

Keystroke Biometric Recognition on Long-Text Input: A Feasibility Study

Mary Curtin , Charles Tappert, Mary Villani, Giang Ngo, Justin Simone, Huguens St. Fort, Sung-Hyuk Cha

Abstract— While most previous keystroke biometric studies dealt with short input like passwords, we focused on long-text input for applications such as identifying perpetrators of inappropriate e-mail or fraudulent Internet activity. A Java applet collected raw keystroke data over the Internet, appropriate long-text-input features were extracted, and a pattern classifier made identification decisions. Experiments focused on the system's usability under ideal conditions: copy task, long-text input (600 characters), same keyboard for enrollment and testing, and subjects aware of the nature of the study and instructed to type naturally. Essentially 100% identification accuracy was achieved on 8 subjects typing the same text. This accuracy decreased in going to 30 subjects, on copying different testing texts, and on progressively reducing the length of the testing text. In summary, we found the keystroke biometric effective for identifying up to 30 users inputting text under the following conditions: sufficient training and testing text length, sufficient number of enrollment samples, and same keyboard type used for enrollment and testing.

Index Terms—biometrics, security, keystroke.

I. INTRODUCTION

The keystroke biometric is gaining popularity because keyboards are readily available and keystroke data capture is not invasive. Keystroke biometric systems measure typing characteristics believed to be unique to an individual and difficult to duplicate [1, 8, 9]. The typing dynamics or timing pattern has been measured and used in both verification and identification applications [17]. The first commercial product, BioPassword, was developed for hardening passwords in existing computer security schemes [13]. This is one of the less-studied biometrics and researchers tend to collect their own data, so few studies have compared recognition techniques on a common database. Nevertheless, the published literature is optimistic about the potential of keystroke dynamics to benefit computer system security and usability [16].

The most commonly adopted metrics to evaluate a biometric system's authentication accuracy are the False Reject Rate (FRR) and the False Accept Rate (FAR) that respectively correspond to the two popular metrics of sensitivity and specificity [1, 6, 10]. Leggett and Williams [11] showed that keystroke digraph latencies had potential for a static identity verifier at login time, as well as a

dynamic identity verifier throughout a computer session, and Leggett, et al. [12] conducted similar experiments, reporting 5.0% FAR and 5.5% FRR on a long string of 537 characters. D'Souza's experiment weighted the latencies to reduce false acceptances [3]. Brown and Rogers [2] and Obaidat and Sadoun [14] used short name strings for user verification. Dynamic shuffling was also evaluated as a process applied to training samples for neural networks as a means of enhancing sample classification and reducing false acceptance and rejection rates during keystroke analysis [2]. Finally, Gunnetti and Picardi [7] suggest that if short inputs do not provide sufficient timing information, and if long predefined texts entered repeatedly are unacceptable, we are left with only one possible solution, which is using the typing rhythms users show during their normal interaction with a computer; in other words, deal with the keystroke dynamics of *free text*.

The keystroke biometric is appealing for several reasons. First, it is not intrusive and computer users, for work or pleasure, frequently type on a computer keyboard. Second, it is inexpensive since the only hardware required is a computer. Third, keystrokes continue to be entered for potential subsequent checking after an authentication phase has verified a user's identity (or possibly been fooled) since keystrokes exist as a mere consequence of users using computers [7]. Finally, with more businesses moving to e-commerce, the keystroke biometric in internet applications can provide an effective balance between high security and ease-of-use for customers [17].

Generally, a number of measurements or features are used to characterize a user's typing pattern. These measurements are typically derived from the raw data of key press times, key release times, and the identity of the keys pressed. From key-press and key-release times a feature vector, often consisting of keystroke duration times and keystroke transition times, can be created [17]. Such measurements can be collected from all users of a system, such as a computer network or web-based system, where keystroke entry is available, and a model that attempts to distinguish an individual user from others can be established. For short input such as passwords, however, the lack of sufficient measurements presents a problem because keystrokes, unlike other biometric features, convey a small amount of information. For example, there is little information from two consecutive keystrokes, basically the elapsed time between the release of the first key and the depression of the second (the so-called digraph latency) and the amount of time each key is held down (the keystroke duration). Moreover, this information tends to vary for different

Manuscript received March 10, 2006.

The authors are with the School of CSIS, Pace University, Pleasantville, New York, 10570 USA; email (in author order): mcurtin@pace.edu, ctappert@pace.edu, villanmv@farmingdale.edu, gn90004n@pace.edu, js97978w@pace.edu, hs78586w@pace.edu, scha@pace.edu; first author phone: (914-773-3736; fax:914-773-3533).

keyboards, different environmental conditions, and different entered texts [7]. For these reasons we focus our studies on long text input where more information is available.

In this paper, we develop a system specifically designed for long-text input and prove the effectiveness of the system under three favorable conditions: the users input prescribed texts, they use the same keyboard for entering both enrollment and testing keystroke data, and they know that their keystroke data are being used for identification purposes (in fact, they are asked to type consistently in their normal manner for these experiments). With the effectiveness of the system established, we then extend these baseline experiments to less favorable conditions: the users enter arbitrary input texts, and they use different keyboards for enrollment and testing, and they are unaware that their keystroke data are being used for identification purposes (they are given no instruction on how to enter their data).

Although there are many possible applications of the keystroke biometric for long-text input, we are primarily interested in two. The first is an identification (one-of-n) application. A potential scenario for this application is a company environment in which there has been a problem with the circulation of inappropriate (unprofessional, offensive, or obscene) e-mail from easily accessible desktops in a work environment, and it is desirable to identify the perpetrator. The second is an authentication application where the system makes a binary accept/reject decision (yes, you are the person you claim to be, or no you are not). A potential scenario for this application is to verify the identity of students taking online quizzes or tests. This is an important application because the student population of online classes is increasing and instructors are becoming more concerned about evaluation security and academic integrity.

The remainder of the paper is organized as follows. Section 2 describes our keystroke biometric system, having components for data capture, feature extraction, and classification. Section 3 describes the experimental design and section 4 presents the experimental results.

II. KEYSTROKE BIOMETRIC SYSTEM

The keystroke biometric system consists of three components: raw keystroke data collection, feature extraction, and pattern classification.

A. Data Capture

A Java applet was developed to enable the collection of keystroke data over the Internet (Figure 1). The user is required to type in his/her name, although no data is captured on this entry. Also, the submission number is automatically incremented after each sample submission, so the subject can immediately start typing the sample to be collected. If the user is interrupted during data entry, the “Clear” button will blank all fields, except name and submission number, and allow the user to redo the current entry.

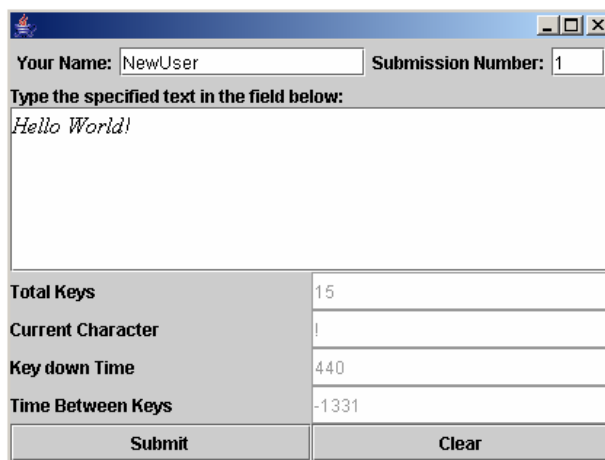


Figure 1. Java applet for data collection.

The raw data file recorded by the application contains the following information for each entry:

- key’s character
- key’s code text equivalent
- key’s location on the keyboard (1 = standard, 2 = left, 3 = right)
- time the key was pressed (milliseconds)
- time the key was released (milliseconds)
- number of left-mouse click, right-mouse click, and double left-mouse click events during the session (note that these are events in contrast to key presses)

Upon pressing submit, a raw-data text file is generated, which is delimited by the ‘~’ character. The aligned version of the raw data file for the “Hello World!” example is shown in Figure 2.

Entry #	Key	Keycode	Location	Press	Release
Num 1	?	Shift	2	1114450735680	1114450736962
Num 2	H	H	1	1114450735991	1114450736311
Num 3	e	E	1	1114450737653	1114450738144
Num 4	l	L	1	1114450738735	1114450739256
Num 5	l	L	1	1114450739786	1114450740277
Num 6	o	O	1	1114450740998	1114450741399
Num 7	Space		1	1114450742090	1114450742420
Num 8	?	Shift	2	1114450743542	1114450745004
Num 9	W	W	1	1114450743872	1114450744263
Num 10	o	O	1	1114450745755	1114450746216
Num 11	r	R	1	1114450747017	1114450747437
Num 12	l	L	1	1114450748138	1114450748549
Num 13	d	D	1	1114450749310	1114450749771
Num 14	?	Shift	2	1114450751373	1114450753776
Num 15	!	!	1	1114450752445	1114450752885
Left Clicks		0			
Right Clicks		0			
Double Clicks		0			

Figure 2. Aligned version of the raw data file for “Hello World!”

In this study, a raw-data file must contain a minimum of 10 keystrokes in order to reliably undergo subsequent processing.

B. Feature Extraction

The system extracts a feature vector from the information in a raw data file. The features are statistical in nature and specifically designed to characterize an individual’s keystroke dynamics over writing samples of 200 or more characters. Most of these features are averages and standard deviations of key press duration times and of transition times

between keystroke pairs, such as digraphs [13, 15]. We measure the transitions between keystrokes from the release of the first key to the press of the second.

While key press duration and transition times are typically used as features in keystroke biometric studies, our use of the statistical measures of means and standard deviations of the key presses and transitions is uncommon and only practical for long text input. As additional features we use percentages of key presses of many of the special keys. Some of these percentage features are designed to capture the user's preferences for using certain keys or key groups – for example, some users do not capitalize or use much punctuation. Other percentage features are designed to capture the user's pattern of editing text since there are many ways to locate (using keys – Home, End, Arrow keys – or mouse clicks), delete (Backspace or Delete keys, or Edit-Delete), insert (Insert, shortcut keys, or Edit-Paste), and move (shortcut keys or Edit-Cut, Edit-Paste) words and characters.

For this study, the feature vector consists of the following 58 measurements:

- The average and standard deviation of the key press durations for the eight most frequent letters of the alphabet (e, a, r, i, o, t, n, s) [5], together with those of the space character and the shift key (20 measurements).
- The average and standard deviations of the transition times between the eight most common letter pairs of the alphabet (in, th, ti, on, an, he, al, er) [5], together with those of a space-to-any-letter and any-letter-to-space (20 measurements).
- The total number of key presses for Space, Backspace, Delete, Insert, Home, End, Enter, Ctrl, and each of the four arrow keys (12 measurements).
- The number of key presses for the left and right keyboard locations of the Shift key (2 measurements).
- The total time to enter the text (1 measurement).
- The number of left mouse clicks (1 measurement).
- The number of right mouse clicks (1 measurement).
- The number of double left mouse clicks (1 measurement).

Two preprocessing steps are performed on the feature measurements, outlier removal and feature standardization. Outlier removal consists of removing any duration or transition time that is far (more than two standard deviations) from the subject's mean duration or transition, respectively. After outlier removal, averages and standard deviations are recalculated. The system can perform outlier removal one or more times, or not at all, and this is a parameter experimentally optimized. Outlier removal is particularly important for these features because a keyboard user could pause for a phone call, for a sip of coffee, or for numerous other reasons, and the resulting outliers (usually overly long transition times) could skew the feature measurements.

After performing outlier removal and recalculation, we standardize the measurements by converting raw measurement x to x' by the formula,

$$x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$$

where min and max are the minimum and maximum of the measurement over all samples from all subjects [4]. This provides measurement values in the range 0-1 to give each measurement roughly equal weight.

C. Classification

A Nearest Neighbor classifier, using Euclidean distance, compares the feature vector of the test sample in question against those of the samples in the training set. The author of the training sample having the smallest Euclidean distance to the test sample is identified as the author of the test sample.

III. EXPERIMENTAL DESIGN

We design three identification experiments to verify that high recognition accuracies can be obtained from this keystroke biometric system under the most favorable conditions: cooperative subjects aware that their keystroke data is being used for identification purposes and a copy task where the subjects enter predefined texts. The subjects are asked to key in the texts as naturally as possible using their normal typing speed and pattern so as to provide representative and consistent input samples. They are also asked to correct any input errors, potentially providing error-correction keystrokes to assist in the identification task. Finally, subjects are asked to leave at least a day between entering samples, so that the samples are spread out over time similar to what might be expected in the data taking of an actual application environment.

For these experiments, subjects enter three copy-task texts, each several times. The first two texts are approximately 600 characters long and the third 300 characters long. The first text is used for training. The first experiment uses the leave-one-out method to test recognition accuracy of new samples of the same text used for training. The second experiment uses the first text for training and the second for testing, and the third experiment uses the first text for training and the third for testing. This structure enables us to compare recognition accuracies on new input of the same text, on input of a different text of the same length, and on input of a different text of shorter length.

IV. RESULTS AND CONCLUSIONS

We conducted preliminary experiments on 8 subjects who entered each of the three copy-task texts approximately 10 times. For the first experiment, using the leave-one-out procedure, the Euclidean distance was computed between each entry of the first text and every other entry of the same text. Recognition accuracy of 100% was achieved. For the second experiment, Euclidean distance was computed between samples of the first text (training) and samples of the second text of roughly equal length (testing).

