ANNALS OF THE FACULTY OF ENGINEERING HUNEDOARA

2005, Tome III, Fascicole 3

TYPING PATTERNS ANALYSIS – A BIOMETRIC TECHNIQUE FOR USER AUTHENTICATION

MUSCALAGIU Ionel[1], BALAN Mihai[2], BALAN Lucia[2]

[1] UNIVERSITATEA "POLITEHNICA" TIMIŞOARA
FACULTATEA DE INGINERIE HUNEDOARA
[2] DENMARK TECHNICAL UNIVERISTY
LYNGBY, DENMARK

REZUMAT/ ABSTRACT

User authentication in computer systems has been a research topic since the birth of multi-user computing environment. Authentication via different biometric techniques is a subset of this topic.

Typing pattern analysis is an example of a biometric technology based on behavioral characteristics. Most subjects possess certain patterns of behavior, which are unique to them. One of these is the way in which the subject types.

In general, typing pattern analysis determines a number of features of a subject's typing, such as: dwell time, flight time and error patterns. These features are recorded during a training session - the subject either repeatedly types a given text, or works normally at the computer over a longer period of time.

After the training session, an algorithm is used to reduce the patterns to a description in terms of a number of suitable parameters. When the subject uses the computer at a later time, the subject's typing behavior is compared against this description. If large deviations from the previously observed behavior are detected, access to the computer is denied or restricted.

1. INTRODUCTION

"Biometrics" is a science that uses one or more bodily characteristics to uniquely identify individuals. This can be achieved by using biological traits, such as those based on retinal or iris scanning, fingerprints, face recognition, voice recognition, DNA matching, ear lobe analysis, typing-pattern recognition and other physical characteristics of individuals.

Biometric techniques can add significant complexity to a system setup. This is due to the need of security and privacy of the database biometric information, optimum-storing places for the biometric characteristics, etc. The privacy of the biometric patterns is a very important issue. Biometric characteristics are not stored as a fingerprint coy or a retina image, but as a mathematical representation of the item e.g. a mathematical formula not the retina picture itself. Thus, people have to be convinced about the privacy of these techniques.

This paper analysis and discuss the results obtained during several measurements of the typing patterns for an authentication process on a Linux – based system. Information about the Dwell Time, Flight Time and the Interval Time from six persons is collected. These results are used to calculate the mean and standard deviation for all the characters in the input text. In this way one can determine which individual is typing a particular text by checking how much is he/she deviating from the standard deviation.

2. ANALYSIS

A text that contains all the English alphabet letters is selected for the training process. Thus a precise measurement of the individual typing pattern is obtained [1]. The following text is used: *"a quick brown fox jumps over the lazy dog"*.

The input text is typed ten times by each person. The *xlisten* program is used to captures the timings for key presses and releases. This program has been developed by Adrian Perrig of Carnegie-Mellon University. The outputs of the program for all typings are stored in 10 files given name from *userName0.txt* to *userName9.txt.* An algorithm that checks the user's patterns for authentication purposes is implemented. It check firstly whether or not a particular user enters a correct text. If errors are detected, the user will be asked to type the text once again for the erroneous file. In this way it take care not to have erroneous input when the mean and the standard deviation are computed.

The algorithm goes through all the files and gathers the information from each file. Then, the dwell time, flight time and the interval time for each character in the given string are calculated.

The *Dwell Time* for a particular key is calculated by adding all the delays between pressing the key and releasing it. This is done for all the ten files and the results are added, to calculate the mean dwell time for the particular key.

The *Flight Time* for a particular key is calculated by adding all the delays between releasing the key and pressing the next one. This is done for all the ten files and the results are added, to calculate the mean flight time for the particular key.

The *Interval Time* for a particular key is calculated by adding all the delays between pressing the particular key and pressing the next one. This is done for all the ten files and the results are added, to calculate the mean flight time for the particular key.

The mean dwell time, mean flight time and mean interval time, are computed by using the previous calculated sums. The next step consists in computing the standard deviation for each character in the given string. Formula 1 is used for the calculating the standard deviation.

The user is authenticated if the algorithm doesn't detect large deviations from the calculated results.

$$SD = \sqrt{\frac{\sum_{i=1}^{n}(x_i - \mu)^2}{n}},\ \mu = \frac{\sum_{i=1}^{n} x_i}{n} \quad (1)$$

3. DESIGN

The *xlisten program* is used to capture the key pressing and releasing delays. The output result of xlisten program is redirected in a simple text file. Thus ten text files are created for each of 6 participants in the experiment.

The name of the person whose pattern is to be analyzed is given as a command line argument in the analysis program developed in C++. All the relevant data from a particular file are gathered in a data structure defined in the program. The length of the input string is known so an array of structure is created. It holds the relevant data as following:

```
struct XListenData
{
  unsigned int event;          // key PRESS or RELEASE
  char* keycode;               // character symbol
  unsigned int delaytime;      // delay
};
XListenData ArrData[MAXLENGTH]; // Array of structure
```

where MAXLENTH = 82 as there are 41 letters in the string. Therefore, there are total of 82 key presses and key releases.

In the next step the algorithm checks if the user enters a valid input or not by comparing the data gathered from the file and the given input string.

Three different arrays are defined for storing the sum of dwell, flight and interval times of characters as:

```
int DwellTime[41];
int FlightTime[41];
int IntervalTime[41];
```

Another data structure is defined to hold all the calculated timings for each character 10 times. This structure is needed in order to calculate the standard deviation by using formula 1.

```
int DwellTable[41][10];
int FlightTable[41][10];
int IntervalTable[41][10];
```

The pseudo code for calculating the Dwell, Flight and Interval times is depicted bellow:
(This code has an outer for loop of 10 iterations for calculating 10 measurements of Dwell, Flight and Interval timings for a particular user.)

```
int index = 0;
int sum = 0;
int tempindex = 0;
int ArrIndex = 0;
int FlightArrIndex = 0;
int IntervalArrIndex = 0;
while (1){
   if (ArrData[index].event == PRESS){
        //----------------------------
        // calculate the Dwell time
        //----------------------------
        sum = 0;
        tempindex = 0;
        tempindex = index+1;
        while(1){
             sum = sum + ArrData[tempindex].delaytime;
             if ((ArrData[tempindex].event == RELEASE) &&
           (strcmp(ArrData[tempindex].keycode,ArrData[index].keycode
     ) == 0))
                break;
             tempindex++;
        }
        // adding the calculated Dwell time with the previously
calculated
        // Dwell times.
        DwellTime[ArrIndex]+= sum;

        // storing all the dwell times for particular user file, to
be used in
        // calculating standard deviation
        DwellTable[ArrIndex][fileindex] = sum;
       ArrIndex++;
        //----------------------------
        // calculate the Interval time
        //---------------------------
        sum = 0;
        tempindex = 0;
        tempindex = index+1;
        while(1){
             sum = sum + ArrData[tempindex].delaytime;
             if                            ((ArrData[tempindex].event
==PRESS)||(IntervalArrIndex == 40))
                break;
             tempindex++;
        }
        IntervalTime[IntervalArrIndex] += sum;
       IntervalTable[IntervalArrIndex][fileindex] = sum;
        IntervalArrIndex++;
      }// end if PRESSED
      else if (ArrData[index].event == RELEASE){
          //----------------------------
          // calculate the Flight time
          //----------------------------
          sum = 0;
          tempindex = 0;
          tempindex = index+1;
          while(1){
             sum = sum + ArrData[tempindex].delaytime;
             if                            ((ArrData[tempindex].event
==PRESS)||(IntervalArrIndex == 40))
                break;
             tempindex++;
          }
          FlightTime[FlightArrIndex] += sum;
          FlightTable[FlightArrIndex][fileindex] = sum;
          FlightArrIndex++;

    }// end else RELEASE
    index++;
    if (index > 81)
       break;
    }// end while
```

The Mean time for all the readings is calculated by following:

```
for (index=0;index<41;index++){
    MeanDwellTime[index] = DwellTime[index]/10;
    MeanFlightTime[index] = FlightTime[index]/10;
    MeanIntervalTime[index] = IntervalTime[index]/10;
}
```

Standard Deviation for of 41 characters in 10 readings is calculated by following:

```
for (index = 0; index<41; index++)    // for each character{

    sumD = 0;
    sumF = 0;
    sumI = 0;
    valD = 0;
    valF = 0;
    valI = 0;

    for (j= 0; j<10; j++){ // for each of 10 readings
       valD =  DwellTable[index][j] - MeanDwellTime[index];
       sumD = sumD + (valD*valD);

       valF =  FlightTable[index][j] - MeanFlightTime[index];
       sumF = sumF + (valF*valF);

       valI =  IntervalTable[index][j] - MeanIntervalTime[index];
       sumI = sumI + (valI*valI);
    }// end j for loop

    SDDwellTime[index] = sqrt(sumD/10);
    SDFlightTime[index] = sqrt(sumF/10);
    SDIntervalTime[index] = sqrt(sumI/10);
}
```

4. RESULTS INTERPRETATION

After calculating the mean dwell, flight, interval time and standard deviation for all the six persons, a person is chosen randomly among the persons that have participated in the experiment. He has to type the input text once again. The chosen person is subject 3. The subject no.3’s Mean Dwell, Flight and Interval and the corresponding standard deviations are depicted in fig. 1, 2 and 3.

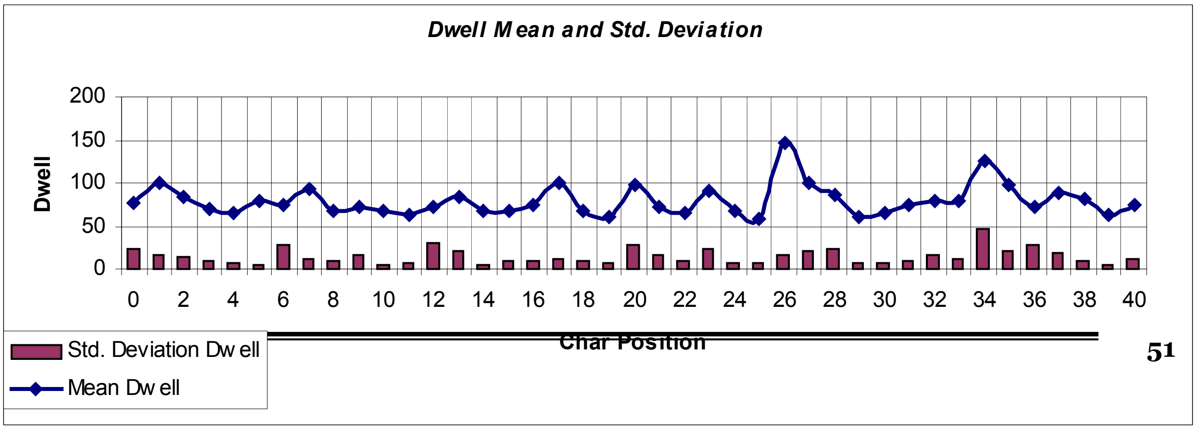

Fig 1. Mean Dwell and Standard Deviation for Subject no. 3

Table 1

Subjects	Sum of ones (Sum of matches)	Reliability (%)
Subject no. 1	18	44 %
Subject no. 2	12	29 %
Subject no. 3	**35**	**88 %**
Subject no. 4	14	34 %
Subject no. 5	12	29 %
Subject no. 6	16	39 %

Then, the dwell, flight and interval time for each character in the input text are calculated. The next step consists in matching all the previous calculated mean and standard deviation dwell time for the six persons with the results of the testing text. If a person lies in between the standard deviation of mean dwell time for a particular character a one (1) is placed for that character; otherwise, a zero (0). At the end, the percentage of ones (1s) is computed. The results pointed that subject no.3 is the person that has typed the testing text – and this is true. His matching results are 88%. The others lie between (30%, 44%). The results are depicted in Table 1.

The flight and interval time means cannot be considered reliable indicators as it can be clearly seen in the graphs.

5. CONCLUSION

A certain person can be identified from the dwell time by knowing how much it is deviating from the standard deviation. Standard deviation should be small for a precise authentication. If the standard deviation is not so small it is difficult to identify a particular person because the interval will be large and other person can be situated in the same interval.

The flight and interval times can be different for various reasons, e.g. while typing, user can answer to the phone and press the next key after 2 minutes. Thus, the flight and interval times are not reliable authentication parameters. The normalization will not increase the reliability of indication as various keys can take longer time to be pressed than others. The fraction of the total time is not so good in that case.

Dwell Time can be a reliable indicator of a person identity in any circumstances e.g. when a person types a password or when he/she is typing a regular document that has no connection with his authentication.

User authentication that takes place by typing a username/password pair is one of the most common user authentication methods. This method is based on the secrecy of the password and maybe username. Its disadvantage is that all authentication tokens are easily transferred among users. Biometrics can add a new layer of security on top of the existent one. Therefore, once can suggest the use of biometric techniques such as: *typing pattern analysis* on top of the username/password authentication protocol. This biometric technique is a very cheap one that doesn’t require any additional hardware investments. As users are already acquainted to typing username/password pair, adding typing pattern analysis authentication will come on no const to the end user.

REFERENCES

[1.] PFLEEGER, C. P., PFLEEGER, S. L.: Security in Computing, 3rd edition, Prentice Hall, 2003.

[2.] STALLINGS, W.: Cryptography and Network Security, 3rd edition, Prentice Hall, 2003.

[3.] ASHBOURN, J.: Biometrics Advanced Identify Verification. The Complete Guide, Springer London, 2000.

[4.] http://www.eg3.com/

[5.] CHIRILLO, J., BLAUL, S., CHIRILLO, J.: Implementing Biometric Security, Wiley Publishing, 2003.

[6.] BOLLE, R., CONNELL, J., PANKANTI, S., RATHA, N., SENIOR, S.: Guide to Biometrics, Springer Professional Computing.

[7.] http://www.wikipedia.org/