

Keystroke Authentication with Adaptive Dynamics

Annapurna P Patil, Aman Mehta, Ayush Kumar, Bineet Kumar, Deepak Jayaprakash
Department of Computer Science and Engineering
M S Ramaiah Institute of Technology
Bangalore -560054, India
annapurnap2@msrit.edu
deepkajpkodlekere@gmail.com

Abstract

The demand for modern tools and techniques to restrict access to applications and services which contain delicate data is increasing exponentially each year. Traditional methods such as PINs, tokens, or passwords fail to keep up with the challenges presented because they can be lost or stolen, which compromises the system security.

On the other hand, biometrics based on “who” is the person or “how” the person behaves, present a significant security advancement to meet these new challenges.

But even the most powerful cryptographic systems fail to prevent unauthorized access since they are static and identify the validity based on a phrase that the user must have provided. Therefore an intuitive, reliable and precise way of recognizing the identity of users based on intrinsic parameters is being researched.

Biometrics, defined as the physical traits and behavioral characteristics that make each of us unique, are a natural choice for identity verification. Biometric attributes become the most optimal and ideal candidates for authentication since they cannot be stolen, lost or impersonated.

Keywords : Keystroke dynamics, Adaptive models, Distance based measures, Leave-One Out-Method.

1. Introduction

Two main causes which pose threat to data, digital network and computer system security are fraud and impersonation. Many web based authentication systems have been proposed to safeguard commercial transactions and to secure data. Ideas such as user ID's and passwords, IP address filtering, message digest authentication, etc. are the popular ones.

However, these systems are far from perfect. For example, if a password becomes compromised it is no longer adequate for authenticating its rightful owner. In the

hope of improving on this, there exists ongoing research into utilizing the idiosyncrasies of a user's interaction with a computer as a form of authentication.

The most promising approach has been Keystroke biometrics which refers to the **habitual patterns or rhythms** an individual exhibits while typing on a keyboard input device. Compared to other biometric schemas, keystroke has the primary advantages that:

- 1.No external hardware like scanner or detector is needed. All that is wanted is a keyboard.
- 2.The “rhythm” or the pattern of the users is a very reliable statistic.
- 3.It can easily be deployed in conjunction with existing authentication systems.

The keystroke authentication approach has been divided into two. Most of the existing approaches focus on “**static**” verification, where a user types specific pre-enrolled string, e.g., a password during a login process, and then their keystroke features are analyzed for authentication purpose. The second one is called as “**free-text**” dynamics which does not have a pre-determined strong. It adapts to the typing pattern on the go. For more secure applications, free text should be used to continuously authenticate a user.

2. Literature Survey

Keystroke Dynamics has become a widely researched and active area due to the increasing importance of cyber security and computer or network access control. Most of the existing approaches focus on static verification, where a user types specific pre-enrolled string, e.g., a password during a login process, and then their keystroke features are analyzed for authentication purposes^[5]. Only a few research studies address the more challenging problem of keystroke biometrics using “**free text**”, where the users can type

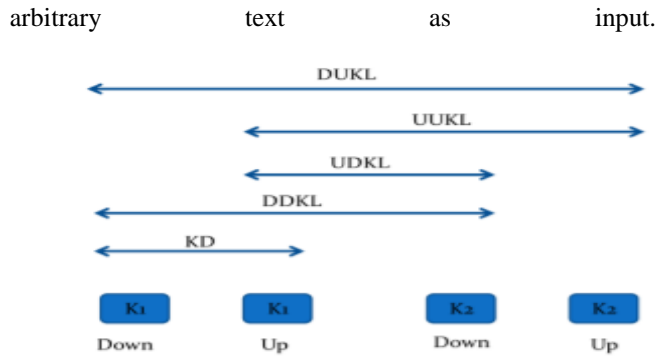


Fig 2.1: Keystroke timing features

Keystroke dynamics features are usually extracted using the timing information of the key down/hold/up events. The hold time or dwell time of individual keys, and the latency between two keys, i.e., the time interval between the release of a key and the pressing of the next key are typically exploited.

The features extracted from keystroke dynamics pattern in most of researches are timing features. Fig shows the extracted timing features:

1. **Key Hold (KD):** key delay between pressed and key released.
2. **Down-Down Key Latency (DDKL):** time in between two consecutive presses.
3. **Up-Up Key Latency (UUKL):** time between two successive releases.
4. **Up-Down Key Latency (UDKL):** time in between the current key release and the next key press.
5. **Down-Up Key Latency (DUKL):** time between the current key press and the next key release.

Research work on keystroke dynamics all originated from Gaines et al. [8] who did a preliminary study authentication using the T-test on digraph features.

Monrose and Rubin [22] few years later extracted keystroke features using the mean and variance of both digraphs and trigraphs.

Then there were statistical Euclidean distance metrics with Bayesian-like classifiers identified 92% for their small dataset containing 63 users correctly. Over the years, keystroke biometrics research has been implemented in many existing machine learning algorithms and classification techniques.

Researchers have presented their work on choosing different distance metrics, such as the Euclidean distance, the Mahalanobis distance and the Manhattan distance and have been explored their suitability on the biometric authentication. For the implementation both classical and advanced classifiers have been used, including K-Nearest Neighbours (KNN) classifiers [4], Bayesian classifiers, K-

means methods [12], Fuzzy logic, neural networks , and support vector machines (SVMs).

A promising research effort in applying keystroke dynamics as a static authentication method originated from the work of Joyce and Gupta [14]. Their approach is relatively simple and yields impressive results.

The main idea of our work is to allow users to access different systems by typing their own usernames and passwords as usual. Then, the users' typing styles features are extracted from their passwords, so there is no additional text required for authentication.

3. Proposed System

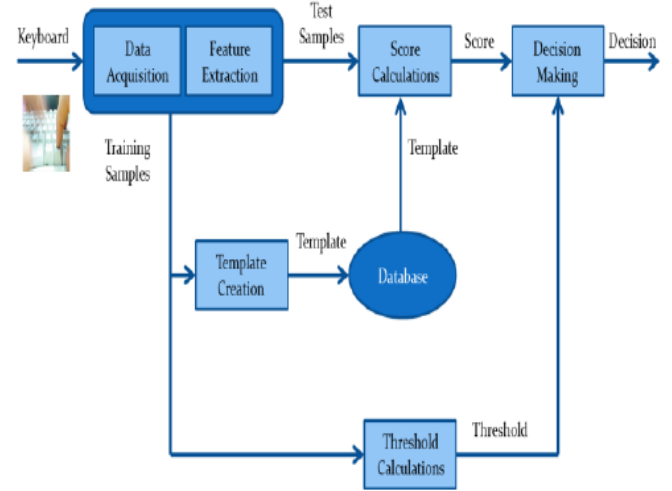


Fig 3.1: Model proposed

The problem with the existing work and implementations related to keystroke authentication based on static text is that the statistic chosen and the model built are not very accessible and compatible with each other. Therefore we propose an easier and a much simpler model and metric to achieve the desired classification which has better interpretability as shown in **Fig 3.1**.

The whole model can be divided four distinct steps. These are listed as follows:

1. The individual register their name and password with the database. Then the user has to type his username and train the machine for six times.
2. Features are extracted when individuals press and release keys. More specifically the delay between the key-down and key-up time.
3. The algorithm is applied and the threshold is generated based on the variations that the user has done while typing the 6 training set. Hence, the adaptiveness.
4. Calculate the Euclidean distance between training and the test samples to get the user's score.
5. Finally, the user's score is compared against its threshold to make the decision. If the Euclidean

measure generated from the test sample is too high when compared to the training set then the user is classified to be an imposter.

4.1 Some key steps and their brief description :

4.1.1 Data and feature extraction:

A dataset is created to evaluate the proposed system. A software application validates the entered data at the time of registration and the credentials are implemented to acquire samples from individuals and extract their features. The user has to simply type his username and passwords that they can comfortably type and the rhythm of which they can easily remember.

Time stamps of each key press (Down) and release (Up) are stored in a logout file and used to calculate KD, DDKL, UUKL, UDKL, and DUKL. For our model we take the key delay between the key up of the current stroke and the key up of the next stroke.

These differences become the attributes of our data set and determine the class labels of our machine learning algorithm. Typically these key delays are stored in a comma separated value fashion.

4.1.2 The metric or the statistic chosen for comparison:

For testing the efficiency and the correctness the two statistics that we chose were Manhattan distance and the Euclidean distance.

Manhattan distance:

The score is calculated as in **Eq. (1)** represents Manhattan distance

$$M = \sum_i^n (x_i - y_i) \quad (1)$$

Where $x = (x_1, x_2, \dots, x_n)$ represents test vector and $y = (y_1, y_2, \dots, y_n)$ represents the mean vector of the training samples

Euclidean Distance

The score is calculated as the squared Euclidean distance between the test vector and the mean vector as in the following **Eq (2)**

$$E = \sqrt{\sum_i^n (x_i - y_i)^2} \quad (2)$$

(2)

Other optimal choices for the distance measures could also be

1. Manhattan with Standard Deviation Distance:

The standard deviation of each feature is calculated as well. Eq. (3) will be in the form:

$$Ms = \sum_i^n (x_i - y_i) / \alpha_i \quad (3)$$

2. Mahanabolis Distance:

The standard deviation of each feature is calculated, where the Mahanabolis distance is presented by Eq. (4)

$$Mh = \sqrt{\sum_i^n ((x_i - y_i) / \alpha_i)^2} \quad (4)$$

4.2 Threshold Calculation:

The threshold calculation is what makes the model adaptive and different than other existing models and algorithms. The window for error is the space in which he is permitted to cause any errors. This is decided by a method called **Leave One-Out-Method (LOOM)**. This method is explained below in some detail in steps:

1. Out of the n samples, divide the training space of (n) samples to one sample used as test sample, and $(n-1)$ samples used to create the training sample.
2. Apply a distance measure (Euclidean in our model) to calculate the distance between the selected test sample and the mean vector of the $(n-1)$ training samples.
3. Iterate the step 2 for (n) times to produce (n) different thresholds for each feature vector.
4. The average of these (n) thresholds is calculated to produce the one single threshold that would represent the effective measure of all the thresholds in total.
5. These steps are repeated to calculate the individual thresholds for the other three distance measures.

4.3 Visual Equivalence:

This threshold calculation and also the working of the algorithm can be clearly stated as a visual representation as follows. Assume that the space is made up of 6 data points each of them which would stand for a set of attribute array in our database. Now pick a data point at random and calculate the distance of this particular point from each of the rest of data points.

That would give the threshold that must exist for this data point. Now choose another data point and repeat this calculation. At the end we would end up with six different difference vectors. Now take the average of the vectors and conclude to a single point in space. This point is cumulative distance equivalent of all the vectors combined.

Imagine a sphere centered at this point. The threshold calculated would be the radius of this sphere. If a test data point arrives, we plot this point in space. We then check if this point is inside the so formed sphere. If it does then it is equivalent to a data set which is of a valid user and its delay array is within bounds of error. If it doesn't, then it would mean that the data set belongs to that of an imposter and the delay discrepancy is beyond the margin allocated.

4.4 Decision making:

The proposed system's is evaluated using two statistics. These metrics are listed as follows:

1. **False Rejection Rate (FRR)**, which is the refused fraction of genuine individuals, and
2. **False Acceptance Rate (FAR)**, which is the accepted fraction of imposter individuals.

Eq. (5) and (6) shows FRR and FAR respectively.

$$FRR = \frac{\text{Number of refused genuines}}{\text{Total number of genuines}} \quad (5)$$

$$FAR = \frac{\text{Number of accepted imposters}}{\text{Total number of imposters}} \quad (6)$$

The biometric system performance could be measured using Equal Error Rate (EER) which refers to the point on the ROC (Receiver Operating Characteristic) curve where the FAR and the FRR are equal

5. Implementation

5.1.1 Design and technology

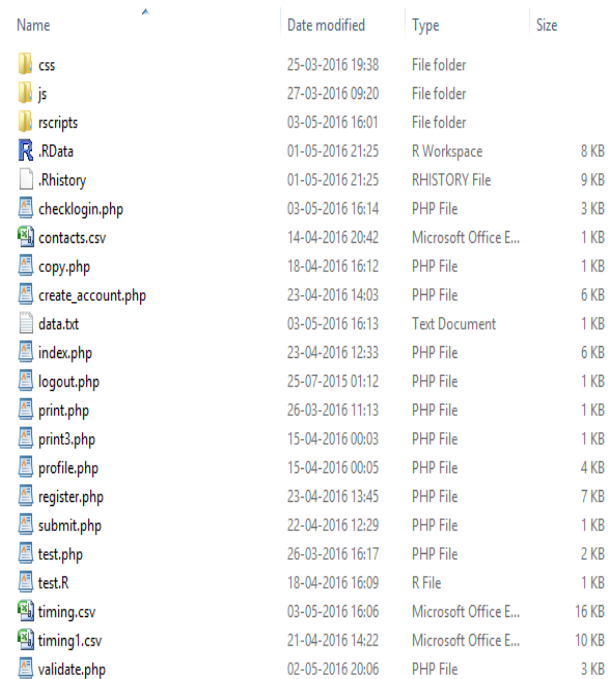
Using the design described in the above section the experiment software was successfully built, tested and used to gather data. Rather than give an in-depth analysis of the code, we shall provide an more informative overview of the technologies we used, and describe the interaction with the software that volunteers experienced.

We built the experiment software as a web application using the following technologies:

1. **HTML 5, jQuery, Bootstrap:** For the front end and creation of forms to accept the data from the user
2. **JavaScript:** To perform the front end validations and also to gather the key up time from the user
3. **Ajax:** To parse the timing data from the front-end JavaScript to the PHP
4. **PHP:** To perform all the file operations and IO
5. **R:** To build the model and to calculate the threshold.
6. **MySQL:** To store the user id's and the passwords and to maintain session information

The software was designed to be very modular. This means that if similar experiments are required, the software can very easily be re configured to with different groups, passphrase and schedules. The volunteer was authenticated with the site using their username and a password. (This is same phrase that we use in learning). Once authenticated they were directed to a page containing a javascript client which allowed them to perform the experiment.

The **directory structure** used for the project is as shown in **Fig 5.1**.



Name	Date modified	Type	Size
css	25-03-2016 19:38	File folder	
js	27-03-2016 09:20	File folder	
rscrip	03-05-2016 16:01	File folder	
.RData	01-05-2016 21:25	R Workspace	8 KB
.Rhstory	01-05-2016 21:25	RHISTORY File	9 KB
checklogin.php	03-05-2016 16:14	PHP File	3 KB
contacts.csv	14-04-2016 20:42	Microsoft Office E...	1 KB
copy.php	18-04-2016 16:12	PHP File	1 KB
create_account.php	23-04-2016 14:03	PHP File	6 KB
data.txt	03-05-2016 16:13	Text Document	1 KB
index.php	23-04-2016 12:33	PHP File	6 KB
logout.php	25-07-2015 01:12	PHP File	1 KB
print.php	26-03-2016 11:13	PHP File	1 KB
print3.php	15-04-2016 00:03	PHP File	1 KB
profile.php	15-04-2016 00:05	PHP File	4 KB
register.php	23-04-2016 13:45	PHP File	7 KB
submit.php	22-04-2016 12:29	PHP File	1 KB
test.php	26-03-2016 16:17	PHP File	2 KB
test.R	18-04-2016 16:09	R File	1 KB
timing.csv	03-05-2016 16:06	Microsoft Office E...	16 KB
timing1.csv	21-04-2016 14:22	Microsoft Office E...	10 KB
validate.php	02-05-2016 20:06	PHP File	3 KB

Fig. 5.1: Directory Structure

5.2 Experiment

The user first chooses a user name and a pass phrase as shown in **Fig 5.2.1** which would be the same password he would be using to train the machine. The user is then logged into a page and is asked to go to the logistics page where the training phase happens. Then once he logs out. The next time the user tries to login the algorithm runs

and the test data that is the current attempt is recorded and tested with the database as shown **Fig 5.2.2**

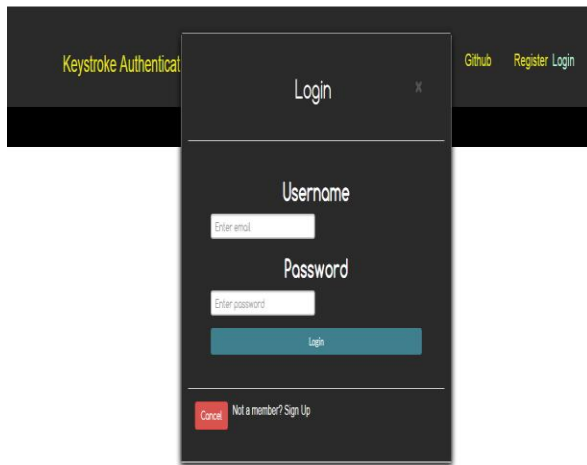


Fig 5.2.1: Login Page

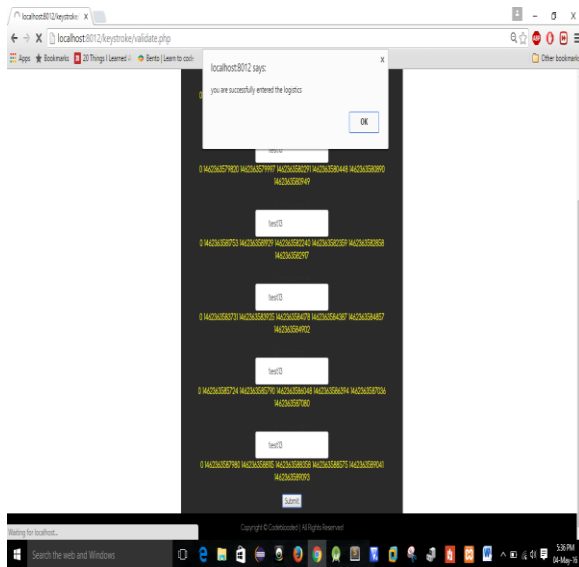


Fig 5.2.2: Entry of Statistics

6. Comparison with existing work

Four datasets were used for evaluation of the comparative efficiency of our system; the first one is by Yu Zhong (2012); the second one being CMU by Kevin S. Killourhy (2009); the third being Shima I. Hassan (2013) and the fourth one being our model.

Killourhy and Maxion [2] used 14 keystroke dynamics anomaly detector to authenticate users, 11 of which were previously proposed, and 3 were classic recognition patterns using various distance statistic models. (Euclidean, Manhattan, and Mahalanobis distance measures). Their dataset composed of 51 users, who typed the password for 400 times along 8 sessions, i.e., 50 times per session, out of which 200 samples were taken for training the model, and the rest were used for testing the

model built. The features from each sample included DDKL, UDKL and KD and achieved an EER of 9.6%.

Yu Zhong et al. [1] evaluated a keystroke authentication based on a new distance metric, i.e., by combining Mahalanobis distance and Manhattan distance on the keystroke dynamics dataset created in (CMU Dataset) [2]. They used Nearest Neighbor classifier with their new distance metric to authenticate the user to achieve an EER of 8.4%.

Shima I. Hassan, Mazen M. Selim, and Hala H. Zayed [3] used the concept of majority voting [4]. If there are n features, the input sample is assigned an identity when at least k of the features agree on that identity, where $k = (n/2) + 1$ if n is even and $k = (n+1) / 2$ if n is odd. It first authenticated based on feature separately, and UDKL produced 8.8% for EER using Manhattan with standard deviation.

Finally, individuals are authenticated based on majority voting (MV), the best results is 7.0 % for EER using Manhattan with standard deviation and MV.

Our model used two distance statistics: Manhattan distance and Euclidean distance. Our dataset composed of 40, who typed the password for 6 times and authenticated using the model for 21 times during 3 sessions.

The model build consists of two distance measures :

1. **Manhattan distances:** Takes the average of all the training data set and compares with a threshold of 150 ms.

i) This model is less adaptive as irrespective of the user, the threshold is fixed and the model is not properly built according to the typing pattern of the user.

ii) As per our results, this model is more flexible when compared to the other model. The major reason behind that is while training the model, the user types the same password six times and hence the training model built using Euclidean distances is very small and precise, this means that during authentication, the user must type with the same pattern without even milliseconds of variation in the pattern which is quite inhuman. The acceptance sphere will be small as shown in **Fig 6.1**.

2. **Euclidean distances:** This distance measure is quite adaptive compared to the other model, that is the threshold completely depends upon the training data and is not fixed to some constant value.

i) It takes the first sample as test and the rest five as training sets and finds the anomaly distance. The same process is repeated for different test sample (second sample, third sample, etc.) and the rest as training example. The anomaly distance of all these six models are taken and the average of them gave the threshold of our model.

ii) Major advantage of this model is when the sample given during the training differs, i.e., in a humane or a natural way, not significantly change as if a different user is typing (that will oppose the objective of our project), the acceptance sphere will become large compared to the Manhattan distance model.

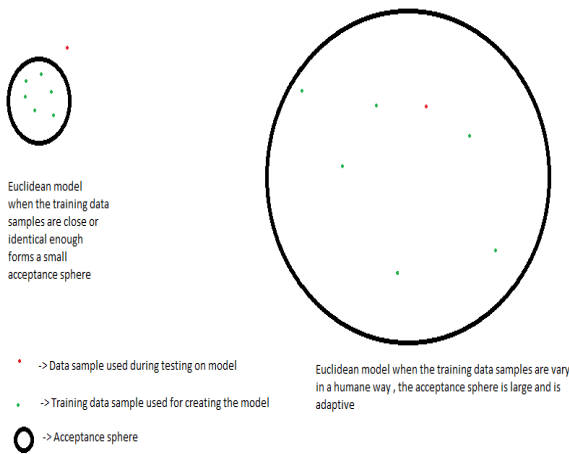


Fig 6.1: Euclidian Distance Model

Table 6.1: EER for the two distance models for our proposed system:

Distance measure	EER
Manhattan distance	8.9
Euclidean distance	9.8

Table 6.2: Comparison of models:

System	EER
Yu Zong (2012) [1]	8.4
Kevin S. Killourhy (2009) [2]	9.6
Shimaa I. Hassan(2013)[3]	7.0
The proposed System	8.9

7. Conclusions and future work

We studied the characteristics of keystroke dynamics for user authentication and proposed a new adaptive model and statistic which would change the threshold according to the user's dissimilarity in his typing patterns. As outliers and data correlations are typical in keystroke dynamics data, it is not surprising that classifiers using the new distance metric outperform existing top performing keystroke dynamics classifiers which use traditional distance metrics.

Although we applied the new combination of distance metric and adaptive model to the problem of matching keystroke dynamics features, there existed a few anomalies and false predictions. This can be attributed to the problem of over-fitting the data onto the model. If the typing pattern of the user is very much similar in each test case then the acceptance sphere formed has very less radius of threshold due to which the user may be asked to enter the logistics a couple of times. But there was an instance in which an imposter was recognized as a valid user. All the false predictions attributed the problem of over-fitting.

Therefore this problem can be overcome by ensemble learning methods. In our future work, we would

present an algorithm which would learn how to assign weights to an average model and the Euclidian model based on the case of over-fitting the threshold.

References

- [1] L. C. F. Ara'ujo, L. H. R. Sucupira, M. G. Liz'arraga, L. L. Ling, and J. B. T. Yabu-uti. . "User authentication through typing biometrics features", In Proc. 1st Int'l Conf. on Biometric Authentication (ICBA), volume 3071 of Lecture Notes in Computer Science, pp. 694–700, 2004.
- [2] F. Bergadano, D. Gunetti, and C. Picardi, "User Authentication through Keystroke Dynamics", ACM Trans. Information and System Security, 5(4), pp. 367–397, 2002.
- [3] S. Cho, C. Han, D. H. Han, and H. Kim. "Web-based keystroke dynamics identity verification using neural network", Journal of Organizational Computing and Electronic Commerce, 10(4):295–307, 2000.
- [4] Yu Zong and Yunbin Deng, Anil K Jain, "Keystroke Dynamics for User Authentication", Int'l Joint Conf. on Biometrics (IJC), pp. 1-5, 2011.
- [5] John Leggett and Glen Williams. Verifying identity via keystroke characteristics. **International Journal of Man-Machine Studies**. Journal of Organizational Computing and Electronic Commerce, 10(6):, 2000.
- [6] Kevin S. Killourhy and Roy A Maxion, "Comparing Anomaly-Detection Algorithms for Keystroke Dynamics Proc. of the 3rd Int. Conf. on Intelligent Information Hiding and Multimedia Signal Processing, pp. 61-64, 2007.
- [7] R. Joyce and G. Gupta, "Identity authentication based on keystroke latencies", Communications of the ACM, 33(2):168–176, 1990.
- [8] K. S. Killourhy and R. A. Maxion, "Comparing Anomaly Detectors for Keystroke Dynamics", in Proc. 39th Annual Int'l Conf. on Dependable Systems and Networks (DSN2009), pp. 125-134, 2009.
- [9] A. K. Jain, S. Pankanti, S. Prabhakar, L. Hong, and A. Ross, "Biometrics: a grand challenge", Proc. Int'l Conf. on Pattern Recognition, vol. 2, pp. 935–942, August 2004.
- [10] J. Leggett and G. Williams, "Verifying Identity via Keystroke Characteristics", Int'l J. Man-Machine Studies, vol. 28, no. 1, pp. 67–76, 1988.
- [11] Y. Li, B. Zhang, Y. Cao, S. Zhao, Y. Gao and J. Liu, "Study on the Beihang Keystroke Dynamics Database", Int'l Joint Conf. on Biometrics (IJC), pp. 1-5, 2011.
- [12] S. Prabhakar, S. Pankanti, and A. K. Jain, "Biometric Recognition: Security and Privacy Concerns", IEEE Security and Privacy Magazine, Vol. 1, No. 2, pp. 33-42, 2003.
- [13] T. Sim and R. Janakiraman, "Are digraphs good for freetext keystroke dynamics?", IEEE CVPR, pp. 17-22, 2007.

[14] R. O. Duda, P. E. Hart, and D. G. Stork. Pattern Classification. John Wiley & Sons, Inc., second edition, 2001.

[15] G. Forsen, M. Nelson, and R. Staron, Jr. Personal attributes authentication techniques. Technical Report RADC-TR-77333, Rome Air Development Center, October 1977.

[16] S. Haider, A. Abbas, and A. K. Zaidi. A multi-technique approach for user identification through keystroke dynamics. IEEE International Conference on Systems, Man and Cybernetics, pages 1336–1341, 2000.

[17] H.-j. Lee and S. Cho. Retraining a keystroke dynamics based authenticator with impostor patterns. Computers & Security, 26(4):300–310, 2007.

[18] J. Montalvao, C. A. S. Almeida, and E. O. Freire. Equalization of keystroke timing histograms for improved identification performance. In 2006 International Telecommunications Symposium, pages 560–565, September 3–6, 2006, Fortaleza, Brazil, 2006.

[19] PC Tools. Security guide for windows—random password generator, 2008. <http://www.pctools.com/guides/password/>.

[20] A. Peacock, X. Ke, and M. Wilkerson. Typing patterns: A key to user identification. IEEE Security and Privacy, 2(5):40–47, 2004.

[14] D. Mahar, R. Napier, M. Wagner, W. Lavery, R. Henderson, M. Hiron, Optimizing digraph-latency based biometric typist verification systems: inter and intra typists differences in digraph latency distributions, Int. J. Human-Comp. Stud. 43 (1995) 579–592.

[15] J. D. Woodward, N. M. Orlans, and P. T. Higgins. “Biometrics: Identity Assurance in the Information Age”, McGraw-Hill, New York, USA, 2003.

[16] S. Bleha, C. Slivinsky, and B. Hussien. Computer access security systems using keystroke dynamics. IEEE Transactions on Pattern Analysis and Machine Intelligence, 12(12):1217–1222, 1990.