An Enhanced Non-Static Biometric Keystroke Dynamics Model for Strengthening the Information Content Security

Saleh M. Abu-Soud
Princess Sumaya University for Technology
Department of Software Engineering
Amman 11941 Jordan

Abstract

As the Internet becomes an integral part of our lives, global access to information and resources is incredibly increasing. This in turn has increased the chances of malicious attacks. In this paper, a new biometric keystroke dynamics model is presented. In our approach, the user's typing rhythm is studied to extract the unique characteristics it has. In this model, three metrics, namely; Key press duration, Latencies between keystrokes and Key even order are combined into a new metric called DLO, which has proved to be a strong metric that showed better results compared with other models. In this research, we also added a new metric to the model, namely; the password overall typing speed which, when combined with DLO forms a new metric called DLOS. DLOS improves the results significantly. In addition, the way of calculating some measurements in the model has been improved.

Keywords: Biometrics, Keystroke, Duration, Latency, Key Event Order, DLOS, Content Security

1. Introduction

As computers and internet become widespread and used by most people, our lives became easy, but the chance for pernicious intrusions becomes high. There are different methods of authentication that can be classified into two categories: physiological features as retina, iris, fingerprints, face geometry, and voice [2]; while the second is adopting artificial measures either physical objects as cards, keys, tokens or personal private information as passwords [3].

User passwords remain the dominant technique and the most popularly adopted way to authenticate users, despite the fact that they have been shown to be a weak mechanism of user authentication [4] and have many deficiencies [5]: too short password is easy to discover, too long password is easy to forget; while wrote password easy to be stolen; somebody can watch the user when he enters the password and know the password; stored password can be revealed by hackers. To overcome these deficiencies, the user is asked to change the password from time to time, or not to use known and easy guessed words that are related to him or to his relatives as wife's name, or child's birthday, etc. these precautions are good and can reduce the danger but cannot eliminate it at all.

Since each person has a different way of typing and has a unique pattern, it is a good idea to rely on how a user types his password instead of relying on what he types. This is called keystroke biometrics based on typing patterns. The idea behind keystroke dynamics has been around since World War II when telegraph operators on U.S. ships could recognize the sending operator according to his keying rhythm [6].

In keystroke dynamics-based authentication, novelty detection methods have been used since only the valid user's patterns are available when a classifier is built. After a while, however, impostors' keystroke patterns become also available from failed login attempts. Keystroke biometrics is not more than data collection (through a keyboard) and manipulation software, so it is cheap since no additional hardware is needed more than a normal keyboard, it is easy to implement and use since no additional training is required, and easy to replicate. But on the other hand, other physiological biometrics such as fingerprints, retinas, etc. remain consistent over long periods of time while for biometrics, a user may change his way of writing depending on the type of the keyboard he is using or on the environment and his physical conditions as if he fatigued or not, standing or sitting, drowsed or not.

But, in spite of these limitations, biometrics remains an effective technique of authentication that is considered as a natural choice of authentication mechanism for computers and network security that is not parasitical and receives a wide user acceptance.

Verification through keystroke dynamics can be categorized as static or non-static [7]. In static verification, the keystrokes are analyzed only at specific times as at login time and cannot detect substitution of the user after the initial verification as the usage of fingerprints, hand geometry, iris or retina, and face, while non-static verification provides continuous verification throughout the session, as the usage of handwriting, voice, lip movements, or keyboard typing patterns.

In keystroke dynamics, many metrics can be used to distinguish one user from another, as key press duration, latency between successive keystrokes, key event order, shift key usage, overall typing speed, finger placement, and pressure typing on the keys.

Many approaches of keystroke dynamics systems have been proposed and developed by researchers. Monrose et al [4] proposed a novel approach in which the user's typing patterns as durations of keystrokes and latencies between keystrokes are combined with the user's password to generate a hardened password. This approach is similar to approaches that use password "salting" for user login. In these approaches the user password is attached with a random number (the salt) that depends on the personal user pattern which leads to increase the complexity and combinations of the password [8], [9].

Some researchers proposed methods based on a statistical model [7], [10], [11]. Some others used a neural network model [12], [13], [14], Fuzzy algorithms [15], Support Vector Machines (SVM) [16], [17], [18], [19], multi-class SVM [5] for username/password access control systems using keystroke biometrics.

One of the earliest analysis techniques of keystroke authentication was proposed by Gaines et al [20]. This approach is based on latency analysis of adjacent keystrokes among seven users each provided two samples of a 3-paragraph text. Another approach which is based on digraph latency was improved by [21]. Then in a later time, Brown et al [13] proposed another parameter beside keystroke latency that is the duration to authenticate users via statistical analysis Euclidean distance-vectors and neural networks. Recently, a system developed by [22] which utilizes this issue with standard web technologies as CGI and Java applets.

Few methods have been designed to collect the intruders' patterns which help in achieving higher accuracy and efficiency. [23], [24], [25].

Some other approaches have been proposed to replace the password method, as physiological features such as iris, voice, fingerprints, retina, and face, etc. [6], [2], [26] where these methods are sometimes difficult to implement because additional hardware needed. Other methods suggested using physical objects held solely by the user as keys, tokens, magnetic cards, etc. [3].

But the enhanced login-problem, where the common username-password paradigm authenticates a user if the password entered is correct disregarding who typed it, remains the most predominant approach, due to its low cost and software simplicity, in addition to the fact that no extra hardware is needed. In this approach, the access may be denied if the password is entered by another user, or access still denied even if the password entered by its owner and even it was correct, if he/she did not match the rightful typing rhythm.

Many researchers as [27] and [28] proposed statistical models that rely mainly on the latency between a pair of consecutive keystrokes as the main metric to distinguish users. One of the interesting models done on keystroke-enhanced login is the implementation of a statistical analysis model proposed by [29]. In this model, they studied and tested four metrics for distinguishing users, namely; key press duration, relative key event order, relative keystroke speeds, and shift key usage patterns. Unfortunately, one of the drawbacks of this model is that these four metrics are examined separately.

In this paper, we present a novel approach to improving the security of passwords and authenticating users accessing the resources of any system against both online and offline attackers. Our approach is a non-static biometric technique based on a statistical model, which aims to identify users based on analyzing habitual rhythm patterns in the way they type.

An early attempt of similar calculations has been done in our previous model [1] which explored the best combination of three keystroke metrics ; namely; key press duration, latency between successive keystrokes, and key event order jointly in one model called DLO. In this paper, we added a new metric which is the speed of typing the overall password, to the previous ones. The new model is called DLOS. In addition, the way of computing some measurements in the previous model were modified. The experiments show that the new model recorded significant improvements over the previous one.

2. Basic Concepts

In this section, we develop certain key concepts upon which the rest of the paper is based. The proposed method involves collecting data that represents the biometric rhythm of a user and converting this biometric data into a form that can be mathematically manipulated. This is followed by application of statistical methods to find an expression that defines the user's biometric traits, against which he can then be authenticated in future. The algorithm aims to develop a system that has low rate of rejecting the owner of the account to enter his system and at the same time to also minimize or hopefully to totally prevent adversaries from entering the system even though they know the password. These biometrics standards are called False Rejection and False Acceptance Rates consecutively [28], [30], and are explained as follows:

1) FAR (False Acceptance Rate), which represents the percentage of times by which an adversary was given access to the rightful owners account. This happens when a non-registered user is incorrectly identified by the system to be an authentic user. It involves an unauthorized person gaining access to protected resources and hence is potentially a far more serious threat to security.

2) FRR (False Rejection Rate), which represents the percentage of times that a valid user was rejected and labeled as an adversary.

In an attempt to provide optimum results, both of the above mentioned standards must be minimized, allowing the adversary to be denied access, while providing it anytime a rightful owner tries to login his account. But it is worthwhile to mention that failing in minimizing FRR which causes more adversaries to enter the system is much dangerous than failing in minimizing FAR which delays the user from entering his system.

2.1 The Metrics Used in the Proposed Model

Five metrics will be considered in our proposed model, these metrics are explained as follows:

1) Key Press Duration

The key press duration is a technique used to identify users by calculating the time interval between the key pressed and the key released.

2) Latency between Adjacent Pairs of Keystrokes

Latency is defined as the time interval between a consecutive pair of keystrokes. In other words, it calculates the time interval by which a user releases the first letter and presses the second letter in a certain pair of keystrokes.

3) Relative Key Event Order

The key press and the key release are each called a "key event", this model's general proposal suggests that some users may type, for example, the word "hello" by first pressing the letter 'h' then releasing it, and then pressing the letter 'e', releasing it and so on. While other users while pressing the letter "h" they press the letter "e" before releasing the letter "h". In other words, the key event order is the sequence in which a user deals with the key press and the key release events when typing a certain word. It is hypothesized that while the pattern in which users press and release keys differ from user to user, the key event ordering would remain consistent for each user.

4) Password Overall Typing Speed

Overall password typing speed measures the time from the key press of the first password character to the key release of the last character. This metric has been added to this version of the model. Actually, this metric has been considered here because it has been noted that it is convergent for the same user while it is slightly divergent for different users and thus can be used as an effective metric to classify users, especially when combined with other metrics. For simplicity, this metric henceforth will be called Speed.

5) DLOS

DLOS is a combination of the above four metrics according to some criteria. That is an entry is accepted by DLOS if it is accepted in all other metrics.

One of the most important characteristics of a metric to be used effectively in this context is that it should be consistent over the same sample when a specific user is taken into consideration whilst it is divergent over different users. It is proved experimentally that the above four metrics relatively satisfy this characteristic.

3. The Proposed Model: DLOS

Since keystroke biometrics indicates that each user types with uniquely characteristics, these characteristics are captured and saved for each user in a "reference template", so that if an intruder tried to hack the rightful owner's account, he/she should be denied access. So, the proposed model starts with collecting the four metric values for the owner of the system; say *user X*; in many trials, and then a series of statistical calculations are applied on the collected data to produce the *user X*'s acceptable boundaries. Then, when *user X* or any other users try to enter the system, the values of the four metrics of their entered password values will be computed and compared with *user X*'s values. The entered password will be accepted if these values lie within the acceptable boundaries.

According to what has been stated previously, five metrics are being implemented and studied in our proposed model by which a user can be authenticated. These are:

1- Key press duration
2- Latency
3- Key event order
4- Password overall typing speed
5- DLOS

The model consists of a series of statistical calculations. These calculations are explained in the following paragraphs.

In order for the statistical calculations of the above five metrics to take place, two events should be recorded for each character in the password: key press time and key release time. From these two events, one can measure the five metrics as follows:

The **key press duration** of a certain character is defined as the time interval by which a user maintains a press on the key, and is calculated as follows:

KPD = Key Released Time – Key Pressed Time ***(1)***

Where:
KPD represents the duration of pressing a key,
Key Released Time represents the key release time, and
Key Pressed Time represents the key press time.

While the **latency** is defined as the time interval between a consecutive pair of keystrokes, and is computed as follows:

Latency = NKey Pressed Time – Pkey Released Time ***(2)***

Where:
Latency represents the time interval between two keys,
NKey Pressed Time represents the key press duration of the next character, and
PKey Released Time represents the key release duration of the present character.

Taking a closer look at the statistical authentication process, it is firstly noted that it is a technique mostly dependent on two measures, the mean and the standard deviation. It is necessary to mention that there are two kinds of means to be calculated among samples of each individual, a mean of press durations for each character (DMean) and a mean of latencies for each consecutive pair of keystrokes (LMean). Two other values; called alphas; should also be calculated, one for the press duration for each character, called Dalpha, and the other is for the latency for each consecutive pair of keystrokes, called Lalpha. These alphas are the similarity bands representing the threshold by which an acceptance rate is determined. Actually, they represent the distance between the value considered for a character and the mean of values of this character in the profile.

In dealing with the key press duration metric, to calculate the similarity band for a certain character, firstly, the differences between the DMean of that character and its actual durations in all the samples in the profile is calculated as follows:

DDiff = |KPD – DMean| ***(3)***

Where,
DDiff denotes the difference,
KPD denotes the actual press duration of a certain character.
DMean denotes the mean of all press durations for this character.

Afterwards, the mean and the standard deviation of those differences are derived, and the following formula to produce the similarity band (Dalpha) is applied:

Dalpha = DMalpha + DSTDalpha ***(4)***

Where,
Dalpha denotes the similarly band of durations.
DMalpha denotes the mean of the differences of durations.
DSTDalpha denotes the standard deviation of the differences of durations.

Likewise, the same procedure is applied to the latency metric to calculate its similarity band for a certain pair of keystrokes (Lalpha), according to the following formula:

Lalpha = LMalpha + LSTDalpha ***(5)***

Where,
Lalpha denotes the similarly band of latencies.
LMalpha denotes the mean of the differences of latencies.
LSTDalpha denotes the standard deviation of the differences of latencies.

This implies that for a user to be accepted, his/her input (Whether the model being used is the key press duration or the latency) should fall within the range of the similarity band, in accordance with the following formula:

Mean - Similarity Band <= Input <= Mean + Similarity Band ***(6)***

Where,
Mean represents the mean of the values in the dataset (DMean or LMean).
Similarity Band represents the Dalpha or the Lalpha.
Input represents the current data value. (Whether it's the duration of a certain character or the latency between a certain pair of keystrokes)

The calculations explained above are only concerned with the first two approaches (the key press duration and the latency). Whilst when dealing with the third approach -the key event order- a slightly different technique is applied.

When it comes to dealing with the **key event order** approach, only the latency of consecutive pairs of keystrokes is considered. This is done by determining the percentage of positive and negative latencies of each pair in the user samples. Where, a positive latency of a certain pair of characters indicates that the first character has been pressed and released before the second character has been pressed. Whilst a negative latency indicates that the first and the second characters were pressed before the first character has been released. Figure 1 depicts the idea of negative and positive latencies when a user enters the word "HELLO".

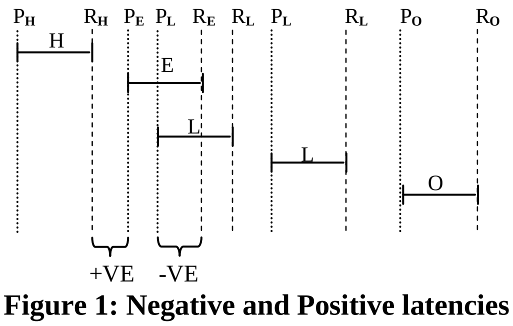

Figure 1: Negative and Positive latencies

As depicted in Figure 1, P_H, for example, represents the press time of the character H, while R_H represents the release time of the same character, and P_E represents the press time of character E, and so on. As noted in the figure, E, for example, is pressed after H has been released. So, $P_E - R_H$ is a positive latency. While the letter L has been pressed before the letter E is released. So, $P_L - R_E$ represents a negative latency. The acceptance privilege here is handed to a certain pair of keystrokes, when the latency of this pair falls within the higher percent (regardless whether it is a positive or a negative percent), as shown below:

- Let $\mathbf{L_i}$ denotes the actual latency of a certain pair of keystroke i.
- Let $\mathbf{PosLat_i}$ denotes the percent of the positive latency for the pair i.
- Let $\mathbf{NegLat_i}$ denotes the percent of the negative latency for the pair i.

Then

- ***If $PosLat_i$ >= 0.5 and L_i >0 then the pair i is accepted.***
- ***If $PosLat_i$ >= 0.5 and L_i <0 then the pair i is rejected.***
- ***If $NegLat_i$ >= 0.5 and L_i <0 then the pair i is accepted.***
- ***If $NegLat_i$ >= 0.5 and Li >0 then the pair i is rejected.*** ***(7)***

After showing the basis by which a certain character or a certain pair of keystrokes is considered acceptable, now, it is important to discuss the technique by which a user is granted access regarding his/her password as a whole. By spotting the light on the first metric; i.e. **key press duration**, it is realized that when determining whether a certain user sample is accepted or not, the measure of the degree by which this sample matches the other samples in the profile has to be calculated (In reference to the mean of this sample).This is achieved by implementing a counter which returns the number of accepted characters in the current password being tested using formula (6), where, the counter value should satisfy a predefined value Dcount as follows:

Dcount >= PasLength * Weight ***(8)***

Where,
Dcount represents the number of accepted characters in the password.
PasLength represents the number of all characters in the password.
Weight represents the predefined acceptance percentage

The same procedure is applied to the other two metrics i.e. **latency** and **key order**, where the counter is used to return the number of accepted pairs of keystrokes in the current password being tested (each approach according to its logic) using formulas 6 and 7, needless to say that the counter value should also satisfy a predefined value Count as follows:

Count >= PairLength * Weight ***(9)***

Where,
Count represents the number of accepted pairs in the password (Latency count or Order count).
PairLength represents the number of all pairs in the password.
Weight represents the predefined acceptance percentage.

For the **password overall typing speed**, it is calculated simply as follows:

SPD = LastKeyReleasedTime – FirstKeyPressTime ***(10)***

Where:
SPD represents the time period, by which a person enters the password,
LastKeyReleasedTime represents the key release time of the last character of the password, and
FirstKeyPressedTime represents the key press time of the first character of the password.

As what has been done for duration and latency, the similarity band for a certain character should be calculated. Firstly, the differences between the SMean of each trial and its actual speed in all the samples in the profile is calculated as follows:

SDiff = |SPD – SMean| ***(11)***

Where,
SDiff denotes the difference,

SPD denotes the actual overall time of a certain trial of *user X*.
SMean denotes the mean of all overall times of all trials of *user X*.

Afterwards, the mean and the standard deviation of those differences are derived, and the following formula to produce the similarity band (Salpha) is applied:

Salpha = SMalpha+ SSTDalpha ***(12)***

Where,
Salpha denotes the similarly band of Speeds.
SMalpha denotes the mean of the differences of Speeds.
SSTDalpha denotes the standard deviation of the differences of Speeds.

This implies that for a user to be accepted, his/her speed should fall within the range of the similarity band, in accordance with the following formula:

SMean - Similarity Band <= Input <= SMean + Similarity Band ***(13)***

Where,
Similarity Band represents the Salpha.
Input represents the overall time of entering the current password.

Now, as had been done previously, to determine if an entry is accepted or not, the following formula should be satisfied:

Dcount >= No. of Trials * Weight ***(14)***

Where,
Dcount represents the number of accepted trials.
No of Trials represents the number of all trials of *user X*.
Weight represents the predefined acceptance percentage.

The fifth metric, **DLOS**, is expected to be the most effective one among others. This is because it considers the password is accepted if it is accepted in all other four metrics.

It has been pointed out earlier that each approach sets a predefined acceptance percentage (sometimes referred to as weight). The determination of these percentages in each approach were conducted and computed in the experimental analysis of several tests being carried out through our research.

3.2 Experiments

Experiments are conducted on *user X*, test user X, and 10 adversaries. *User X* is the owner of the password. He will enter the password 10 times, from which data about durations, latencies, key event order, and speeds will be collected and computed according to the steps of the model. This computed data will be used as a benchmark for others to decide whether their entries are accepted or not. Test user X is the same as *user X*, but he will enter the password another 10 times, and for each entry, the system will check it with his previously entered entries to decide on the percentage of FRR for the five metrics. Then 10 adversaries will enter the password each 10 times, and the average of the 10 times will be considered in the model and will be tested against the computed values of *user X* to decide on the percentage of FAR for the five metrics.

Before conducting the above mentioned experiments, an experiment is conducted to show the behavior of our model and to create the benchmark on which other users are tested. This experiment is conducted on data collected from *user X* who represents the user who possesses the password. Data collected from this experiment represent the latency, duration, and key event order for each character, in addition to the speed of entering the password. *User X* will enter the password 10 times. Let the password be the following string: “houseroad91” which is composed of 11 characters. We assume no mistakes and no corrections for these mistakes are allowed, the password is not case sensitive, and all users will use the same keyboard in all trials.

To decide on the best acceptance percentage (weight), another experiment has to be conducted on the entries of *user X* regarding the four metrics: durations, latencies, key event order, and speeds.

All these experiments and the results are discussed in details in the consecutive sections.

4. An Illustrative Example

To illustrate the processes of the model, a sample of ten trials for *user X* is taken into consideration. All these trials were entered using the same keyboard but in different times, and we assume no errors are allowed. This sample represents the key press duration (KPD) values (equation 1) for each character in the password and latencies (equation 2) for each pair of consecutive characters of the password are calculated for this user when he entered the password “houseroad91”. These values are listed in Tables 1 and 2 consequently (all values are in milliseconds).

Table 1: The Key Press Durations for the *User X* When Entered the Password houseroad91

Trial	h	o	u	s	e	r	o	a	d	9	1
1	140	128	135	65	155	90	150	105	145	140	107
2	140	135	155	90	105	110	140	170	85	176	89
3	136	105	145	87	130	85	111	115	145	155	154
4	182	155	115	96	155	150	134	127	140	135	123
5	143	142	135	90	150	123	117	118	185	110	124
6	156	156	100	87	197	100	121	100	150	150	100
7	165	103	153	79	100	100	105	110	150	175	96
8	120	155	170	89	150	50	126	150	95	143	102
9	135	139	195	134	190	150	134	123	150	156	123
10	143	138	130	76	150	108	127	164	165	147	132
Dmean	**146**	**135.6**	**143.3**	**89.3**	**148.2**	**106.6**	**126.5**	**128.2**	**141**	**148.7**	**115**

Table 2: The Latencies of Each Two Consecutive Characters for the *User X* When Entered the Password houseroad91

Trial	h-o	o-u	u-s	s-e	e-r	r-o	o-a	a-d	d-9	9 _ 1
1	-11	20	-20	-10	33	-10	19	-20	28	-10
2	-22	13	-8	-10	18	-20	20	-10	20	-22
3	8	20	-10	0	26	-13	21	-20	31	-13
4	-9	-12	-23	-20	10	-2	13	-20	30	0
5	12	20	0	-10	30	-10	27	-10	21	-14
6	-13	18	-10	0	42	10	10	0	38	-24
7	-24	10	-20	-10	32	-12	20	-10	30	-7
8	15	-9	-8	-23	28	-18	33	-13	29	7
9	-8	20	-13	-7	20	9	10	-20	19	0
10	-21	23	-20	-10	30	-10	20	-10	20	-13
LMean	**-7.3**	**12.3**	**-13.2**	**-10**	**26.9**	**-7.6**	**19.3**	**-13.3**	**26.6**	**-9.6**

Figures 2 and 3 show that the key press durations and the latencies of *user X*'s entries are both consistent and convergent. This is normal even though the user entered the password in different times because each person should have a unique way of writing. But, as will be seen in the next section, this should not be true when different adversaries enter the same password.

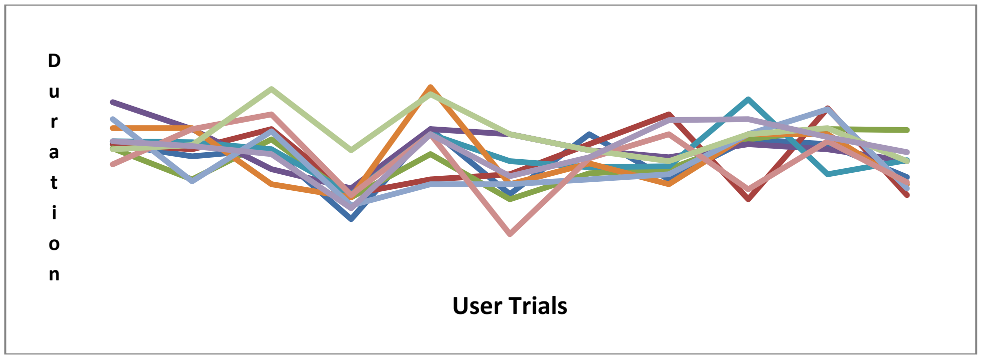

Figure 2: The Key Press Duration Entries for *user X*

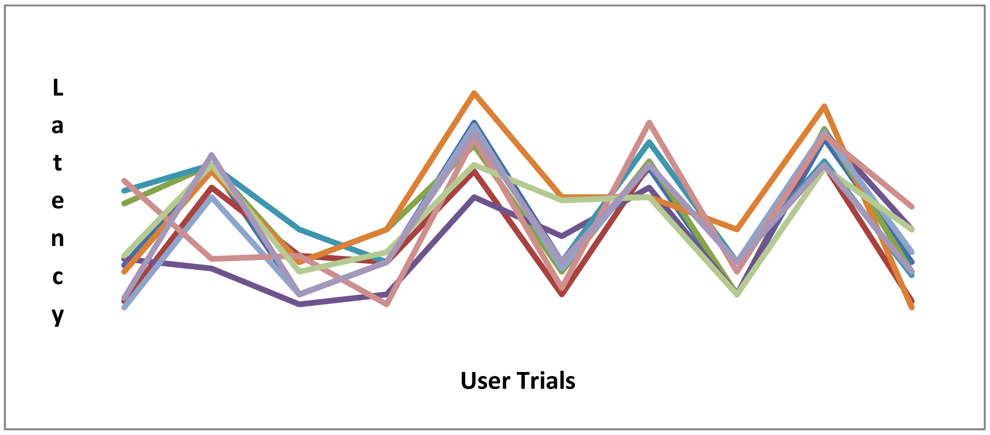

Figure 3: The Latency Entries for *user X*

Now, according to equation (3), the mean of each character in the **duration** values of the ten trials are calculated as appeared in the bottom row of Table 1. These values are needed to calculate the difference between them and each corresponding actual entry. These differences are appeared in the leftmost eleven columns of Table 3.

Table 3: The DDiff and Dalpha Values

h	o	u	s	e	r	o	a	d	9	1	DMalpha	DSTDalpha	Dalpha
6	7.6	8.3	24.3	6.8	16.6	24	23.2	4	8.7	8	12.45454545	7.841219757	20.29577
6	0.6	12	0.7	43.2	3.4	14	41.8	56	27.3	26	20.92727273	19.28953555	40.21681
10	30.6	1.7	2.3	18.2	21.6	16	13.2	4	6.3	39	14.76363636	11.98292725	26.74656
36	19.4	28	6.7	6.8	43.4	7.5	1.2	1	13.7	8	15.63636364	14.38355121	30.01991
3	6.4	8.3	0.7	1.8	16.4	9.5	10.2	44	38.7	9	13.45454545	14.52324782	27.97779
10	20.4	43	2.3	48.8	6.6	5.5	28.2	9	1.3	15	17.30909091	16.30603904	33.61513
19	32.6	9.7	10.3	48.2	6.6	22	18.2	9	26.3	19	20.03636364	12.25522523	32.29159
26	19.4	27	0.3	1.8	56.6	0.5	21.8	46	5.7	13	19.8	18.58138854	38.38139
11	3.4	52	44.7	41.8	43.4	7.5	5.2	9	7.3	8	21.18181818	19.44365286	40.62547
3	2.4	13	13.3	1.8	1.4	0.5	35.8	24	1.7	17	10.38181818	11.56683346	21.94865

According to equation (4), the mean (DMalpha) and the standard deviation (DSTDalpha) of each row of Table 3 are calculated and added to each other. The result is called Dalpha or the similarity band of durations. These three values are appeared in the three right most columns of Table 3.

Likewise, the same procedure is applied to the **latency** metric to calculate its similarity band for a certain pair of keystrokes (Lalpha). The results are appeared in Table 4.

Table 4: The LDiff and Lalpha Values

h-o	o-u	u-s	s-e	e-r	r-o	o-a	a-d	d-9	9 _ 1	DLalpha	DSTDLalpha	Lalpha
3.7	7.7	6.8	0	6.1	2.4	0.3	6.7	1.4	0.4	3.55	3.041655251	6.591655251
14.7	0.7	5.2	0	8.9	12.4	0.7	3.3	6.6	12.4	6.49	5.416938455	11.90693845
15.3	7.7	3.2	10	0.9	5.4	1.7	6.7	4.4	3.4	5.87	4.316904755	10.18690476
1.7	24.3	9.8	10	16.9	5.6	6.3	6.7	3.4	9.6	9.43	6.711855183	16.14185518
19.3	7.7	13.2	0	3.1	2.4	7.7	3.3	5.6	4.4	6.67	5.748439402	12.4184394
5.7	5.7	3.2	10	15.1	17.6	9.3	13.3	11.4	14.4	2.41	4.671913717	7.081913717
16.7	2.3	6.8	0	5.1	4.4	0.7	3.3	3.4	2.6	4.53	4.717355662	9.247355662
22.3	21.3	5.2	13	1.1	10.4	13.7	0.3	2.4	16.6	10.63	8.134432986	18.76443299
0.7	7.7	0.2	3	6.9	16.6	9.3	6.7	7.6	9.6	6.83	4.793525726	11.62352573
13.7	10.7	6.8	0	3.1	2.4	0.7	3.3	6.6	3.4	5.07	4.387368232	9.457368232

As stated earlier, to decide if an entered character by the user or an adversary is accepted or not according to duration, it must fall between (DMean – Dalpha) and (DMean + Dalpha) of that character as computed above. For example, let the duration of the entered character "h" is 140ms, and since it lies between 125.7042 and 166.2958, it is accepted.

We repeat the same formula for all characters in the entered password, and the number of accepted characters is stored in Dcount. Table 5 shows the results of computing the number of accepted characters in the entered password according to duration for the first trial of Table 1.

Table 5: Accepted Characters According to Duration for the First Trial

	h	o	u	s	e	r	o	a	d	9	1
KPD	140	128	135	65	155	90	150	105	145	140	107
Dmean	146	135.6	143.3	89.3	148.2	106.6	126.5	128.2	141	148.7	115
Ddiff	6	7.6	8.3	24.3	6.8	16.6	23.5	23.2	4	8.7	8
Dmean-Dalpha	125.7042	115.3042	123.0042	69.00423	127.9042	86.30423	106.2042	107.9042	120.7042	128.4042	94.70423
Dmean+Dalpha	166.2958	155.8958	163.5958	109.5958	168.4958	126.8958	146.7958	148.4958	161.2958	168.9958	135.2958
Accepted?	yes	yes	yes		yes	yes			yes	yes	yes

DMalpha	12.45455
DSTDalpha	7.84122
Dalpha	20.29577

Dcount	8
PasLength	11
Weight	0.7

As shown in Table 5, for the weight say 0.7 (this value is determined experimentally and will be discussed later), 8 characters out of the password's 11 characters are accepted. So, the value of Dcount for this trial becomes 8.

When it comes to dealing with the **key event order** approach, only the latency of consecutive pairs of keystrokes is considered. This is done by determining the percentage of positive and negative latencies of each pair in the user samples as depicted in Table 6.

Table 6: Percentage of Positive and Negative Latencies of the 10 Trials of *User X*

Trial	h-o	o-u	u-s	s-e	e-r	r-o	o-a	a-d	d-9	9-1
1	-11	20	-20	-10	33	-10	19	-20	28	-10
2	-22	13	-8	-10	18	-20	20	-10	20	-22
3	8	20	-10	0	26	-13	21	-20	31	-13
4	-9	-12	-23	-20	10	-2	13	-20	30	0
5	12	20	0	-10	30	-10	27	-10	21	-14
6	-13	18	-10	0	42	10	10	0	38	-24
7	-24	10	-20	-10	32	-12	20	-10	30	-7
8	15	-9	-8	-23	28	-18	33	-13	29	7
9	-8	20	-13	-7	20	9	10	-20	19	0
10	-21	23	-20	-10	30	-10	20	-10	20	-13
$PosLat_i$	**0.3**	**0.8**	**0.1**	**0.2**	**1**	**0.3**	**1**	**0.1**	**1**	**0.3**
$NegLat_i$	**0.7**	**0.2**	**0.9**	**0.8**	**0**	**0.7**	**0**	**0.9**	**0**	**0.7**

To decide on a character whether it is accepted or not, its latency should satisfy one of the cases of formula (7). For example the pair h-o in trial 1 has the latency -11, and since it is less than 0 and the corresponding $NegLat_i$ is >= 0.5 this pair is accepted. Table 7 shows number of accepted pairs for trial 1 in Table 6.

Table 7: Number of Accepted Key Event Order Pairs

	h-o	o-u	u-s	s-e	e-r	r-o	o-a	a-d	d-9	9-1
L_i	-11	20	-20	-10	33	-10	19	-20	28	-10
Accepted?	yes	yes	yes	yes	yes	yes	yes	yes	yes	yes
$PosLat_i$	0.3	0.8	0.1	0.2	1	0.2	1	0.1	1	0.3
$NegLat_i$	0.7	0.2	0.9	0.8	0	0.8	0	0.9	0	0.7

A fourth metric has been added to this model, which is the **password overall typing speed** (for simplicity it will be called speed). This metric can be used to distinguish users from each other. So it can be used as an effective metric in our model. It is simply the time difference between the KeyPressedTime of the first character of the password and the KeyReleasedTime of the last character for each trial. Table 8 shows the speed of *user X* in the ten trials in addition to their average.

Table 8: The Password Overall Typing Speed of *user X*

Trial	1	2	3	4	5	6	7	8	9	10	Average
Speed	1380	1375	1428	1482	1507	1497	1346	1390	1659	1490	**1455.4**

Now, the similarity band should be calculated for all the speeds of the 10 trials of *user X*. These values are shown in Table 9. As shown on Table 9, the speed of the first trial, for example, is 1380ms. Since it falls between 1330.43 and 1580.37, it is accepted. So, as appeared on the table, 9 trials of *user X* are accepted, and the value of Scount becomes 9. This means that for any other user, to be accepted, his speed value should lie between these two values.

Table 9: The Speeds of All Trials of *User X* and the Similarity Band

Trials' speeds	1380	1375	1428	1482	1507	1497	1346	1390	1659	1490
Diff	75.4	80.4	27.4	26.6	51.6	41.6	109.4	65.4	203.6	34.6
Smean-Salpha	1330.43									
Smean+Salpha	1580.37									
Accepted?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes		Yes

Smean	1455.4
SMalpha	71.6
SSTDalpha	53.3654
Salpha	124.965

Scount	9
PairLength	10
Weight	0.7

To proceed with the calculations, we have to decide experimentally on the value of the **weight**. The weight is used as a part of formulas (8), (9) and (14) to decide if the entry is accepted or not. For example, for the case of key event order, as appeared in Table 7, the entry is accepted for the weight %70 since Dcount (= 7) and it is greater than or equal 7 (PairLength*weight) while it is not accepted for the weight %90 for example, because 7 is not greater than 9. Table 8 shows different weight values with their effect on the results for the four metrics used. It shows, for instance, that %10 of *user X* attempts using durations were rejected when the weight value is %75. It is noted also that the highest weight value with the best results for all metrics, i.e. zero FRR; is 70%. This value as well as all the similarity bands obtained from the ten trials of *user X* will be used to judge for a password entry is acceptable or not when the owner of the system *(user X)* enters his system in later times (this measures the FRR) or adversaries attempt to enter the system (this measures the FAR). The experiments along with their results will be shown in the following section.

Table 10: The Percentage of FRR for *User X* the Three Metrics with Respect to Different Weight Values

weight	duration	Latency	Order	Speed
0.69	0	0	0	0
0.7	***0***	***0***	***0***	***0***
0.75	0.1	0.2	0.2	0
0.8	0.1	0.2	0.2	0
0.85	0.7	0.7	0.6	0
0.9	0.7	0.7	0.6	0
0.95	1	1	0.6	1
1	1	1	0.6	1

5. Results

An experiment had been conducted to check the accuracy of our model. In this experiment, *user X* attempted to access his own account; as a test user; 10 times, and 10 different adversaries tried to enter the system 10 times each. All attempts were done using the same keyboard in different times. We computed the FAR for the adversaries and the FRR for the user attempts to access his own account for all metrics i.e. the duration, the latency, the key event order, the speed and the DLOS.

Table 11 shows the percentages of FRR and FAR for all metrics when test user X and the ten adversaries attempt to enter the system. As shown on the table, the test user has been rejected 50% of the times from accessing his own account using the duration metric alone, while the adversaries succeeded 60% of times to access the system using the same metric.

While test user X has been rejected 40% of time using the latency metric, and the adversaries failed to access the system 60% of the times in their attempts using the same metric. As shown in the third column of the table, using the key event order metric, the user is rejected 40% of the times and the adversaries accessed the system 30% of the times. For speed metric, test user X failed to access his system 40% of the times, while the adversaries succeeded to enter the system 50% of the times. The last column shows the results of our proposed metric, the DLOS. It depicts that the user is rejected 10% of the times while the adversaries failed to access the system all the times. These results may also be shown more clearly in Figure 4.

Table 11: FRR and FAR for the Test User X and Adversaries

	Duration	Latency	Key Event Order	Speed	DLO	DLOS
FRR for test user	50	40	40	40	20	10
FAR for adversaries	60	60	30	50	0	0

Since our aim is to minimize both FRR and FAR, it is apparent that the best results are obtained when DLOS is used; 10% for FRR and 0% for FAR. So, even though FAR is somewhat good for key event order (30%), it is higher for FRR (40%), and similarly FRR is not bad for latency (40%), it is bad for FAR (60%). It is noted also that speed got 40% for FRR which is the same as latency and key event order and better than duration, while it obtained 50% for FAR, which is better than duration and latency.

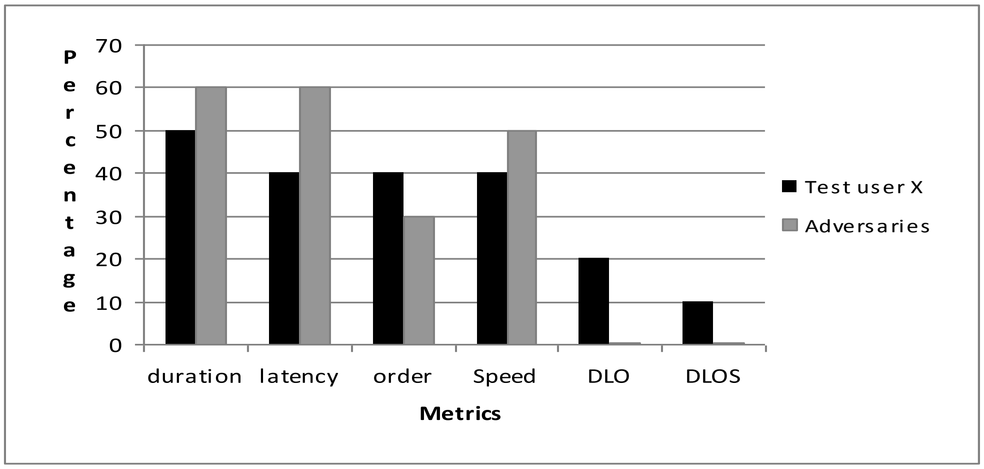

Figure 4: FRR and FAR for All Users

The above results show that the use of our Hybrid Key Stroke Authentication technique (DLOS) successfully overcomes the problems of all other individual implementations of each metric, and even records a significant improvement on the original model (DLO). It is noted also that Speed metric did not contribute much when it is employed individually, but it has a significant effect when used within DLOS.

6. Conclusions

In this research, a new hybrid authentication model has been proposed and tested. This model studied and tested four metrics, namely: Duration, Latency, Key event order, and speed.

It is apparent from the obtained results that the best implementation of the statistical approach would include combining the four mentioned models together since the implementation of each one separately was found to be inconvenient, in reference to the FAR and FRR.

The results obtained showed that the proposed model; DLOS; succeeded in maintaining the highest level of security compared to other models.

Although the proposed system has managed to increase the level of security, nevertheless, some other problems couldn't be solved, such as the differences in personal keyboards, mental and psychological state of the user at that time that will causes gradual changes in the user pattern when entering the password in different times or in different moods and atmospheres. We are working now on a new model that takes these gradual changes into consideration.

9. References

1. S. Abu-Soud , A Hybrid Key Stroke Authentication System, The Proceeding of the Second International Conference on Informatics Engineering & Information Science (ICIEIS2013), Malaysia. Nov. 12-14, 2013. pp 84-95.
2. N. Ratha, A. Senior and R. Bolle, "Tutorial on automated biometrics", Proceedings of International Conference on Advances in Pattern Recognition, Brazil, 2001.
3. N. Ahituv, Y. Lapid, and S. Nuemann, "Verifying the authentication of an information system user", Computer Security, 6 2, 1987.
4. F. Monrose, R. Michael, and W. Susanne, "Password hardening based on keystroke dynamics", Intr. Journal of Information Security 1(2), 2002.
5. S. Giovanni, C. Giovanni, and T. Massimiliano, "Cumulative and ratio time evaluation in keystroke dynamics to improve the password security mechanism", Journal of Computer and Information Technology, 2(1), 2011.
   A. Jain, R. Bolle, and S. Pankanti, Biometrics: personal identification in networked society, Kluwer, Norwell, 1999.
6. F. Monrose and A. Rubin, "Keystroke dynamics as a biometric for authentication", Future Generation Computer System 16(4), 2000. pp 351-359
7. R. Morris and K. Thompson, "Password security: a case history", Communication of the ACM, 22(11), 1979.
8. U. Manber, "A simple scheme to make passwords based on one-way functions much harder to crack", Computers and Security, 15(2), 1996.
9. L. Araujo, L. Sucupira Jr., M. Lizarraga, L. Ling, and J. Yabu-Uti, "User authentication through typing biometrics features", IEEE Transactions on Signal Processing 52(2), 2005.
10. S. Bleha, C. Slivinsky, and B. Jussein, "Computer-access security systems using keystroke dynamics", IEEE Transactions on Pattern Analysis and Machine Intelligence 12(12), 1990.
11. M. Obaidat and B. Sadoun, "Verification of computer users using keystroke dynamics", IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 27(2), 1997.
12. M. Brown and SJ. Rogers, "User identification via keystroke characteristics of typed names using neural networks", Int. J. Man-Mach. Stud, 39(6), 1993.
13. T. Alexandre, Biometrics on Smartcards: "An approach to keyboard behavioral signature", Second Smart Research and Advanced Applications Conference, 1996.
14. B. Hussein, R. McLaren and S. Bleha, "An application of fuzzy algorithms in a computer access security system", Pattern Recognition Letter, 9, 1989.
15. B. Schlkopf, R. Williamson, A. Smola, J. Shawe-Taylor, and J. Platt, Support vector method for novelty detection, Solla SA, Leen TK, Muller K-R, editors, Advances in Neural Information Processing Systems, vol. 12, MIT Press, 2000.
16. M. de Oliveira, V. Kinto, E. Hernandez, and T. de Carvalho, "User authentication based on human typing patterns with artificial neural networks and support vector machines", SBC, 2005.
17. Y. Sang, H. Shen, and P. Fan, Novel imposters detection in keystroke dynamics using support vector machines, LNCS Springer Berlin/Heidelberg, 2004.
18. K. Sung and S. Cho, "GA SVM wrapper ensemble for keystroke dynamics authentication", Intr. Conference on Biometrics, Hong Kong, 2006.
19. R. Gaines, W. Lisowski, S. Press, and N. Shapiro, Authentication by keystroke timing: some preliminary results, Rand Report, 1980.
20. G. Legget, J. Williams, and M. Usenick, "Dynamic identity verification via keystroke characteristics", Intr. J. of Man-Machine Studies, 1991.

21. T. Marino and A. Juan, Fuzzy keystroke biometrics on web security. Universidad Autonoma de Madrid, 2000.
22. D. Tax and R. Duin, “Support vector data description”, Machine Learning, 54, 2004.
23. H. Lee and S. Cho, “SOM-based novelty detection using novel data”, Proceedings of Sixth International Conference on Intelligent Data Engineering and Automated Learning, Lecture Notes in Computer Science, 3578, 2005.
24. H. Lee and S. Cho, “Retraining novelty detector with imposer patterns for keystroke dynamics-based authentication”, IAPR International Conference of Biometrics, Hong Kong, 2006.
25. R. Luis-Garcia, C. Albertola-L'ope, O. Aghzout, and J. Ruiz-Alzola, “Biometric identification systems”, Signal Processing, 83, 2003.
26. X. Ke, R. Manuel, M. Wilkerson, and L. Jin, “Keystroke dynamics: a web-based biometric solution”, 13th USENIX Security Symposium, 2004.
27. F. Wong, A. Supian, and A. Ismail, “Enhanced user authentication through typing biometrics with artificial neural networks and k-nearest neighbor algorithm”, IEEE, 2001.
28. L. Edmond, L. Xia, X. Chen, and Y. Xiao, “Enhanced user authentication through keystroke biometrics”, Final Project Report, MIT, 2004.
29. V. Kacholia and S. Pandit. "Biometric Authentication using Random Distributions (BioART)", Canadian IT Security Symposium. 2003.