Continuous Authentication Based on Language Redundancy Analysis and Keystrokes Dynamics

Dwijen Rudrapal 1, Smita Das 2
1,2 Dept. of Computer Sc. & Engineering, National Institute of Technology, Agartala, India
Email: dwijen.rudrapal@gmail.com, dwijen_m@rediffmail.com

Abstract—In this paper, we have proposed a method by which we can continuously authenticate one claimant person through keystroke values of frequent letters in text. Language redundancy in human languages describe that all the letters are not evenly distribute in any text. Few letters in a text occurs most frequently and keystroke dynamics of those letters are taken into consideration as a measure for continuous authentication. A letter may distribute in many places in the text but more or less the keystroke timing of that letter is same. Comparison of a letter keystroke pattern values will authenticate user on regular basis. The proposed method was applied and evaluated on a sophisticated number of genuine users to prove that the authentication performance effective and satisfactory.

Index Terms— Biometrics, Keystroke Dynamics, Continuous Authentication, Language Redundancy Analysis

I. INTRODUCTION

User name and password of some user is the current method for authenticating [1] that user in some computer system. But once the user is logged on to the system who can say that the same user is on that system throughout the whole session? Possibility of local or remote threat from some unauthorized user is higher if the authorized user does not logout from the terminal [2] properly. Therefore, it is necessary to verify the user continuously from the point of view of security and one of possible and useful solution of this problem may be found in biometrics.

There are two categories of biometrics namely, physiological [3] and behavioural. Physiological biometrics [3] is used to measure different things such as, fingerprints, the face, hand (palm) geometry and the iris. Physiological biometrics, which habitually depends on a solitary instant in which user measurements are examined, is further appropriate for a static authentication. Besides, iris scan, fingerprints etc. are extraneous for continuous authentication, since they are invasive, costly and not suitable for Web applications. Alternatively, behavioural biometrics [4] is related to a particular behaviour of a user while performing some task, such as typing, speaking, verifying signature etc. One of such behavioural biometrics is Keystroke Dynamics which is very appropriate for dynamic authentication and it is also reasonably priced. Moreover, in any kind of application related to computer keyboard is very essential and easy to handle with respect to other physiological biometrics. Several studies verified that keystroke dynamics is very appropriate for continuous authentication [5-8].

In these studies it has been found that people have dissimilar typing modes and by analyzing the timings of keystrokes, a person can be recognized. Keystroke dynamics [5] deals with the series of key press and key release events timings while the user types a string. Although current keystroke dynamics methods [9-10] for continuous authentication acquire fairly excellent detection performance, they necessitate long sessions and
considerable amounts of keystrokes to consistently verify users. In this paper, we have used an approach of keystrokes dynamics based on the statistical properties of some language required for verification, helps us to decrease the time necessary to detect an intruder.

II. MOTIVATION

In the course of this paper work we have studied and analysed the active authentication systems using keystroke pattern. We have gone through a situation where we have found that suppose someone has logged onto a machine and writing some text. Now a situation can also arise where without the authentic user’s knowledge an imposter may write some text from the true user’s account. Even it may also happen that in an online exam system, a true user has logged into client machine under a remote server. How can the server track continuously that the same user is using the client machine or some other user is typing the answer instead of the true user. In this scenario we have found keystrokes pattern analysis in continuous authentication is very useful. We have used keystroke dynamics in the phases of authentication and verification of true user as well as imposters. If we capture the keystroke pattern analysis of each and every key pressed in a system, then volume of data will be very high. Therefore, instead of full analysis we have chosen partial analysis of some specific keys. For this we had to depend on statistical properties of language. To find out better keystroke credentials and matching result, in this paper we combined standard deviation of keystroke duration and degree of disorder for keystroke latency as well as disorder for keystroke duration.

III. BACKGROUND

A. Keystroke Dynamics

While a person is typing on a computer using a keyboard, Keystroke Dynamics provides the comprehensive timing information that illustrates exactly when each key was pressed and when it was released. Three timing events namely: key down event, key up event and key press event are generated when a key is pressed.
user authentication can be done successfully. The features that are considered to be account on for authenticating purpose are:

- Keystrokes latencies (fight)
- Duration of a specific keystroke (dwell)
- Pressure (Force of keystrokes)
- Typing speed
- Frequency of error
- Overlapping of specific keys combinations
- Method of error correction.

B. Statistical Techniques for Language Analysis

Natural language is not arbitrary in nature and single alphabetic based replacement does not cover the statistical properties of language. The most frequent latter in English is ‘E’, followed by ‘T’, ‘A’, ‘O’ and ‘I’ whereas the least recurrent are ‘Q’, ‘Z’, ‘X’ [11].

The most common Diagrams [12] are as follows:

```
th, he, in, en, nt, re, er, an, ti, es, on, at, se, nd
or, ar, al, te, co, de, to, ra, et, ed, it, sa, em, ro.
```

Figure 3: Most common Diagrams

The most common Trigrams [12] are as:

```
the, and, tha, ent, ing, ion, tio, for,
nde, has, nce, edd, tis, oft, sth, mon
```

Figure 4: Most common Trigrams

Statistical outlines in a language can be detected by tracing the redundancy of the text in the language. It has been comprehended that various worldwide regularities differentiate text from dissimilar domains and languages. The best-known rule used is Zipf's law [13] on the basis of distribution of word frequencies. According to Zipf's law the frequency of terms in a compilation reduces inversely to the rank of the terms. Zipf's law has been found to apply to collections of written documents in virtually all languages.

IV. RELATED WORK

The revolutionary research work prepared in keystroke dynamics was in 1980 inspired by individual inimitable rhythm when they sent telegraphs. In 1985, Umphress and Williams [9] conducted a more methodical experiment and gave more acceptances to the idea that keyboard dynamics was workable. Garcia's 1986 [14] patent described a system where users used to type their names in order to authenticate. In 1989, a patent was granted to Young and Hammon [15] for their description of a keystroke authentication method. This patent talks about the use of keystroke latencies and keystroke pressures as important dimensions of keystroke behaviour. After the initial level of exploration concluded that keystroke authentication were feasible, researchers set in motion designing experiments that would formulate keystroke dynamics a more practical tool. In 1994, Brown and Rogers [10] took a more realistic approach in their study on keystroke verification. Their research was also the primary one to scrutinize the use of neural networks as a process of classifying claimant vectors.

Then Monrose and Rubin [5] in 1997 proposed identifying users based on free text. They used two chronological features, the interval time and the dwell time of digraphs, in order to build a user summary. These user outlines underwent a pre-processing to perceive and eliminate outliers. Afterwards, the profiles were used to recognize a new session based on one of three anticipated classifiers: Euclidean distance classifier, non-weighted probability classifier and weighted probability classifier. In order to confirm users based on their typing rhythm, Nisenson et al [7] suggested using the Lempel-Ziv algorithm [16], which was originally designed to compress sequences by means of variable rate coding. Gunetti et al. [6] projected a new approach based on the interval of an n-graph, which is the elapsed time between the depression of the first and of the nth (the last) key of a series of n keystrokes. After extracting the temporal features of the n-graphs, based on the following two proposed measures, they calculated the remoteness between two sessions.
that share n-graphs. The first measure, named the "R" (relative) measure, calculates the distance between two sessions as the normalized disorder between them. The second method, which is referred to as the "A" (absolute) measure, considers the absolute values of the temporal features

V. PROPOSED METHOD

In our proposed method we have considered an online exam system conducted by a remote server under which there may be a number of clients. Candidates appearing at the examination may be treated as a valid user with some login id and password. Now, user can login to a client machine under the server and server has to validate continuously that whether the same user is typing the answer or he/she is being assisted by some other invalid user. This can be identified by the keystroke pattern analysis. The steps for user continuous verification are as follows:

1. Valid user will log in to client machine with user preset username and password.
2. An application at the server side will be activated to continuously authenticate the valid user.
3. Throughout the typing session, while user will type text, system will trace occurrences of most frequent alphabets i.e. 'T', 'A', 'O', 'I' and will be captured their key pressed time i.e. keystroke duration for each letter up to a certain number occurrence.
4. Key pressed values for one letter will be saved in database for pre-defined number of times. From these values Standard Deviation (SD) will be calculated.
5. Repetition of the above point for a text will have several SD (say SD₁, SD₂, …,SDₙ) for one letter.
6. For continuous authentication these SD values will be analyzed. Ignoring few cases one true user must have SD with very little deviation.

In this paper we subcategorized our proposed method into following sections namely:

A. Analysis Of Text And Find The Most Frequent Single Letters:

If we check each and every alphabet in a certain text for finding keystroke dynamics, it will be of huge volume and will require large session time. Therefore, we have chosen to check for the most frequent English alphabets i.e. 'T', 'A', 'O' and 'I' occurring in the sample text. In our paper we have chosen only single alphabets. Proposed method will trace occurrence of above alphabets and will store keystroke duration of that alphabet.

B. Capturing Keystroke Features:

A set of commands is sent from the keyboard to the computer while entering a character on a keyboard. Each time a key is pressed down or released, an event is activated. According to key press or key release, this event will have a timestamp which consists of the number of microseconds. The timestamp values are Key-pressed time stamp, Key Code value (the character code), Key released time stamp.

Fig 5 shows plotted values of key pressed duration of characters while typed by a user.

Figure 5: Graphical representation of key-pressed timing.

C. Calculation of Standard Deviation

Standard deviation is a measure of the diffusion of a set of data from its mean. The more spread apart the data, the higher the deviation. Standard deviation is calculated with the following formula:
\[
\sqrt{\frac{1}{N} \sum_{i=1}^{N} (x_i - x')^2} \quad \text{where} \quad x' = \frac{\sum_{i=1}^{N} x_i}{N}
\]  

(1)

Here in this paper, suppose the key pressed durations of alphabet T are 80, 70, 80, 80, 80 in one set and 70, 60, 80, 80, 70. As we considered 5 occurrences of a character Standard deviation of key pressed duration for first set is:

\[
\text{SD\_KP} = 4.
\]

D. User Authentication

The authentication step is mainly executed on the basis of SD calculation. Based on 5 key pressed values for each alphabet one standard deviation will be calculated. This way for one large text typed by the user will have few SD which will show the actual deviation. If the deviation value is within the threshold limit user will be validate as true user. In the following table we have shown the values captured for one small text:

<table>
<thead>
<tr>
<th>Set</th>
<th>KP Value 1</th>
<th>KP Value 2</th>
<th>KP Value 3</th>
<th>KP Value 4</th>
<th>KP Value 5</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>80</td>
<td>70</td>
<td>80</td>
<td>80</td>
<td>80</td>
<td>4</td>
</tr>
<tr>
<td>2</td>
<td>80</td>
<td>70</td>
<td>70</td>
<td>80</td>
<td>80</td>
<td>4.89</td>
</tr>
<tr>
<td>3</td>
<td>80</td>
<td>80</td>
<td>70</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Above table has plotted based on one free text “this sample text is written to evaluate the frequency of alphabet, most frequent letter will be considered with its key pressed duration”. In the text there are 13 occurrences of ‘t’. So, last row is uncompleted.

VI. SIMULATION AND RESULT ANALYSIS

The simulation of the proposed method has done using .Net framework. Every registered user will have own username and password. Through a login screen user will enter and type long text. While typing System will trace alphabets and will store their key pressed values. From set of values System will also calculate the standard deviation. Fig 6 shows calculation values.

Figure 6: Calculation of SD from typed text.
In the above experiment screenshot, only two SD values of alphabet ‘t’ key pressed value calculated. For a large text there will be more SD values of each letter i.e. most frequent occurred letter as cited in this method. All these SD values for corresponding letter analyzed to check continuity of authentication.

VII. CONCLUSION

Taking into consideration, the recent Internet-based attack, it is of no doubt why many organizations are so resolute on firewall and web server security. In our proposed method we have made efforts in making stronger login authentication proposal. In this paper we have set up the requisite of verifying users based on their keystroke dynamics, especially after the verification procedure has successfully finished. Instead of verifying large volume of keystroke data, we have used the keystroke analysis on specific very frequent alphabets of English for volume reduction used for each user separately. We have applied our method using Standard Deviation Calculation as an absolute measure of temporal features. This is further enhanced with the use of threshold value.

VIII. LIMITATION AND FUTURE WORK

In this work we have provided a sample fixed text for all users and different texts also. Though we have obtained satisfactory results from this study but several concerns are there for future research. Here, Standard Deviation along with threshold value can only find absolute measurement of temporal features of keystroke patterns. For e.g. in the preliminary phase of any examination, candidate may type slowly, but later stages the same candidate may type faster in order to complete the answer in time. Thus, for the same user, typing may deviate which can’t be managed by only Standard Deviation calculation.

In future, for relative measurement of temporal features of keystrokes, degree of disorder on keystroke duration can be used. Diagrams and Trigrams distribution in text also can be important measure to enhance the dynamic authentication process in future work.

REFERENCES


