Pointer Based Authentication System

Binit Singh

***Abstract*—Up till now Science and technology has introduced a pointing device at almost every place where user interfaces with digital world. For example Mouse in desktop Computers, touchpad in laptops, touch screen in smart phones. In this paper a pattern extraction and matching based behavioral Biometric system which recognizes pattern made by the user using his pointing hardware device is proposed. The system distinguishes the authenticated from mimicked signature by the longestcommon subsequence (LCS) pattern matching technique. The system does not require any extra scanning hardware too.**

***Index Terms*—Pointing devices, pattern matching, biometric, longest common subsequence (LCS), authentication, verification.**

I. Introduction

Biometrics, the application of statistical analysis to identify individuals through their biological or physiological characteristics, is emerging as a key aspect in new security systems.Using biometrics it is possible to avoid pitfalls encountered with traditional security systems where users are required to keep information, such as passwords, safe [1]. Biometric data can be classified as physiological or behavioral [2]. Physiological data remains stable over time (barring injury), examples include fingerprints [3], iris and retinal scans [4], [5], and hand geometry measurements [6]. Behavioral data may change over time typical examples include signatures [7], [8], [9], [10], voice prints and typing styles.

In this paper, I present a behavioral biometric verification system that will be used inplace of standard password match system. The biometric system improves upon the security level provided by password matching while greatly reducing the risk of dictionary-based attacks. The system uses no specialized equipment, requiring only a pointing device such as mouse, touch-pad or touch screen and a computer or a laptop or a touch-screen mobile device; other systems require specialist equipment such as scanners (e.g., fingerprint, iris, retinal) and microphones.

A. Basic Concept

Basic concept is to draw a pattern using mouse or touch pad which you can draw again later. The pattern can be anything like your signature, somebody's name, a drawing or a symbol. Later on you will draw the same thing and get authenticated. The system will recognize it as a sequence of patterns known as Tokens. These tokens have a unique token code which is saved in the sequence in which they occurred, for future authentication.

Although exactly the same pattern cannot be made every time but the sequence of occurrence of curves, turns, lines, circles, clicks etc in the pattern will be similar and the system bothers about that only.For example, see Fig. 1.0; a) it is made by touch screen and b) is made by a touch-pad. Though they both do not look alike because in touch-screen we can release the mouse and start drawing from another point but in touch-pad system, when we release the mouse and move to another location on the touch-pad, the screen pointer do not move to another place, it remains there where we left it. So it starts from the same position where we left. But the sequences of making the shapes are same in both the cases. The sequence is;

1) Draw the outer circle.
2) Draw the left eye.
3) Draw the right eye.
4) Draw a nose (line).
5) Draw a smile.

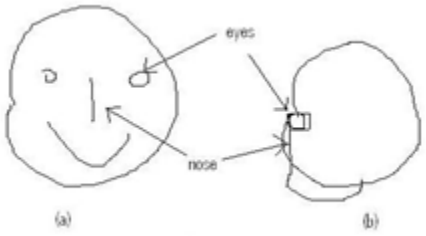

Fig. 1

II. System Overview

System is based on the writing style of the user. As shown in Fig. 2.0, (a) shows the signature on paper, (b) shows the same word on computer using mouse and (c) shows the same word on computer using touch pad. All these have approximately same subsequence of tokens.

Fig. 2

Unlike other systems [12], [7], [8], [13], [14], [15], [16], [17], [18], we use the mouse as the input device. We take this pattern as input to our biometric

System and match it with the original pattern in our database.

As shown in Fig. 2.1 the system takes the coordinates i.e. (x, y) coordinate of the pointer and then do pre-processing in which the change in angle of the vector joining the original pointer position and the previous pointer position and the

Manuscript received June 16, 2012; revised August 2, 2012.

B. Singh is with the Department of Information Technology Shri Ramswaroop Memorial College of Engineering and Management Uttar Pradesh Technical University, Lucknow, India (e-mail:binit_singh111@yahoo.com)

vector joining the original pointer position and next pointer position is recorded. This recorded data is sent to the next stage of feature extraction where we analyze and convert the pattern into token (pre-define standard simpler patterns). This refined data is sent for template generation. In template generation phase token codes are analyze and template file is generated which is stored for future authentication. When any data comes for authentication we call this file from store into matcher and perform matching with current data.

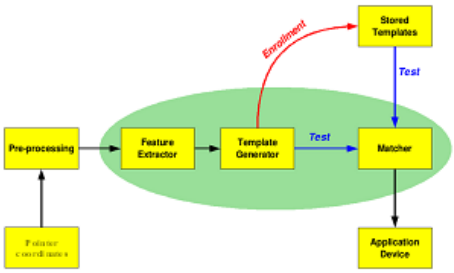

Fig. 2.1

A. System Function

The system functions in two distinct modes: registration and verification. During the registration phase, new users are required to select a username and input a chosen pattern multiple times (5 in our assumption). The gathered biometric data is processed to extract salient information. The details of the salient information, specifically, the feature points used for the authentic user are then stored in a template file. During verification, the user logs into the system by a username and signature. The user template file, retrieved from the store, contains details of the authentic user's salient features; these features are then extracted from the input biometric data and sent to matcher for verification.

B. Data Acquisition

Data acquisition is done in form of pointer coordinates (x, y) which keeps on refreshing as the pointer moves. So we can easily track the position of pointer and get to know the patterns.

C. Preprocessing

Here system extracts tokens from the pattern in a sequential order irrespective of whether they are large or small and a rough time taken to make it. In this step we analyze the behavior of the pointer i.e. in which direction it is turning.

As you may see in Fig. 2.2 that we are analyzing the movement of pointer by the change in angle of the vector joining the original pointer position and the previous pointer position and the vector joining the original pointer position and next pointer position is recorded. The angle between v1 and v2 decides the movement of pointer. This data tells us what token it is building so that in next step we could easily build the token based replica of this pattern.

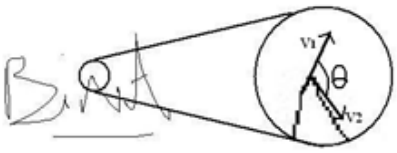

Fig. 2.2

D. Feature Extraction

It is possible to represent a pattern using all information obtainable from the raw pattern trace.This is, however, undesirable because much of the data will not provide a significant degree of uniqueness or consistency. The usage of such information could, therefore, prove to be contradictive. Storing all of the information is also costly (in terms of space) and has implications for processing overheads when verifying signatures. A signature may be represented by a set of extracted features rather than all of the raw data. This system adapts a technique of token generation. In this the data is coded in form of already know pattern (refer to table I) which makes our matching work consistent.

TABLE I

Pattern code	Pattern	Pattern name	Pattern type	Pattern code	Pattern	Pattern name	Pattern type
0		Line	Simple	8		Bend left	Simple
1		Curve right	Simple	9		Curve left	Simple
2		Bend right	Simple	A		U-turn right	Simple
3		Turn right	Simple	B		U-turn left	Simple
4		Hill right	Simple	C		Oval	Compound
5		Pin	Simple	D		Single click	Simple
6		Hill left	Simple	E		Double click	Compound
7		Turn left	simple	F		Disjoint	Compound

In Fig. 2.3 token interpretation of a pattern is shown. This signature involves 28 sequential tokens and the pattern code for the signature will be "0B02AB021595D504065B0596FF0D".

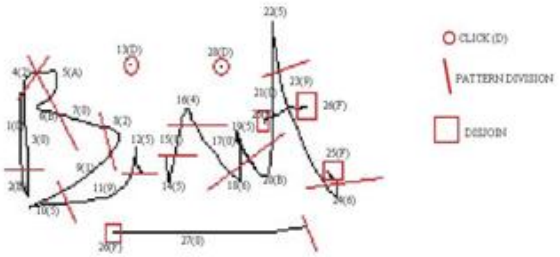

Fig. 2.3

E. Template Generation

The signature code obtained from feature extraction process is actually a hexadecimal code which is easy to store.

During registration part, system asks for the same pattern five times from the new user. Of course the signature patterns will differ from each other. The System will take the Longest Common Subsequence of their pattern code and keep the weight of the LCS data into another array, let's call it as Weight[]. Weight[] array contains 1's in the beginning. Every time when Compute-Weight() function is called it will increment the respective bits of LCS in Weight[] by 1. This means that in the end, the pattern bits which are common in all 5 signatures will have it'sWeight[]bits equal to 5. Lets call it as strong bit and the Weight[] bits which are equal to 1 as

fault bit as it has not been encountered even twice.

The ordinary LCS algorithm cannot be used here because there can be more than one longest common subsequence in the pattern code, so in that case the ordinary LCS algorithms [25],[26],[27],[28] will take the one which occurred in the end but we have to take the one with greater sum of LCS-weight. So I modified and priorities it on the basis of weight. In Fig. 2.4 the procedure followed by LCS-weight() algorithm is shown; the LCS obtained is [B→D→A→B] whose length is 4 and weight is 8 but in ordinary LCS algorithm the LCS obtained would be [B→C→B→A] whose length is also 4 and weight is 3, which is not acceptable.

Let X and Y be two samples of signature, then;

```
LCS-Weight(X, Y )
1.  m ← length[X]
2.  n ← length[Y]
3.  for i ← 0 to m
4.  do c[i, 0] ← 0
5.      b[i,0]="←"
6.  for j ← 0 to n
7.  do c[0, j ] ← 0
8.      b[0,j]="↑"
9.  for i ← 1 to m
10. do for j ← 1 to n
11.     do if xi = yj
12.         then c[i, j ] ← c[i − 1, j − 1] + weight[i]
13.              b[i, j ] ←"↖"
14.         else if c[i − 1, j ] ≥ c[i, j − 1]
15.         then c[i, j ] ← c[i − 1, j]
16.              b[i, j ] ← "↑"
17.         else c[i, j ] ← c[i, j − 1]
18.              b[i, j ] ← "←"
19.  return c and b
```

LCS-Weight() algorithm will compute the longest common subsequence with maximum weight. The running time complexity of LCS-weight() algorithm is O(mn). The following method will increment the respective Weight[] bits of the LCS.

```
Compute-Weight(b, X, i, j, Weight[] )
1.    ifi= 0 or j = 0
2.  then return
3.    ifb[i, j ] = "↖"
4.  thenWeight [xi]+=1
5.            Weight-LCS(b, X, i − 1, j − 1)
6.    elseifb[i, j ] = "↑"
7.  thenWeight-LCS(b, X, i − 1, j )
8.    elseWeight-LCS(b, X, i, j − 1)
```

The running time complexity of Compute-weight() algorithm is O(m). Next algorithm is Merge-array() which is used during registration phase. This algorithm merges two pattern codes who's LCS has been computed in order to reduce data loss due to LCS. The running time complexity of Merge-arrays() algorithm is O(k), where k is m+n-l. In this way the next LCS will be computed between the merged pattern code and new pattern code.

```
Merge-arrays(X, Y, b)
1.  i ← length[X]
2.  j ← length[Y]
3.  k= m+n-(LCS-length(X,Y))
4.  while(k!= 0)
5.  do if b[i,j]= "↖"
6.     then c[k]= X[i]
7.          weight[k]= weight[i]
8.          i--
9.          j--
10.   else if b[i,j]= "←"
11.   then c[k]= y[j]
12.        weight[k]= 1
13.        j--
14.   else c[k]= X[i]
15.        weight[k]=weight[i]
16.        i--
17.   k--
```

Repeating these algorithms for all five pattern codes, at last we get a combined pattern code and computed weight[].The running time complexity of whole registration phase will be O(mn). For reducing time and memory complexity one may avoid fault bits in combined pattern code, but for the purpose of this research paper I leave it intact now.

The overall signature will be termed strong if:

1>it has at least 3 strong bits in Weight[].

2>the length of the pattern code is >12 bits.

If the pattern satisfies these two conditions then it's time to create a file containing user's salient features.

```
File (user, X, weight[], time)
{
    USER_NAME = user
    PATTERN_CODE = X
    PATTERN_WEIGHT = weight[]
    TIME_LIMIT = time
}
```

This file is stored and later on called by the user name field for authentication.

F. MATCHING

The system asks for the user-name and signature each time the user logs on to the application. The user-name is searched in the store for a match, if match is found then the respective file is extracted for verification. Now user will enter his signature which goes from stage 2.2, 2.3, 2.4, 2.5 to 2.6 i.e. till matcher. In template generation phase (i.e. 2.5) only LCS-weight() algorithm is called. Here LCS-weight() algorithm is used in different manner than during registration period. Taking a close look at the algorithm you will find that c[n,m] value is actually containing the sum of the longest common subsequence. So LCS-weight() serves two purpose, firstly, finding the LCS and secondly, calculating the sum of LCS-weight bits.

Matcher-LCS() algorithm return's a Boolean value which determines whether the user is authentic or fake. It compares the sum of LCS weight bits with the total sum of the weight[] array. It should be minimum 85% of the total Sum of all Weight[] values. Moreover it also checks that all strong bits are matched. The running time complexity of Matcher-LCS() algorithm is O(m).

```
Matcher-LCS(file, Y, c, b, time)
1.  s = sum(file.PATTERN_WEIGHT)
2.  m = length(file.PATTERN_CODE)
3.  n = length(Y)
4.  iffile.TIME_LIMIT ≈ time
5.  then if c[m,n] ≥ (0.85 x s)
6.       thenif Is-SPM(b,file.PATTERN_WEIGHT,m,n)
7.            return TRUE
8.  return FALSE
```

Is-SPM() algorithm checks for all Strong Points Matched.

```
Is-SPM(b, weight, i, j )
1. if i = 0 and j = 0
2.    then return TRUE
3. if b[i, j ] = "↖"
4.    then Is-SPM(b, weight, i − 1, j − 1)
5. elseif b[i, j ] = "↑"
6.    then if weight[i] = 5
7.           then return FALSE
8.    else Is-SPM(b, weight, i − 1, j )
9. else Is-SPM(b, weight, i, j − 1)
```

The System do not expect that whole pattern will match. it see that:

1> All the strong bits are matched.

2> The Sum of all Weight[] values corresponding to the LCS (Output of LCS-weight()) should be 85% or above of the total sum of the Weight[] array values.

3> Time taken in making the pattern should be approximately near to the original time taken.

If these three conditions satisfy, we say that the signature is authentic and access is granted.

III. Applications

The authentication system can be customized with any desktop based, internet based and mobile based applications requiring secure authentication. In addition to it cryptography and other biometric system or password authentication can also be customized with it.

The performance of implementation of the system will be best with touch-screen applications because the accuracy level in taking the signature is very high and the use of stylus makes it moreover like writing with pen. The performance of implementation with mouse will be poor. A mouse-based signature test done by Ross A.J. Everitt and Peter W. Mcowan [19], achieves a fraudulent access rate of≈4.4 percent, while authentic users access with a rate of≈99 percent. They made use of ranking and genetic approach for their findings.

The data acquisition and preprocessing has language based constrains, so that may be a problem area while customizing with any application. As the application can be in any language so integration is also tuff. So on the basis of statistical analysis of patterns on the particular application we may change the features extraction and matching constrains.

IV. Compare and Contrast with Other Biometric System

This Biometric system do not require any extra scanning device as required by fingerprints [3], iris and retinal scans [4], [5], and hand geometry measurements [6]. The pointer devices which are required are generally in-build with desktop, laptop and touch screen devices. So this seems to be the best authentication system we may have in near future.

The on-paper signature is easy to copy because it is visible, so one may practice and may get perfect in copying it, but the pointer signature are neither visible nor has it copy saved. The user signs on blank screen with no mark of the pointer movement. It seems like the user is shaking his mouse like that only.

It is protected from Dictionary based attacks and key loggers, so it provides better authentication than password protection. Moreover it will provide better level of security in doing transactions in cyber cafes.

It provides extreme level of flexibility while choosing the signature. The signature can be anything- symbol, name or any language.

It do not require series of test or more than five time input during registration phase, as in Ross A.J.Everitt and Peter W. Mcowan [19] menuscript, which take 25 times the same signature in learning phase.

V. Conclusion

This paper has introduced a new approach for providing secure access to the daily used computer and mobile systems using biometric verification. It authenticate the user in O(mn) time complexity. The system does not use any sequence of tests [24] or asks for any password, so this makes it user-friendly. The system is novel because it is to the best of the authors' knowledge, the only Pointer-based authentication system for desktop, internet and mobile devices to be proposed at this point in time. The system is specifically designed for use in a potentially hostile real world environment with uncontrolled and non standard equipment.

The benefits over other biometric system make this approach better amongst all. So it may be predicted that the False Acceptance Rate (FAR) and False Rejection Rate (FRR) of the proposed system will also be better as compared to other biometric systems.

Up till now paper-signatures authenticate your access over physical world entities but after the implementation of this system, digital world will also be accessed by our signatures.

References

[1] R. Clarke, "Human identification in information systems: Management challenges and public policy issues," *Information Technology and People*, vol. 7, no. 4, pp. 6-37, Dec. 1994.

[2] B. Miller, "Vital signs of identity," *IEEE Spectrum*, vol. 31, no. 2, pp. 22-30, Feb. 1994.

[3] A. Roddy and J. Stosz, "Fingerprint features—Statistical analysis and system performance estimates," *Proc. IEEE*, vol. 85, no. 9, pp. 1390-1422, Sept. 1997.

[4] S. Gordon, "Ocular biometrics: For your eyes only," *Opto and Laser Europe*, no. 84, May 2001.

[5] Y. Zhu, T. Tan, and Y. Wang, "Biometric personal identification based oniris patterns," *Proc. Int'l Conf. Pattern Recognition*, vol. 2, pp. 805-808, 2000.

[6] R. Sanchez-Reillo, C. Sanchez-Avila, and A. Gonzalez-Marcos, "Biometric identification through hand geometry measurements," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 22, no. 10, pp. 1168-1171, Oct. 2000.

[7] S. Hangai, S. Yamanaka, and T. Hamamoto, "On-line signature verification based on altitude and direction of pen movement," *Proc. IEEE Int'l Conf. Multimedia and Expo*, vol. 1, pp. 489-492, 2000.

[8] B. Herbst and D. Richards, "On an automated signature verification system," *Proc. IEEE Int'l Symp. Industrial Electronics*, vol. 2, pp. 600-604, July1998.

[9] G. B. Hesketh, "Countermatch: A neural network approach to automaticsignature verification," *Proc. IEE Colloquium on Neural Networks for Industrial Applications*, pp. 2/1-2/2, Feb. 1997.

[10] J. Higashino, "Signature verification system on neuro-computer," *Proc.11th IAPR Int'l Conf. Pattern Recognition*, vol. III-C, pp. 517-521, 1992.

[11] S. A. Bleha and M. S. Obaidat, "Dimensionality reduction and featureextraction applications in identifying computer users," *IEEE Trans. Systems, Man, and Cybernetics*, vol. 21, no. 2, pp. 452-456, Mar./Apr. 1991.
[12] J. Brault and R. Plamondon, "Segmenting handwritten signatures at theirperceptually important points," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 15, no. 9, pp. 953-957, Sept. 1993.
[13] L. Lee, "Neural approaches for human signature verification," *Proc. ThirdInt'l Conf. Document Analysis and Recognition*, vol. 2 pp. 1055-1058, 1995. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, September 2003 1171
[14] R. Martens and L. Claesen, "Automatic on-line signature verification discrimination emphasised," *Proc. Fourth Int'l Conf. Document Analysis and Recognition*, pp. 657-660, 1997.
[15] M. Mingming and W. Wijesoma, "On-line signature verification based on multiple models," *Proc. IEEE/IAFE/INFORMS Conf. Computational Intelligence for Financial Eng*., pp. 30-33, Mar. 2000.
[16] T. Wessels and C. Omlin, "A hybrid system for signature verification," *Proc. South African Telecommunications Networks and Applications Conf*., pp. 5509-5514, 2000.
[17] Y. Xuhua et al., "A study on signature verification using a new approachto genetic based machine learning," *Proc. IEEE Int'l Conf. Intelligent Systems for the 21st Century*, vol. 5, pp. 4383-4386, Oct. 1995.
[18] K. Yue and W. Wijesome, "Improved segmentation and segment association for on-line signature verification," *Proc. IEEE Int'l Conf. Systems, Man, and Cybernetics*, vol. 4, pp. 2752-2756, Oct. 2000.
[19] A. J. Everitt and W. Mcowan "Java-based internet biometric authentication system" *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 9, Sep. 2003.
[20] A. Apostolico, "String editing and longest common subsequences," in: G. Rozenberg, A. Salomaa (Eds.), Linear Modeling: Background and Application, in: *Handbook of Formal Languages*, vol. 2, Springer-Verlag, Berlin, 1997, pp. 361–398.
[21] A. Apostolico, "General pattern matchings," in: *M. J. Atallah (Ed.)*, Handbook of Algorithms and Theory of Computation, CRC, Boca Raton, *FL*, 1998, Chapter 13.
[22] L. Bergroth, H. Hakonen, and T. Raita, "A survey of longest common subsequence algorithms," in: *SPIRE*, A Coruña, Spain, 2000, pp. 39–48.
[23] D. S. Hirschberg, "Algorithms for the longest common subsequence problem," *J. ACM* vol. 24, pp. 664–675, 1977.

Binit Singh was born in Amritsar, Punjab, India on 8th January, 1991. He graduated from Bachelor of Information Technology, from Shri Ramswaroop Memorial College of Engineering and Management, affiliated to Uttar Pradesh Technical University, Lucknow, India in year 2012. His major fields of study are Information Technology, Object oriented Methodology, Algorithms, Parallel computing and Cryptography. He has made projects on JAVA Technology which are Pointer based Authentication System, Attendance Monitoring System and Online Banking System. His area of research work are Cryptography, Biometric System and Object Oriented Systems.