Enhanced User Authentication Through Keystroke Biometrics

Edmond Lau, Xia Liu, Chen Xiao, and Xiao Yu

6.857: Computer and Network Security
Final Project Report
Massachusetts Institute of Technology
{edmond, xialiu, chenx05, xyu}@mit.edu

December 9, 2004

Abstract

In this paper, we look at several metrics for keystroke-enhanced user authentication. We begin by introducing keystroke biometrics and looking at its advantages and disadvantages. We examine a statistical analysis method described in the current literature and show through experimental data how the metric of consecutive keystroke latencies is inadequate for user authentication. The insufficiency gives the motivation for exploring other keystroke metrics. We design, test, and evaluate four different metrics relating to keystroke analysis.

1 Introduction

Keystroke biometrics refers to the art and science of recognizing an individual based on an analysis of his typing patterns. Biometric authentication and classification procedures have traditionally been implemented using physiological traits such as fingerprints, retinas, and face, or using behavioral traits such as voice. The concept of keystroke biometrics has arisen as a hot topic of research only in the past two decades.

1.1 Advantages of Keystroke Biometrics

Biometrics based on typing patterns are distinctive in that they are cheaper to implement, more distributed, and more unobtrusive than conventional biometric procedures. Collecting data regarding a person's typing patterns simply requires a keyboard and some basic software to collect data. Two authors were able to implement basic functionality for this purpose using Java's Swing package in less than 10 hours of combined work. Contrast this relatively cheap investment with fingerprint or retinal scan technology, which requires the manufacturing of expensive hardware for data collection. Moreover, data collection software is easily replicable whereas hardware is not.

Because the primary hardware requirement for keystroke biometrics is a keyboard, keystroke biometrics can be collected from virtually anywhere throughout the world via an Internet connection without requiring an individual to be at certain locations with access to specialized hardware. Keystroke collection software can be distributed via client-side Java applets as proposed by Ke et al. in [1]. Moreover, because each keystroke is captured entirely by the key pressed, the press time, and the release time, the data can also be transmitted using low bandwidth. According to a 2000 U.S. Census report [2], 51.0% of U.S. households have a computer and 41.5% have Internet access. These numbers have continued to skyrocket in the past four years. The growth of Internet connectivity thus makes distributed mechanisms for authentication increasingly feasible and attractive.

A final advantage of keystroke biometrics

is that it is a relatively unobtrusive measure. Fingerprint, retina, and face scans all inconvenience the user by requiring him to place a particular body feature either within or in front of some machinery. By contrast, typing on a keyboard is already a daily activity for many people; thus, keystroke biometrics can be easily integrated into a person's daily routine.

1.2 Disadvantages of Keystroke Biometrics

The advantages of keystroke biometrics come at the tradeoff of significant variability and an uncontrolled authentication environment. Unlike other physiological biometrics such as fingerprints, retinas, and facial features, all of which remain fairly consistent over long periods of time, typing patterns can be rather erratic. Even though any biometric can change over time, typing patterns have an intrinsically smaller time scale for changes. Not only are typing patterns rather inconsistent as compared to other biometrics, a person's hands can get tired or sweaty after prolonged periods of typing, often resulting in major pattern differences over the course of a day.

Another substantial problem is that typing patterns vary based on the type of the keyboard being used, the keyboard layout (i.e. qwerty or dvorak), whether the individual is sitting or standing, the person's posture if sitting, etc. The fact is that the distributed nature of keyboard biometrics also means that additional inconsistencies may be introduced into typing pattern data.

1.3 The Problem of Keystroke-Enhanced Login Systems

In our project, we consider the problem of keystroke-enhanced login systems. The username-password paradigm is common throughout virtually all web services and system login programs. Currently, a user's account is entirely compromised if an adversary somehow discovers the user's username and password. Under a keystroke-enhanced login system, the system would still be able to reject the adversary if his typing patterns for the two phrases differed from the pattern stored for that user. The system could then conceivably prompt the user with a personal question if the typing patterns differed. Thus, a keystroke-enhanced authentication scheme would provide an additional layer of security.

The keystroke-enhanced login problem differs from *classification* problems in that a system cannot merely say which known user the typist most likely resembles but must decide whether the typist is sufficiently close to a stored template of his claimed identity to minimize false positives and false negatives. The *authentication* problem therefore appears more difficult.

The keystroke-enhanced login problem differs from other keystroke authentication problems, such as continuous user authentication, in that the training data and the test sample available is relatively small. Problems such as continuous user authentication have available hours or days of training data per user. By contrast, a keystroke-enhanced login system must be capable of learning a user's typing pattern for a short username and password pair within 10-15 samples of roughly 20-40 characters each, over a course of 3-5 minutes. The small training set means that more patterns must be mined from the samples to generate adequate information to produce a template of the user's typing pattern.

The rest of this paper proceeds as follows. Section 2 describes an implementation of a statistical analysis model for keystroke-enhanced login and motivates a need for more refined metrics to distinguish between users. Section 3 then discusses four proposed metrics for distinguishing users and supports the metrics with data analyses from experimental data. We summarize in Section 4 related work that has been done in this area of research, and we conclude with a list of our research contributions in Section 5.

2 A Proposed Model for User Authentication

In [1], Ke describes the statistical analysis model as a viable solution to the keystroke-enhanced login problem. The model takes the

latencies between adjacent keystrokes among several samples of a user and then computes a vector of means and standard deviations for the latencies between each pair of keystrokes. The vector of means and standard deviations then represents the user's profile.

The model then sets two parameters:

1. How close each latency must be to the mean.

2. What percentage of the latencies must fall within the "similarity band."

The model sets the first parameter by setting the number of standard deviations that each element in the latency vector of a new input sample must fall within the mean. Consequently, a band of allowed latencies form around the mean of each pair of consecutive keystrokes. The model sets the second parameter, a similarity threshold, by setting the number of keystroke latencies in the input vector that must fall within the allowed band. To gauge the effectiveness of the statistical model, two commonly used standards in biometrics were employed:

- False Rejection Rate (FRR) - the percentage of times that a valid user is labelled as an adversary and denied access; also known as the false negatives rate.

- False Acceptance Rate (FAR) - the percentage of times that an adversary gains access as a user; also known as the false positive rate.

Seeking to minimize both the FRR and FAR, Ke et al. tweaked the two aforementioned parameters (number of standard deviations and percent similar) of the statistical model. They conducted experimental tests on the model, but only on a samples size of 4 users. When analyzing the data of 4 users, their model maximized performance when the band pass was 2.0 standard deviations and the percent similar was 75%. At these settings, the model achieved 3% FRR and 2% FAR.

Since Ke only used samples from 4 users, we wanted to verify the results of the statistical model to see if it scaled to more users. We implemented a similar statistical model, using latencies between every other keystroke, but extended the sample size to 15 users. We collected 10 samples from each user, each typing in "`A quick brown fox jumps over the lazy dog.`" To calculate the FFR, we took all possible combinations of 8 samples for each user and calculated the statistics on those samples. Then we used the remaining two samples for cross-validation to simulate a user attempting to access his own account. To calculate the FAR, we used all 10 samples of a particular user to build his statistical profile and used the remaining 140 samples of 14 users (10 each) to simulate an adversary trying to access the user's account. We varied the two parameters, standard deviation and the similarity threshold, to find the best parameters for minimizing FRR and FAR. Table 1 and 2 show our results.

	1.0	1.5	2.0	2.5
.50	7%	.074%	0%	0%
.55	17%	.37%	0%	0%
.60	31%	3%	.22%	.074%
.65	45%	6%	.96%	.67%
.70	61%	12%	3%	1%
.75	77%	26%	6%	2%
.80	88%	46%	16%	6%
.85	96%	67%	34%	15%
.90	99%	84%	58%	35%
.95	100%	95%	89%	75%
1.0	100%	100%	100%	100%

Table 1: FFR of various parameter values (similarity threshold on the left column vs number of standard deviations on the top row).

From our experimental results, we observe that the statistical model performs the best with the number of standard deviations set to 1.5 and the percent similar set to 70%. We achieve a FFR of 12% and a FAR of 10%. However, even at these values, the statistical model does not perform as well as described in [1]. While several factors could account for this discrepancy, we believe that the statistical model running on a larger sample size shows that the statistical model runs into scalability problems for the keystroke-enhanced

	1.0	1.5	2.0	2.5
.50	20%	61%	81%	87%
.55	9%	50%	74%	83%
.60	3%	33%	64%	77%
.65	1%	20%	51%	71%
.70	.52%	10%	37%	62%
.75	.095%	4%	22%	47%
.80	0%	1%	11%	29%
.85	0%	.43%	4%	14%
.90	0%	0%	.86%	5%
.95	0%	0%	.095%	.62%
1.0	0%	0%	0%	0%

Table 2: FAR of various parameter values (similarity threshold on the left column vs number of standard deviations on the top row).

login problem. Intuitively, as the the number of users increase, FAR would potentially increase because there are more potential adversaries. The results also lead us to believe that looking at latencies between keystrokes may be too general of a metric when faced with a large pool of users each with a small amount of training data. Poor performance of the statistical model motivates us to explore more specific metrics.

3 Metric Proposals

In this section, we propose and assess experimentally the soundness of four metrics: one based on key press duration, a second based on whether a key is released prior to pressing the next key, a third based on relative keystroke speeds, and a fourth based on `Shift` key usage patterns.

3.1 Key Press Duration

Most experiments described in previous research literature did not allow users to correct their errors by using the 'delete' or 'backspace' buttons and forbade users to pause while providing a typing sample. Statistical models used by Ke [1], and Wong [6] relied on *digraph latency* (the latency between a pair of consecutive keystrokes) to profile users, and accurate profiling of users depended heavily upon little errors in the data. We hypothesize that key press duration does not suffer from the same limitations as keystroke latency, and conduct the following experiment to test key press durations robustness to user correction on erroneous input.

3.1.1 Key Press Duration Parameters

For each user, we compute the mean and standard deviation of the key press duration for each key they typed from a predetermined sentence. We then compare typing samples between users, specifying that a threshold percentage of key presses for the unknown sample must fall within a some standard deviation of the mean for a known user. Samples that do not meet the percentage threshold do not successfully authenticate the user and are rejected.

3.1.2 Experiment

We collect typing samples from 15 different users. Each user typed the following 43-key-press passphrase either 10 or 11 times: "`A quick brown fox jumps over the lazy dog.`" During data extraction, we require that each user type the sentence exactly as shown, but that they may use backspace for any corrections. We filter each user's typing samples for common key presses, and then apply the statistical model to find the False Acceptance Rate (FAR) and False Rejection Rate (FRR) for key press duration profiling.

3.1.3 Results of Key Press Duration

In general, we can observe qualitatively that a user's key press durations are consistent across several samples, and that key press durations vary across different users. An example of this consistency can be see in Figure 1 and Figure 2, which depict the typing patterns of User 15 in 11 total samples of the sample sentence. For readability, the x-axis denotes the *x-th* key in the sequence of key presses to complete the sample sentence. Figure 3 shows the variations for mean key press

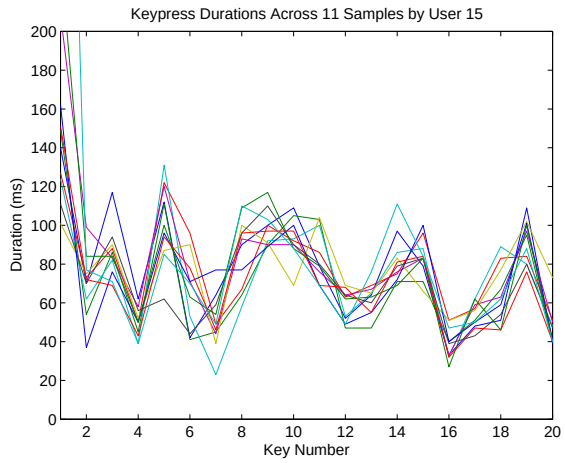

Figure 1: Key press durations for User 15 on the first half of the sample sentence: "`A quick brown fox` "

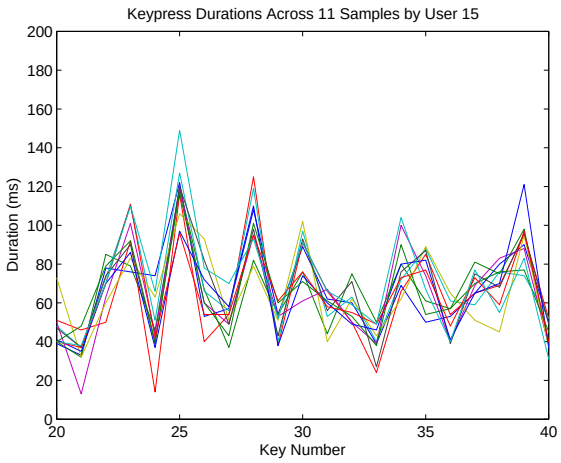

Figure 2: Key press durations for User 15 on the second half of the sample sentence: "`jumps over the lazy dog`"

duration between 5 different users for the first 10 characters of the sample sentence.

The results for False Acceptance Rate and False Rejection Rate using key press duration are shown in Tables 3 and 4, respectively. Our results are consistent with previous research and tests performed on the statistical analysis model by Ke [1] and D'Souza [3]: as the number of standard deviations increases the FAR increases and FRR decreases. Standard deviation measures the range of key-press durations that is acceptable for each particular key in the sample sequence, thus increasing this range also increases the likelihood that an unknown user's typing samples will overlap with the distribution of a legitimate user's key press duration distribution and causes the FAR to increase. Likewise, variations within the user's own typing samples are also more likely to fall within an increased range, thus FRR decreases.

Another trend to note is that as the threshold percentage increases, FAR decreases and FRR increases. Threshold percentage measures the proportion of the intruder's sample that must match a user's reference pattern in order to impersonate the user successfully. Thus, increasing the threshold percentage imposes more stringent requirements for successful authentication, which means that a user is also more likely to be rejected due to natural

	1.0	1.5	2.0	2.5
0.75	0%	0.61%	8.26%	25.0
0.80	0%	0%	3.45%	16.79%
0.85	0%	0%	1.01%	7.39%
0.90	0%	0%	0%	2.01%

Table 3: False Acceptance Rate for users typing "`A quick brown fox jumps over the lazy dog.`"

variations in his or her own typing.

From the our results, we conclude that a threshold of 0.8 and standard deviation of 2.0 is the best combination of parameters for aiding the authentication problem using key-press duration metrics. Even though thresholds of 0.75 and 0.8 (standard deviation=2.0) minimizes FRR and FAR at 1.25% and 1.01%, respectively, the corresponding values for FAR and FRR are unacceptable for these parameters.

3.2 Relative Key Event Order

The relative order in which users press and release keys can vary greatly from user to user, especially while typing words or phrases in which each user has a more established typing pattern. We call the press of a key and

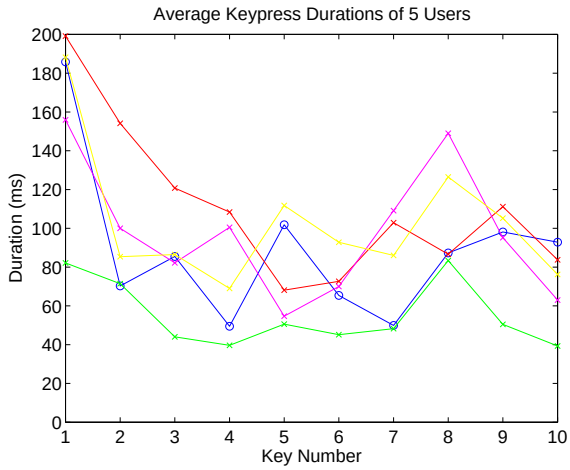

Figure 3: Graphical representation of variation in average key press durations for the first 10 keystrokes of the sample sentence of 5 users.

	1.0	1.5	2.0	2.5
0.75	96.35%	31.41%	1.25%	0%
0.80	99.43%	52.47%	8.95%	0%
0.85	100%	84.85%	24.16%	4.03%
0.90	100%	94.97%	64.01%	19.59%

Table 4: False Rejection Rate for users typing "A quick brown fox jumps over the lazy dog."

the release of the key each a *keyevent*.

We hypothesize that while the pattern in which users press and release keys differ from user to user, the key event ordering would remain consistent for one user. For example, when typing the word "the," one user may release may press t, release t, press h, release h, press e, release e, in that order, another user may press all three keys quickly in sequential order before releasing all three keys at the end.

3.2.1 Distance Metric And Authorization Scheme for Key Press and Release Orderings

A distance metric for two typing samples, or trials, was developed to compare their similarity. Given two samples a and b, the distance between sample a and sample b equals the number of key events that are swapped between the two samples. For instance, if one user types "h-i" by pressing h, releasing h, pressing i, and releasing i, while the second user presses h, presses i, releases h, releases i, the distance between the two samples is 1.

To find the distance between two trials, we first order the key events of each trial by time. Second, we find the sum of the absolute value of the difference in the positions for each key event in the two trials. Since every swap will cause two key events to be out of position, divide this sum by two to find the distance between the trials. Two trials with distance equal to one would feature very similar key orderings. A distance equal to the total number of letters typed would indicate a very different typing pattern.

Consider a trial t and a data set S of trials of one user to which we compare t. Let the distance between the elements of S to other elements of S have mean and standard deviation m and n. If the average distance from t to the elements of S is at most one standard deviation greater than m, then we say that t is a member of S.

3.2.2 Experiment

For this experiment, data samples were collected from 15 users. Each user typed the sentence "A quick brown fox jumps over the lazy dog" up to eleven times and the key press and release times were recorded. The delete operations and irrelevant shift key presses were eliminated from the data.

The analysis portion of this experiment is broken up into three parts. The first two were to find the false acceptance rate and false failure rates according to the data found. The last portion involves examining the average distance between all trials.

3.2.3 Results for Key Press and Release Comparisons

The distance between the trials of the same user has a mean and standard deviation of 6.72 and 3.48, while the distance between trials of different users has a mean and standard deviation of 12.88 and 8.51. Figures 4 and 5 show histograms of the distances between

same user data and different user data, respectively.

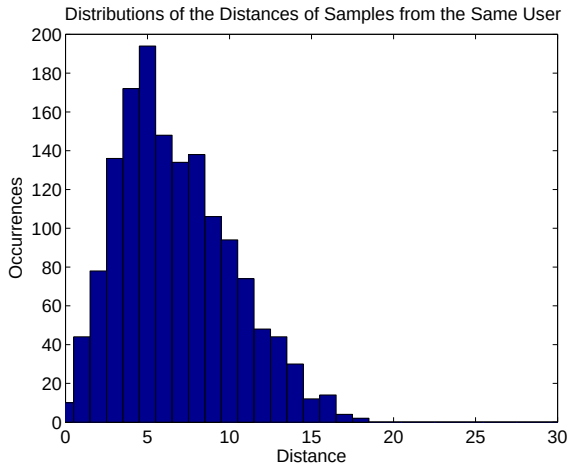

Figure 4: Distributions of Distances from Same User

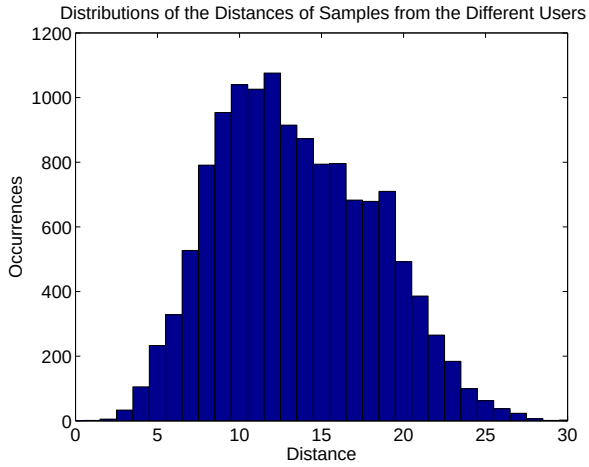

Figure 5: Distributions of Distances of Different Users

These results show that the high inconsistency between the data of different users prevent a general mean and standard deviation model from being used. Instead, the mean and standard deviation for each user was computed. The false failure rate of each user was found by using the identification scheme proposed earlier on each trial of that user. The group that each trial is compared to consists of all other data from that user. The max failure rate was found to be 1/10. That is, a user will be incorrectly rejected at most once out of ten tries.

The false acceptance rate of each user was found by first finding the mean and standard deviation of User A. Compare every trial of the other users to User A's data. For one standard deviation, while many users had a FAR of less than 5% (a few even had a FAR of 0.0), the false acceptance rate of some grew as high as 57%. The possible FAR for this metric is too high to be of use on its own, but the great rates for some users indicate that this metric could be a valuable addition to other schemes. While the FAR was very high, when user data was compared separately, it was observed that there was only four cases in which the mean distance to the same user's data was less than the mean distance to other user's data.

As seen from the FAR data, the limitation of key event orderings is for a given correctly typed phrase, the number of possible variations of key presses and releases is small. Thus, by itself, key event orderings cannot be trusted with the entire authorization problem. However, the low FAR's of several trials indicate that key event orderings could be a good support metric.

With more data, this scheme could be modified to account for specific swaps, not just the total number of swaps. An average distance could be recorded for each keyboard event. Another variation which requires longer input data would record the mean and standard deviation for every letter individually.

3.3 Relative Keystroke Speeds

Significant variability can occur between two consecutive typing samples even if the user has not undergone any observable psychological or physiological change. A user's typing speed likely fluctuates rather than remains constant, even for the same typing sample.

Many keystroke recognition models in the current literature, however, fail to adequately address this variation. Models such as those used by D'Souza [3], Ke [1], and Joyce and Gupta [4] all develop a profile using a statistical analysis method involving means and standard deviations of latencies between consecutive keystrokes. Under these types of statistical models, if a user were to type each key

much faster than usual, then he would most likely be rejected because the timing measurement of each of his pairs of consecutive keystrokes would fall beyond the stored mean of his trained profile.

We hypothesize that even if a user's absolute typing speed of a particular phrase varied between typing samples, the speed with which he typed particular keys relative to other keys in any given sample may actually remain consistent. For example, a user who types the word "fruit" at different speeds each time may still consistently type the letter pair r-t faster than f-r, the pair f-r faster than u-i, the pair u-i faster than i-t, and the pair i-t faster than r-u.

If this hypothesis is true, then it may be feasible to adopt a metric that analyzes the relative speed of keystroke pairs rather than the absolute speed with which the user typed each letter pair. We call such a metric a metric for *relative keystroke speeds*. A good metric for this purpose would be one that identified consistency across the relative typing speeds of particular keystrokes for a given user, but that also distinguished one user's relative speeds of particular keystrokes from another user.

3.3.1 Distance Metric for Relative Keystroke Speeds

To explore our hypothesis, we adopted a distance metric proposed by Bergadano [5] to quantify the similarity or difference between two typing samples, based purely on the relative latencies between every other keystroke, though we modify the metric to apply to latencies between consecutive keystrokes instead because consecutive pairs of keys are more intuitive.

Bergadano implemented a system that achieved an FRR of about 4% and an FAR of less than 0.01%, but with a text length of 683 characters. For the keystroke-enhanced login problem, such a large body of text is infeasible for the length of a username and password, and we wanted to test whether the distance metric also worked well for short phrases.

To compute the distance between two typing samples S and S', we represent each sample as a vector of key pairs, filter out all key pairs that are not shared between the samples to handle backspaces and deleted characters, and then sort the remaining pairs in each sample based on their latencies (the press time of the second key minus the release time of the first key). Letting $S[i]$ denote the location of key pair i in the sorted sample S, we compute the distance between S and S' as:

$$\text{distance} = \sum_i |S[i] - S'[i]|$$

We also normalize the distance to a number between 0 and 1 by dividing the greatest possible distance of the two samples of length $|S|$.

A distance of 0 between two samples means that the two samples have very consistent relative keystroke speeds, while a distance of 1 means that the two samples have very inconsistent relative keystroke speeds.

3.3.2 Experiment

For our experiment, we collected typing samples from 15 different users. Each user typed the following 42-character passphrase either 10 or 11 times, for a total of 153 typing samples: "`A quick brown fox jumps over the lazy dog.`" The length of this passphrase is approximately the length of a username-password pair. We then compared the distance between each pair of samples from the same user and each pair of samples from different users.

3.3.3 Consistency Results of Relative Keystroke Speeds

The distance between any two samples of the same user has a mean of 0.3192 and a standard deviation of 0.0732; the distance between any two samples of different users has a mean of 0.5290 and a standard deviation of 0.0828. These results are illustrated graphically in Figure 6.

Figures 7 and 8 show histograms of the distances between any two samples of the same user and different users, respectively. We observe that the distance between two samples of the same user ranges primarily from 0.2 to 0.45, whereas the distance between two samples of the different users ranges from 0.35 to 0.7.

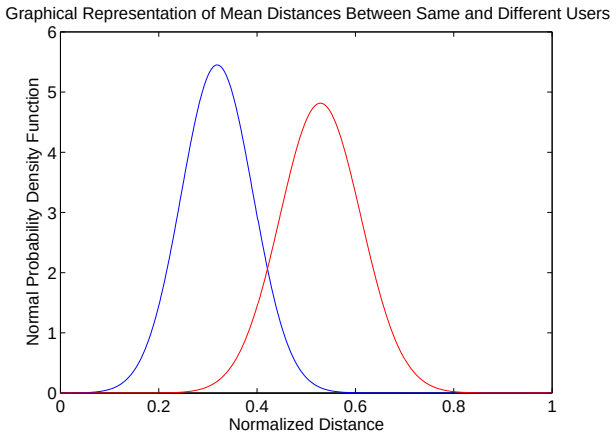

Figure 6: Graphical representation of distribution of mean distances between typing samples of the same user and different users.

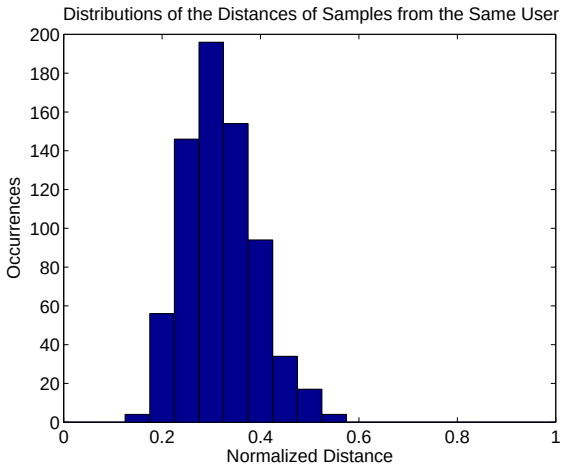

Figure 7: Histogram of distances of samples from same user.

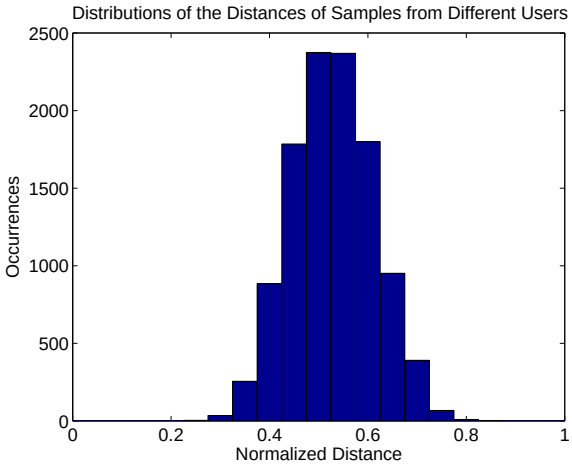

Figure 8: Histogram of distances of samples from different users.

From these experimental results, we can make two important conclusions regarding the use of relative keystroke speeds as a metric. First, we find that a user's relative typing speed of particular keys across different typing samples does in fact exhibit some degree of consistency and that different people indeed have different relative typing speeds across particular keys; otherwise, the two histograms would have had similar distributions. Bergadano was able to achieve a higher degree of separation between the two distributions, and we attribute the larger overlap in the distance distributions in our experiments to the shorter passphrase text length that we use.

Second, for a given typing sample S, if we approximate the sorted key pairs of another user S' as being randomly sorted with respect to S, then we have confirmed Bergadano's observation that the distance distribution of randomly sorted vectors is non-uniform; instead, it has a form similar to the one shown in Figure 8, therefore allowing us to use relative keystroke speeds as a metric to distinguish users.

We conclude that relative keystroke speeds can be an effective metric for a keystroke-enhanced authentication system for distinguishing users when the distances between typing samples are either low (less than 0.35) or high (greater than 0.45). A distance value that falls within the overlapping metric would require additional metrics to aid in authentication.

3.4 Classes of Shift Key Usage

For each key used in conjunction with the `Shift` key, the user can either use the right or left `Shift` key. For users who habitually type on a keyboard, consistent trends in the same `Shift` key used for a particular letter, number, punctuation, or other symbol may be used to distinguish users.

3.4.1 Hypotheses for `Shift` Key Usage Patterns

We hypothesized prior to any investigation that users would consistently use the same `Shift` key (left or right) for each different key, and we expected this trend to carry through all of a user's typing samples. For example, it seemed reasonable to expect that a user may regularly use the left `Shift` key to type "A", while typing "P" with the right `Shift` key. In addition, we postulated the existence of four different categories of `Shift` key users: strictly left-shift users, strictly right-shift users, opposite-shift users, and erratic users. Strictly left-shift users and strictly right-shift users consistently use the respective `Shift` key regardless of the key combination. Opposite-shift users press the `Shift` key opposite the combination key to maximize typing efficieny, and erratic users exhibit no pattern in `Shift` key usage.

Surveying the current literature, we did not find any models that use `Shift` key patterns as a metric. If our hypothesis proved to be correct, we therefore hoped that an analysis of users' `Shift` key habits and separate the users into different categories. Since there a finite number of categories, the classfication as the only metric in user authentication is not viable. However, the classification as a metric can be valuable as an added layer in user authentication. A good metric for this purpose hinge upon the fact that users use the same Shift keys across samples and these Shift key patterns form distinct categories.

3.4.2 Experiment

To test our hypothesis, we looked at the use of `Shift` keys strictly for the capitalization of letters. We decided to focus on the capitalized letters because letters occur more commonly in typing and users are more likely to have developed Shift key preferences for letters. In our experiment, we recorded typing samples from 15 different users. Each user typed in the following sentence 5 times: "`Another Quick Brown Fox Jumps Over The Lazy Dog Yet Round Cats Eat Plain Goldfish Heartily In Maine Not Kansas Under Some Vain Zealous Xena Warrior.`" The sentence contains capitalized versions of each of the 26 letters in the alphabet. We studied the `Shift` key used for each capitalized letter and then compared the `Shift` key sequence between samples of the same user. From this analysis, we created a taxonomy of users based on the their `Shift` key usage patterns.

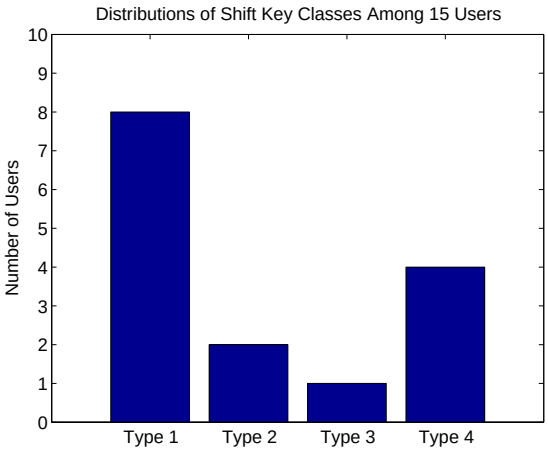

Figure 9: Histogram of user taxonomy classified by `Shift` key patterns.

3.4.3 Actual Taxonomy of `Shift` Key Usage Patterns

Figure 9 shows a histogram of the distribution of the `Shift` key classes among the 15 users. In the experiment, we observed four types of users: Type 1 users strictly pressed the left `Shift` key; Type 2 users strictly pressed the right `Shift` keys; Type 3 users utilized both right and left `Shift` keys but were consistent with respect to a particular letter; Type 4 users, however, were consistent on a majority of letters, but on a small subset of the letters, pressed both left and right `Shift` keys across the 5 samples.

From the experiments, we observe that although 4 classes of `Shift` key users do exist, the experimental categories differ from those hypothesized. While strictly left and strictly right `Shift` users exist, Type 3 users do not correspond exactly to the hypothesized opposite-shift category. While Type 3 users do use both right and left `Shift` keys, the

choice was not determined solely on the location of the target key. In addition, Type 4 users were not truly erratic and were inconsistent only on some subset of the letters.

3.4.4 `Shift` Key Patterns as a Viable Metric

Based on the existence of these four categories, we conclude that the `Shift` key metric is a viable metric for user authentication. By classifying a user as a particular `Shift` key user, an authentication scheme can reject users who do not type their password using the correct `Shift` key pattern. For example, if a user presses only left `Shift` keys for eight different capitalized letters but right `Shift` keys for all the remaining letters, an adversary who obtains the user's username and password must also know to use the correct `Shift` key pattern. Moreover, in the case of Type 1, Type 2, and Type 3 users, the `Shift` key metric is actually quite reliable; thus, the `Shift` key metric provides an additional layer of user authentication.

However, we should note two weakness in the `Shift` key metric. First, in most usernames and passwords, capital letters do not represent a large proportion of the characters. Based on the capital letters present, the authentication scheme may not be able to extract sufficient information to classify the person at login. For example, consider a Type 3 user who uses left `Shift` for the first half of the alphabet and right `Shift` for the second half of the alphabet. The Type 3 user has the password "Apple." An adversary who is Type 1 and has the user's password will be able to spoof as the user because both the user and the adversary use the left `Shift` key to type "A." Second, once a user's `Shift` key category is known, except in the case of the category being Type 3, the adversary can easily spoof as the user. In the case of Type 3 users, knowing the category is not sufficient, the adversary must also use the correct `Shift` key associated with each letter. Therefore, for dictionary attacks where the adversary does not have any additional knowledge about the user, the `Shift` key metric still does add a layer of protection.

The `Shift` key metric opens up several areas of research and exploration. Future experiments should take more samples and test on more users in order to look at further refining the categories, specifically in the classification of Type 3 and Type 4 users. We may find that users consistently type part of the alphabet with the left `Shift` key and the other part with the right `Shift` key. We may also find that users are always inconsistent on particular letters. Additionally, we can look at the particular letters that are inconsistently being capitalized and use these characteristics to label the user. Furthermore, we can extend testing to explore the use of the `Shift` key with symbols and punctuation marks. Differences in such usages may prove useful in categorizing users.

4 Related Work

In addition to the analysis techniques cited in previous sections of this paper, several earlier works on keystroke biometrics have already adopted approaches based on different metrics, sampling methodologies, and data analysis techniques. One of the first studies on keystroke authentication was keystroke latency analysis by Gaines et al. [11] in 1980, in which seven users each provided two samples of a three-paragraph text, 4 months apart. In 1985, Umphress and Williams pioneered the digraph latency approach for keystroke analysis, and this approach was again improved upon by Leggett and Williams [10] in 1988. A few years later, Joyce and Gupta adopted an approach for rejecting and accepting samples based on threshold parameters and applied cross-validation within a user's own sample set to test more accurately for False Rejection Rates.

In the mid-nineties, researchers gradually augmented pure statistical analysis of keystroke latencies with newer classification techniques and algorithms. Marcus Brown and Samuel Rogers [9] use keystroke latency and duration to authenticate users via statistical analysis on Euclidean distance/vectors and two different kinds of neural networks. Unlike previous work, Brown and Rogers per-

formed their analysis on much shorter text samples and were able to produce fairly low FARs and FRRs.

Recently, several papers propose techniques for username/password access control systems using keystroke biometrics. Tapiador and Sigenza [8] propose a system that utilizes keystroke biometrics with standard web technologies such as CGI and Java applets to obtain user data in nonintrusive ways. Stanford Research Institute (SRI)s BioPassword provides the first commercial client-server implementation of keystroke biometrics that enhances log-on security on Windows NT and Windows 2000 platforms.

5 Contributions

In our project, we have:

- Verified that a naive, statistical analysis model cannot, by itself, handle well the limited training and testing data available for a keystroke-enhanced login program.
- Proposed four new keystroke metrics for identifying consistency within a user's typing patterns and for distinguishing one user from another.
- Experimented with and analyzed typing samples to validate and assess the goodness of each proposed metric.

References

[1] X. Ke, R. Manuel, M. Wilkerson, and L. Jin. "Keystroke Dyamics: A Web-based Biometric Solution." *13th USENIX Security Symposium.*

[2] E. C. Newburger. "Home Computers and Internet Use in the United States: August 2000." US Census Bureau, Sept. 2001. `http://www.census.gov/prod/2001/pubs/p23-207.pdf`

[3] D. C. D'Souza. "Typing Dynamics Biometric Authentication." Oct. 2002. `http://innovexpo.itee.uq.edu.au/2002/projects/s373901/thesis.PDF`

[4] R. Joyce and G. Gupta. "Identity Authentication Based on Keystroke Latencies." *Communications of the ACM*, Vol. 33, No. 2, p168-176, 1990.

[5] F. Bergadano, D. Gunetti, and C. Picardi. "User Authentication through Keystroke Dynamics." *ACM Transactions on Information and System Security*, Vol. 5, No. 4, Nov., p. 367-397, 2002.

[6] F. W. M. H. Wong, A. S. M. Supian, and A. F. Ismail. "Enhanced User Authentication through Typing Biometrics with Artificial Neural Networks and K-Nearest Neighbor Algorithm." *IEEE*, p. 911-915, 2001.

[7] Net Nanny Software International, Inc. "Technical Report BioPassword Keystroke Dynamics." Net Nanny Software International, Inc. 2000-2001. `http://www.biopassword.com/home/technology/BP\%204.5\%20Technical\%20Paper.pdf`

[8] Marino Tapiador and Juan A. Sigenza. Fuzzy Keystroke Biometrics On Web Security. Universidad Autnoma de Madrid, 2000.

[9] M. Brown and S. J. Rogers. User identification via keystroke characteristics of typed names using neural networks *International Journal of Man-Machine Studies*, 1993.

[10] G. Leggett and J. Williams, M. Usnick. "Dynamic Identity Verification via Keystroke Characteristics." *International Journal of Man-Machine Studies*, p.859-870, 1991.

[11] R. Gaines, W. Lisowski, S. Press, and N. Shapiro. "Authentication by Keystroke Timing: some preliminary results." *Rand Report*, 1980.