Free Typed Text Using Keystroke Dynamics for Continuous Authentication

Paulo Pinto, Bernardo Patrão, Henrique Santos

Free typed text using keystroke dynamics for continuous authentication

Paulo Pinto, Bernardo Patrão and Henrique Santos

Watchful Software, Coimbra, Portugal
Universidade do Minho - Dep. de Sistemas de Informação, Braga, Portugal
prpinto@itgrow.pt,bernardo.patrao@watchfulsoftware.com,
hsantos@dsi.uminho.pt
http://www.watchfulsoftware.com,http://www.uminho.pt

Abstract. *Information is increasingly in a digital-only format and almost everything we do nowadays depends on a digital identity for authentication and authorization. No matter how strong the authentication system is, after the authentication phase, there is no continuous verification that user is still the same human being that successfully logged in, thus leaving the system unprotected. This paper presents a usable breakthrough approach for continuous authentication with free typed text through the use of biometric techniques based on keystroke dynamics. Our main purpose is to achieve a reduction of the required sample size, while improving (or at least not worsen) precision performance, by adapting and improving parameters on a keystroke dynamics algorithm.*

Keywords: Identity verification; User authentication; Biometrics; Keystroke dynamics; Host-based intrusion detection; Security.

1 Introduction

In a world governed by digital information, computing is the main activity and user authentication plays an important role in access control. One of the most common situations of intrusion occurs when a worker leaves his/her workstation unlocked or when someone knows a user password and tries to use a false identity to do something malicious (for example, to send an e-mail, to change a document, to write on facebook or twitter)[1, 2].

Keystroke dynamics, as a biometric for authentication, can be used to mitigate the above threat, continuously detecting intrusions[3]. But false alarms in such intrusion detection systems are quite common[4] and the european standard for access-control systems (EN-50133-1) specifies a false alarm rate of less than 1% for this type of solution [5]. For that reason, it is crucial to optimize algorithms to achieve low false rejection rate (FRR) and false acceptance rate (FAR).

In this paper, we present an heuristic optimization of an algorithm based on keystroke dynamics and results obtained from a real experiment, for validation purposes.

The rest of the paper is organized as follows. Section 2 is dedicated to the study of keystroke dynamics and how it can be used to verify user identity and section 3 is the state of the art. In section 4, we start defining main functions used for the intrusion detection and we also define the decision criterion. In section 5 we present techniques that will allow us to improve the algorithm as well as how these techniques can be implemented. On the same section we describe a new decision criterion that will allow us to get FRR lower than 1% (validated with results on a real environment) and in section 6 we write the main conclusions of this work.

2 Biometrics and keystroke dynamics

Using biometrics, each individual can be uniquely identified by physical and behavioural characteristics and, unlike passwords, biometrics can not be lost, stolen or copied. There are several biometric techniques such as fingerprints, the way you walk, your eye geometry or even the way you speak[6]. Each one can be used to identify or to authenticate. In a very simplified way, identification involves the comparison of a given biometric pattern with all of previously stored patterns. Authentication is similar but involves only one comparison with the pattern belonging to someone claiming identity checking. In both cases, one main concern is to avoid false rejections and false acceptances: for access control the objective of the application is to not allowing access to unauthorized individuals under all circumstances while granting access to all legitimate users. It is clear that the surveillance software has to be set up with a very low FAR even if it comes at the price of a higher FRR. On the other hand, identification within surveillance software has to be set up with a low FRR even if FAR gets higher.

Keystroke dynamics is a technique aiming to find patterns based on timing information from pressed and released keys when a person is typing at a keyboard. Most common features are dwells (time between key down and key up of a specific key), flights (time between key up and key down of two consecutive keys), digraphs, trigraphs and fourgraphs[3]. Weather conditions, fatigue, stress or any sort of influence, can drastically impact the result but with a constant user profile update, even the effect of these weird behaviours can be drastically reduced.

3 State of the art

Keystroke dynamics is a behavioural based furtive biometric technique[7] that does not require any special resources (hardware), besides a normal keyboard and the support low-level software usually available in any PC like system. These properties make it a good candidate for continuous authentication[8]. Recently this research topic received some important contributions, being evident that one of the main issues is the training data used. Solami et. al present a generic model for a Continuous Biometric Authentication System (CBAS), discussed some proposed solutions and propose a classification based on the type of target

scenario: class I, when legitimate users' profiles are available and the identities of all possible impostors are known (i.e., a closed system); class II, otherwise[9]. In this work we are targeting a class I system, aiming to improve only its performance over the internal user universe.

4 Software design and algorithm

4.1 Architecture

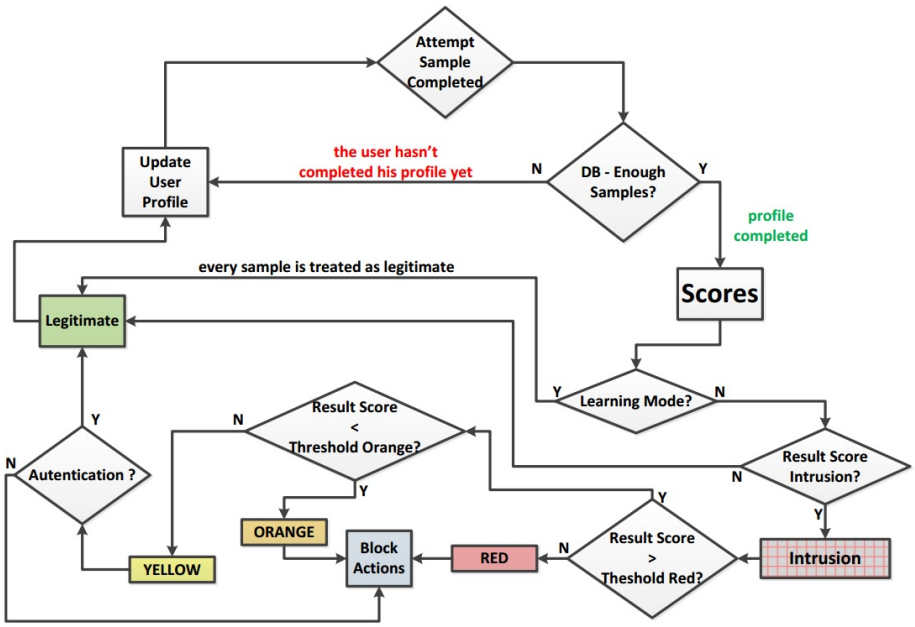

Fig. 1. Algorithm architecture.

The base of the software architecture is the one proposed in[8, 10]. It is composed of a central server, a database where user profiles are saved and a constant validation mechanism for each sample produced by the user (see above references for more detailed information).

The mechanism receives an attempt sample and needs to decide whether it is an intrusion or not (see Fig. 1). The decision algorithm is the heart of the mechanism and, as we will see later, with a generalization of the scores function proposed in [3] (by defining new parameters and new metrics), varying acceptance thresholds and implementing a dynamic decision criterion, it is possible to substantially decrease the FRR, the FAR and the attempt sample size.

4.2 Absolute scores

The user profile, stored in the database, consists essentially in one merged sample with all characteristic user features. This profile is constructed and constantly updated every time the user produces a new valid sample.

Let us define $AS(l)$ as the set of all features from an ***attempt sample*** with length l and DB as the set of all features from the ***merged sample*** stored on the database. Let us also define, if exist, $DW(x(y))$ as the average of $x(y)$ where

$$x(y) \in I_{DW}(y) = \{all\ dwells\ in\ y\}$$

and

$$y \in J = \{AS(l), DB\}.$$

Intuitively, we define $FL(x(y))$, $DI(x(y))$, $TR(x(y))$ and $FR(x(y))$ for flights, digraphs, trigraphs and fourgraphs, respectively, and the corresponding sets $I_{FL}(y) = \{all\ flights\ in\ y\}$, $I_{DI}(y) = \{all\ digraphs\ in\ y\}$, $I_{TR}(y) = \{all\ trigraphs\ in\ y\}$ and $I_{FR}(y) = \{all\ fourgraphs\ in\ y\}$.

For each feature x on the merged sample ($x(DB)$), we define the ***acceptance neighbourhood*** as

$$V(x(DB)) = [(2 - p(z))z(x(DB)), p(z)z(x(DB))],$$

where

$$z(x(y)) = \begin{cases} DW(x(y)) & if\ x\ is\ a\ dwell\ in\ y, \\ FL(x(y)) & if\ x\ is\ a\ flight\ in\ y, \\ DI(x(y)) & if\ x\ is\ a\ digraph\ in\ y, \\ TR(x(y)) & if\ x\ is\ a\ trigraph\ in\ y, \\ FR(x(y)) & if\ x\ is\ a\ fourgraph\ in\ y. \end{cases}$$

and $p(z) \in [1, 2]$ is a parameter that defines the interval around $z(x(y))$. Note that we write z without arguments in $p(z)$ because we are referring to the type of feature and not to some specific feature, meaning that features of the same type (for example dwells) all have the same parameter value to define his acceptance neighbourhood.

For each feature $x \in AS(l)$ that is shared with the merged samples, the acceptance feature function is defined as

$$A(x) = \begin{cases} 1 & if\ z(x(AS(l))) \in V(x(DB)), \\ 0 & otherwise. \end{cases}$$

Defining $N(z)$ as the number of shared features between $AS(l)$ and DB of the type z (for example, if $z = DW$ then $N(z)$ is the number of all shared dwells between the attempt sample and the merged sample), we write the ***absolute score*** as

$$Ab(AS(l)) = \sum_{z \in Z} \left(w(z) \left(\frac{1}{N(z)} \sum_{x \in I_z} A(x) \right) \right),$$

with $\sum w(z) = 1$, where $w(z)$ is the weight parameter associated to each type of feature. Note that the formula proposed by Monrose and Rubin[3] is a particular case of the absolute score defined here.

4.3 Relative scores

The relative score value is based in time disorder divided by the maximum disorder. For each type of features in our $AS(l)$, a list is created ordering all features by time. Then this order is compared with the one from DB. Let us define $D(z, AS(l))$ as the function that give us the disorder between all shared features of the type z in $AS(l)$ with DB(see [8], 5.3.2 for more details). Then, the ***relative score*** is written as

$$Rl(AS(l)) = 1 - \sum_{z \in Z} \frac{D(z, AS(l))}{D_m(z, AS(l))},$$

where $D_m(z, AS(l))$ is the maximum possible disorder between $AS(l)$ and DB for some z type of feature.

4.4 Decision criterion

The value to compare with a fixed threshold is a linear combination of the two quantities defined before:

$$S(AS(l)) = w_{Ab} Ab(AS(l)) + w_{Rl} Rl(AS(l)), \tag{1}$$

where w_{Ab} and w_{Rl} are weights.

We define 3 thresholds for different intrusion levels: yellow, orange and red level represented by th_y, th_o and th_r, respectively. The intrusion is detected when

$$S(AS(l)) < th_y.$$

4.5 Parameter space

The parameter space is composed by $p(z)\ \forall z \in Z$, $w(z)\ \forall z \in Z$, th_y, w_{Ab}, w_{Rl} and l. The ***trivial configuration*** is the configuration where each event has the same weight as well as each score function.

5 Validation

All results presented in this section come from a real environment composed, most of the times, by more than 10 users. The users are, for the most part, software developers, meaning their input can result from coding and/or writing, both formally and informally, in English and Portuguese, in several distinct software environments. In some sense it is the worst case scenario to classify behaviour since there is a great variety on the user actions.

According to initial requirements we assume no other users have access, i.e., intruders are internal users trying to circumvent access control rules.

5.1 Tool description for artificial attacks

On the database, each user has his own merged sample with all characteristic features. This merged sample is the user profile previously saved and constructed using the user samples also saved on the database.

With all samples from the database we can test the algorithm. For a particular user, we can simulate authentications using the samples from the database against the merged sample of the same user. In that way we can calculate the FRR. Also, we can calculate the FAR of the entire group using all samples against each merged sample. It is an *artificial attack* because people are not intentionally attacking but it is the easiest way to simulate a large number of attacks and it is according to the scenario specification.

5.2 Acceptance neighbourhoods study

Different configurations for the acceptance neighborhoods of each type of feature

	p(DW)	p(FL)	p(DI)	p(TR)	p(FR)
A	1.12	1.22	1.20	1.18	1.12
B	1.14	1.23	1.22	1.20	1.15
C	1.15	1.25	1.25	1.25	1.18
D	1.17	1.27	1.27	1.27	1.20
E	1.15	1.30	1.26	1.22	1.18

increasing security

Threshold \ p(z)	A	B	C	D	E
0,06	5,64% / 0,90%	4,10% / 1,15%	3,59% / 1,24%	3,08% / 1,62%	3,08% / 1,32%
0,075	2,05% / 1,84%	1,54% / 2,26%	2,05% / 2,48%	1,54% / 2,95%	1,54% / 2,52%
0,09	2,05% / 3,16%	1,03% / 3,68%	1,54% / 3,89%	0,51% / 4,66%	1,03% / 3,97%
0,105	1,03% / 4,87%	0,51% / 5,30%	0,00% / 5,77%	0,51% / 7,09%	0,51% / 5,98%
0,12	0,00% / 7,22%	0,00% / 7,86%	0,00% / 8,85%	0,00% / 10,04%	0,00% / 9,10%

FRR values
FAR values

Fig. 2. Summary of the acceptance neighbourhoods study. All FRR and FAR values, for each fixed configuration (A, B, C, D *and* E) and for each fixed threshold, were obtained from 11 users each with 15 samples of 750 keys.

Acceptance neighbourhoods are controlled by the parameters $p(z)$. Considering $w(z) = 0.2\ \forall z \in Z$, $w_{Ab} = 0.5$ and $w_{Rl} = 0.5$ (trivial configuration), what we study first is the "best" (heuristically speaking) acceptance neighbourhoods for each type of features looking at the FRR and FAR calculated from artificial attacks. Fig. 2 is a summary of the best scenarios tested with $l = 750$. The first

column of the second table represents different thresholds to detect intrusion. As an example, for the value 0.06, th_y is defined as $th_y = (S(DB) - 0.06)$, where $S(DB)$ is the score from the user profile.

As we can see from Fig. 2, when we increase the threshold we get a high FRR and when we decrease the threshold the FAR tends to increase. Situation A produces a very high FRR with high security level (threshold= 0.06) and situation E produces a very low FAR with low security level (threshold= 0.12). The table shows that situation B is the one that produces the best results to minimize FAR and FRR.

5.3 Transients study for sample reduction

This part of the study allows us to understand if it is possible to reduce the size of the attempt sample and how much we can reduce. The more information we have, the more accurate are our decisions. On the other hand, the fewer keys the intruder types, the less damage the intruder does before being detected. The aim is always to find best of both parts.

Transient is a common term in differential equations theory and essentially represents states that actually are not the common states on some dynamical system. A simple example is throwing a rock to a lake: The waves produced by the rock is a transient state since most of the time the lake has no waves. The classical example for transients is the harmonic oscillator[11]. In our mechanism, we have

$$S(AS(l))_{l \to \infty} \longrightarrow a$$

with a representing the characteristic value for the user (in theory close to 1). Here, the transients are the values we get when l is too small and, consequently, are not user representative. Identifying the minimum l for which we do not have any more transients means identifying the minimum size of the sample to get a reasonable value for a.

Fig. 3 shows the transients study using 3 samples with 750 keys each from 3 different users. The horizontal axis represents sample reduction process and the vertical axis is the score defined in (1). In each iteration of the reduction process, we erase 11 or 12 features from the samples (on the same proportion as they exist for any sample with length l). This figure shows that after 11 sample reductions, the value of $S(AS(l))$ starts to be a bit unstable. Using 11 as the maximum number of reductions for the trivial configuration means that we can reduce the sample size in 40% (around less 300 keys) but 450 keys is still a considerable number for an attempt sample.

The question we would like to answer is: It is possible to get a better reduction result with a different parameter configuration? Next section we answer this question with an heuristic study of the weights.

5.4 Weights study

The study of weights has three distinct phases. The first one is the absolute and relative weights study. With the good weights from the first phase a study

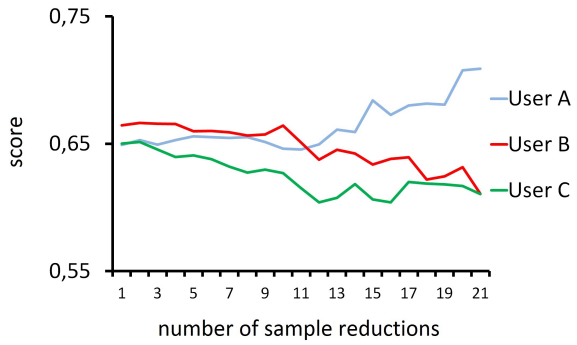

Fig. 3. Transients study. The horizontal axis represents the number of reductions that were made on each sample. The vertical axis represents the scores defined in (1) for each sample in different reduction phases. The more reductions we do, the smaller is the sample. The initial sample size is 750 keys and this picture shows samples from 3 different users, the reduction process and the score in each reduction phase.

of the weights of each type of feature in the absolute score is done (the reason we are only considering the absolute score is justified by the first phase study). Finally, we check if, after the second phase, we still have the same weight results for absolute and relative scores. There might be more efficient ways to conduct this study like using genetic algorithms[12] but, apart from the fact that this approach takes much less time, even using this simplified technique, the results should be very similar to the ones from a genetic algorithm strategy.

In Fig. 4, the chart on the left shows how the gap between the average intrusion score and the average user score increases at the same time as we increase the weight of the absolute score. On the horizontal axis, we start with $S(AS(l))$ calculated using only the relative score. Then we increase the weight of the absolute score until we have $S(AS(l))$ calculated using only the absolute score. The gap represents how much separate is the intrusion score region and the user score region. The chart on the right shows some stability when the absolute score weight is more than 70%. We conclude, among all the distinct combinations, that $70\% - 30\%$ or even $80\% - 20\%$ are the best configurations for $w_{Ab} - w_{Rl}$ to get an higher gap and, at the same time, to consider the relative score in our evaluation.

Fig. 5 shows a summary of many simulations, using $80\% - 20\%$ for $w_{Ab} - w_{Rl}$, for many different parameters $w(z)$ $\forall z \in Z$. From here we observe an important fact: dwells, flights and digraphs are the most efficient type of features to identify or to authenticate the user. On the other hand, in a real environment, when we try to isolate the most important type of feature (dwell) the FAR typically tends to increase because the validation mechanism tends to be more sensible. We conclude that $(42\% - 24\% - 16\% - 10\% - 8\%)$ is, heuristically speaking, the

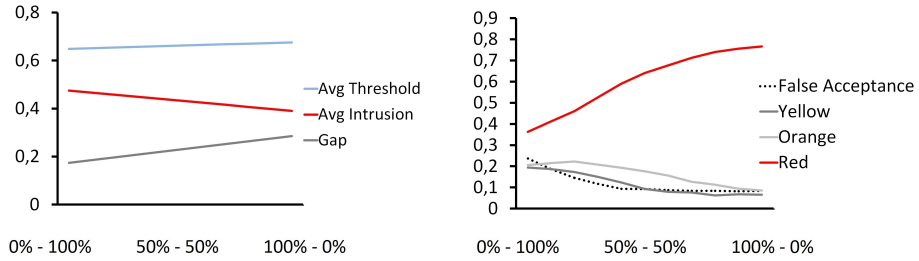

Fig. 4. Absolute and relative weights study for user samples with 750 keys. The gap between the typical intrusion score and the typical user score increases whenever we increase the importance (weight) of the absolute score. On the other hand, it is important to have more than one measure to evaluate identities, we should not ignore the relative score at all. Instead of ignoring it, we should decrease his importance.

best configuration for $w(z)$ and with this configuration the results from the first phase (Fig. (4)) were exactly the same.

weights (dwell-flight-di-tri-four)	20-20-20-20-20	40-25-18-12-5	20-40-23-12-5	8-22-40-22-8	5-12-23-40-20	5-12-18-25-40
Avg Threshold	0,67	0,70	0,67	0,65	0,64	0,63
Average Intrusion	0,40	0,40	0,40	0,41	0,41	0,40
Gap	0,26	0,30	0,26	0,24	0,23	0,23

weights (dwell-flight-di-tri-four)	60-15-10-8-7	50-20-12-10-8	42-24-16-10-8	34-27-19-11-9
Avg Threshold	0,74	0,72	0,70	0,69
Average Intrusion	0,39	0,40	0,40	0,40
Gap	0,34	0,32	0,30	0,29

Fig. 5. Features weights study.

Finally, the transient study was repeated but this time taking into account the previous best configurations (the best acceptance neighbourhood parameters and the best weights previously presented). Fig. 6 shows $S(AS(l))$ for different values of l. As we can see, the value stays stable during all the reduction process (starting with $l = 750$). After 20 reductions, the value shows no relevant fluctuations. This means a reduction of 80%, equivalent to a reduction of around 600 keys. At this point we are able to use attempt samples with 150 keys.

5.5 ROC curve

The Receiver Operating Characteristics (ROC) curve illustrates the performance as its discrimination threshold is varied. We present, in Fig. 7, a pseudo ROC curve using the best parameters configuration presented here and attempt samples with 250 keys ($l = 250$). At this point we are able to produce rates close to 2%. Next section we present a simple way to reduce even more these rates.

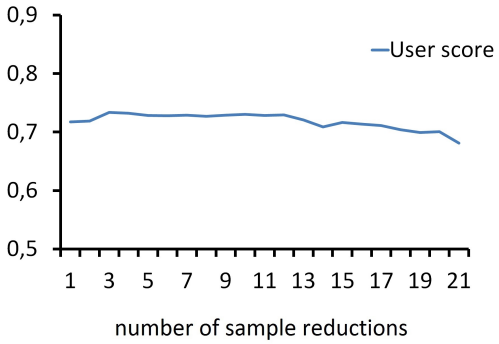

Fig. 6. Transients study with an improved configuration and considering only one user. The initial sample size is 750 keys.

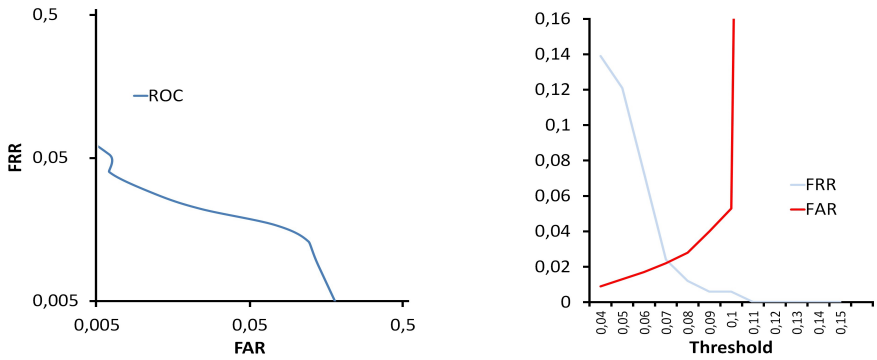

Fig. 7. Pseudo ROC curve (left) with logarithmic scale and DET curve (right) for an improved configuration. The sample size is 250 keys and all simulations were done using a group of 11 real users.

5.6 Evaluations scheme

One of the main focus in any continuous detection system using keystroke dynamics is to reduce the size of the required sample to detect intrusions and, at the same time, not increase FRR and FAR. For a static biometric system it is important to know how often a wrong decision is made but the purpose of a performance evaluation for a continuous biometric authentication system is not to see if an impostor is detected, but how fast he is detected[13].

An interesting way to detect intrusions fast is dividing the evaluation process in more than one part. Let us suppose that, after each l keys, we want intrusion detection probability p. So, p is what we want after l keys and the question is: What should we have in $l/2$ keys to achieve p in l keys?

To simplify our example, let us divide the process in only two parts and let us assume that $l/2 \in N$. If we define $X =$ "$number\ of\ authentications\ in\ 2\ attempts$" then $X - Bi(2, m)$ with

$$P(X = k) = \binom{2}{k} m^k (1-m)^{2-k},\ k = \{0, 1, 2\}$$

If we define $Y =$ "$detect\ intrusion$" then we want to know m for $P(Y) \approx p$:

$$P(X \geq 1) \approx p \Leftrightarrow 1 - P(X = 0) \approx p \Leftrightarrow 1 - (1-m)^2 \approx p \Leftrightarrow m \approx 1 - (1-p)^{1/2}.$$

So, $m < p$ for $p \neq \{0, 1\}$ meaning that we do not need p probability after each $l/2$ keys but usually much less! As a consequence, we do not need to be so accurate with $l/2$ keys to have probability p in l keys and we have the opportunity to catch the intrusion in less than l keys(in our example, in $l/2$). Another direct consequence of not being so accurate is the fact that the FRR decreases.

As a conclusion, if we divide the process in two parts then we just need to have intrusion probability m (less than p) and we automatically have a lower FRR. Also, during this process, the yellow (th_y) and orange (th_o) warnings are ignored and only red warnings are considered intrusion. At the end of the process (in this example after each 2 evaluations) some particular situations with yellow and orange warnings are considered intrusion. All these considerations on the yellow and orange warnings will help us on the FRR reduction.

We already have some good and stable results using this approach and considering evaluations after each 125 keys but it is still a work in progress. Fig. 8 shows DET curve for 125 keys and for 250 keys (process divided in two parts) from a real scenario of users (11 users) and considering the best configuration, presented here, for the parameters.

6 Conclusions

The purpose of this paper is to present improvements of some of the keystroke dynamics biometrics to identify/authenticate users and to detect intrusions, using data from a real environment to validate.

The main conclusions of this work are the huge importance of dwells (time difference between key up and key down for any key) to identify/authenticate

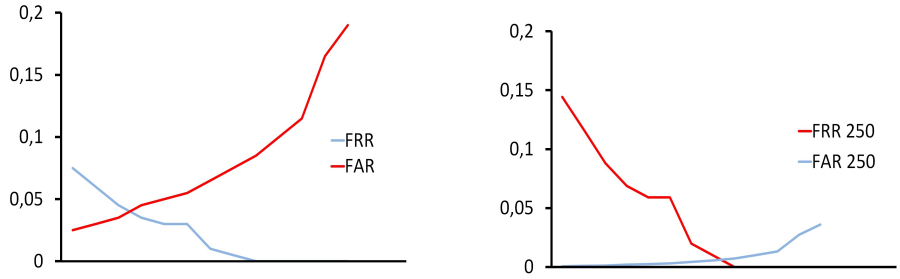

Fig. 8. DET curve at the middle of the evaluation process and with 125 keys (left) and DET curve at the end of the process and with 250 keys (right). In this example the process was only divided in two parts.

users. Also, the importance of the right weights on all features to reduce the size of the attempt sample (sample used to verify user identity or to detect intrusion) and to reduce FAR and FRR at the same time. In our particular scenario, we were able to get an amazing reduction of the sample size from 750 keys to 150 keys with FRR and FAR close to 2%. Also, using an evaluation scheme we were actually able to get an impressive FRR and FAR lower than 1% with the strong possibility to detect the intrusion after only 125 keys.

It is important to refer that in [8] all the presented simulations are with samples of 750 keys and what we present here is a huge reduction of the sample size which is crucial to have an earlier intrusion detection. Also, Kevin Killourhy and Roy Maxion [4] referred, in a recent study, that at present no anomaly detector has archived a false alarm rate specified in the european standards which makes our results even more interesting.

We believe that the strategy presented here to calculate the weights in order to improve the FAR, FRR and attempt sample reduction is something that can be implemented in any group of users. Also, the weights study can be done periodically, continuously calculating the best parameters for an specific group of users.

Acknowledgements. This work has been supported by FCT - Fundaçaão para a Ciência e Tecnologia within the Project Scope: PEst-OE/EEI/UI0319/2014. We would like also to thank Watchful for supporting this work and the Software Development Team for helpful comments and data. The authors had financial support from Watchful Software.

References

1. Lynch, D.M., Securing Against Insider Attacks, Information Security Journal: A Global Perspective, volume 15, page 39–47 (2006)

2. Schulz, E.E., A Framework for Understanding and Predicting Insider Attacks, Computers and Security, volume 21, page 526–531 (2002)
3. Monrose, F., Rubin, A., Keystroke dynamics as a biometric for authentication, Future Generation Computer Systems, volume 16, page 351–359 (2000)
4. Killourhy, K., Maxion, R., Comparing Anomaly-Detection Algorithms for Keystroke Dynamics, Dependable Systems and Networks, page 125–134 (2009)
5. CENELEC European standard EN 50133-1, Alarm systems. Access control systems for use in security applications, European Committee for Eletrotechnical Standardization (2002)
6. Bergadano, F., Gunetti, D., Picardi, C., User authentication through keystrokes dynamics, ACM Transactions on Information and Systems Security, volume 5, page 367–397 (2002)
7. Magalhes, S., Keystroke dynamics stepping forward in authentication, GESTS International Transactions on Computer Science and Engineering, volume 29 (2006)
8. Ferreira, J., Santos, H., Patrão, B., Intrusion detection through keystroke dynamics, 10th European Conference on Information Warfare and Security (2011)
9. Al Solami, E., Continuous Biometric Authentication: Can It Be More Practical?, 12th IEEE International Conference on High Performance Computing and Communications, page 647–652 (2010)
10. Ferreira, J., Santos, H., Keystroke dynamics for continuous access control enforcement, Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge, page 216-223 (2012)
11. Serway, R.A., Jewett, J.W.: Physics for Scientists and Engineers (2003)
12. Mitchell, M.: An Introduction to Genetic Algorithms(Complex Adaptive Systems) (1998)
13. Boruc, P., Continuous keystroke dynamics: A different perspective towards biometric evaluation, Information Security Technical Report, volume 17, page 36–43 (2012)