Dynamic Sample Size Detection in Continuous Authentication using Sequential Sampling

Ahmed Awad E. Ahmed
Department of Electerical and Computer Engineering
University of Victoria
Victoria, BC, Canada
aahmed@ece.uvic.ca

Issa Traore
Department of Electerical and Computer Engineering
University of Victoria
Victoria, BC, Canada
itraore@ece.uvic.ca

ABSTRACT
Continuous Authentication (CA) departs from the traditional static authentication scheme by requiring the authentication process to occur multiple times throughout the entire logon session. One of the main objectives of the CA process is to detect session hijacking. An important requirement about designing or operating a CA system is the need to achieve the quickest detection while maintaining rates of missed and false detections to predetermined levels. We introduce in this paper a new approach for detection based on the sequential sampling theory that allows balancing appropriately between detection promptness and accuracy in CA systems. We study and illustrate the proposed approach using an existing mouse dynamics biometrics recognition model and corresponding sample experimental data.

Keywords
Continuous Authentication, Biometrics, Security Monitoring, Sequential Sampling, Mouse Dynamics.

1. INTRODUCTION
User authentication is the process of verifying whether the identity of a user is genuine prior to granting him access to resources or services in a computer system. Traditionally, authentication is performed statically at the point of entry to the system (e.g., login). A successful authentication, however, at the beginning of a session does not provide any remedy against the session being hijacked later by some malicious user. One of the solutions proposed to address this shortcoming is *continuous authentication (CA)*. CA consists of the process of positively verifying the identity of a user in a repeated manner throughout a computing session to ensure that the claimed identity (at login time) remains the same as the monitored one. In this case, the detection delay or length of an individual authentication period can be captured in terms of either the time or the amount of data involved. In this paper, we refer to the detection delay as the *time-to-detect (TTD)*. One of the important questions that will be raised while designing a CA system is when to stop data collection and detect or, in other words, what should be the detection delay? For some systems, the answer to the above questions can be simple if the data arrival rate is known in advance as well as the entropy of the collected data. A model can be tuned to obtain an acceptable level of error for an optimal TTD. For other systems such as mouse dynamics, the rate of data arrival is unknown and the effectiveness of the collected data is unpredictable. One user can take 5 minutes to produce 1000 actions that cover only a small part of the detectable factors, while another user can take double the time to produce a lower number of actions covering all of the detectable factors. For such systems, there is a risk of achieving poor accuracy if the aim is to minimize the TTD. In this paper, we present a new approach that can be used to solve this dilemma. The approach is based on the sequential sampling technique [12] and provides a systematic procedure to determine the time needed to make the decision. The technique assures a predetermined level of error and has the flexibility to tune it according to the planned security needs. Although, we illustrate and validate the approach using mouse dynamics biometric modality, we believe that the approach could readily be extended to some other behavioral biometric modalities such as keystroke dynamics. The remainder of the paper is structured as follows. In Section 2, we summarize and discuss related work. In Section 3, we provide some background on sequential sampling technique and summarize an existing mouse dynamics biometric model and dataset used in our study. In Section 4, we introduce our technique for dynamic sample size detection based on sequential sampling. In Section 5, we present the experimental evaluation of the proposed technique. Finally, in Section 6, we make some concluding remarks and discuss future work.

ACSAC'11, December 5–9, 2011, Orlando, Florida USA.

2. RELATED WORKS
In the last decade, continuous authentication has been dealt with *indirectly* in various areas including user command sequence monitoring [7], free-text detection of keystrokes dynamics biometrics [3,4], and mouse dynamics biometrics recognition [1,9]. The basic assumption underlying such systems is that illegitimate activity can be discriminated from normal user behavior when based on human-computer interactions. For instance, in the case of command string monitoring the recognition task consists of classifying sequences, or blocks, of user commands as pertaining to "self" (i.e. legitimate user), or "nonself" (i.e. unauthorized user) [7].

CA has also been directly the focus of several proposals in the recent literature. Ikehara and Crosby proposed a continuous identity authentication system (CIAS) that uses multimodal sensors to monitor and protect access to critical computer systems [5]. The proposed system is intended for the protection of critical systems with a limited number of users (e.g., 1 to 3 users). The multimodal sensors consist of a set of passive sensors built into a computer mouse measuring the pressures applied to the mouse during clicking when performing a task of varying difficulty. According to Ikehara and Crosby, a pilot study involving six participants yielded 95% accuracy after two clicks when using discriminant analysis.

Sim *et al*. proposed a multimodal CA system that combines a digital camera-based facial recognition scheme with a mouse-based fingerprint recognition scheme [10]. The system involves a feedback mechanism into the operating system that automatically locks up the computer by delaying processes or suspending them entirely. The mouse sensor is a modified computer mouse that incorporates an optical fingerprint scanner at the place where the user would position their thumb when using the mouse. The authors also proposed a holistic fusion technique using Hidden Markov Model (HMM) that integrates the face and fingerprint across modalities and time for CA. It was claimed that the proposed fusion scheme captures more suitably the characteristics of CA systems than traditional ones. Azzini and Marrara proposed a multimodal biometric CA system that combines face and fingerprint recognitions using a fuzzy controller [2]. After the initial authentication, the user identity is continuously checked on the sole basis of the facial recognition, until the computed score falls below a certain threshold, at which point fingerprint recognition is triggered.

Although the need for balancing the goals of detection promptness and detection accuracy has largely been recognized in the CA literature (including the above proposals), we have not come across any work that directly addresses this problem from a CA perspective. This issue, however, has already been tackled in the intrusion detection field as a sequential decision problem from different angles. In [6], Jung *et al*. introduced an approach for accurate portscans detection with minimal delay based on sequential sampling theory. In the proposed approach, accesses to local IP addresses are modeled as a random walk on one of two stochastic processes corresponding to traffic originating either from benign hosts or scanners, respectively. In [11], Tartakovsky *et al*. used sequential change-point detection theory for timely and accurate detection of abrupt changes in the network traffic, which potentially may be linked to attacks such as denial-of-service or portscans. While sequential hypothesis testing allows differentiating between normal and malicious behaviors, sequential change-point detection determines the specific point where normal behavior starts becoming malicious. Although both types of problems are relevant to continuous authentication, our primary focus in this paper is on the former. In this regard our approach is closely related to the work of Jung *et al*., although we target different types of attacks and use different types of data.

3. BACKGROUND

3.1 On Mouse Dynamics Biometric Analysis

Mouse dynamics correspond to the actions generated by the mouse input device for a specific user while interacting with a graphical user interface. Many characteristics of the mouse movement can be used for biometrics analysis. In [1], the following characteristics for each mouse movement are considered: Type of action, Traveled distance (in pixels), Elapsed Time (in seconds), and Movement Direction. Three types of actions are considered, namely: point-and-click (PC), drag-and-drop (DD), and regular mouse movement (MM). The movement direction is determined based on the angle of the movement. Eight directions, numbered from 1 to 8, are considered. Each direction covers an area of 45 degrees moving clockwise from the forward direction. In [1], the following seven biometric factors were extracted from the raw mouse data and used for biometric identification:

- The Movement Speed compared to Traveled Distance (denoted MSD) factor: computed by approximating the raw mouse data to a curve using Neural Networks.
- The Average Movement Speed per Movement Direction (denoted MDA) histogram.
- The Movement Direction Histogram (MDH): corresponds to the distribution of actions in each of the eight directions considered.
- The Average Movement Speed per Type of Action (denoted ATA).
- The Action Type Histogram (ATH): corresponds to the distribution of the performed actions over the different types of actions considered.
- The Traveled Distance Histogram (TDH): represents the distribution of the number of actions performed by the user within different distance ranges.
- The Movement elapsed Time Histogram (denoted MTH): represents the distribution of the number of actions performed by the user within different time ranges.

To build a biometric profile for the user, 39 features, each corresponding to a point judiciously chosen from the above curve and histograms were used. To compare some monitored behavior against a reference profile, a feed-forward neural network consisting of three layers was used.

The neural network takes as input the 39-dimensional feature vector and computes as output a percentage referred to as the confidence ratio (CR), which represents the degree of similarity of the compared behaviors. During enrolment, the above same neural network architecture is trained for each user, and a profile is created for each user based on the weights of the neural network. During the detection mode, the weights of the neural network are restored from the profile database and the newly collected mouse samples are applied to the network. The output is then compared to a preset threshold in order to decide on the similarity or dissimilarity of the compared behaviors. The above scheme was evaluated through free and controlled experiments involving 22 and 7 users, respectively. In the free experiment, the participants were given an individual choice of operating conditions and applications. Consequently, data was collected using a variety of hardware and software systems. The experiment ran for 9 weeks, and allowed the collection of 284 hours of raw mouse data over 998 sessions, with an average of 45 sessions per participant. By analyzing the data, an Equal Error Rate of 2.46% was obtained when setting the CR threshold to 50%. The objective of the controlled experiment was to study the impact of confounding

factors involved in the free experiment. Participants were asked to perform the same set of actions using a customized application deployed on the same operating environment (i.e. same machine, mouse and operating system). Data was collected over 63 sessions, 9 sessions per participants, and each session consisting of 100 mouse actions. A FAR of 2.245% and a FRR of 0.898% was achieved, when the threshold was set to CR=50%.

3.2 On Sequential Sampling Technique

To our knowledge, previous CA models were all based on classical sampling where the sample data is predetermined.

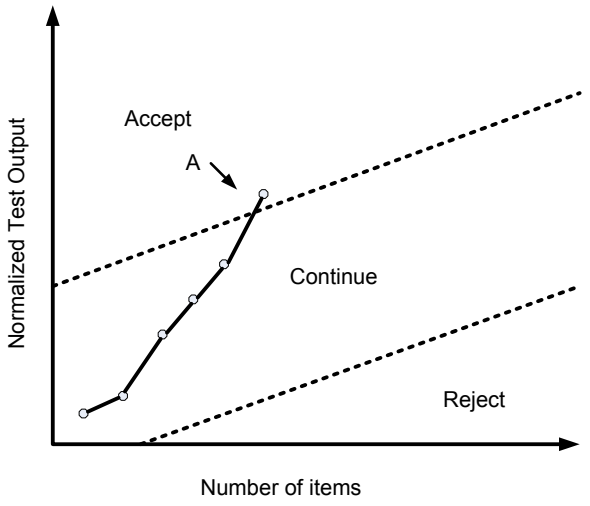

Figure 1. Sequential sampling plan.

In classical statistical techniques, the sample size is set according to the expected level of features prevalence over the data and the degree of precision required. At the end of the data collection process, a decision will be made to accept or reject the null hypothesis. The sequential sampling technique, originally proposed by Wald, combines (simultaneously) data collection and data analysis into a single process (sampling plan) [12]. In this approach, the sample size is not fixed — the size increases over time until a decision is made. Hypothesis testing can occur every time data is available (item by item unit-step sequential sampling) or over incremental batches of data (group sequential sampling). Applying the sequential sampling technique requires building a sampling plan consisting of three areas (*Accept*, *Continue*, and *Reject*). A test will be conducted on the data collected up to the current sampling period and, according to the result of this test, the decision will be made. Figure 1 shows the sampling plan. The plan is partitioned into three regions. The x-axis represents the number of items included in the evaluation, and the y-axis represents the output of the test performed on the sampled data. Each time a sample is tested, a point will be placed on the plan and the status will be evaluated according to the location of the point on the plan. The sampling process will continue (adding more items to the sample and testing again) if the point falls in the *Continue* region, and will stop if it falls in the *Accept* or *Reject* regions.

The sampling plan is built according to the following parameters:

α = acceptable type I error or false rejection rate (FRR).

β = acceptable type II error of false acceptance rate (FAR).

p_1 = lower threshold limit (as proportion).

p_2 = higher threshold limit (as proportion).

The two lines representing the borders between the *Accept*, *Continue*, and *Reject* areas can be described as:

$$N_{CR} = -h_1 + sn \quad \text{(Rejection line)}$$

$$N_{CR} = h_2 + sn \quad \text{(Acceptance line)}$$

where n is the test number (x-axis) and N_{CR} is the normalized value of the confidence ratio (CR) calculated for test number n; $N_{CR} = CR_n \times n/100$.

Parameters h_1, h_2 and s can be computed as follows:

$$h_1 = \left(\ln \frac{1-\alpha}{\beta} \right) \Big/ \left(\ln \frac{p_2(1-p_1)}{p_1(1-p_2)} \right);$$

$$h_2 = \left(\ln \frac{1-\beta}{\alpha} \right) \Big/ \left(\ln \frac{p_2(1-p_1)}{p_1(1-p_2)} \right);$$

$$s = \left(\ln \frac{1-p_1}{1-p_2} \right) \Big/ \left(\ln \frac{p_2(1-p_1)}{p_1(1-p_2)} \right).$$

On the sampling plan, the variables h_1 and h_2 affect the distance between the two border lines while s represents their slope.

4. DYNAMIC SAMPLE SIZE DETERMINATION

We present in this section our approach for dynamic sample size determination using sequential sampling. Before describing the technique, let us try to evaluate how the system accuracy is affected with respect to the time needed for the data to arrive. Figure 2 shows how CR changes as the number of actions included in the evaluation increases. Each point on the curve represents the CR computed by processing $n \times N$ mouse actions (from the dataset collected in [1]), where n is the test number and N is the number of actions in a sampling period. Exactly 50 tests were conducted; therefore the last point on the curve represents the CR resulting from processing 2500 actions ($n = 50$ tests, $N = 50$ actions).

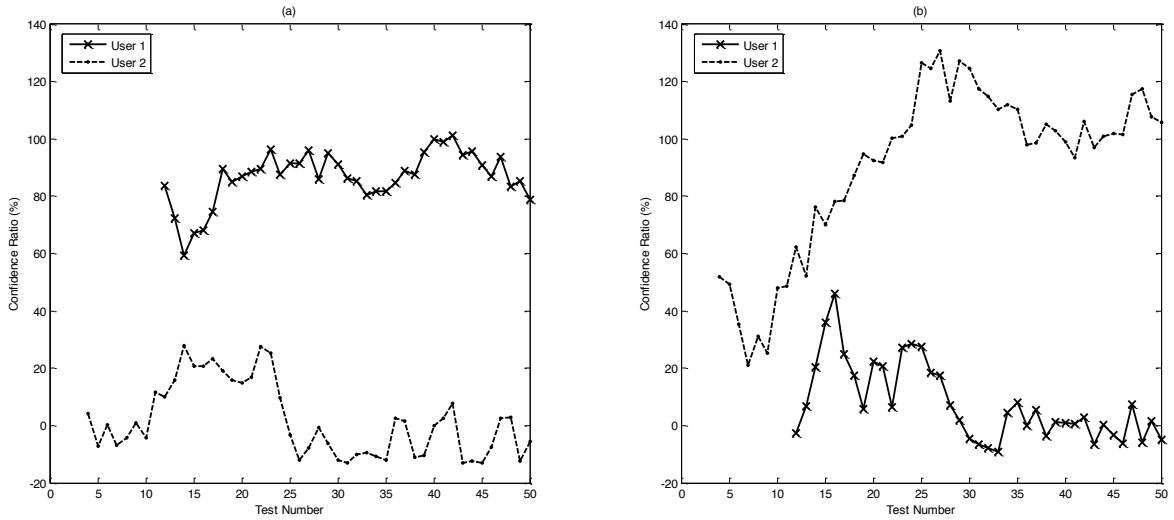

Figure 2. Evolution of CR as the number of actions increases; earlier test points are missing because the sample size is too small to compute the CR. a. Confidence ratio resulted from testing user 1's and 2's data against User 1's reference signature. b. Confidence ratio resulted from testing User 1's and 2's data against User 2's reference signature.

Figure 2-a shows two curves for two different users — User 1 and User 2 — both compared to User 1's reference signature. Figure 2-b shows the results when the data is compared to User 2's reference signature. Note that for both users the earlier tests failed to compute the CR because of the low number of data available; this explains why corresponding test points are missing. Considering a simple detection algorithm that will base its result on whether the resulting CR is below or above a threshold limit of say CR= 50%, we notice that three out of the four curves give a correct result near the beginning of testing when only a low number of actions are included. We also notice an increasing pattern in the CR as the number of action increases when the session belongs to the reference user, and a decreasing pattern when the current session is not for the legitimate user. As for the fourth curve representing User 2's results in Figure 2-b, there is a probability for false rejection if the decision is made based on the tests where $n < 10$ as the resulting CR will be lower than 50%. Figure 3 illustrates the use of the sequential sampling technique on the two users' data presented in Figure 2.

The CR calculated for each test will be normalized and plotted on the plan. As the number of actions increases, the algorithm will give more weight to the calculated CR and at the same time will increase the limit required to be passed to accept and decrease the one needed to reject. The decision is made as soon as the normalized CR value crosses one of the limits (example: point A on Figure 1); at that time, the sampling process will stop and a new sampling process will start for the data to come after. Note that the technique avoided the problem mentioned above regarding the evaluation of user 2's data on Figure 2-b before $n = 10$, which can lead to a false rejection. As shown in Figure 3-b, the technique waited until $n = 23$ to make the correct decision.

Table 1 demonstrates how the decisions are made for the two sampling processes presented in Figure 3-a. User 1's session is accepted when the normalized CR reaches 25.911, passing the 24.949 limit set by the algorithm to stop sampling at step number 27. The algorithm was able to classify and reject User 2's session at step number 25 when his normalized CR value reached -0.83718 , falling in the reject area.

Table 1. Data collected through the sampling process (depicted by Figure 3-a) and the decisions taken in reference to the rejection and acceptance limits. Earlier test points are missing because the sample size is too small to compute the CR.

Test Number	Number of Actions	Reject Limit	Accept Limit	User 1's Normalized CR	User 2's Normalized CR	Decision
1	50	-10.949	11.949	X	X	X
2	100	-10.449	12.449	X	X	X
3	150	-9.9494	12.949	X	X	X
4	200	-9.4494	13.449	X	0.16447	X
5	250	-8.9494	13.949	X	-0.36659	X
6	300	-8.4494	14.449	X	0.009224	X
7	350	-7.9494	14.949	X	-0.48361	X
8	400	-7.4494	15.449	X	-0.35372	X
9	450	-6.9494	15.949	X	0.080125	X
10	500	-6.4494	16.449	X	-0.43563	X
11	550	-5.9494	16.949	X	1.2663	X
12	600	-5.4494	17.449	10.043	1.203	X
13	650	-4.9494	17.949	9.4186	2.0601	X
14	700	-4.4494	18.449	8.3071	3.9064	X
15	750	-3.9494	18.949	10.059	3.109	X
16	800	-3.4494	19.449	10.898	3.302	X
17	850	-2.9494	19.949	12.66	3.9447	X
18	900	-2.4494	20.449	16.109	3.4102	X
19	950	-1.9494	20.949	16.14	3.0154	X
20	1000	-1.4494	21.449	17.371	2.9798	X
21	1150	-0.9494	21.949	18.611	3.509	X
22	1200	-0.4494	22.449	19.664	6.0904	X
23	1250	0.05059	22.949	22.137	5.781	X
24	1300	0.5506	23.449	21.046	2.3659	X
25	1350	1.0506	23.949	22.847	-0.83718	Reject User 2
26	1400	1.5506	24.449	23.806	X	X
27	1450	2.0506	24.949	25.911	X	Accept User 1

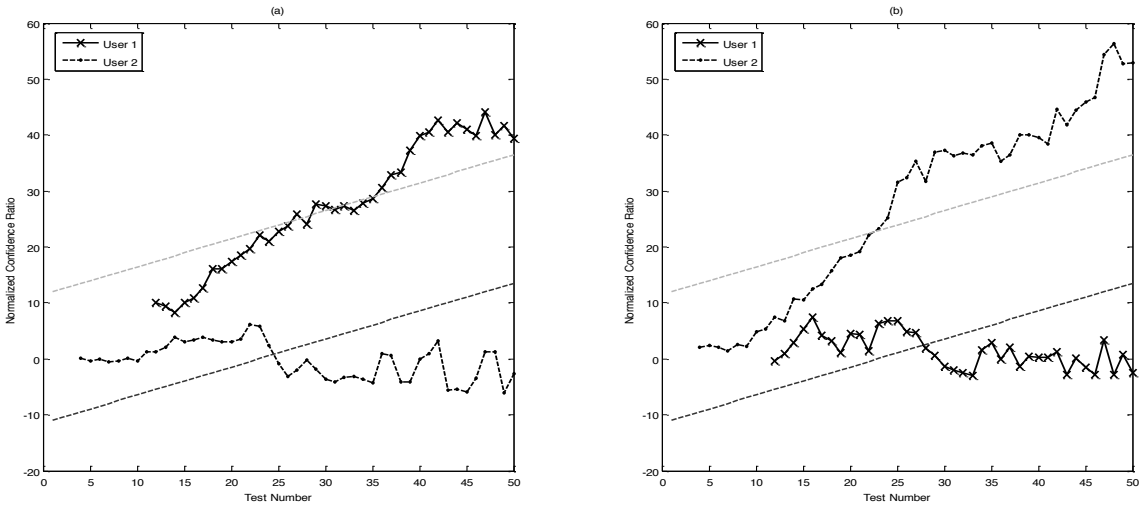

Figure 3. Sequential sampling plans showing: (a) test results for User 1 and User 2 when compared to User 1's reference signature; (b.) test results for User 1 and User 2 when compared to User 2's reference signature. Earlier test points are missing because the sample size is too small to compute the CR.

The sampling plan is established by choosing values for the parameters that properly describe the classification criteria and the acceptable level of error. For example, in our first test we chose $N = 50$, $\alpha = 0.01$, $\beta = 0.01$, $P_1 = 0.45$, and $P_2 = 0.55$. This means that the algorithm will accept the hypothesis that the user session is legitimate if the calculated CR passes the 55% threshold limit, and will reject the hypothesis indicating an intrusion if the CR value is lower than 45%. If the CR falls in the range between the two limits, a decision cannot be made and the sampling will continue until CR reaches one of the two limits. In addition, according to those parameters, there is a probability of 1% that the algorithm will misclassify a session (due to the fact that $\alpha = \beta = 0.01$).

5. EXPERIMENTAL EVALUATION

The sequential sampling technique described above can help in decreasing the time needed to make a decision about the provided session data; a factor that can be very critical in CA applications. The technique also can help in increasing the accuracy since a decision is made incrementally and based on a number of repetitive evaluations, not only a single one. Prior to the application of this technique, it is very important to select the proper sequential sampling parameters to ensure the achievement of the potential goal. The selection of these parameters can be made theoretically by setting the aimed values for the accuracy (FAR, FRR); however, it is not possible to set an expected value for the TTD. Because the sequential sampling process involves repetitive processing of the data, another important factor to be considered is the CPU usage. In order to study the effect of the sequential sampling parameters on the factors mentioned above, we conducted an experimental evaluation using the mouse dynamics data collected in [1]. We describe in this section the experimental method, and present and analyze the results.

5.1 Methods

Our evaluation of the processing requirement of the tests indicated a very high demand for processing power. Around 40 days are needed to finish processing the data for one test on an Intel dual Xeon machine with 2 GB of RAM. In order to avoid falling into this problem, we considered a small subset of the data where we randomly selected 5 users and included only 20,000 actions for each of them. Similar to [1] a one-hold-out cross-validation test was performed on the data, where 2000 user actions per user were used to calculate the users' signatures in each round. The rest of the data were used to simulate legitimate/attack sessions. In each round, one of the users was considered as an impostor, while the other four users played the role of insiders. A profile was built for an insider by using 2000 actions from their own data for positive training while 2000 actions from each of the three remaining insiders were used for negative training. FAR was computed by comparing impostor data against insiders' profiles, while FRR was computed by comparing each insider's profile against their own data. The main difference between this evaluation and the one performed in [1] is that the session size is not fixed. In this evaluation the decision is not made based on a specific number of actions in a session. Instead, the data is sequentially examined until the normalized CR reaches one of the decision limits. By then, a new session will start. With this limitation, it takes an average of 8 hours to process the data for each test with specific values for the sampling parameters and calculate the resulting factors.

5.2 Results

Figure 4 shows the results obtained from the tests for different values for the parameters N, α, β, P_1, and P_2. We started our evaluation by fixing the parameters α and β to 0.01 in order to study the effect of the other parameters N, P_1, and P_2. We considered five different pairs of values for (P_1, P_2), and for each pair we considered three different values for the sample size $N = 25, 50, 100$. So, in total, we conducted 15 different tests. Each point on the graphs represents a test (15 tests in total). For each test, we calculated the FAR, FRR, Mean TTD (denoted MTTD), expressed as the number of actions needed to make the final decision and the average CPU usage needed to process the data in each of the sampling tests. The aim was to lower these values as much as possible or to balance between them according to the prospective application.

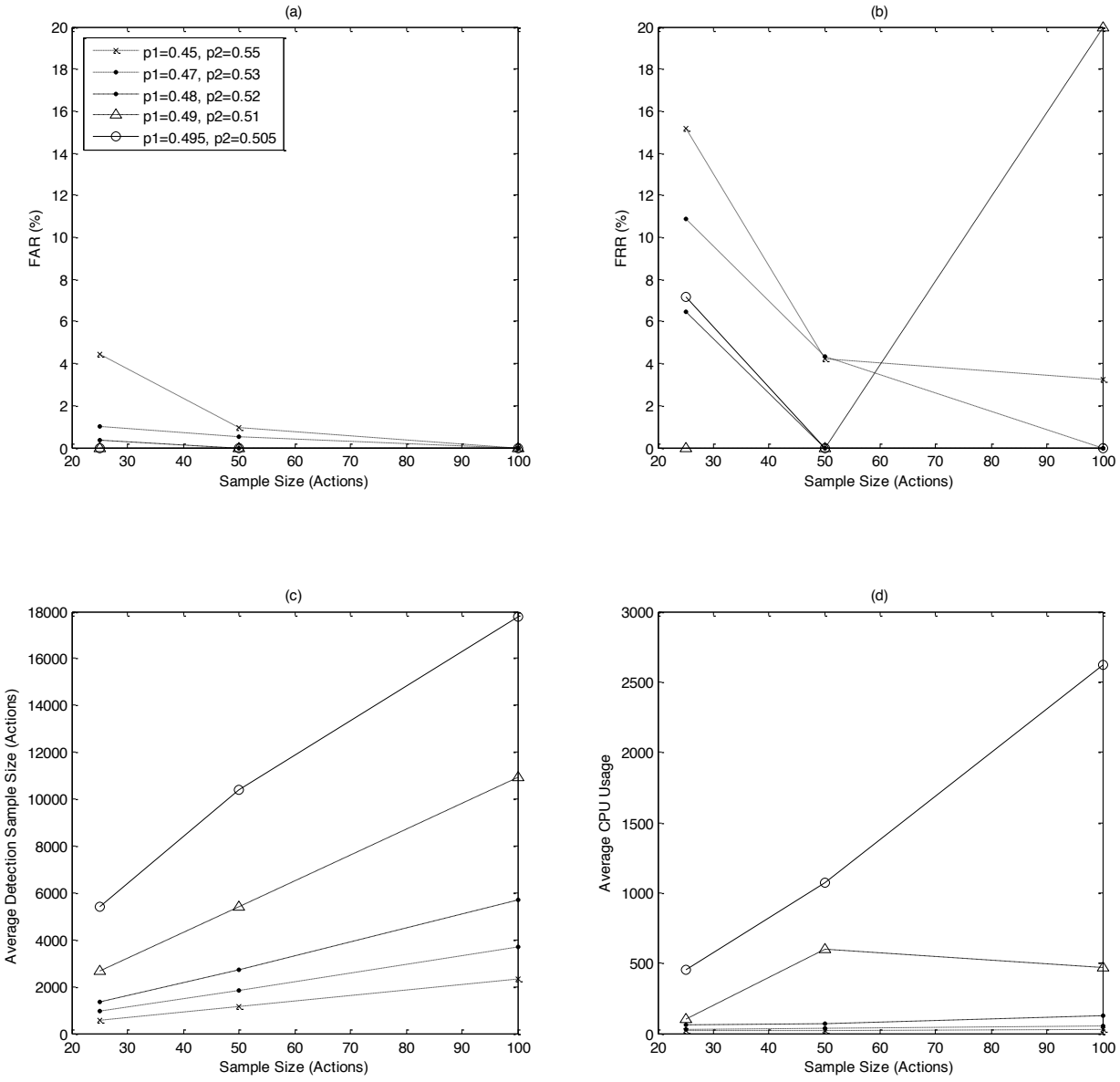

Figure 4. Tuning sequential sampling parameters by studying their effect on: (a) FAR, (b) FRR, (c) MTTD, and (d) CPU usage.

Looking at Figure 4-a, we notice a decreasing pattern for the FAR as the number of actions in the sampling window (i.e. parameter N) increases. We also note that the range that FAR swings in also decreases as the parameter N increases. The same comments apply to FRR (Figure 4-b), except for the test where ($P_1 = 0.49$, $P_2 = 0.51$) where FRR jumps from 0 to 20% as N increases from 50 to 100 actions. For all tests, the values of FRR are always greater than or equal to the corresponding values of FAR.

From Figure 4-c, we note that the MTTD (mean of the number of actions in all detection sessions) increases as the parameter N increases and as the difference between P_1 and P_2 decreases. Only 5 tests out of the 15 resulted in a MTTD lower than 2000 actions. The CPU usage (Figure 4-d) also increases as N increases, and becomes very high only for N greater than 25 actions and when the difference between the parameters P_1 and P_2 is 0.2 or lower.

Table 2 shows the resulting values for FAR, FRR, MTTD and CPU usage for the tests shown in Figure 4. The table also shows the minimum and maximum TTD obtained for all of the tests.

Through analysis of this table, and by considering the acceptable ranges for these values, we can select the tests that satisfy these limits. These values can change according to the requirements of the detection system. If the aim is to detect intrusions as soon as possible, then decreasing MTTD or Min TTD will be the goal, and if the aim is to increase the accuracy then lowering FAR and FRR is the goal. Ideally, we might also like to achieve the lowest possible CPU usage.

Table 2. FAR, FRR, MTTD, and CPU usage achieved for different values of the parameters P_1 and P_2 while setting α and β to 0.01 for N = 25, 50, and 100 actions. Highlighted rows are candidates for improvement towards achieving the selected criteria.

Test Number	Sample Size N (Actions)	P_1	P_2	α	β	FAR (%)	FRR (%)	Min TTD (Actions)	Max TTD (Actions)	Mean TTD (Actions)	CPU Usage (Seconds)
1a	25	0.45	0.55	0.01	0.01	0.04437	0.15172	75	1825	550.24	17.912
2a	25	0.47	0.53	0.01	0.01	0.01033	0.1087	275	2375	933.25	25.165
3a	25	0.48	0.52	0.01	0.01	0.00377	0.06451	300	2875	1350.2	57.718
4a	25	0.49	0.51	0.01	0.01	0	0	1175	5150	2657.9	104.28
5a	25	0.495	0.505	0.01	0.01	0	0.07142	2625	8375	5381.7	451.1
6a	50	0.45	0.55	0.01	0.01	0.00967	0.04225	400	2900	1160	21.82
7a	50	0.47	0.53	0.01	0.01	0.00518	0.04347	800	4450	1816.5	39.337
8a	50	0.48	0.52	0.01	0.01	0	0	1250	5500	2705.2	71.172
9a	50	0.49	0.51	0.01	0.01	0	0	3600	10150	5394.3	599.98
10a	50	0.495	0.505	0.01	0.01	0	0	7000	16450	10405	1071.9
11a	100	0.45	0.55	0.01	0.01	0	0.03225	1000	5600	2328.3	29.4
12a	100	0.47	0.53	0.01	0.01	0	0	1300	7900	3709.2	50.509
13a	100	0.48	0.52	0.01	0.01	0	0	3700	9300	5682.4	125.32
14a	100	0.49	0.51	0.01	0.01	0	0.2	8200	18600	10950	466.78
15a	100	0.495	0.505	0.01	0.01	0	0	15100	19600	17789	2622.4

For instance, if the main requirement for the system is to meet the European Standard for Access Control, which requires FAR less than 0.001% and FRR less than 1% [8], the parameters for test 11a (from Table 2) will be the best pick. In addition to satisfying the above requirement, this particular test achieves the lowest MTTD (2328.3 actions) and CPU usage (29.4 seconds) among all other candidates. In addition to the above criteria, if we add another requirement — setting the MTTD to be less than 2000 actions — we notice that none of the tests shown in Table 2 will satisfy such criteria.

The other parameters α and β, which were not considered in the previous tests, can be used to enhance the results obtained by either reducing the TTD or reducing FAR, FRR, or both. A reduction in the TTD will be at the expense of FAR and FRR. Furthermore, each of these parameters has its own effect on FAR and FRR. Lowering α should help in lowering FAR while lowering β should help in lowering FRR. TTD tends to decrease as α and β increase.

Table 3 shows the results of the tuning process. More tests are performed to illustrate the studied effect. Several (test) candidates, which were close to the above criteria as highlighted in Table 2 (e.g. test 3a), are the targets of this new round of tuning. The original tests copied from Table 2 are highlighted in Table 3 and are followed and preceded by the new tests. In the table, trials to decrease FAR and FRR are done by lowering the values of α and β to 0.005 and then to 0.001, while other trials to decrease the TTD are done by increasing α and β to 0.015. The trials in Table 3 succeeded in satisfying the previous criteria in two of the performed tests (Tests 3b and 6c). Considering the fact that the CPU usage for test 6c is 58% lower than the CPU usage for the other test (3b), our best pick for the mentioned criteria will be the parameters used for test 6c ($N = 50$, $\alpha = 0.001$, $\beta = 0.001$, $P_1 = 0.45$, and $P_2 = 0.55$).

Table 3. Tuning the parameters α and β aiming to minimize FAR, FRR, TTD for the highlighted tests to satisfy the selected criteria.

Test Number	Sample Size N (Actions)	P_1	P_2	α	β	FAR (%)	FRR (%)	Min TTD (Actions)	Max TTD (Actions)	Mean TTD (Actions)	CPU Usage (Seconds)
3a	25	0.48	0.52	0.01	0.01	0.00377	0.06451	300	2875	1350.2	57.718
3b	**25**	**0.48**	**0.52**	**0.005**	**0.005**	**0**	**0.07017**	**300**	**3075**	**1523.3**	**60.950**
3c	25	0.48	0.52	0.001	0.001	0.00574	0.02439	925	4625	1995.5	87.233
4b	25	0.49	0.51	0.015	0.015	0.00714	0.03030	975	4725	2407.1	376.72
4a	25	0.49	0.51	0.01	0.01	0	0	1175	5150	2657.9	104.28
4c	25	0.49	0.51	0.005	0.005	0	0	1275	5475	3030.9	129.12
6a	50	0.45	0.55	0.01	0.01	0.00967	0.04225	400	2900	1160	21.82
6b	50	0.45	0.55	0.005	0.005	0.00361	0.11111	350	2900	1291.2	18.181
6c	**50**	**0.45**	**0.55**	**0.001**	**0.001**	**0**	**0.02127**	**450**	**3350**	**1650.8**	**25.997**
7a	50	0.47	0.53	0.01	0.01	0.00518	0.04347	800	4450	1816.5	39.337
7b	50	0.47	0.53	0.005	0.005	0	0.10811	900	4300	2054.8	35.66
8b	50	0.48	0.52	0.015	0.015	0	0.03333	1050	6500	2479.3	48.177
8a	50	0.48	0.52	0.01	0.01	0	0	1250	5500	2705.2	71.172
11a	100	0.45	0.55	0.01	0.01	0	0.03225	1000	5600	2328.3	29.4
11b	100	0.45	0.55	0.005	0.005	0	0.03333	1300	5600	2640	27.251
11c	100	0.45	0.55	0.001	0.001	0	0	1800	7300	3451.7	41.772
12b	100	0.47	0.53	0.015	0.015	0	0	2100	7000	3519.1	44.652
12a	100	0.47	0.53	0.01	0.01	0	0	1300	7900	3709.2	50.509
12c	100	0.47	0.53	0.005	0.005	0	0.05882	2000	6700	4337.4	63.68

6. CONCLUSIONS

We have presented in this paper a new technique for dynamic sample size determination in continuous authentication based on the sequential sampling approach. Experimental evaluation shows that the proposed technique can be used to minimize the length of the detection period while controlling the detection error rates. Although we have illustrated our approach using mouse dynamics biometric, we believe that it can potentially be adapted for several other types of biometric-based continuous authentication systems. Our goal is to investigate this possibility in our future work by adapting and applying the proposed approach for continuous authentication based on keystroke dynamics biometric.

7. REFERENCES

[1] Ahmed A. A. E. and I. Traore 2007. A New Biometrics Technology based on Mouse Dynamics. *IEEE Transactions on Dependable and Secure Computing*, 4, 3 (July-September 2007), 165-179.

[2] Azzini A. and S. Marrara 2008. Impostor Users Discovery Using a Multimodal Biometric Continuous Authentication Fuzzy System. In *KES 2008*, I. Lovrek et al., Eds. Springer-Verlag Berlin Heidelberg, Part II, LNAI 5178, 371-378.

[3] Dowland, P., Singh, H., and Furnell, S. 2001. A Preliminary Investigation of User Authentication using Continuous Keystroke analysis. In *Proc. 8th IFIP Annual Working Conf. on Information Security Management and Small System Security*, Las Vegas, Nevada.

[4] Gunetti D., and C. Picardi 2005. Keystroke Analysis of Free Text. *ACM Transactions on Information and System Security*, 8, 3 (Aug. 2005), 312-347.

[5] Ikehara C. and M. Crosby 2004. Continuous Identity Authentication using Multimodal Physiological Sensors. In *Biometrics Technology for Human Identification*, A. K. Jain and K. Nalini, Eds. *Proc. of the SPIE*, 5404, 393-400.

[6] Jung J., V. Paxson, A.W. Berger, and H. Balakrishnan 2004. Fast Portscan Detection Using Sequential Hypothesis Testing. In *Proc. IEEE Symposium on Security & Privacy*, 211-225.

[7] Maxion R.A. and T.N. Townsend 2003. Masquerade Detection Augmented with Error Analysis. *IEEE Transactions On Reliability Analysis*, 53, 1 (March 2003), 124-147.

[8] Polemi D. 1997. Biometric Techniques: Review and Evaluation of Biometric Techniques for Identification and Authentication, Including an Appraisal of the Areas where they are most applicable. Technical Report. DOI= ftp://ftp.cordis.lu/pub/infosec/docs/biomet.doc

[9] Pusara, M., Brodley, C.E. 2004. User Re-Authentication via Mouse Movements. In *Proceedings ACM VizSec/DMSEC'04*, October 29, 2004, Washington, DC, USA.

[10] Sim T., R. Jankiraman, and S. Kumar 2007. Continuous Verification Using Multimodal Biometrics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29, 4 (April 2007), 687-700.

[11] Tartakovsky A.G., B.L. Rozovsky, R. B. Blazek, and H. Kim 2006. A Novel Approach to Detection of Intrusions in Computer Networks via Adaptive Sequential and Batch-Sequential Change-Point Detection Methods. *IEEE Transactions on Signal Processing*, 54, 9 (Sept. 2006), 3372-3382.

[12] Wald A. 1947. *Sequential Analysis*. J. Wiley & Sons, New York.