# K-Medoids-Based Random Biometric Pattern for Cryptographic Key Generation

H.A. Garcia-Baleon, V. Alarcon-Aquino, and O. Starostenko

Deparment of Computing, Electronics, and Mecatronics,
Universidad de las Americas Puebla
72820 Cholula, Puebla, Mexico
{hectora.garciabn,vicente.alarcon}@udlap.mx

**Abstract.** In this paper we report an approach for cryptographic key generation based on keystroke dynamics and the k-medoids algorithm. The stages that comprise the approach are training-enrollment and user verification. The proposed approach is able to verify the identity of individuals off-line avoiding the use of a centralized database. The performance of the proposed approach is assessed using 20 samples of keystroke dynamics from 20 different users. Simulation results show a false acceptance rate (FAR) of 5.26% and a false rejection rate (FRR) of 10%. The cryptographic key released by the proposed approach may be used in several encryption algorithms.

**Keywords:** keystroke dynamics, biometrics, cryptography, k-medoids.

## 1 Introduction

The combination of biometrics and cryptography has attracted the attention of some researches due to the fact that this combination can bring together the better of the two worlds. The idea of combining biometrics and cryptography is not new; however, the concept is poorly developed because several biometric cryptosystems require maintaining the biometric information in a centralized database. This fact has a serious impact in the social acceptance of the biometric cryptosystems. The first practical system that integrates the iris biometrics into cryptographic applications is reported in [3]. A system that works using fingerprint identification based on a token is presented in [1]. A successful combination of face biometric and cryptography for key generation is also reported in [2]. Another research that uses on-line handwritten signatures to generate cryptographic keys is reported in [4]. Other approaches have also been reported in [8,9]. However, these approaches have also reported a poor FAR and FRR. Both performance metrics are crucial in determining if the combined system can be implemented in real scenarios.

Keystroke dynamics can be defined as the timing data that describes when a key is pressed and when a key is released as a user types at the keyboard. The recorded timing data can be processed through an algorithm to determine a primary timing pattern (PTP) for future verification. The PTP is used to verify

E. Bayro-Corrochano and J.-O. Eklundh (Eds.): CIARP 2009, LNCS 5856, pp. 85–94, 2009.

the identity of the individual. The design of the proposed approach considers three security factors, namely, a user password, a behavioral biometric sample, and a token. It works using a 3D random distribution of the biometric data that assures also the randomness of the cryptographic key released. The 3D pattern is extracted from the 3D random biometric pattern using the k-medoids algorithm tested for different types of distances that measure similarity, namely, Manhattan, Euclidean, Chebyshev and Markowski distance. The rest of the paper is organized as follows. In Section 2, the k-medoids algorithm is described. Section 3 presents the Minkowski distance for measuring similarity. Section 4 presents the keystroke dynamics and shows how the PTP is extracted to work with the proposed approach. In Section 5, the design of the proposed approach is explained, whereas in Section 6 simulation results are reported. Finally, conclusions are reported in Section 7.

## 2 K-Medoids Algorithm

The k-medoids algorithm is a clustering algorithm based on the k-means algorithm and the medoidshift algorithm. Both, k-means and k-medoids, algorithms break the dataset up into $k$ clusters [5, 6]. Also, these algorithms attempt to minimize squared error. The squared error can be defined as the distance between points labeled to be in a cluster and a point designated as the center of that cluster. The k-medoids algorithm chooses datapoints as centers instead of computing the centers as the k-means algorithm does. The k-medoids algorithm is a partitioning technique of clustering that clusters the data set of $n$ objects into $k$ clusters known a priori. The k-medoids algorithm is more robust to outliers and noise compared to the k-means algorithm [6]. A medoid is defined as that object of a cluster whose average dissimilarity to the rest of the objects in that cluster is minimal. The partitioning around medoids (PAM) algorithm describes a common realization of the k-medoid clustering algorithm. The PAM algorithm is as follows:

1. *Arbitrary selection of $k$ objects as medoid points out of $n$ datapoints ($n > k$).*
2. *Associate each data object in the given data set to the most similar medoid to form clusters. The similarity in this step can be computed using distance measure. The distance measure used can be Euclidean, Manhattan, Chebyshev, or Minkowski distance.*
3. *Randomly select a non-medoid object named R' for each cluster.*
4. *Compute the total cost S of swapping initial medoid object to R'.*
5. *If $S < 0$, then swap initial medoid with the new one. Otherwise, the initial medoid remains.*
6. *Repeat steps 2 to 5 until there is no change in the medoids.*

The PAM algorithm is based on an iterative optimization process that evaluates the effect of swapping between the initial medoid object and the non-medoid object randomly selected. The principle of the PAM algorithm resides in step 5. It can be seen that it may require trying all objects that are currently not medoids.

Thus it represents an expensive computational cost, $Cost(k(n-k)^2)$, in each iteration. The PAM algorithm results in high quality clusters, as it may try every possible combination, working effectively for small datasets. However, due to its computational complexity, it is not practical for clustering large datasets [5,6].

## 3 Minkowski Distance

Formally, a similarity function aims at comparing two entities of a domain $M$ based on their common characteristics. Similarity can be measured in several ways depending on the scale of measurement or data type. Based on the vector representation the similarity can be calculated using the concept of distance. In this paper, we use the Minkowski distance to do so. The selection of the Minkowski distance is due to the fact that it is easy to implement in software and hardware, its computational cost is lower compared with more complex distances as Mahalanobis distance, and it fits better with the characteristics of the proposed approach considering the type of data used. In general, the distance $d_{ij}$ between any two points, $P = (x_1, x_2, ...x_n)$ and $Q = (y_1, y_2, ...y_n) \in \Re^n$, in n-dimensional space may be calculated by the equation given by Minkowski as follows [7]:

$$d_{ij} = \left( \sum_{i=1}^{n} |x_{ik} - x_{jk}|^p \right)^{\frac{1}{p}} \tag{1}$$

with $k$ being the index of the coordinates, and $p$ determining the type of distance. There are three special cases of the Minkowski distance:

- $p = 1$: this distance measure is often called city block distance, or *Manhattan distance*.
- $p = 2$: with $p$ equalling 2 the Minkowski distance is reduced to the well-known *Euclidean distance*.
- $p = \infty$: with $p$ equalling $\infty$ the Minkowski distance is reduced to the *Chebyshev distance*. In the limiting case of $p$ reaching infinity, the resultant equation is as follows:

$$d_{ij} = \lim_{p \to \infty} \left( \sum_{i=1}^{n} |x_{ik} - x_{jk}|^p \right)^{\frac{1}{p}} = max|x_{ik} - x_{jk}|_{i=1}^{n} \tag{2}$$

## 4 A Behavioral Biometric: Keystroke Dynamics

Keystroke dynamics is defined as the timing data that describes when a key is pressed or released as the user types at the keyboard. This behavioral biometric uses the manner and the rhythm in which a user types characters. The keystroke rhythms of a user are measured to develop a unique biometric pattern of the users typing for future verification. The recorded timing data can be processed through an algorithm to determine a PTP for future verification. The PTP is used to verify or even try to determine the identity of the individual who is

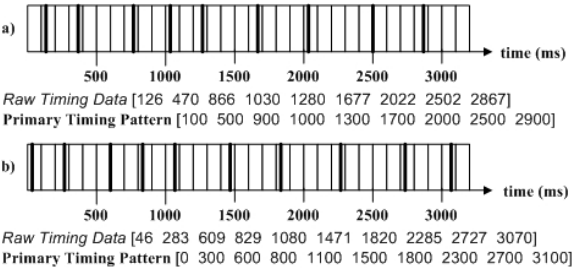


**Fig. 1.** Acquisition stage, a) Key pressing events pattern, b) Key releasing events pattern

producing those keystrokes. This is often possible because some characteristics of keystroke production are as individual as handwriting or a signature.

The technique used to extract the PTP used in this paper considers partitioning the acquisition time in time slots. The size of the time slot affects directly the FAR and FRR metrics. Several experiments performed showed that a size of $100ms$ for the time slot is good enough to minimize the FAR and FRR metrics as it is shown in Section 6. Figure 1 shows the timing data of an individual. In the top part, the timing data from the key pressing events is shown. The bottom part shows the timing data from the key releasing events. As can be seen, the key pressing process produces 9 events represented by the bold lines. The key releasing process produces 10 events also represented by the bold lines. It is important to notice that the first key pressed launches the acquisition stage and also the timer. It is assumed that the first key pressed event is located at zero in the time scale and thus the event is not considered in the computing of the PTP. The rest of events are located in the time scale according to the value that the timer has when the events take place. Figure 1 also depicts that the events can occur at any time within a determined time slot however the time value is rounded to the closest time slot value given in $ms$. This fact assures that the extracted pattern only comprises a combination of the possible discrete time values otherwise the possible time values that the event could take are infinite.

## 5 Proposed Approach

The successful recovering of the random cryptographic key depends on a correct combination of the user password, the behavioral biometric sample and the token, which stores the user password hash, the encrypted random distribution vectors (RDVs) used to reconstruct the 3D random biometric pattern, and the 3D pattern hash. The design presented here ensures that compromising two factors at most will not let to the attacker reveal the random biometric key.

The proposed approach is divided in two stages, namely, *training-enrollment* and *user verification*. Figure 2 shows a detailed representation of the approach. The first stage is executed when an individual is going to be enrolled for the first time to the biometric cryptosystem. This stage produces through a simple training process the information needed to verify the user in the second

stage. The training process uses the keystroke dynamics biometric information obtained from the user at his enrollment. The first stage can also be executed each time that the random biometric key needs to be revoked or renewed for any security concern. The second stage, user verification, is executed each time that the user needs to be identified before the biometric cryptosystem. The training-enrollment stage consists of the following steps:

1. *A 10-character password is required to the user. The user password p is hashed using Message-Digest Algorithm 5 (MD5). The hash result H(p) is then directly stored in the token.*
2. *The PTP is extracted as explained in the previous section. As a result of the timing pattern extraction, two dataset are obtained, namely, key pressing pattern and key releasing pattern. A third dataset is created taking the ASCII values of the password characters. These datasets form a universe of 900 possible different combinations. Notice that the PTP may vary even when the biometric information comes from the same user. This is due to the fact that the user is not used to input the chosen password or external factors affect his typing. Then, a training process is needed to overcome these difficulties. The purpose of the training process is to make converge and to generalize the PTP. The training process is as follows:*
   - *The user is required to input ten times the 10-character password chosen. Each time the PTP is extracted as explained previously.*
   - *The 10 PTPs are compared each other point by point. If the two compared points are separated each other for more than 4 time slots when the comparison takes place, that timing pattern is automatically discarded.*
   - *If at least six timing pattern survive this comparison process, the mean is calculated for each point and the result is rounded to the nearest time slot value. Otherwise, the training process must be restarted. Practical experiments showed that a user used to type a password generates the same PTP at least 6 out of 10 tries.*
   - *The resultant PTP obtained from this training process is considered as the global PTP to be used with the proposed approach.*
3. *Three random vectors are generated of 160 values each one. The formed datasets by the key releasing pattern and the ASCII password values are distributed according to the generated RDV which contain pseudorandom values drawn from the standard uniform distribution on the open interval (0, 10). The dataset that correspond to the key pressing pattern is also distributed over generated RDV which contains pseudorandom values drawn from the standard uniform distribution on the open interval (0, 9). Each of the three random vectors corresponds to a coordinate in a 3D plane. Figure 3 shows a 3D random biometric pattern generated using a specific behavioral biometric with a determined random distribution vector. The 3D pattern computed is formed for the resultant eight points obtained of performing the k-medoids algorithm over the random distribution of datasets.*
4. *Once the k-medoids algorithm converges and the 3D pattern is extracted, the pattern, k, is hashed using MD5 and the hash result H(k) is also saved into the token.*
5. *The RDVs used to construct the 3D random biometric pattern are encrypted using the advanced encryption standard (AES) and stored in the token. The MD5 hash*

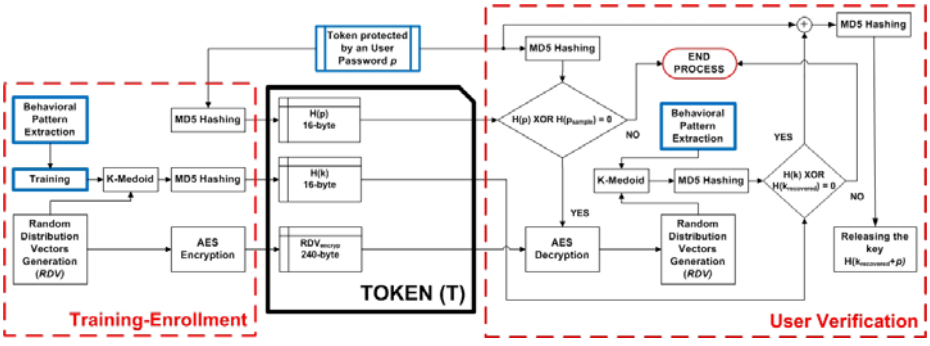


**Fig. 2.** The three security factor, user password, biometric behavioral sample and token, for the proposed approach

*of the user password is used as the 128-bit key that the AES algorithm needs to work. The training-enrollment stage can thus be defined as follows:*

$$\langle p, RDV, k\rangle \xrightarrow{training-enrollment} \begin{bmatrix} H(p) \\ H(k) \\ RDV_{enc} \end{bmatrix} \tag{3}$$

Now, we proceed to a detailed description of the user verification stage. It must be assumed that the user has the token with the three parameters stored in it.

1. *A user password, $p_{sample}$, is required to the user who is claiming the identity. Then the password provided for the user is hashed using MD5, $H(p_{sample})$, and compared with the hash stored in the token $H(p)$. If both hashes do not match, the stage ends. Otherwise, the stage continues to step 2.*
2. *To perform AES decryption over the encrypted RDVs stored in the token using as a key the MD5 hash of the password of the user already authenticated.*

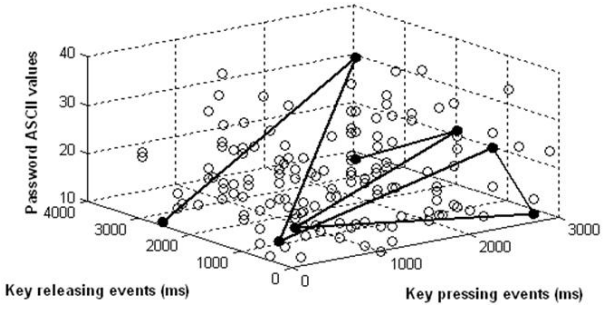


**Fig. 3.** A random biometric pattern generated using the biometric information with a determined random distribution is shown. The bold line shows the convergence points, 3D pattern, after performing the k-medoids algorithm.

3. *To extract the PTP, of a keystroke dynamics sample presented by the user as described previously.*
4. *To build the 3D random biometric pattern using as datasets the biometric information obtained in step 3 and password in step 1 and as distribution the decrypted RDVs obtained in step 2.*
5. *To apply the k-medoids algorithm over the 3D random biometric pattern built in the previous step to extract the 3D pattern.*
6. *The 3D pattern recovered, $k_{recovered}$, in the previous step is hashed using MD5, $H(k_{recovered})$ and compared to the hash stored in the token H(k). If both hashes do not match, the stage ends. Otherwise, the stage continues to step 7.*
7. *The 3D pattern is added to the ASCII values of the password of the user, $k_{recovered}+p$. The result is hashed using MD5 $H(k_{recovered}+p)$ to obtain a 128-bit random biometric key, k'. This is the cryptographic key that is released and belongs to the verified user. The user verification stage can thus be defines as follows:*

$$\langle p_{sample}, PTP_{sample}, T\rangle \xrightarrow{\text{user verification}} k' \tag{4}$$

Notice that in both, training-enrollment and user verification, stages is crucial that PTPs, random biometric keys and decrypted RDVs used along the stages must be securely crashed and not retained in memory.

## 6 Simulation Results

In this section, the performance results of the architecture discussed in the previous sections are reported. To illustrate the performance of the three security factors architecture, a Keystroke Dynamics Database was created. This database contains the timing data of 20 different users. It was collected 20 raw timing data samples total per user without any discretization process. Then, the database contains a global total of 400 timing data to be used to compute the FAR and FRR metrics and the computational cost. The 20 samples collected per user fulfill the training criterion stated previously. Even when the k-medoids algorithm presents several advantages as resistance to noise and outliers compared with other clustering algorithm, it also represents a high computational cost, $Cost(k(n-k)^2)$ due to the fact that it may try every point in the dataset before converging. Table 1 summarizes the maximum, minimum and the mean number of iterations needed to converge using different types of distances. As can be seen, a minimum of 2 iterations are need to make converge the 3D pattern for all distances. However, the Manhattan distance is the most effective distance because it needs at most 4 iterations to converge compared with the 6 iterations that the Euclidean and Chebyshev distance may need or with the 5 iterations that the Minkowski distance evaluated in 3 and 4 may need. Also, the computed mean using the Manhattan distance is the closest value to the minimum number of iterations which assures that the frequency of convergence with 2 iterations is higher compared with the rest of the distances. Then, Manhattan distance is the best choice for the architecture proposed because it needs less iterations to converge and its computational cost, with k=8 and n=160, is considerably lower compared with the rest of the tested distances.

**Table 1.** Iteration comparison of the k-medoids algorithm working with different types of distance

| **Distance** $p$ | **Maximum** | **Minimum** | **Mean** |
|---|---|---|---|
| 1 | 4 | 2 | 2.39 |
| 2 | 6 | 2 | 2.44 |
| 3 | 5 | 2 | 2.41 |
| 4 | 5 | 2 | 2.43 |
| $\infty$ | 6 | 2 | 2.49 |

**Table 2.** Performance of FAR and FRR metrics for different time slots

| **Time slot** $(ms)$ | **FAR** (%) | **FRR** (%) |
|---|---|---|
| 25 | 2.63 | 30 |
| 50 | 4.74 | 15 |
| 100 | 5.26 | 10 |
| 200 | 10.52 | 5 |
| 300 | 17.10 | 5 |

In training-enrollment stage, it is selected randomly a user then the 20 timing data samples of that user are used in the training process to extract the PTP. Once the PTP has been extracted as explained previously, the proposed architecture generates and stores in the token the information needed in the user verification stage. The FAR and FRR metrics were obtained testing an extracted PTP against the 400 timing patterns stored in the Keystroke Dynamics database. Given that the 20 timing data samples of the user fulfill the training criterion, it may be expected that only the 20 timing data samples that corresponds to the user who is claiming the identity should be accepted as legitimate. The rest, 380 timing data samples of other users, should be rejected by the architecture proposed. However, the FAR obtained in this work is 5.26% because 20 out of 380 timing data samples that do not belong to the user who claims the identity before the proposed architecture were accepted as authentic when they were not. Also, the FRR obtained is 10% because 2 out of 20 timing data samples that in fact belong to the user who claims the identity were rejected even when they represented accurately a timing data sample used to generate the user verification data stored in the token. The FAR is high compared with the combined system reported in [3,4]. However, the FRR has good performance if it is compared to [1,2,4] but it is still high compared to [3]. Table 2 shows how the FAR and FRR metrics are affected by changing the size of the time slot. As the time slot increases the extraction process is less selective this makes that PTPs from different users look similar. This fact then affects FAR negatively. Increasing the size of the time slot affects positively the FRR metric due to the fact that the architecture is able to identify PTP from the same user when the PTP do not differ so much each other.

As can be seen, there is a compromise between the FAR and FRR metric. The size of the time slot must be carefully chosen to save the equilibrium between

the metrics. The reason of choosing the 100ms time slot size is due to the fact that the absolute value of the difference of both metric is the minimum among the rest of the data of Table 2 which assures that the uncertainty level is also minimized.

## 7 Conclusions

In this paper, we have proposed an architecture based on the keystroke dynamics and the k-medoids algorithm. The proposed approach comprises three security factors, namely, user password, behavioral biometric sample and token. It assures that if an attacker compromises at most two factors, he is not going to be able to derive the random cryptographic key. The good performance of the FAR reported in this paper is directly related to the correct selection of the time slot however it is still high compared with the one obtained in systems that combine biometrics and cryptography reported in [3,4]. The FRR has good performance if it is compared with [1,2,4] but it is still not good compared with [3].

The idea behind the three security factor architecture reported in this paper is not limited to work with the PTP as it is extracted here. The extraction technique may be more sophisticated to improve the FAR and FRR and the rest of the approach remain unchanged. Instead of only considering the key pressing pattern and key releasing pattern, it could be added other parameters as the total typing time or the tendency of using certain keys by the user to make even more personal the biometric data. Also, one of the most notable advantages of the proposed approach is that it is not necessary to maintain a centralized database with the biometric information. This fact impacts positively in the social acceptance of the biometric cryptosystems. The proposed three security factor approach is a very secure system because the distribution of the 3D random biometric pattern is randomly generated. Also, if an attacker could compromise the all three factors, the cryptographic key can be easily revoked and renewed by executing the training-enrollment stage again. In the case of that the attacker could somehow derived the cryptographic key, he could compromise the key of that specific user but not the keys of a group or a corporation that could happen in the case of maintaining a centralized database with the biometric information of all users.

## References

1. Clancy, T.C., Kiyavash, N., Lin, D.J.: Secure smart card-based fingerprint authentication. In: Proceedings of the 2003 ACM SIGMM Workshop of Biometrics Methods and Application (2003)
2. Goh, A., Ngo, D.C.L.: Computation of cryptographic keys from face biometrics. In: Lioy, A., Mazzocchi, D. (eds.) CMS 2003. LNCS, vol. 2828, pp. 1–13. Springer, Heidelberg (2003)
3. Hao, F., Anderson, R., Daugman, J.: Combining Cryptography with Biometrics Effectively, Computer Laboratory, University of Cambridge, Technical Report Number 640 (2005)

4. Hao, F., Chan, C.W.: Private key generation from on-line handwritten signatures. Information Management & Computer Society 10(4) (2002)
5. Zhang, Q., Couloigner, I.: A New and Efficient K-Medoid Algorithm for Spatial Clustering. In: Gervasi, O., Gavrilova, M.L., Kumar, V., Laganá, A., Lee, H.P., Mun, Y., Taniar, D., Tan, C.J.K. (eds.) ICCSA 2005. LNCS, vol. 3482, pp. 181–189. Springer, Heidelberg (2005)
6. Barioni, M.C.N., Razente, H., Traina, A.J.M., Traina Jr., C.: Accelerating k-medoid-based algorithms through metric access methods. Journal of Systems and Software 81(3), 343–355 (2008)
7. Kardi, T.: Minkowski Distance of order $\lambda$ (2009), `http://people.revoledu.com/kardi/tutorial/Similarity/MinkowskiDistance.html`
8. Garcia-Baleon, H.A., Alarcon-Aquino, V.: Cryptographic Key Generation from Biometric Data Using Wavelets. In: Proceedings of the IEEE Electronics, Robotics and Automotive Mechanics Conference, CERMA 2009 (September 2009)
9. Garcia-Baleon, H.A., Alarcon-Aquino, V., Ramirez-Cruz, J.F.: Bimodal Biometric System for Cryptographic Key Generation Using Wavelet Transforms. In: Proceedings of the IEEE Mexican International Conference on Computer Science, ENC 2009 (September 2009)