Authentication via Keystroke Dynamics

Fabian Monrose
New York University

Aviel Rubin
Bell Communications Research

Abstract

In an effort to confront the challenges brought forward by the networking revolution of the past few years, we present improved techniques for authorized access to computer system resources and data. More than ever before, the Internet is changing computing as we know it. The possibilities of this global network seem limitless; unfortunately, with this global access comes increased chances of malicious attack and intrusion. Alternatives to traditional access control measures are in high demand. In what follows we present one such alternative: computer access via keystroke dynamics.

A database of 42 profiles was constructed based on keystroke patterns gathered from various users performing structured and unstructured tasks. We study the performance of a system for recognition of these users, and present a toolkit for analyzing system performance under varying criteria.

Keywords: Biometrics, keystroke dynamics, pattern recognition, computer security.

1 Introduction

Todays' society depends heavily on computers. They are an integral part of nearly every aspect of our lives. They control the communications, aviation, and financial services. We entrust them with vital information such as medical and criminal records, and use them to manage our taxes, pay our bills, and write personal letters. However, this increasing dependency on computers coupled with the growing emphasis on global accessibility, has unveiled new threats to computer system security. Traditional measures such as passwords and PINs are no longer adequate and we need to investigate more advanced safeguards against unauthorized access to computer resources.

One such safeguard is keystroke dynamics. As the name implies, this method analyzes the way a user types at a terminal by monitoring the keyboard inputs thousands of times per second, and aims to identify users based on certain habitual typing rhythm patterns [Mil94].

We argue that the use of keystroke rhythm is a natural choice for computer security. This argument stems from observations that similar neuro-physiological factors which make written signatures unique, are also exhibited in a user's typing pattern [JG90]. When a person types, the latencies between successive keystrokes, keystroke durations, finger placement and applied pressure on the keys can be used to construct a unique signature for that individual. For well-known, regularly typed strings, such signatures can be quite consistent. Furthermore, keystroke dynamics is not intrusive, making it very applicable to computer access security as the users will be typing at the keyboard anyway.

It is widely accepted that certain behavioral patterns, such as handwritten signature dynamics can be used reliably as identity verifiers. In this paper, we investigate the possibility of using yet another behavioral characteristic for identification purposes - a person's keyboard typing rhythm.

1.1 Searching for Clues

Traditional security measures used in current computing systems are simply inadequate. Security breaches to restricted documents are now commonplace events. With the dramatic increase of interest in computer security over the past few years, scientists and engineers are now turning their efforts to biometrics as a means of identification.

1.1.1 Its in your Hands, Eyes and Face

Biometrics, the physical traits and behavioral characteristics that make each of us unique, are a natural choice for identity verification. Unlike passwords and PINs biometrics can not be lost, stolen or overheard, so they offer a potentially foolproof way of determining an individual's identity. Physiological characteristics, such as fingerprints, are good candidates for verification, because they are unique across a large section of the population. Moreover, they are hard to forge without causing severe trauma to the impersonator.

Biometrics are gaining popularity because when used in conjunction with methods built upon what an individual knows (like a password), or what s/he possesses (example an ID card), they provide an extra level of security. Some of the identifying biometric features being used for identification based systems include hand geometry, thermal patterns in the face, blood vessel patterns in the retina and hand, finger and voice prints, and handwritten signatures. Today, a few devices based on these biometric techniques are commercially available. However, some of the techniques being deployed are easy to fool, while others like iris pattern recognition, are too expensive and invasive.

CCS 97, Zurich, Switzerland
Copyright 1997 ACM 0-89791-912-2/97/04 ..$3.50

1.1.2 Not What You Type, But How You Type

Keystroke dynamics, while more of a behavioral trait than a physiological characteristic, is also a good sign of identity. Moreover, unlike other biometric systems which may be expensive to implement, keystroke dynamics is almost free. The only hardware required is the keyboard.

The application of keystroke rhythms to computer access security is not entirely new. There has been some sporadic and episodic work done in this arena. We now survey related work that has appeared in the literature over the past few years.

2 Related Work

Joyce and Gupta [JG90] present a comprehensive literature review of work related to keystroke dynamics. Readers interested in the psychology of keystroke rhythms should consult Leggett et al. [LW88]. We briefly summarize these efforts and other material that has been undertaken since their report.

In 1980, Gaines et al. [GLPN80] conducted experiments with seven secretaries in which they were asked to retype the same three paragraphs at two different times over a period of four months. Keystroke latency timings were collected and analyzed for a limited number of digraphs and observations were based on those digraph values that occurred more than 10 times. A test of statistical independence was carried out using the *T-Test* under the hypothesis that the means of the digraph times at both sessions were the same, and with the assumption that the two variances were equivalent. Although their results were encouraging, the sample population was too small and the amount of data required to build the reference profiles was unacceptable.

Similar experiments were carried out by Leggett et al. [LW88] with 17 programmers. The authors reported an identity verifier which validated the results of [GLPN80]. Leggett et. al. report a system with false alarm rate of about 5.5 percent and importer pass rate of approximately 5.0 percent. In their experiments a latency was considered valid if it fell within 0.5 standard deviations of the mean reference digraph latency, and "accepted" a user if more than 60 percent of the comparison between the test signature and the mean reference latencies were valid. However, as in the previous studies there was a major limitation in that the amount of data required was excessive; for example, a total of more than 1000 words was needed from each participant. Such a static authentication system would not fair well in practice. Furthermore, the impostor pass rate was too high.

There has also been some work done by Garcia[Gar86], and Young and Hammon [YH89]. Garcia's approach utilizes the covariance matrix of the vectors of reference latencies as a measure of the consistency of the individual's signature. Then the Mahalanobis distance function is used to determine similarity between reference and verification profiles. By contrast, Young and Hammon used the Euclidean distance between the two vectors for comparing a number of attributes which may include keystroke pressures and time to type a predefined number of words. Unfortunately, no data is available about the performance of these systems as they were developed for commercial ventures.

One of the more promising research efforts in applying keystroke dynamics as an authentication method is the work done by Joyce and Goyal[JG90]. Their approach is relatively simple and provides very impressive results. It is based on using keystroke information obtained during the login process of a modified login sequence. Their system requires new users to provide eight reference signatures by typing their usernames, passwords, first and last names eight (8) times. Their approach to the removal of outliers is discussed in Section (5).

Joyce et. al. represent the mean reference signature as:

$$\mathcal{M} = \{\mathcal{M}_{username}, \mathcal{M}_{password}, \mathcal{M}_{firstname}, \mathcal{M}_{lastname}\}$$

At verification time, the user provides a test signature $\mathcal{T}$ which is compared with $\mathcal{M}$ to determine the magnitude of difference between the two profiles. Given $\mathcal{M} = (m_1, m_2, ..., m_n)$ and $\mathcal{T} = (t_1, t_2, ..., t_n)$ where n is the total number of latencies in the signature, the verifier computes the magnitude of difference as the L_1 norm. Positive identification is declared when this difference is within a threshold variability of the reference signature. The mean and standard deviation of the norms $||M - S_i||$, where S_i is one of the eight training signatures, are used to decide the threshold for an acceptable difference vector between a given $\mathcal{T}$ and $\mathcal{M}$.

Some neural network approaches have also been undertaken in the last few years [HMB89][BR93][Ale96]. While the back-propagation models used yield favorable performance results on small databases, neural networks have a fundamental limitation in that each time a new user is introduced into the database, the network must be retrained. Moreover, for applications such as access control, the training requirements may be prohibitively expensive and time consuming. To overcome the problem of continuous retraining, most systems based on neural network approaches partition the database into smaller groups of subjects. However, in situations where there is a high turnover of users, the down time associated with retraining can be significant.

The ideas presented in the work of Joyce et. al. were used as a foundation for our research. Our main goals were to:

- Extend the basic research on keystroke dynamics previously studied by Gaines et al. [GLPN80], Leggett et al.[UW85] [LW88][LU89], Bleha [BSH90], and Joyce and Gupta [JG90].
- Examine the use of keystroke durations (i.e., length of time keys are depressed) in addition to keystroke latencies (i.e., time between successive keystrokes).
- Develop robust and general methods which allow for the examination of graphs of varied lengths. To investigate whether certain features are better candidates than others, and to explore methods for dynamically updating these graphs in the user's profile.
- Examine the use of "Free-Style" (i.e., non-structured) text as opposed to constant chosen phrases like user names (used in previous studies).
- Incorporate a larger sample set taken from users of varying ages, nationalities and backgrounds. In addition, to provide a framework upon which further research can be based.

In the next section we discuss some basic fundamental notions in authentication and pattern recognition, before proceeding to discuss the goals outlined above.

3 User Authentication and Recognition.

Strictly speaking, authentication requires the person being identified to lay claim to an identity, so that the system may decide on either accepting or rejecting the claim. As with any security system, given that the claimant is, or is not, a true instance of the user, there are four possible outcomes; Acceptance of Authentic (AA), Acceptance of Impostor (IA), Rejection of Authentic (RA), and Rejection of Impostor (RI) [Dau93]. Since the first and fourth outcomes are the desired results, one's main goals in designing an authentication system are to maximize the conditional probabilities of (AA) and (RI), while minimizing the likelihoods of (IA) and (RA). In practice, these goals are not always achievable and the level of acceptance for type I errors is application specific.

The overall picture is slightly different for the case of profile recognition. The main focus here is to examine a set of stored features about some sample of the population, and to select the one that yields the best possible match to the unknown profile being presented. This paper concentrates on the problem of recognition, as it is strictly more difficult than authentication. We believe any success in recognition is directly transferable to authentication and therefore focus our efforts on the problem of profile recognition.

4 Data Collection

The performance results reported here are based on a database of profiles collected over a period of 7 weeks. The data was collected simultaneously on Sun workstations running `SunOS 4.1.x` at New York University and Bell Communications Research. Typing proficiency was not a requirement in this study although almost all participants were familiar with computers. Unlike previous studies in which the observers had complete control over the collection of the data [BSH90], participants ran the experiment from their own machines at their convenience. Binaries where made available, and participants downloaded and executed the experiment on their local machines. The results were automatically uuencoded and electronically mailed back to us. Figure (1) is an example of a profile received for a user in the data set.

Approximately 60% of the participants were aware of the purpose of the experiments. The subjects were asked to retype a few sentences from a list of available phrases and/or to type a few sentences on the spur of the moment. The text to be typed was placed in the top half of the display and the participants keystrokes were displayed in the bottom half. As in the case of [BSH90] it was not necessary for participants to shift their focus between the text to be typed and their keystrokes.

4.1 System Toolkit

As an aid to understanding the behavior of each of the classifiers, we developed a C++ toolkit for analyzing the data. The interface was built using the `xview` library routines, and serves as a frontend to the main recognition engine.

The toolkit was extremely valuable in diagnosing system behavior. The ability to sort the output based on a number of different criteria, simplified the analysis phase and helped in determining faults in the classifiers. The graphs automatically generated for the `Matlab` and `Gnuplot` systems proved very useful when investigating strange results. For example, an examination of the resulting latency graphs for a particular user quickly explained why the imposter pass rate on this profile was extremely high; s/he was a *very* inconsistent typist. In addition, the toolkit allowed us to quickly pinpoint those profiles which contained erroneous timing values (as discussed in Section (5)), and discard them from the experiments.

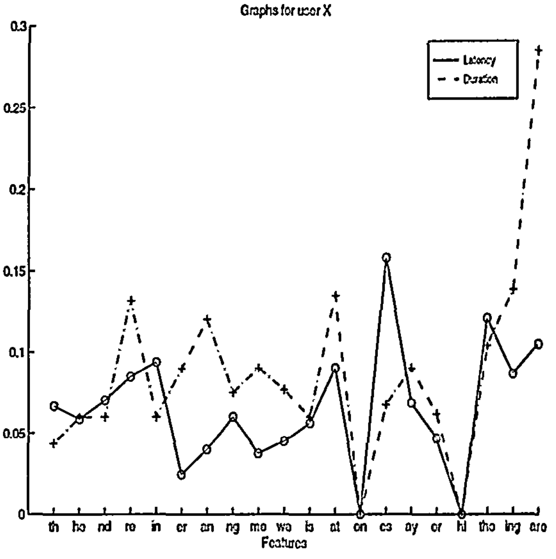


Figure 1: Example reference profile. The top n most frequent features in the pattern vector are shown on the X-axis. The users keystroke latencies, as well as keystroke durations, are graphed above. The graphs show that on average, the user suppresses keys for a longer period than it takes him/her to type them.

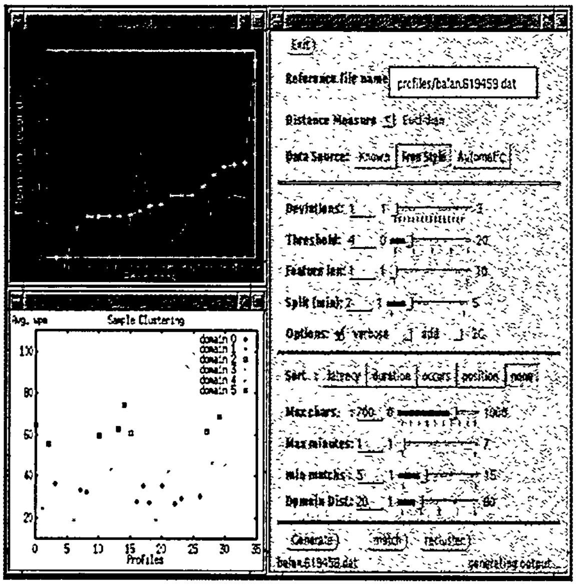
Figure (2) is from the main panel of the interface. The toolkit also includes a lexer and parser written in `flex` and `bison` respectively. The lexer allows us to examine features of lengths between 1 and 8 characters anywhere in the data. The parser builds and maintains a tree whose internal nodes point to the features of interest in the raw data, so that actual timing values can be easily calculated. We believe the functionality of this toolkit will be very useful in further research, as it provides a platform for automatically incorporating new, viable, profiles into the sample sets.

4.2 Clustering Method

Correctly recognizing a user from a stored set of characteristics can be computationally expensive. Therefore, to reduce the search time it is necessary to partition the data into cluster domains. Partitioning also provides a quick way for establishing rough properties on the data set.

Usually, a clustering criterion represents a heuristic scheme, or is based on the minimization (or maximization) of a certain performance index. We have chosen a heuristic approach that is guided by intuition; we cluster profiles based on the typing speed of the users, i.e, the number of words per minute (wpm) typed in the given profile. This idea partitions the database into subsets whose in-class members are "similar" in typing speeds and whose cross-class members are dissimilar as in the corresponding sense.

The heuristic used in calculating the cluster domains is a slight variation of the *maximin-distance* algorithm [TG81]. Initially, we arbitrarily select one of the profiles as the first cluster center c_1. Next, in accordance with the algorithm, we determine the farthest sample from the chosen center, and label it as cluster center c_2. The distance from the remain-



Reference file name: profiles/balan619459.dat

Distance Measure: Euclidean

Data Source: Known | Free Style | Automatic

Deviations: 1 1 — 3

Threshold: 4 0 — 20

Feature len: 1 1 — 30

Split (min): 2 1 — 5

Options: verbose add 2C

Sort: latency | duration | occurs | position | none

Max chars: 700 0 — 1000

Max minutes: 1 1 — 7

min matchs: 5 1 — 35

Domain Dist: 20 1 — 80

balan619459.dat generating output...

Figure 2: The system toolkit is a very robust utility that assists in the visualization, tuning, and overall analysis of the data. It is very useful as it provides a graphical display that helps in understanding and interpreting results. The various options allow the user to diagnose the performance of each of the classifiers in detail. The above is a snapshot from the main panel of the interface.

ing profiles to centers c_1 and c_2 is computed, and for every pair of these computations, we save the minimum distance. Then, the maximum of these minimum distances is selected. If this distance is an reasonable fraction of the distance to the established cluster centers, we denote the corresponding profile as the next cluster center. Otherwise the algorithm is terminated. The distance for the remaining profiles to each of the cluster centers is computed, and the above procedure repeated. Figure (3) shows a sample clustering based on this approach.

When presented with an unknown profile $\mathcal{U}$ with given wpm $\mathcal{U}_{wpm}$, we attempt to recognize $\mathcal{U}$ with all members of a set of up to k neighboring clusters to the domain that $\mathcal{U}_{wpm}$ belongs.

Clustering profiles based on this approach has limitations in that as the system is used some reclustering has to be performed. With each successful authentication, the pattern vector for user i will get updated. In addition, for non-consistent typists their wpm will almost always vary. This variation may change the membership of the profile from cluster c_i to c_j, and require updating and recalculating some neighboring clusters. Undoubtedly, there exist more elegant and efficient clustering techniques which may require less reclustering. We are currently incorporating a modified K-Nearest Neighbor [Dud73] approach to replace the *maximin* algorithm in our implementation.

5 Experimental Method

The profiles collected over the course of the experiment were represented as N-dimensional feature vectors. Pairwise similarities and differences were calculated using the Normalized Euclidean distance (1), and weighted (4) and non-weighted (2) maximum probability measures. The reference profile, called the $\mathcal{R}$ file, was processed in a method similar to that of Joyce et al. [JG90].

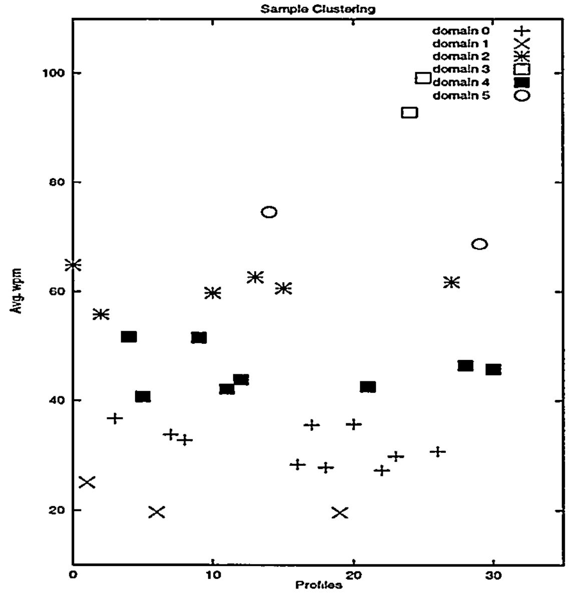


Figure 3: Clustering of user profiles based on the number of words per minute. In order to reduce the search space the database is partitioned into cluster domains whose in-class members are "similar" in typing speeds, and cross-class members dissimilar in the corresponding sense. Here, a profile was labeled as a new cluster center if its distance was greater than one-tenth the average distance to all other established cluster centers.

First, the mean reference profile was computed by calculating the mean and standard deviations of both the latencies and durations for all features in the data. Each latency and duration is compared with its respective mean, and any measurements greater than T standard deviations above the mean (i.e. outliers) discarded. The means of the remaining latencies and durations are then recalculated.

The aforementioned produced better results than the averaging and shuffling techniques reported by [Ble88]. Figure (4) shows the effect of varying T against the number of outliers. With $T = 0.5$, on average, more than 50% of the data is discarded.

Our initial sample size of 42 was reduced to 31 as we were forced to eliminate certain profiles due to erroneous timing results. Most of these unacceptable patterns were a consequence of the users providing their reference profile on systems which relied on an `XDM`[1] host. Since these machines relied on a host for their timings, the values gathered were not representative of the users patterns.

Each of the remaining profiles were divided into a learning and training sets; every n minute(s)[2] a new pattern vector v was created from the data in profile $\mathcal{R}$. Markers inserted in the data during collection allowed us to categorize what the data represented (i.e., structured or unstructured text). In a manner similar to [Dix79], one of the v vectors is selected from the data set and the program pretends v is "unknown".

We are now faced with the problem of recognizing v among m other patterns each of N dimensions, but with the possibility of some of the features in each pattern being blank. Figure (5) illustrates what is meant by a "blank" feature. This procedure is repeated until every vector has played the part of "unknown".

[1] X Display Manager

[2] $n = 1$ was used in results reported here.

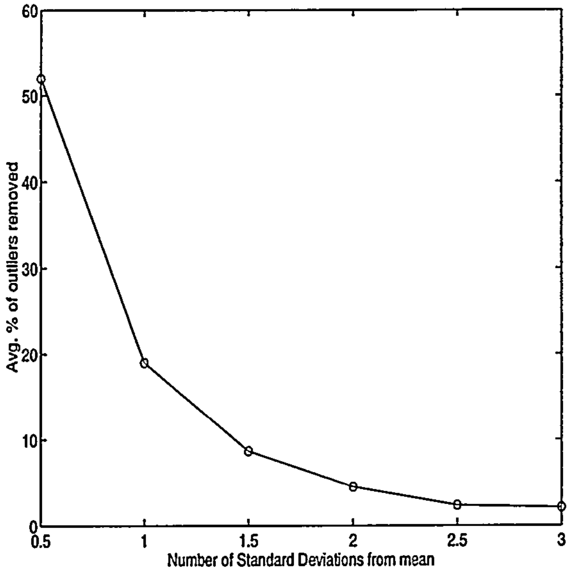


Figure 4: The figure above shows the effect of varying T with respect to the number of outliers removed from the data. At $T = 0.5$ standard deviations above the mean, more than 50 % of the data in each reference profile is discarded. This threshold value is clearly too high for the majority of the users in the database. The number of outliers steadily decreases as T increases.

	Acceptance	Acceptance	Acceptance
T	Known vs Known	Free vs Known	Free vs Free
0.5	62.9%	37.7%	insufficient data
1.0	79.1%	45.0%	21.3%
1.5	78.7%	33.4%	17.1%

Table 1: The Euclidean distance is used as a similarity measure between the vectors. With T set to 1.0 standard deviations above the mean, approximately 80% of the users were correctly recognized based on samples derived from structured text.

6 Classification Algorithms

In what follows we discuss each of the classifiers used in recognition. Results and observations are also reported in each section.

6.1 Euclidean Distance Measure

Our first metric of "similarity" was based on the Euclidean distance between the vectors. Let $R = [r_1, r_2, ..., r_N]$ and $U = [u_1, u_2, ..., u_N]$ then the Euclidean distance between the two N dimensional vectors U and R, is defined as

$$D(R, U) = \left[\sum_{i=1}^{N} (r_i - u_i)^2 \right]^{1/2} \qquad (1)$$

For an "unknown" U the pairwise Euclidean distances $D(R_i, U), i = 1, 2, ..., n$ where n = number of patterns vectors in the database, were rank ordered and the profile with the minimum distance to U was chosen. Similar work is done by Harmon et al. [HKLR81]. Our findings are reported in Table (1).

The values in column (2) represent the Acceptance of Authentic (AA) when the "unknown" from the learning set (generated from structured text) was the best match to a training pattern generated from the same user typing different structured text. With $T = 0.5$ deviations above the mean, the recognition level was 62.9%. This relatively low level can be attributed to the large loss of data (more that 50% on average) in each of the profiles. With $T = 1.0$ we observe an increase in AA. However, as T increases, so does the probability of impersonation. This is reflected in the drop in AA when T was increased to 1.5.

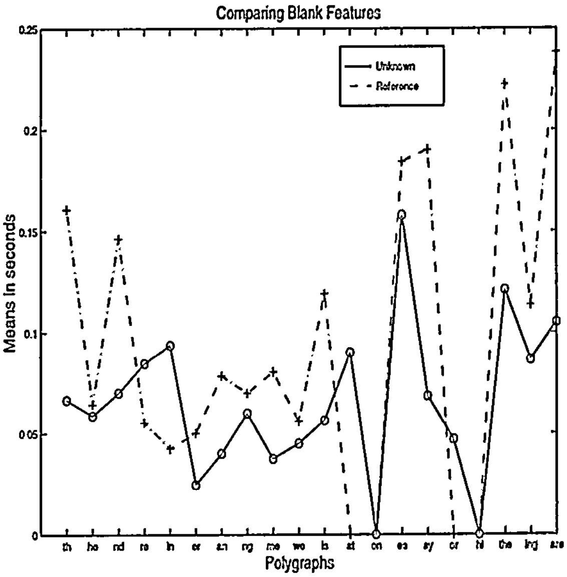


Figure 5: Some of the vectors being compared may have blank features as depicted above. Features are considered blank if the mean value in either the Reference or "Unknown" profile is 0.0. A feature is considered blank if it's occurrences are less than a threshold value, set via the data analysis toolkit. In the above example, **at**, **on**, **or** and **hi** are blank features.

Column (2) depicts correct identification when U and R were from the same user performing *different* tasks. There appears to be a relatively larger variation in keystroke rhythm when comparing the same user performing structured verses unstructured tasks. This is exemplified in the low AA. One factor that may have affected some patterns in the "free style" category was that these users were probably unsure of what to type, and so their patterns may not have been a true reflection of their normal typing rhythm. However, for very consistent typists, correct identification was still possible, which is encouraging. Even with 1.5 standard deviations above the mean, the imposter pass rate on those consistent typists was near zero. Figure (6) graphs the closest match to one of the "better" typists.

Column (4) tabulates the results when both u_i and r_i were generated from patterns vectors of users performing unstructured tasks. Only the most consistent typists were correctly recognized. This is primarily due to the lack of data available in this section. However, we believe its performance will be comparable with that of column (1) given adequate amounts of data. We expect to collect further samples in this category, and investigate this claim in the coming weeks.

It should also be pointed out that while these percentages reported here are lower than some previous efforts, we believe they reflect the nature of the experiments; unlike past studies where the observers had complete control over the experiments, our data gathering phase resembled more closely a real environment. We now discuss the second classifier

	Acceptance	Acceptance	Acceptance
T	Known vs Known	Free vs Known	Free vs Free
0.5	68.4 %	40.4 %	insufficient data
1.0	85.6 %	48.9 %	21.5 %
1.5	84.3 %	36.5 %	18.2 %

Table 2: Non-weighted probability measure. By setting a threshold value for the number of standard deviations above the mean to 1.0, approximately 85.6% of the users were correctly recognized.

used in our experiments.

6.2 Non-Weighted Probability

In this section, let U and R be N-dimensional pattern vectors as defined previously. Furthermore, let each component of the pattern vectors be the quadruple $\langle \mu_i, \sigma_i, o_i, X_i \rangle$, representing the mean, standard deviation, number of occurrences, and actual data values for the i^{th} feature. Assuming that each feature for a user is distributed according to a Normal distribution, we calculate the score between a reference profile R and unknown profile U as

$$Score(R, U) = \sum_{i=1}^{N} S_{u_i} \tag{2}$$

where

$$S_{u_i} = \frac{1}{o_{u_i}} \Big[\sum_{j=1}^{o_{u_i}} Prob\Big(\frac{X_{ij}^{(u)} - \mu_{r_i}}{\sigma_{r_i}} \Big) \Big] \tag{3}$$

and $X_{ij}^{(u)}$ is the j^{th} occurrence of the i^{th} feature of U.

In other words, the score for each u_i is based on the probability of observing the value u_{ij} in the reference profile R, given the mean (μ_{r_i}) and standard deviation (σ_{r_i}) for that feature in R. Intuitively we assign higher probabilities to values of u_i that are close to μ_{r_i} and lower probabilities to those further away. The "unknown" vector is then associated with the *nearest neighbor* in the database, i.e., to the person who maximizes the probability of the feature vector. Table (2) summarizes our findings using the approach outlined above. It is of the same format as Table (1).

We observed that on average, this method performs better than the previous classifier, with a slight increase in computation. By setting the threshold deviation value at 1.0 the acceptance of authentic increased by more than 6% over (1). However, the decrease in AA shown in column (3) compared to Table (2) further reflects the large variation that was observed from the same user performing structured verses unstructured tasks. This lead to the incorporation of weights and other optimizations, discussed in the next section.

6.3 Weighted Probability Measure

Empirical tests showed that it may be reasonable to attach weights to the features. Some features are more reliable than others simply because they come from a larger sample set or have a relatively higher frequency in the written language; example in English er, th, re should constitute greater weights than qu or ts. Thus, we incorporated the notion of weights, and the score between profiles R and U is

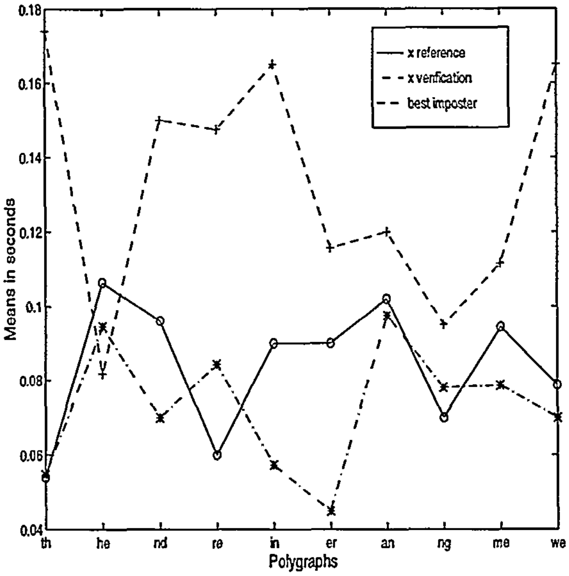


Figure 6: The graph above shows the closest match to one of the more consistent typist. The imposter past rate for these users was essentially zero.

now computed as

$$Score(R, U) = \sum_{i=1}^{N} \Big(S_{u_i} * weight_{u_i} \Big) \tag{4}$$

where

$$weight_{u_i} = \begin{bmatrix} 0 & \text{if } u_i \text{ or } r_i \text{ is blank} \\ \frac{o_{u_i}}{\sum_{k=1}^{N} (o_{u_k})} & \text{otherwise} \end{bmatrix}$$

In essence, the weight of u_i is the ratio of its occurrences relative to all other features in the pattern vector U. Features that are based on many occurrences are considered more reliable and weighted higher than those features that come for a smaller sample set.

	Acceptance	Acceptance	Acceptance
T	Known vs Known	Free vs Known	Free vs Free
0.5	71.9 %	39.4 %	insufficient data
1.0	90.7 %	44.1 %	23.0 %
1.5	84.6 %	33.7 %	18.2 %

Table 3: Weighted probability measure. Each feature is weighted based on its relative occurrence in the feature space. The level of recognition for authentic users was approximately 90%.

Table (3)[3] outlines our observations using this weighted probability measure. This method performed better than both of the previous classifiers and showed an overall increase of more than 10% in the acceptance of authentic users compared to (1). While there is still much room for improvement, we believe these results provide motivation for additional studies.

[3] the results reported here reflect improvements since the submitted version of this paper

7 Applications

Keystroke dynamics has many applications in the computer access security arena. Some immediate applications include:

- Access to privilege services - example, root level access to the master server hosting a Kerberos [SNS88] key database can be authorized via keystroke dynamics. Any user accessing the server is asked to type a few words or a pass phrase in conjunction with his/her username and password. Access is granted if his/her typing pattern matches within a reasonable threshold of the claimed identity. This safeguard is effective as there is usually no remote access allowed to the server, and the only entry point is via console login.
- Access to highly restricted documents or execution of tasks in an environment where the user must be "alert" at all times, for example air traffic control. Keystroke dynamics may be used to detect uncharacteristic typing rhythms (brought on by drowsiness, fatigue etc.) in the user and notify third parties when appropriate.

8 Discussion and Further Areas of Research

8.1 Left versus Right Handedness : A Preliminary Report

We now present some preliminary observations on the recognition of left versus right handers. Figure (7) represents the pattern vector for a left handed user $\mathcal{W}$ sorted by keystroke duration.

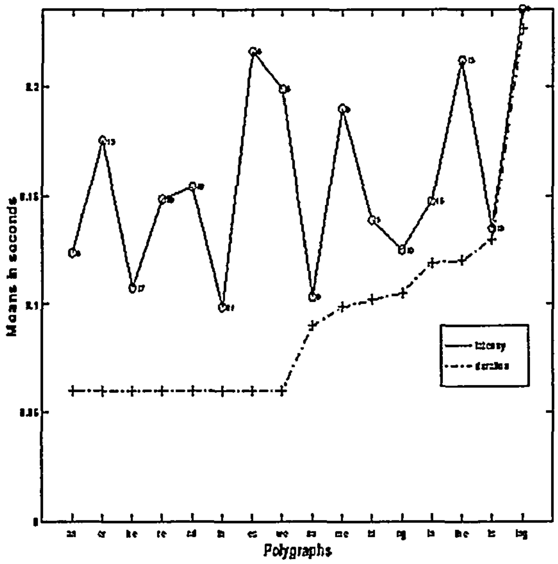


Figure 7: Reference profile for a left handed user. We split the keyboard based on the standard hand positioning. The features are sorted based on keystroke durations. The graphs show both durations and latencies for each feature. The values for those features that fall on the left hand side of the keyboard are much lower than those on the right hand side. Furthermore, the durations for those keystrokes are almost identical.

The accumulated score for the left-handed features (those features like er, es, re, we) is much lower than those of the right hand. This may be because $\mathcal{W}$ has more control with his left hand and as such, does not suppress keys for as long a duration as with the right hand. Similar results, albeit not as dramatic, are observed for other left handed users.

It is plausible that information from keystroke durations may be used in tilting the decision to accept or reject a claimant during authentication. However, no concrete conclusion may as yet be made as the total number of left handers accounted for a small percentage of the sample set.

8.2 Dynamic Identity Verification

Results from this and other experiments provide incentive for building a recognition system which (passively) authenticates the user during the lifetime of a login session. However, to accomplish this task, alternative methods of recognition need to be explored. Because of the continuous nature of the problem, i.e. new measurements arriving each second, if we tried to apply the normal methods of least squares estimation, we would fall hopelessly behind. What we intend to pursue for computation in real time is the method of weighted recursive least squares estimation [PTVF92].

An alternative method is the work by Leggett et al. [LU91], which involves the application of sequential statistics theory. Their approach to solving the problem of computing the test mean digraph values *before* all the digraphs have been accepted, is based on computing the probability density functions and zone of acceptance for the test digraphs. We are planning on conducting comparative studies on both of these techniques in the next phase of our experiments. This second phase of experimentation entails incorporating a dynamic authentication system on a network of Linux PCs, and gathering data over a period of a few weeks.

8.3 Breaking PGP

Pretty Good Privacy (PGP) originally developed by Phil Zimmerman, is a software package for secure electronic mail. Its security is based on cryptographic techniques including data encryption, digital signatures, and one-way hash functions. We refer interested readers to [Zim95] for further insight on PGP.

One future undertaking involves attacking the seed generation mechanism of this popular electronic mail security program. We believe keystroke dynamics can be used to exploit a weakness in the protocol. During the key generation phase, the user's keyboard latencies are measured. If the pass-phrase is long enough, PGP does not prompt the user for additional input, and uses the measured latencies as a source of randomness for the initial seeds for the encryption and signature routines. It is the generation of this seed value we intend to exploit.

We will investigate the possibility of discovering the seed used by PGP, by having one of the users in the database generate a PGP public and secret key on his/her own. In an effort to discover this key pair, we will attempt to calculate a range value for the seed using the profile we have previously built for the participant. Next, we will substitute values (in parallel) in this range as the seed to the PGP routines, and compare the resultant public key with the real public key. As we will not have access to the data typed by the participant during the key generation phase, this attack is particularly important.

8.4 Privacy Issues

Our data collection techniques require us to record every key typed by the user. Besides timing information, we use the actual characters for our data analysis. In some applications, such as login authentication, this is perfectly reasonable. However, privacy becomes an issue once passive monitoring of users is involved.

If we could ensure the user that the data we collect can only be used for keystroke analysis, and not to reconstruct what they type, then privacy would probably not be violated. However, there is no obvious way to do this. Passive monitoring will only work in environments where users have no expectation of privacy.

9 Conclusions

We have investigated the performance of automatic techniques for user recognition based on keystroke dynamics. As the names implies, this method analyzes a user's typing pattern by monitoring the keyboard inputs thousands of times per second, and aims to identify users based on habitual patterns in their typing rhythm.

In the implementation described, three classifiers were considered. The first is based on a Euclidean distance measure (1), in which an "unknown" profile is associated with the reference profile in the database which minimizes the distance over all reference profiles in the database (i.e. the best score). Probabilistic measures (2), in which we assumed a Normal distribution for each of the features in the pattern vector, were investigated and used as the second classifier. The idea here is to maximize the probabilistic score for a given "unknown" profile against the database of reference profiles. The third classifier (4) involved optimizations on the second, including the incorporation of weighted scores. The scores are adaptive and depend on the frequency distribution of the individual features. The weights emphasize the classification power of the most reliable features.

The correct identification rate using the weighted probabilistic classifier was approximately 90%, which represents improvement with respect to the rates of the other classifiers. We believe significantly higher results can be achieved in more controlled environments, such as in its application to console logins on Kerberos servers.

It must be stressed that the data collection and feature extraction phase were performed in an uncontrolled setting on purpose, in an effort to more closely simulate a real environment. Participants down-loaded and executed the experiments from their machines at their leisure, and the results were automatically encoded and electronically mailed back to us. As such, the recognition performance achieved in the experiments are clearly specific to our data collection method and reflect some of the obstacles posed to keystroke dynamics in such uncontrolled environments. We expect that the incorporation of additional nearest neighbor classifiers, together with a modified recursive least squares estimation technique, will yield higher performance rates, albeit at some computational expense.

The results obtained so far will be verified on a larger and more meaningful database in the near future. A more difficult problem that we did not consider is the reliable rejection of profiles which were not in the database. An important capability of the use of multiple classifiers (as is intended in the next phase of experiments) involves the need to reject the input data when it cannot be matched with *significant* confidence to any of the database entries. While this does not present a problem for user verification[4], where our methods would perform well, such results are important for profile recognition given that we may need to automatically add profiles to the database. We expect the data presented here to be the first results of a project which will compare several techniques for the automatic recognition of users based on a common database of users, thereby providing quantitative information on the performance of different strategies.

We are performing more detailed studies related to robustness to [network] timing accuracy, automatically learning new profiles, and the use of a limited number of features for each individual. Tradeoffs between reducing imposter pass rates while not significantly increasing the rejection of authentic (*RA*) also warrant further studies. Work is currently underway to provide conclusive results to our claim that the use of free style text (i.e. unstructured text) will perform comparable to that which can be attained with structured text. This is important, as we hope it will provide the necessary foundation for the design of a dynamic authentication system.

Our work also presents a unique toolkit for analyzing the performance of each of the classifiers, based on various tunable parameters. The toolkit, which provides a graphical interface to the main recognition engine, will be of great value in future experiments as it allows for the effortless incorporation of new profiles into the database.

We believe keystroke dynamics can be used effectively as a safeguard to unauthorized access to computer resources and sensitive data. There are numerous applications which can benefit from its success, and additional studies will further validate its use as an identity verifier.

10 Acknowledgments

The authors would like to express their gratitude to all those who participated in the experiments. We also thank Martin Garcia, Kouji Ouchi and Marek Teichman for their many helpful suggestions along the way, and Peter Wyckoff for his comments on drafts on this paper.

References

[Ale96] T. Alexandre. Biometrics on Smartcards: An approach to keyboard behavioral signature. *Second Smart Card Research & Advanced Applications Conference*, 1996.

[Ble88] S. Bleha. *Recognition Systems Based on Keystroke Dynamics.* PhD thesis, Univ. Missouri - Columbia, May 1988.

[BR93] M. Brown and S.J. Rogers. User Identification via Keystroke Characteristics of Typed Names using Neural Networks. *International Journal of Man-Machine Studies*, 39(6):999–1014, 1993.

[BSH90] S. Bleha, C. Slivinsky, and B. Hussein. Computer-Access Security Systems Using Keystroke Dynamics. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-12(12):1217–1222, December 1990.

[Dau93] J.G. Daugman. High Confidence Visual Recognition of Persons by a Test of Statistical Independence. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11), November 1993.

[4] a threshold deviation can be set per user

[Dix79] J. Dixon. Pattern Recognition with Partly Missing Data. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-9(10):617–621, October 1979.

[Dud73] R. Duda. *Pattern Classification and Scene Analysis.* John Wiley & Sons, New York, 1973.

[Gar86] J. Garcia. *Personal identification apparatus.* Patent No. 4 621 334, U.S. Patent and Trademark Office, 1986.

[GLPN80] R. Gaines, W. Lisowski, S. Press, and Shapiro N. Authentication by Keystroke Timing: some prelimary results. *Rand Report R-256-NSF. Rand Corporation*, 1980.

[HKLR81] L.D. Harmon, M.K. Khan, R. Lasch, and P.F. Ramig. Machine Identification of Human Faces. *Pattern Recognition*, 13(2):97–110, 1981.

[HMB89] B. Hussien, R. McLaren, and S. Bleha. An application of fuzzy algorithms in a computer access security system. *Pattern Recognition Letters*, 9:39–43, 1989.

[JG90] R. Joyce and G. Gupta. Identity Authorization Based on Keystroke Latencies. *Communications of the ACM*, 33(2):168–176, February 1990.

[LU91] G. Leggett, J. Williams and M. Usnick. Dynamic Identity Verification via Keystroke Characteristics. *International Journal of Man-Machine Studies*, 35(6):859–870, 1991.

[LU89] G. Leggett, J. Williams and D. Umphress. Verification of User Identity via Keystroke Characteristics. *Human Factors in Management Information Systems*, 89.

[LW88] J. Leggett and G. Williams. Verifying Identity via Keystroke Characteristics. *International Journal of Man-Machine Studies*, 28(1):67–76, 1988.

[Mil94] Benjamin Miller. Vital Signs of Identity. *IEEE Spectrum*, pages 22 – 30, 1994.

[PTVF92] W.H Press, S. Teukolsky, W. Vetterling, and B.P. Flannery. *Numerical Recipes in C: 2nd Edition.* Cambridge University Press, New York, 1992.

[SNS88] J.G Steiner, B.C Neuman, and J.I Schiller. Kerberos: An Authentication Service for Open Network Systems. *USENIX Conference Proceedings*, pages 191–201, Feb. 1988.

[TG81] J.T. Tou and R.C. Gonzalez. *Pattern Recognition Principles.* Addison-Wesley, 1981.

[UW85] D. Umphress and G. Williams. Identity Verification Through Keyboard Characteristics. *International Journal of Man-Machine Studies*, 23(3):263–273, 1985.

[YH89] J.R. Young and R.W. Hammon. *Method and apparatus for verifying an individual's identity.* Patent No. 4 805 222, U.S. Patent and Trademark Office, 1989.

[Zim95] P.R. Zimmerman. *The Official PGP User's Guide.* Boston:MIT Press, 1995.