difference should be large, highlighting attacks.

Once the keyboard and mouse features have been extracted and absolute differences created, classification is performed using a supervised machine learning classifier. Since we want to maximize accuracy detection, we propose to train a independent model per user. Since scalability is also a requirement in this setting, a fast training algorithm is needed.

For the stated reasons, we have chosen a Random Forest (RF) classifier to discriminate legitimate users from attackers. After parameter tuning of the model, we propose to train a RF with 100 estimators. Furthermore, we train each RF in a few-shot fashion. This means that for each training process we are expecting between 3 and 7 sessions of about 30 seconds each one. The method we used to perform few-shot learning works by augmenting only the legitimate user sessions while attacker sessions are learned without augmentation. Notice that we do not need to augment attacker sessions because we have enough attacks, all of those extracted randomly from login attempts performed by other users (for further details refer to Section III-B). Augmentation of the minority class is performed by over-sampling with multiple copies (10x) of the original positive features. In this way, we increase the information of legitimate users in the training set. This oversampling method has proven to keep the model robustness in classification tasks [18].

We defined as positive labels the difference vector calculated from the history of a legitimate user. To create negative samples, we randomly choose sessions performed by other users and compare (create the difference vector) against the history of the legitimate user we are training the model for. We thus train a model using a balanced data set in which we oversample the minority class (i.e. legitimate sessions). The number of positive samples used to train the model is n while the number of negative samples is $n * 10$ (random sessions performed by other users).

B. Experimental Design

In order to evaluate our approach, we propose a set of experiments with multiple set-up scenarios where model performance is measured. These experiments are inspired by the research questions stated above.

For the proposed task, each user is requested to login into a web page using their assigned credentials. As our goal is to train and test a single model for each user, we would ideally need attacks for each individual user's credentials to evaluate the accuracy of our approach. However, it is impractical to collect multiple login attacks for every distinct username and password combination at large (that is, multiple targeted attacks by different users that are typing a victim's particular credentials). To address this challenge, we propose two different login interactions to generalize the attacks for all users.

Login type 1: Every user is requested to perform n login sessions writing the same pre-defined username and password

TABLE I: Experiment 1 - Ideal Scenario for Biometric Characterization.

	Legitimate user class	**Attack class**
Train	n sessions of u_i writing $\langle U, P\rangle_{\text{fixed}}$	n random sessions from U_j writing $\langle U, P\rangle_{\text{fixed}}$
Test	The remaining m sessions of u_i writing $\langle U, P\rangle_{\text{fixed}}$	All the remaining sessions from U_j writing $\langle U, P\rangle_{\text{fixed}}$

This is the ideal scenario where all users are writing the same login credentials for both the training and testing phases. The goal of this experiment is to check if a user's model can differentiate between the biometric behavior of the legitimate user and others. This scenario is crucial to validate that the model is in fact learning from the user's biometric behavior and not from different username and password sequences literally.

TABLE II: Experiment 2 - Pragmatic Training for Targeted Attacks.

	Legitimate user class	**Attack class**
Train	n sessions of u_i writing $\langle U, P\rangle_{\text{fixed}}$	n random sessions from U_j writing $\langle U, P\rangle_{\text{own}}$
Test	The remaining m sessions of u_i writing $\langle U, P\rangle_{\text{fixed}}$	All sessions from U_j writing $\langle U, P\rangle_{\text{fixed}}$

The goal of this experiment is to simulate the model's behavior in a real life scenario where we won't have targeted attacks to train a user's model, therefore other users individual credentials would be used as the training attack class. As for the testing phase, we use targeted attacks, which simulate an attacker who owns the user's credentials.

TABLE III: Experiment 3 - Password Change.

	Legitimate user class	**Attack class**
Train	n sessions of u_i writing $\langle U, P\rangle_{\text{fixed}}$	n random sessions from U_j writing $\langle U, P\rangle_{\text{own}}$
Test	All sessions of u_i writing $\langle U, P\rangle_{\text{own}}$	All the remaining sessions from U_j writing $\langle U, P\rangle_{\text{own}}$

The aim of this experiment is to evaluate how resilient our approach is to changes of legitimate user credentials in the testing phase (i.e. change of password). Since we collected two types of login credentials for each user, we used one for training and one for testing. For the attack class, in order to simulate the production environment, random samples from other users credentials are used.

TABLE IV: Experiment 4 - Training and Testing with Individual User Credentials.

	Legitimate user class	**Attack class**
Train	n sessions of u_i writing $\langle U, P\rangle_{\text{own}}$	n random sessions from U_j writing $\langle U, P\rangle_{\text{own}}$
Test	The remaining m sessions of u_i writing $\langle U, P\rangle_{\text{own}}$	All the remaining sessions from U_j writing $\langle U, P\rangle_{\text{own}}$

To recreate the restrictions of a real production environment, we consider not having targeted attacks neither for training nor for testing. To simulate this scenario, we only used individual credentials from all users (login type 2), to build a valid user's model.

$\langle U, P\rangle_{\text{fixed}}$ while the JavaScript monitor is gathering information from mouse and keyboard events.

Login type 2: Every user u_i is requested to perform n login sessions writing his own username and password $\langle U, P\rangle_{\text{own}}$ while the JavaScript monitor is gathering information from mouse and keyboard events.

Having all users writing the same character sequence $\langle U, P\rangle_{\text{fixed}}$ in the *Login type 1* makes it possible to simulate multiple attacks, since we have the same static text typed by all users. On the other hand, having individual login sessions in *Login 2* with $\langle U, P\rangle_{\text{own}}$, makes it possible to define other experiments and address some of the fine-grained research questions of Section III.

Let: $U = \{u_1, ..., u_n\}$ be the set of all users. Given a legitimate user $u_i \in U$ we denote the set of all users excluding the user u_i as:

$$U_j = \{u_j \in U | j \neq i\}$$

The proposed experiments are described in Tables I to IV.

IV. Evaluation

A. Datasets

The first dataset we analyzed was a pilot dataset with a small number of subjects. We call this dataset *"Small Scale Pilot Dataset"* (SSPD). The dataset was collected among the employees of a company who were informed about the experiment; they were asked to login to a monitored dummy website for data collection. The test subjects used their usual input devices, namely keyboard and mouse, so their typing behavior was as natural as possible. Each user performed two experiments as explained in section III: First, the same username and password $\langle U, P\rangle_{\text{fixed}}$ was collected for all users, and second, they were asked to write a username and password $\langle U, P\rangle_{\text{own}}$ composed of their name and a combination of unique numbers. 20 attempts were recorded per user, that is, 10 inputs per experiment for each one of the 21 test subjects.

To preliminary test the scalability of our approach to more users under a controlled environment, the 'Amazon Mechanical Turk' service was used. With this service, human workers perform a certain task following instructions defined by the task requester. We managed to collect a total of 1374 valid login attempts from 89 subjects. We call this dataset *"Medium Scale Pilot Dataset"* (MSPD). A summary of the average characteristics of the users interaction for each dataset is presented in Table V.

Restrictions on behavior: It is important to note that we imposed restrictions on the behavior of subjects as follows: (1) it was not allowed to use *Tab* to navigate between input fields, (2) it was not allowed to press *enter* to complete the login in order to have more mouse interaction; and finally, (3) it was not allowed for the subjects to *copy-paste* the username or password, to keep the maximal keyboard data.

Although these choices might seem too restrictive when compared to user behavior on the wild, there are several reasons why we chose to impose them. First, we wanted to maximize the keyboard and mouse interactions in order to have more information to build a meaningful model, since the scenario we are tackling its already quite challenging due to the short term of a login attempt. Second, in certain domains similar restrictions are common, such as in banking. In fact, in the banking domain even longer interactions can be recorded when users are asked to enter their pin numbers using a randomized keyboard on-screen. Last, if good results are achieved in spite of the mentioned restrictions, a case can be made to enforce them in applications that want to integrate this technology in order to favor security. We believe that in although these restrictions create some friction with users, they are still reasonable for critical applications.

TABLE V: Pilot Datasets Description

	SSPD N=21	**MSPD N=89**
Avg. login time [sec]	18.69	26.76
Avg. # of keystrokes/login	46	49
Avg. # mouse events/login	106	129

Privacy considerations: In the pilot datasets we did not ask participants to introduce their real login information. We simulated a login website interface where they introduced either the shared artificial credentials, or individually defined artificial login information. No sensitive information was managed or stored.

Raw data was then processed in the back-end to compute features. Note that processed features are relatively hard to revert into raw data (there is no injectivity in the feature calculation function), and thus this representation already provides some degree of privacy. In particular, keystroke features are considered to be secure in the sense of protecting the inputs typed, since only speed related averages are computed for various actions.

Limitations: First, the SSPD test subjects were operational personnel so they are used to working with a computer throughout the day, which might not be a representative sample of users in the wild; secondly, regarding both datasets, it was expected that all logins collected would have been performed within a short time frame; this might have caused an artificial similarity among login attempts. Also, due to privacy reasons we did not ask the test subjects to type their real usernames or passwords. It could be argued that the behavior of each user could change when they write something familiar to them instead of something new, nevertheless, our hypothesis is that these datasets provide a meaningful baseline to work with and approximate login attempts in the wild.

Data validation: In order to increase the number of legitimate sessions tested, and to do cross-validation, we evaluated several *parallel worlds* by randomly selecting n sessions out of the available sessions of a given user to train the model, and test it against the remaining sessions. Since on average there are 10 sessions per user, this gives $\binom{10}{n}$ choices of parallel worlds.

Production dataset: Additionally, in order to evaluate our approach in a less controlled environment, we have preliminary evaluated data captured in a production environment from the banking domain. The data captured corresponds to roughly one week of activity, where a total of 380.000 login sessions were performed. Of those, 2.000 users logged in more than 5 times, which allowed us to evaluate the performance of our method for those users in a total of 14.000 sessions.

In this scenario, for privacy reasons, features are calculated on the client side, to avoid storing raw credential information on the server side.

Hardware The machine used in our tests has an x86_64 architecture, Intel® Core$^{\text{TM}}$ i7-8550U 1.80GHz processor with 4 physical and 4 virtual cores and 23.3GB of RAM.

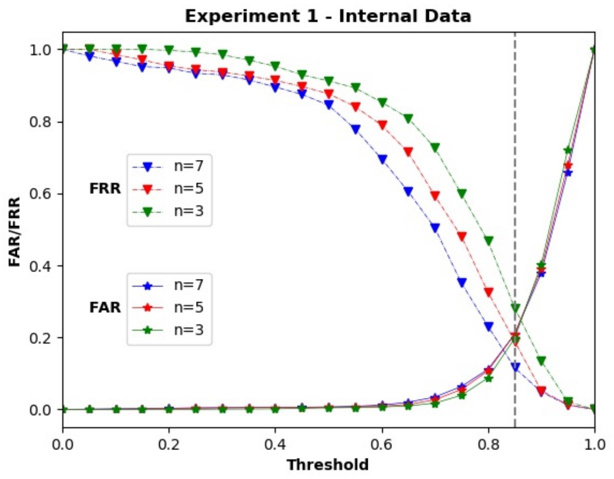
B. Results

We used the experiments described in subsection III-B to evaluate the effectiveness of our model. We perform our evaluation by measuring (1) the False Acceptance Rate *FAR*, which is the proportion of actual users identified as attacks, and (2) the False Rejection Rate *FRR*, which is the proportion of actual attacks identified as users by the model. It is also important to remark that Lower FAR is generally preferred in high security applications, whereas Lower FRR is preferred to in systems which try to improve the user experience.

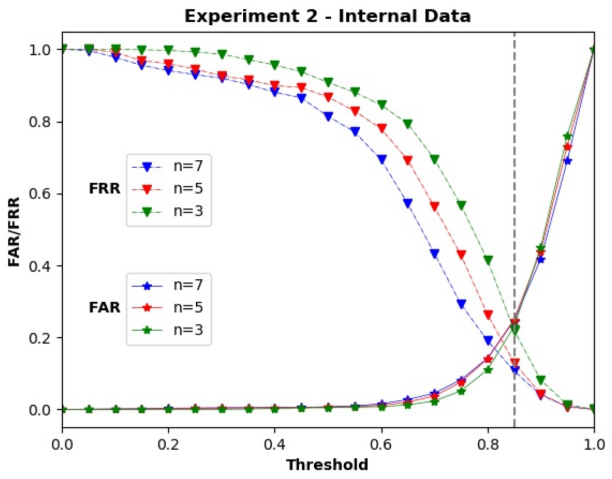
1) Small Scale Pilot Dataset (users = 21): We first tested our proposed model in the Small Scale Pilot Dataset (SSPD). Notice that the SSPD dataset had the most controlled environment among the datasets we collected, were we made sure the users were following the instructions and therefore, SSPD is the highest quality dataset we had to approach our research questions. We fully evaluate all of the 4 experiments on the SSPD dataset to preliminary answer our research questions. The FAR and FRR with different numbers of login sessions for training are shown in Figure 4. For the FRR, the larger the number of login sessions used for training, the more accurate our model is in detecting legitimate users for all thresholds. Remarkably, the number of logins sessions for training does not affect significantly attack accuracy detection (FAR). As a general trend, we observe that 0.85 is the threshold which globally minimizes FRR and FAR simultaneously.

The *first experiment* is the ideal scenario to test the system capability to learn from the users' behavior instead of the static text written by them as stated in Table I. For our selected threshold, namely 0.85, the difference in the FAR score for the three tests on the working point is negligible for this experiment, while the difference in FRR between $n = 7$ and $n = 5$ is around 7% (Figure 4c); it can be observed that considering more users is worth it for this experiment; however, only with 5 logins the metrics are around 19%, which is an acceptable value.

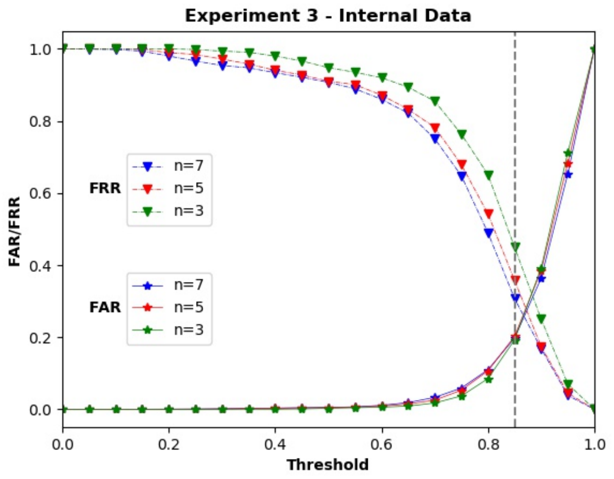
In the *second experiment*, for our selected threshold, a lower FRR is achieved, meaning that lesser friction for the normal user is preferred. In this case, the difference in the FRR between 7 and 5 user logins for training the model is only 2%, meanwhile, between 5 and 3 logins the difference is almost 11%. On the other hand, the difference in FAR between 7 and 3 logins is only 2% (Figure 4a). Being the differences in FRR so considerable, 5 logins could be seen as the preferred case, since the metrics do not fall strongly and fewer logins



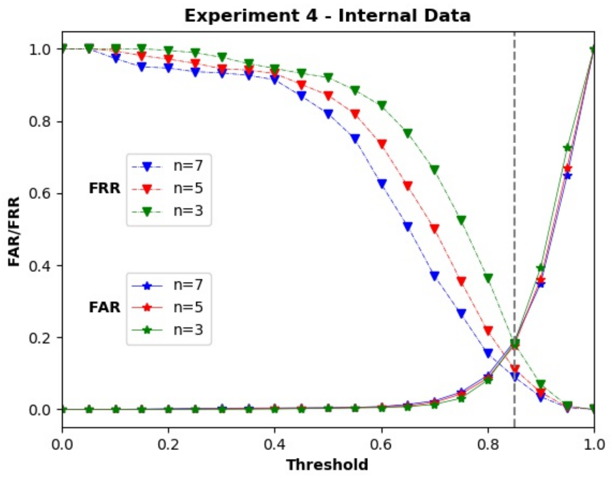
(a) Experiment 1 (Ideal Scenario for Biometric Characterization) results for different number of sessions(n) in training.



(b) Experiment 2 (Pragmatic Training for Targeted Attacks) results for different number of sessions(n) in training.



(c) Experiment 3 (Password Change) results for different number of sessions(n) in training.



(d) Experiment 4 (Training and Testing with Individual User Credentials) results for different number of sessions(n) in training.

Fig. 4: FAR versus FRR plot for different experiments when they are **evaluated on the pilot dataset (users=21)**. Notice that at a threshold cut-off point of 0.85 both FAR and FRR are simultaneously minimized.

are needed, which is easier to acquire. With a FRR of 13% and a FAR of 25% for 5 logins, this is preliminary evidence, regarding **RQ-A**, that our system is capable to learn user's behavior being trained with credentials different to the ones of the user.

The *third experiment* aims to show how robust is the system against password changes from the same user, meaning that no attacks are involved, but the user's experience itself. On our selected threshold point, the FAR of the system is lower than the FRR: For $n = 7$ and $n = 5$ logins the difference in FRR is 4%, compared to the difference between $n = 5$ and $n = 3$ logins of 9% (Figure 4b). The use case parameter of 5 logins delivers a FAR of 21% and a FRR of 36%. For this experiment, the model showed certain resilience after the password change. However, the overall performance decreases in comparison to the original set-up of the trained model. To answer **RQ-B**, we should take into account that for the real-world applications, the most recent login sessions would include user behavior with the new password, further experiments on this question are proposed as future work.

Notice that combining the insights from experiment 1, 2 and 3 we have promising results towards learning accurately behavioral patterns instead of the user interaction(i.e user password) with few training sessions, thus, addressing partially the main **RQ**.

Finally, the *fourth experiment* is related to the production case, where no direct attack data is available, but only logins from other users. The behavior is quite similar to the second experiment, where for the FAR there is almost no difference, but for the FRR the $n = 5$ and $n = 7$ are specially close to each other, with a difference of 1.5%; the advantage of using fewer logins outweights the improvement on FRR for $n = 7$. The values of 11% and 18% show that in this simulation of a real environment, the system is capable of distinguishing between users without attackers data for training.

2) Medium Scale Pilot Dataset (users=89): Next, we tested our Random Forest model in the Medium Scale Pilot Dataset (MSPD) in order to check the scalability of our approach. As the experiments 2 and 4 are related to real training and attack scenarios, this dataset was only evaluated in those scenarios. The FAR and FRR results for different numbers of log-in sessions used for training each user' model are shown in figure 5. At this point, the similarities in the test performance between MSPD and SSPD become evident. In fact, we observe the optimal threshold for MSPD for the two experiments evaluated is around 80-85% for 3, 5 and 7 training sessions (fig. 5), which corroborates the optimal threshold found for SSPD.

From the *second experiment* (fig. 5a), training each user model with 5 logins seams to be enough to maximize their FAR and FRR at the threshold of 0.85, being 24% and 11% respectively, without falling to a FRR of 19% as in the case of 3 log-in sessions and being significantly close to the FAR of 23% and the FRR of 8% resulting from training with 7 log-in sessions. From this analysis, we can get closer to answer **RQ-C** given that it is reasonable to evaluate our proposed framework scalability to thousands of users having 5 log-in sessions as hinted by our previous results. Therefore, the tendencies shown by the experiments with SSPD and MSPD made us go forward into checking if our framework can have a similar performance even in real production environments.

C. Scalability

In order to implement our approach in a production environment some scalability aspects must be considered. For example, execution times and model's sizes become critical. In our proposed architecture, one of the highlights is the fact, that we need only 5 logins per user to be able to identify legitimate users; however, for every user we need to create and train a different model. For SSPD (21 users) data preparation takes approximately 2.67s and training time per user is around 0.094s; for MSPD (89 users) data preparation takes approximately 6.15s and the training time is approximately the same as in the first dataset (0.095s). This result is expected because regardless of the number of users, each model is trained with 5 login sessions (augmented to 50) from the legitimate user and another 50 randomly selected logins from different users as the attackers. The execution time to classify a new login is equally independent of the number of users and around 0.03s. Finally, regarding storage, each model requires approximately 100KB. So in a scenario with 1 million users, the needed storage would be of 100 GB, which is a reasonable storage requirement for such a highly demanded system.

D. Production Environment Evaluation

We note that there are several challenges to correctly evaluate the 2000 users and 14.7k sessions of data captured from the banking domain. Since data is coming from a production environment, we have no control over several factors that may impact the results. First, we have no information on how many of the sessions were in fact performed by the same users. Although we estimate that attacks are relatively rare, there could be malicious login attempts that are unknown to us. Also some accounts might be shared by multiple people (for instance corporate accounts). Last, in this scenario we cannot force users to avoid the use of keyboard actions (such as tabs and enters) and some users might store credentials in password managers.

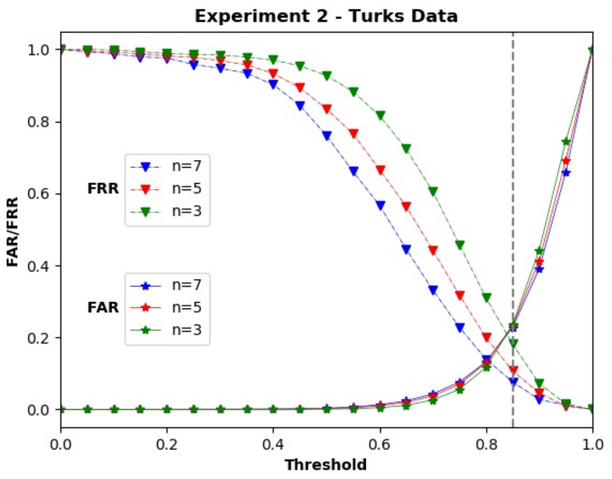
For the purpose of our evaluation we assume that all sessions logins are legitimate. Note that in this scenario the number of legitimate sessions for testing is vastly unbalanced against potential attacks, since we have a mean of 2 sessions left to test after the model has been trained with 5, but for each user we have around 14.000 simulated attacks (logins performed by other users).

In order to handle the test data imbalance, we chose the following strategy. First, we restricted the number of attack sessions per user to 1000 randomly selected logins (out of 14.7k from other users). On the other hand, in order to increase the testing surface on the legitimate class, we assembled several *parallel worlds*. These parallel worlds work by randomly selecting 5 sessions out of the available sessions of a given user to train the model, and then evaluating it against the remaining sessions. Since on average there are 7 sessions per user, this gives $\binom{7}{5} = 21$ possible choices of parallel worlds. We chose 10 parallel worlds, which transformed the system from having only 2 legitimate sessions per user to test (on average) into having $2 \times 10 = 20$ sessions to test.

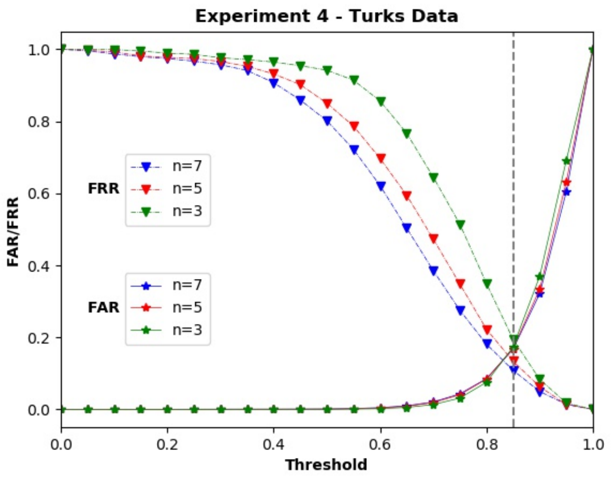
Given this evaluation setting, we make the following preliminary observations. First, the FRR, which was of ca. 13% in the controlled pilots, increases up to 25% (1 out of 4 legitimate sessions) for $n = 5$ logins and a threshold of 0.85. We believe that this can be explained by the fact that some users used shortcuts (Tabs,enter) and password managers, making sessions much shorter in duration and number of events compared to the controlled experiments. On the other hand, as discussed above, some users might be sharing accounts, and there could be unlabeled attacks in the collected data.

Given this evaluation setting, we make the following preliminary observations. First, the FRR, which was of ca. 13% in the controlled pilots, increases up to 25% (1 out of 4 legitimate sessions) for $n = 5$ logins and a threshold of 0.85. We believe that this can be explained by the fact that some users used shortcuts (Tabs,enter) and password managers, making sessions much shorter in duration and number of events compared to the controlled experiments. On the other hand, as discussed above, some users might be sharing accounts, and there could be unlabeled attacks in the collected data.

The FAR however remains interestingly similar to the one



(a) Experiment 2 (Real Attack) results for different number of sessions(n) in training.



(b) Experiment 4 (Production Environment) results for different number of sessions(n) in training.

Fig. 5: FAR versus FRR plot for different experiments when they are **evaluated on the pilot dataset (users=89)**. Notice that at a threshold cut-off point of 0.85 both the FAR and FRR are simultaneously minimized.

of experiment 4 in the controlled pilots, reaching 23% (1 out of 4 attacks is undetected). Note that the FAR of experiment 4 seems to be an upper bound to the FAR of the more realistic attack of experiment 2, but given that they are close, we believe this constitutes a good approximation.

Practicality: In sum, although the model performs somewhat worse in production data for the same 0.85 threshold, the approximate accuracy of 75% for a model based on 5 logins is still reasonably practical in light of the risk vs. usability discussion of the introduction, that we recap here. First, suspicious logins can be challenged by means of 2-factor authentication, and not necessarily manually. This creates friction once every 4th login attempt in average, but also possibly prevents 3 out 4 attacks. Second, the score of our model can be merged with other risk-based models, such as context (IP, browser, login time etc.) as discussed in [30]. Third, although gathering more logins before building a model can potentially yield a more accurate prediction, the risk of an attack being unnoticed and even poisoning data for training increases with every extra login. We plan to further investigate available production data, which is being collected daily, in order to give deeper insights into the generalization of our approach to less controlled scenarios. We will also evaluate our approach on environments that enforce more restrictions on user behaviour (and thus force users to have longer interactions at login time as we did in our pilots).

V. Related Work

Keystroke Dynamics: Banerjee et al. [6] and more recently Raul et al. [25] performed comprehensive reviews of keystroke authentication studies. These reviews mention that n-graphs, gave better classification results for several studies, and therefore these keyboard features were included in our model. Additionally, studies that used random forest as classification models, report results as low as 3.2 FAR and 5.5 FRR for static text conditions, however the amount of data required for classification in the testing phase is still large, with more than 8000 keystroke samples per user [6]. An interesting example is the model proposed by Kim et al. [17], which achieved an EER score of 6.7%, using an RNN with together with an anomaly detection system. The model required only 10 keystrokes per user to perform a classification in the testing phase, however, they still required a set of 500 keystrokes per user for training. Reviews also mention that environmental factors, such as the type of keyboard [29] or behavioral changes over longer periods of time [14] can increase user's variance. The model we propose, takes into account a small number of behavioral records, it would be interesting to explore methods to increase adaptability to such changes in time.

Mouse Dynamics: Diverse methods to extract features from free mouse movements have been proposed. Some authors classify mouse features by registering: distance, action type, frequency, duration and direction of raw mouse events and group them into sessions of a predetermined time window. Studies that implement this approach achieved average EERs of 2.46% [2] and 3.37% [24]. Others proposed to aggregate mouse events occurring between two clicks. For instance, Gamboa et al. analyzed statistical features of the temporal-spatial information between clicks achieving EERs of 2% [12]. The idea of segmenting the movement into metrics that describe a curve has been explored by [28] and [39]. In the proposed model, we process information from both the trajectories and the temporal-spatial characteristics of raw mouse data to extract user's behavioral information.

Mouse biometric classification still demands large amounts of training data. For instance, Zheng et al., used a training dataset consisting of 12500 clicks, with a 0.5 threshold they obtained classification results of 0.86% FRR and 2.96% FAR, they also tested with a smaller training dataset of 500 clicks, and obtained accuracies of 4.57% FRR and 18.79% FAR [39]. This study shows that reducing the training dataset is a challenging problem, since it can rapidly decrease accuracy. Jorgensen et al., stated that most mouse dynamics studies require long data collection sessions to achieve acceptable authentication accuracy [16], which could be impractical for some real world applications.

Multimodal Biometrics with Keystroke and Mouse Features

a) Constrained Tasks: Neha et al. [21] performed a study with both static login and continuous authentication sessions for 60 users. For static authentication, they achieved an accuracy of 95.66% with J48 classification algorithm. However, 50 login entries for each of user were required for the training phase, using a fixed password for login. In this study the multimodal fusion was performed at the decision-level.

In a preliminary study, Khan et al. [4] investigated biometrics of website logins. The training data-set consisted of one user with 65 legitimate dummy website logins and 37 attacks. The highest average accuracy achieved was 97.3% with SVM, using 52 legitimate logins for training.

We note that the available studies do not analyze the problem of static user authentication under real-world constraints such as limited amount of user data and limited information on attacks, it was also found that scalability in production environments was not evaluated.

b) Unconstrained Tasks: In contrast, in a completely uncontrolled setting, Modal et al. [20] required 471 user actions to detect an impostor with 62.2% accuracy, with 25 subjects. Traore et al. [33], simulated a social network website, and performed an experiment with 12 subjects for 8 weeks, that resulted in a EER of 8.21%, they used separate bayesian models for keyboard and mouse, and then fused the scores, a minimum of 200 keystroke records and 2500 mouse dynamics were required for training.

Bailey et al. [5], combined keyboard, mouse and GUI interactions; users performed internet based research tasks and wrote reports. 31 users were tested, samples consisted in 10 minutes sliding windows. With approximately 7 samples for training and 3 for testing, using SVM and Bayes classification algorithms, a final outcome of 2.1% FAR and 2.24% FRR was obtained.

Fridman et al. [11] collected data of 67 users for approximately 56 hours. The study collected mouse, keyboard and stylometry features from article writing and web-searching tasks. They used Naive Bayes classifiers for each feature and binary local decisions were fused for a final classification. Training was performed with 60% of gathered data. Overall they achieved 0.004% FAR and 0.01% FRR after testing with 30s of user interaction. They also found that the highest relative contribution to the classification was related to mouse curve features, which were included in our model.

Other continuous authentication studies were also compared with our study, we found that, given the unconstrained tasks that these studies address, the amount of training data required for the models is much larger, which partially explains why the reported accuracies are higher.

A summary of comprehensive behavioral biometrics studies with mouse and/or keyboard is found on Table VI. We compared samples required for training and testing, number of users in the data-set and model accuracies.

TABLE VI: Related Work - Mouse and Keyboard Biometrics

Study	**Input**	**Training**	**Test**	**FAR**	**FRR**	**No. Users**
Kim et al.[1]	K	500KS	10KS	6.7%[3]	6.7%[3]	120
Zheng et al.[1]	M	12500C 500C	500C 1C	0.86% 4.57%	2.96% 18.79%	30P & 1000F
Traore et al.[1]	K,M	200KS, 2500ME	-	8.21%[3]	8.21%[3]	12
Bailey et al.[1]	K,M, GUI	70 min	30 min	2.10%	2.24%	31
Fridman et al.[1]	K,M, Styl.	33.6 hours	30 sec	0.04%	0.01%	67
Neha et al. [2]	K,M	50 logins	1 login	0.89%	1.20%	60
Ours[2]	K,M	**7 logins** **5 logins** **3 logins**	1 login	22.77% 23.34% 23.51%	7.65% 10.73% 18.17%	109P & 2000F

[1] Studies with unconstrained tasks.
[2] Studies with login tasks.
[3] Values obtained from reported EER.
Abbreviations: K- Keyboard, M- Mouse, KS- Keystrokes, C- Clicks, ME- Mouse Events, GUI- GUI interactions, Styl.- Stylometry, P- Pilot Controlled Dataset, F- Field or Production Dataset.

To the best of our knowledge our study is the first to address static behavioral biometric authentication with a highly restricted amount of training samples. Moreover, compared to previous studies, we present a new approach that considers user's behavioral history. Finally, the evaluation of our study was performed with a large number of samples for both controlled and production environments, which also stands out from previous studies in the field.

VI. Conclusions

In this work we have presented a novel approach for risk-based authentication using behavioral biometrics that aims at having practical accuracy while needing few user sessions to train. We have evaluated our approach in three independent datasets ranging from dozens to thousands of users and shown that the approach is both reasonably accurate and scalable for a training set as small as 5 logins. In the future we plan to evaluate the approach on more available production data as well as to study the consistency of the proposed model against changes in end-user devices as well as login website layout changes.

Acknowledgements

We would like to thank Julián Uribe and his team at AppGate for their support and feedback during this research.

References

[1] A. A. Ahmed and I. Traore, "Biometric recognition based on free-text keystroke dynamics," *IEEE Transactions on Cybernetics*, vol. 44, no. 4, pp. 458–472, 2014.

[2] A. A. E. Ahmed and I. Traore, "A new biometric technology based on mouse dynamics," *IEEE Transactions on dependable and secure computing*, vol. 4, no. 3, pp. 165–179, 2007.

[3] A. Antoniou, A. Storkey, and H. Edwards, "Data augmentation generative adversarial networks," *arXiv preprint arXiv:1711.04340*, 2017.

[4] F. Arif Khan, S. Kunhambu, and K. C. G, "Behavioral biometrics and machine learning to secure website logins," in *Security in Computing and Communications*. Springer Singapore, 2019, vol. 969, pp. 667–677.

[5] K. O. Bailey, J. S. Okolica, and G. L. Peterson, "User identification and authentication using multi-modal behavioral biometrics," vol. 43, pp. 77–89, 2014.

[6] S. P. Banerjee and D. Woodard, "Biometric authentication and identification using keystroke dynamics: A survey," vol. 7, no. 1, pp. 116–139, 2012.

[7] J. Bonneau, C. Herley, P. C. v. Oorschot, and F. Stajano, "The quest to replace passwords: a framework for comparative evaluation of Web authentication schemes," University of Cambridge, Computer Laboratory, Tech. Rep. UCAM-CL-TR-817, Mar. 2012. [Online]. Available: https://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-817.pdf

[8] L. Chen, Y. Zhong, and D. Zhang, "Continuous authentication based on user interaction behavior," *7th International Symposium on Digital Forensics and Security, ISDFS 2019*, no. 2017, pp. 1–6, 2019.

[9] Z. Chen, Y. Fu, Y. Zhang, Y.-G. Jiang, X. Xue, and L. Sigal, "Semantic feature augmentation in few-shot learning," *arXiv preprint arXiv:1804.05298*, vol. 86, p. 89, 2018.

[10] T. G. Dietterich, "Machine learning for sequential data: A review," in *Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR)*. Springer, 2002, pp. 15–30.

[11] L. Fridman, A. Stolerman, S. Acharya, P. Brennan, P. Juola, R. Greenstadt, and M. Kam, "Multi-modal decision fusion for continuous authentication," vol. 41, pp. 142–156, 2015.

[12] H. Gamboa and A. Fred, "An identity authentication system based on human computer interaction behaviour." in *Proceedings of the 3rd International Workshop on Pattern Recognition in Information Systems*, 01 2003, pp. 46–55.

[13] H. Gao, Z. Shou, A. Zareian, H. Zhang, and S.-F. Chang, "Low-shot learning via covariance-preserving adversarial augmentation networks," in *Advances in Neural Information Processing Systems*, 2018, pp. 975–985.

[14] N. Harun, W. L. Woo, and S. S. Dlay, "Performance of keystroke biometrics authentication system using artificial neural network (ANN) and distance classifier method," in *International Conference on Computer and Communication Engineering (ICCCE'10)*. IEEE, 2010, pp. 1–6.

[15] A. K. Jain, A. A. Ross, and K. Nandakumar, *Introduction to Biometrics*. Springer US, 2011.

[16] Z. Jorgensen and T. Yu, "On mouse dynamics as a behavioral biometric for authentication," in *Proceedings of the 6th ACM Symposium on Information, Computer and Communications Security - ASIACCS '11*. ACM Press, 2017, p. 476.

[17] J. Kim and P. Kang, "Recurrent neural network-based user authentication for freely typed keystroke data," jun 2018.

[18] C. X. Ling and C. Li, "Data mining for direct marketing: Problems and solutions." in *Kdd*, vol. 98, 1998, pp. 73–79.

[19] M. Misbahuddin, B. S. Bindhumadhava, and B. Dheeptha, "Design of a risk based authentication system using machine learning techniques," *2017 IEEE SmartWorld, Ubiquitous Intelligence Computing, Advanced Trusted Computed, Scalable Computing Communications, Cloud Big Data Computing, Internet of People and Smart City Innovation*, pp. 1–6, 2017.

[20] S. Mondal and P. Bours, "Combining keystroke and mouse dynamics for continuous user authentication and identification," in *2016 IEEE International Conference on Identity, Security and Behavior Analysis (ISBA)*. IEEE, 2016, pp. 1–8.

[21] Neha and K. Chatterjee, "Continuous User Authentication System: A Risk Analysis Based Approach," *Wireless Personal Communications*, vol. 108, no. 1, pp. 281–295, 2019.

[22] N. Nishiuchi and S. Aoki, "Study on soft behavioural biometrics to predict consumer's interest level using web access log," *International Journal of Biometrics*, vol. 11, no. 3, pp. 243–256, 2019.

[23] Y. Patel, "The state of play – traditional versus behavioural biometrics," *Biometric Technology Today*, vol. 2019, no. 2, pp. 5–7, feb 2019.

[24] M. Pusara and C. E. Brodley, "User re-authentication via mouse movements," in *Proceedings of the 2004 ACM workshop on Visualization and data mining for computer security - VizSEC/DMSEC '04*. ACM Press, p. 1.

[25] N. Raul, R. Shankarmani, and P. Joshi, "A Comprehensive Review of Keystroke Dynamics-Based Authentication Mechanism," 2019, pp. 149–162.

[26] A. A. Ross, K. Nandakumar, and A. K. Jain, *Handbook of Multibiometrics*, ser. International Series on Biometrics. Kluwer Academic Publishers, 2006, vol. 6.

[27] B. Ross, C. Jackson, N. Miyake, D. Boneh, and J. C. Mitchell, "Stronger password authentication using browser extensions," p. 15, 2005.

[28] D. A. Schulz, "Mouse curve biometrics," in *2006 Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference*. IEEE, 2006, pp. 1–6.

[29] P. Smriti, S. Srivastava, and S. Singh, "Keyboard invariant biometric authentication," in *2018 4th International Conference on Computational Intelligence & Communication Technology (CICT)*. IEEE, 2018, pp. 1–6.

[30] J. Solano, L. Camacho, A. Correa, C. Deiro, J. Vargas, and M. Ochoa, "Risk-based static authentication in web applications with behavioral biometrics and session context analytics," in *Security in Machine Learning and its Applications (SiMLA) workshop co-located with Applied Cryptography and Network Security (ACNS)*, J. Zhou, R. Deng, Z. Li, S. Majumdar, W. Meng, L. Wang, and K. Zhang, Eds. Springer International Publishing, 2019.

[31] K. Thomas, F. Li, A. Zand, J. Barrett, J. Ranieri, L. Invernizzi, Y. Markov, O. Comanescu, V. Eranti, A. Moscicki, D. Margolis, V. Paxson, and E. Bursztein, "Data breaches, phishing, or malware?: Understanding the risks of stolen credentials," in *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '17. New York, NY, USA: ACM, 2017, pp. 1421–1434.

[32] I. Traore, *Continuous Authentication Using Biometrics: Data, Models, and Metrics: Data, Models, and Metrics*. IGI Global, 2011.

[33] I. Traore, I. Woungang, M. S. Obaidat, Y. Nakkabi, and I. Lai, "Combining mouse and keystroke dynamics biometrics for risk-based authentication in web environments," in *2012 Fourth International Conference on Digital Home*. IEEE, 2012, pp. 138–145.

[34] P. Waggett, *Risk-based Authentication: Biometrics' Brave New World*. Elsevier Advanced Technology, 2016.

[35] H. Wang, J. Gu, and S. Wang, "An effective intrusion detection framework based on svm with feature augmentation," *Knowledge-Based Systems*, vol. 136, pp. 130–139, 2017.

[36] Y. Wang and Q. Yao, "Few-shot learning: A survey," *arXiv preprint arXiv:1904.05046*, 2019.

[37] R. V. Yampolskiy and V. Govindaraju, "Behavioural biometrics: a survey and classification," *International Journal of Biometrics*, vol. 1, no. 1, pp. 81–113, 2008.

[38] A. Zhao, G. Balakrishnan, F. Durand, J. V. Guttag, and A. V. Dalca, "Data augmentation using learned transformations for one-shot medical image segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2019, pp. 8543–8553.

[39] N. Zheng, A. Paloski, and H. Wang, "An efficient user verification system via mouse movements," in *Proceedings of the 18th ACM conference on Computer and communications security*. ACM, 2011, pp. 139–150.