www.ijemr.net ISSN (ONLINE): 2250-0758, ISSN (PRINT): 2394-6962

Volume-8, Issue-3, June 2018

International Journal of Engineering and Management Research

Page Number: 18-23

DOI: doi.org/10.31033/ijemr.8.3.1

A Study of Person Identification using Keystroke Dynamics and Statistical Analysis

Nikhil Ashok Hegde
Member Technical Staff, AERO Department, NetApp, Bangalore, INDIA

Corresponding Author: nikhilhegde20@gmail.com

ABSTRACT

In this paper, a basic study of closed-set identification using keystroke dynamics and simple statistical analysis has been carried out. Dwell time, flight time and one additional feature called key affinity are used as user-identifying features. The timing information is passed through a statistical layer to produce mean and standard deviation. This information is combined with key affinity to identify a rank-based person list. In conclusion, we compare the performance of this setup with other setups. This work aims to suggest that a keystroke dynamics system relying on pure statistics as its underlying algorithm may not be sufficiently accurate.

Keywords— Biometrics, Closed-set identification, Identity management systems, Keystroke dynamics, Ranking (Statistics)

I. INTRODUCTION

Keystroke dynamics is a type of behavioral biometrics. It uses timing patterns of a person to authenticate or identify a person. Timing patterns may include overall speed, common errors, time between key presses and length of time for which a key is pressed. Unlike other biometrics like retinal and fingerprint-based systems, keystroke dynamics is not as reliable. However, it can be implemented in a system as an additional layer of security. Conventionally, in systems solely protected by a password, a criminal who knows the system password will find it very simple to access and commit illegal activities. However, the criminal will find it more difficult to access a system which is protected by an additional layer of keystroke dynamics.

II. METHODOLOGY

A. Feature Extraction

The following metrics are extracted from user input:

1. Dwell time – The time interval for which a key is pressed down and released,
2. Flight time – The time interval between two key presses,
3. Key affinity – User preference to use shift or caps lock keys for special or uppercase characters.

In general, flight time can be of three types:

1. F1 – Time interval between key press and next key press,
2. F2 – Time interval between key release and next key press,
3. F3 – Time interval between key release and next key release.

In the setup presented in this paper, flight time of type F1 will be extracted from the user input. Key affinity feature is useful in those cases where a criminal is successful in replicating the victim's typing speed but was denied access because the victim preferred the right shift key and the criminal used the left shift key.

B. Person Template

This experimental setup is based on static identification. To train the system, the user typed a fixed pass-phrase text thirty time. The pass-phrase used was of nine characters, with one uppercase character. When the caps lock key was used for an uppercase letter, only the second caps lock key press has been considered. The first caps lock key press was ignored because users were found to look at the screen to check whether caps lock is active. The pattern however, is irregular and to avoid unnecessary variations, the first caps lock key press has been ignored.

Each timing sample consists of ten dwell times (N_d) and nine flight times (N_f). With this information, the mean and standard deviation were calculated for a user to build their template. The key affinity for a user is also part of the template. Left shift, right shift and caps lock key are each considered distinct. A flag is assigned to each of these

keys which is referred to after the statistical layer to obtain the final rank list of identified persons.

a. Mean of dwell time and flight time A person's typing rhythm varies slightly even when typing the same pass-phrase text. Concomitantly, a mean of the set of data is required. However, while training the system, the person's typing rhythm may change significantly due to uncontrolled factors like the environment, etc. Calculating the mean directly would result in an incorrect value because of the possibility of presence of outliers.

b. Combining the Mode and Mean In this setup, the mode of a set of data is defined differently. The mode of data is defined as the set of those samples which have the highest number of neighbors within a constant C_d (= 0.01). The mean is then calculated as the average of this set of timing samples.

Consider five samples of dwell time for user X (see Table 1). In this case, the number of timing samples within C_d for the:

1. First sample (=0.078) is 3,
2. Second sample (=0.069) is 5,
3. Third sample (=0.066) is 4,
4. Fourth sample (=0.066) is 4,
5. Fifth sample (=0.069) is 5.

In this case, the mode data would be 0.069 and the average is also equal to 0.069. This value is pushed into the person's template as the first key's average timing sample μ_c, where c is the character represented by the key. This algorithm repeats for the timing samples of other keys.

c. Standard Deviation Using the mean value for each key as calculated above, the standard deviation σ_c for each key with mean μ_c is calculated for dwell and flight timing samples according to the formula:

$$\sigma_c = \sqrt{(1/N * \sum_{i=1}^{N} (x_i - \mu_c)^2)} \quad (1.1)$$

C. Matching Process

a. Euclidean Distance A user's input timing data must be compared to every person's template in the system's database. In this setup, Euclidean distance is used as a similarity measure between the user input and the templates in the system.

For dwell time,

$$D_d = | \text{InputKey}_d - \text{TemplateKey}_d | \quad (1.2)$$

where,

InputKey_d is the dwell time of the key that the user presses on the keyboard, and TemplateKey_d is the dwell time of the same key that is present in person templates.

Similarly, for flight time,

$$D_f = |\text{InputKey}_f - \text{TemplateKey}_f| \quad (1.3)$$

where,

InputKey_f is the flight time which is the time interval between two key presses, and TemplateKey_f is the flight time of the same key that is present in person templates.

b. Initial Scoring the Euclidean distance that is calculated for each key press is summed up for all key presses.

For dwell time,

$$Score_d = \sum_{k=1}^{10} |InputKey_d^k - TemplateKey_d^k| \quad (1.4)$$

Similarly, for flight time,

$$Score_f = \sum_{k=1}^{9} |InputKey_f^k - TemplateKey_f^k| \quad (1.5)$$

c. Timing Hits For a key press timing sample to match a user's template timing sample, it must fall within a certain distance from the template's timing data. This distance is equal to the standard deviation for that specific key.

To register as a match (or *hit*), the relation between the input key's dwell time and the template's dwell time for the same key must be as follows:

$$D_d <= 1.25 * \sigma_c \quad (1.6)$$

If the condition is satisfied, that key is registered as a hit,

$$hit_d = 1 \quad (1.7)$$

Similarly, to register as a hit, the relation between input key's flight time and the template's flight time must be as follows:

$$D_f <= \sigma_c \quad (1.8)$$

If the condition is satisfied, that key is registered as a hit,

$$hit_f = 1 \quad (1.9)$$

The number of hits is calculated for all key presses against a user template and summed up. The maximum number of hits possible for dwell and flight time is equal to N_d and N_f respectively. A constant value is used as a weight in eq. (1.6) based on trial and error to improve performance.

Values for (1.2) to (1.9) are calculated against all person templates that are available in the system.

D. Final Scoring and Results

The final score for each template is calculated as follows:

$$Score_{template} = (N_d - hit_d) * Score_d + (N_f - hit_f) * Score_f \quad (1.10)$$

After the scores for all templates were calculated, a separate list based on key affinity was determined. Each template has a key affinity associated with it and those

templates (with $Score_{template}$) which have the key affinity value same as the input key affinity value were pushed to the top of the final rank list.

III. PRIOR APPROACH

There has been significant work in the field of keystroke dynamics to authenticate persons. In general, identification algorithms are more demanding than authentication algorithms. This is intuitive, since in an identification system, user input must be compared with all user template known to the system whereas in authentication, input is matched only against the authorized user's template.

Monaco et al [6] explored the application of statistical analysis and artificial neural networks (ANN) in keystroke dynamics-based identification systems. Using pure statistics, they were able to achieve an accuracy of 50.7% for both hands typing. However, when they implemented a fusion algorithm of statistics and ANN, they were able to score an accuracy of 69.4% for 58 users.

Mondal et al [10] used Pairwise User Coupling (PUC) as the underlying concept and were able to achieve an accuracy of 89.7% for both hands typing for 64 users.

Our setup was also tested with the CMU benchmark dataset [2]. This produced an accuracy of 45.6% which was expected. With increasing size of users, a keystroke dynamics-based system is bound to have decreased accuracy.

This paper presented a purely statistical algorithm with the addition of the key affinity feature. The setup was able to accurately identify users 62% of the time. This performance is much below the 89.7% accuracy score benchmark set by Mondal et al [10] and comparable to the performance of Monaco's [6] setup which used a statistical algorithm.

IV. OUR APPROACH

A. Experimental Setup

A total of twenty users were asked to participate in training the system. The data collection process was conducted over a period of three weeks. Users' age varied from 20 to 50 years old and had a gender distribution of 14 males and 6 females. 2 females were from medical field, 1 female from architecture field and others were from engineering field.

Users were prompted to type a fixed pass-phrase text '*.zoroBen1'* thirty times. Incorrect text was ignored, and timing data was extracted only for correct entries. This timing data was used to extract the mean and standard deviation for dwell and flight time of the user. Sample templates for user K and L are shown in tables 2, 3, 4 and 5.

Data was collected in a semi-controlled environment using a pre-allocated keyboard and, in an environment, where they could expect interference from their colleagues or disturbances from other environmental factors. Variations that could be possibly introduced by time of day was also considered with some users' keystroke data taken in the morning while others in the evening. Testing was conducted a month after the training phase in the same environment as the initial training. Each user provided five test samples and consequently, the following results discussion is based on a total of hundred test samples (N_T = 100).

The Cumulative Matching Curve (CMC) is a rank-based performance measurement and is being used as a performance measure of the system. A user is said to be correctly identified if their name appears at Rank 1. The graph is plotted with identification rate of a user at rank-k against rank index. CMC is a discrete function of rank-k. The probability that users are correctly identified at at-least rank-k:

$$p(k) = (\sum_{x=1}^{k}(N_x))/N_T \tag{1.11}$$

where,

N_x is the number of users correctly identified at at-least rank-x.

B. Results

In the first stage, 10 users were asked to provide test samples. The system was able to correctly identify the user at rank-1, 76% of the time. At rank-5, 100% of the time. The CMC curve plotted is as shown in Fig. 1.

In the second stage, 20 users (including the previous 10 users) provided test samples. The system was able to correctly identify the user at rank-1, 62% of the time. At rank-5, 92% of the time. At rank-15, 100% of the time. The CMC curve plotted is as shown in Fig. 2.

It can be seen from the two CMC plots that although the system can correctly identify users more than 50% of the time at rank-1, it gets significantly weaker with increasing size of user database. This behavior can be attributed to the fact that keystroke dynamics is inherently weak in discerning between users having similar typing patterns.

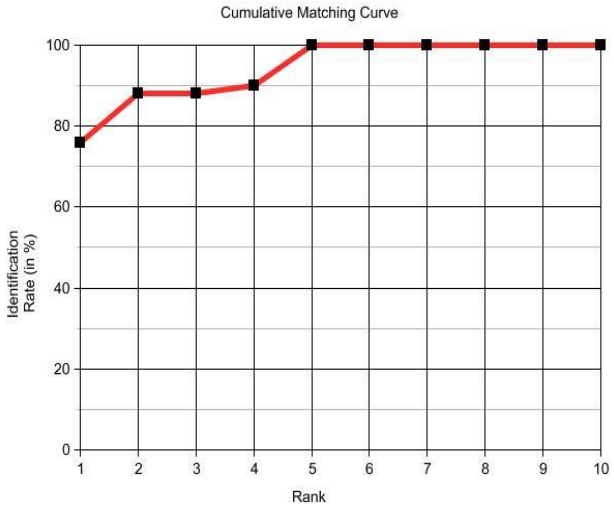

Figure 1: CMC for a sample set containing 10 users

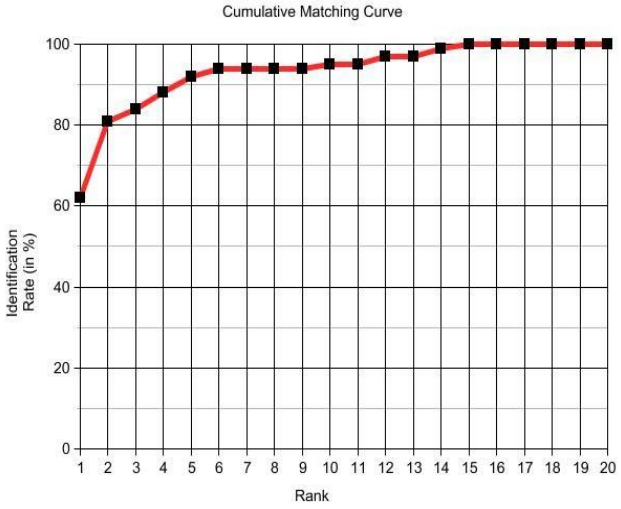

Figure 2: CMC for a sample set containing 20 users

V. CONCLUSION

The performance of the experimental setup mentioned before, and the results of other researchers leads to an intuitive observation. Keystroke dynamics-based systems will find it difficult to use pure statistics as their underlying algorithm.

Experimental setups need to have a larger user database to understand the scalability of the concept. This setup dealt with only 20 users and was able to produce a 62% identification rate at rank-1 while produced a 45.6% accuracy when dealing with 51 users. It is intuitive to extrapolate that this percentage may decrease significantly with a large user database.

The effectiveness of a keystroke-dynamics based system also depends on external factors like time of day, user temperament while typing, user's approach to the system's training phase and other environmental factors that may disturb the user when they are typing. The concept also requires that user systems have key logging software installed on them. This may be a violation of privacy according to laws in many countries. Permissions to implement these softwares on public systems may prove very difficult.

REFERENCES

[1] C. C. Loy, W. K. Lai, & C. P. Lim. (2007). Keystroke patterns classification using the artmap-fd neural network in intelligent information hiding and multimedia signal processing. *IIHMSP 2007. Third International Conference on, 2007, 1*, 61-64. Available at: http://ieeexplore.ieee.org/document/4457493

[2] Kevin Killourhy & Roy Maxion. (2009). CMU benchmark data set. Available at: https://www.cs.cmu.edu/~keystroke/

[3] P. S. Teh, A. B. J. Teoh, C. Tee, & T. S. Ong. (1201). A multiple layer fusion approach on keystroke dynamics. *Pattern Analysis and Applications, 14*(1), 23–36. Available at: https://link.springer.com/article/10.1007/s10044-009-0167-9

[4] F. Monrose & A. D. Rubin. (2000). Keystroke dynamics as a biometric for authentication. *Future Gener. Computer, 16*(4), 351-359. Available at: http://www.cs.columbia.edu/4180/hw/keystroke.pdf

[5] H. R. Lv & W. Y. Wang. (2006). Biologic verification based on pressure sensor leyboards and classifier fusion techniques. *IEEE Transactions on Consumer Electronics, 52*(3), 1057-1063. Available at: http://ieeexplore.ieee.org/document/1706507

[6] J. V. Monaco et al. (2015, May). One-handed keystroke biometric identification competition. *IEEE/IAPR International Conference on Biometrics, Phuket, Thailand*, 58- 64. Available at: http://ieeexplore.ieee.org/document/7139076/

[7] Pin Shen Teh, Andrew Beng Jin Teoh, & Shigang Yue. (2013). A survey of keystroke dynamics biometrics. *The Scientific World Journal, 2013*(4), 1-24. Available at: https://www.hindawi.com/journals/tswj/2013/408280

[8] R. Joyce & G. Gupta. (1990). Identity authentication based on keystroke latencies. *Communications of the ACM, 33*(2), 168-176. Available at: http://www.cs.cmu.edu/~maxion/courses/JoyceGupta90.pdf

[9] S. Hocquet, J. Y. Ramel, & H. Cardot. (2007). User classification for keystroke dynamics authentication. *Advances in Biometrics Proceedings, 4642*, 531-539. Available at: (https://link.springer.com/chapter/10.1007/978-3-540-74549-5_56)

[10] Soumik Mondal & Patrick Bours. (2017, June). Person identification by keystroke dynamics using pairwise user

coupling. *IEEE Transactions on Information Forensics and Security*, *12*(6), 1319-1329. Available at: http://ieeexplore.ieee.org/iel7/10206/7867904/07833085.pdf

[11] W. G. De Ru & J. H. P. Eloff. (1997). Enhanced password authentication through fuzzy logic. *IEEE Expert*, *12*(6), 38-45. Available at: http://ieeexplore.ieee.org/document/642960

APPENDIX

.	z	o	R	o	Shift	b	e	N	1
0.078	0.051	0.098	0.010	0.011	0.061	0.085	0.050	0.049	0.053
0.069	0.083	0.075	0.007	0.010	0.061	0.085	0.050	0.052	0.080
0.066	0.050	0.066	0.050	0.015	0.092	0.102	0.041	0.059	0.056
0.066	0.011	0.057	0.010	0.043	0.104	0.116	0.036	0.069	0.056
0.069	0.083	0.060	0.006	0.035	0.081	0.101	0.077	0.071	0.079

Table 1: Dwell Time Samples

.	z	O	R	o	Shift	b	e	n	1
0.058	0.106	0.071	0.076	0.066	0.028	0.079	0.089	0.068	0.118

Table 2: Mean dwell time for a user K

.	0.010083475124590496
Z	0.013176404936811235
O	0.009577854358761522
R	0.013448157538357089
O	0.01646475737124959
Shift	0.026376794614161272
B	0.013133925536563696
E	0.014074382234901739
N	0.013741307412752424
1	0.02469043634808473

Table 3: Standard deviation dwell time for a user K

. - z	z - o	o - r	r - o	o - Shift	Shift - b	b - e	e - n	n - 1
0.241	0.118	0.109	0.036	0.137	0.309	0.113	0.069	0.188

Table 4: Mean flight time for a user L

– z	0.17572019871174208
z – o	0.021707277175015322
o – r	0.2529480300029097
r – o	0.06495948963867902
o – Shift	0.20345428739356547
Shift – b	0.22324953066283582
b – e	0.021514017318401288
e – n	0.04291784216931914
n – 1	0.05083016705021676

Table 5: Standard deviation flight time for a user L