A Novel Approach to User Authentication Through Machine Learning of Keyboard Acoustic Emanations

Stephen Gould
sgould@stanford.edu

***Abstract*— Recent work by Asonov and Agrawal [1] and Zhuang et al. [2] has shown that acoustic emanations from keyboards can be used to reconstruct the text that is being typed. This, in theory, allows for the retrieval of confidential information such as passwords by covertly recording sound from the keyboard, and thus poses a significant security threat.**

Their techniques work because different keys produce different acoustic signatures when struck by the typist. Classifiers can be built to detect these differences and language models help in establishing priors for the various key combinations.

This paper looks at applying machine learning techniques to the identification and authentication of users based solely on the acoustic waveform generated as the user types on a keyboard. This is possible because, not only do different keys produce different sounds, but the same key sequence can produce different acoustic waveforms when typed by a different user. Classification accuracies of up to 98% are reported on a small set of trial users.

In one novel application, this kind of system can be used to biometrically harden users' passwords by incorporating each user's typing behavior as part of the authentication procedure (see Monrose et al. [3]).

***Index Terms*— Machine Learning, User Authentication, Keyboard Acoustic Emanations, Computer Security**

I. Introduction

THIS report discusses the outcome of research conducted into the identification and authentication of computer system users from keyboard acoustic emanations.

Most users are comfortable with the familiar clicking sounds emanating from their keyboard as they type. However, these sounds, known as keyboard acoustic emanations, have been shown to be a significant security risk in allowing the reconstruction of text from covert audio recordings ([1] and [2]).

A side channel attack in computer and communications security is the ability for an adversary to obtain secret information through indirect observations of the system. For a long time, researchers have known that private information can be retrieved by monitoring electromagnetic radiation emitted from different types of electronic equipment. For example, data transmitted over older modems can be read by monitoring the activity LED on the modem which is highly correlated with the data being sent. Side channel attacks result from weaknesses in a system's physical implementation rather than underlying algorithms or protocols.

Keyboard acoustic emanations are therefore a type of side channel attack. Asonov and Agrawal [1] introduce the idea of training a classifier to recognize keys based on their unique acoustic properties. More recently, Zhuang et al. [2] extended this work by showing that it is possible to reconstruct text that the user is typing after recording only 10 minutes of data. Their results improve accuracy by overlaying a language model which provides priors for keystroke bigrams.

The work outlined in this report shows that it is also possible to identify the users themselves. This can then be used as a novel way to improve computer security, for example, by hardening passwords with biometric information. Monrose et al. [3] have shown that password hardening is possible by direct measurements of user's typing characteristics (through modification of the computer system's device drivers). The approach described below does not rely on direct measurements of keyboard timing, but rather extracts this information as well as other features from the recorded sound. It allows for the incorporation of features such as how hard the user strikes each key, which has been identified by Bergadano et al. [4] as valuable in distinguishing between users. This can be done without the need for specialized computer hardware - the computer's microphone is all that is required. Finally, the work here has other novel applications outside of password hardening which are discussed at the end of the paper.

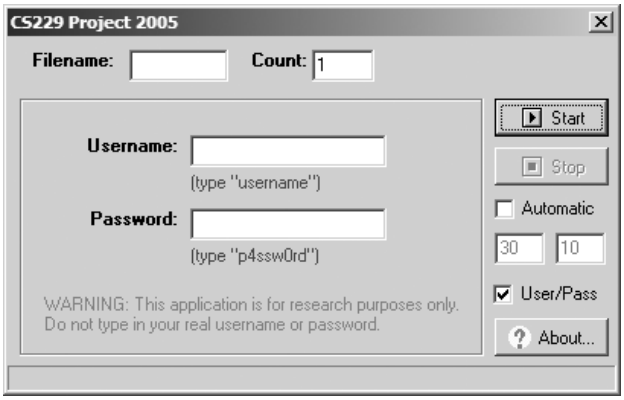

Fig. 1. Screen capture of Windows GUI application developed for capturing acoustic data samples.

II. Data Collection

Since standard datasets of acoustic keystroke data are not readily available, the first stage of this project was to develop a tool for recording and labeling the sound of users typing on keyboards. A Windows GUI application was written to

record and save sample waveforms of different user's typing patterns along with some associated meta-information such as keystroke timing.

The GUI samples the audio signal at a rate of 22.05kHz with 16-bit quantization. In order to facilitate easy processing in Matlab, each sample waveform (training instance) is saved in a separate .WAV file with meta-information in a corresponding .TXT file.

A total of 150 training examples were recorded from six different users under similar environmental conditions (i.e. ambient noise). Each user was asked to type "username" and "p4ssw0rd" as if logging on to a computer system. All the data samples (including those with misspellings) were included in the training set[1]. Basic statistics of the training set are shown in the table below.

TABLE I
TRAINING DATA STATISTICS

	Number of Samples	Avg. Length (sec.)	Avg. Keystrokes
Chris	20	4.35	18.5
Kendra	20	4.95	18.0
Konstantin	20	6.45	17.5
Naomi	30	4.80	18.1
Ross	20	4.55	18.2
Stephen	40	5.18	17.5

All data was collected on a Dell Latitude D400 laptop using the built-in microphone.

III. DATA ANALYSIS

A. Overview

The acoustic signal produced by users as they type on computer keyboards contains a suprisingly large amount of information. Fig. 2 shows a sample audio waveform of four consecutive keystrokes. Clearly present in the waveform are the (i) *touch*, (ii) *hit*, and (iii) *release* peaks corresponding to (i) when the user's finger first touches the key, (ii) when the key reaches the bottom of its stroke, and (iii) when the key rebounds after being released, respectively. This is consistent with the findings of Zhuang et al. [2]. The *touch* and *hit* peaks are sometimes referred to collectively as the *push* peak.

There is typically 100-150ms between a key being pushed and that same key being released, and 200-300ms between consecutive keystrokes corresponding to an average typing speed of 200-300 characters per minute. However, there can be significant shortening in the duration between keystrokes especially when a user is typing familiar text (such as his username and password). In these cases, the *hit* peak from a subsequent keystroke may coincide with the *release* peak from the previous keystroke. Although trivialized by Zhuang et al. [2], this overlapping makes it extermely difficult to reliably detect every keystroke as discussed in the next section.

It is quite obvious that each key click contains rich spectral information that distinguishes it from background noise (see Fig. 3). The critical insight of Asonov and Agrawal [1] is that due to differing mechanical properties, each key on a keyboard will produce a different sound. They use that difference to reconstruct text. Our proposition is that different typing patterns (corresponding, for example, to different users) will produce different sounds for the same key or phrase, and can therefore be used to identify users.

[1]In a real implementation of an authentication system, the login software would, as usual, reject attempts with invalid usernames or passwords before invoking any biometric checking

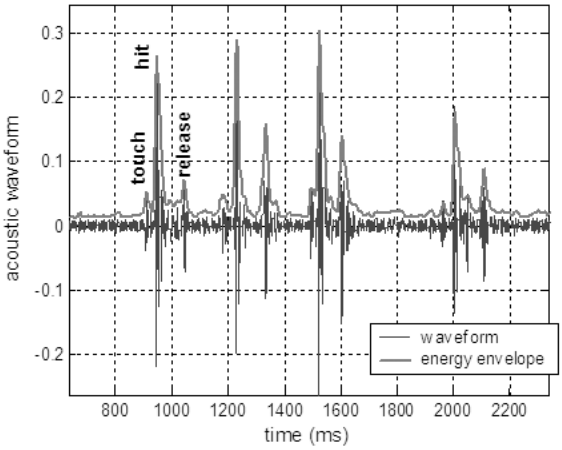

Fig. 2. Captured audio signal for four keystrokes showing the energy envelope used to detect each key press. Also clearly visible are the *touch*, *hit* and *release* peaks of each keystroke.

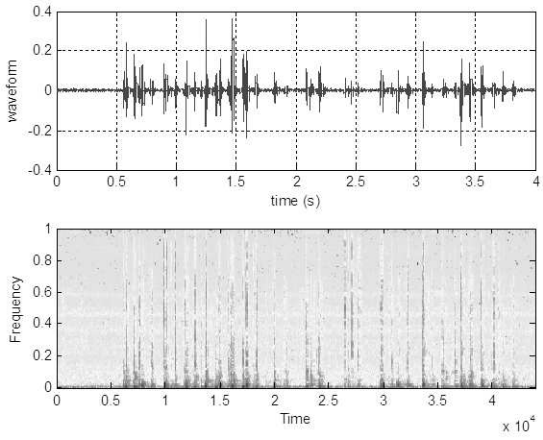

Fig. 3. Keyboard audio signal and corresponding frequency spectrogram showing four seconds of continuous typing.

B. Keystroke Detection

Although seemingly simple, keystroke detection required a significant amount of work to get right. Our first attempt was to replicate the procedure outlined by Zhuang et al. [2] and detect keystrokes by applying a threshold to the energy envelope of the signal (computed by summing Fourier coefficients from 400Hz to 12kHz). Zhuang et al. [2] admit that this sometimes makes mistakes which slightly affects the quality of their results. Our performance was, however, considerably lower than theirs, which is most likely due to the use of the quieter laptop keyboard over standard PC keyboards.

Early experimentation found the method as described to be inadequate for obtaining accurate keystroke detection. The procedure was improved by employing some general techniques described in the speech processing literature [5]. The most successful techniques were:

- High-pass filtering the signal.
- Clipping the audio signal to within two standard deviations of the mean to prevent excessivley high sample values.
- Raising the signal to a large odd power to accentuate peaks.
- Post filtering the detected keystrokes to disqualify any keystroke within 50ms of a previous keystroke.

While these steps gave better results than using the raw signal, the results obtained still contained about a 5% detection error[2], and is something that would require investigation in any further work.

IV. Feature Extraction

After detecting the keystrokes, a window of 20-50ms surrounding each keystroke was used to extract features from the waveform for use by the classifiers. Among the many features tried, four main classes of feature were found to be effective.

1) Inter-key Timing: Inter-key timing, or latency, measures the time duration between consecutive keystrokes. Monrose et al. [3], Bergadano et al. [4] and Lau et al. [6] have all identified this as one of the main features in discriminating between different users' typing behavior. However, due to inaccuracies in keystroke detection from the audio signal[3] and the short input sequences being used for classification, this feature was not as critical as one would think, as will be shown in the results below.

The inter-key time was associated with the second key in each keystroke pair, so another issue to consider was that of boundary conditions, i.e. how to deal with the first key which has no preceding keystroke. Three policies were investigated, all producing similar results:

- Drop the first sample altogether, leaving $m-1$ feature vectors for classifying the user from m keystrokes.
- Assume some nominal constant value for the first keystroke, for example 0ms.
- Set the inter-key time for the first keystroke to be the average of the inter-key intervals for the entire sample.

In the results reported below, the first option was used when inter-key timing was the only feature used, while the second option was used when inter-key timing was used as part of a larger feature vector.

2) Keystroke Energy: Bergadano et al. [4] make the observation that keystroke energy provides good biometric information since users strike keys with different pressure, but that special-purpose keyboards would be required to make such measurements. The audio signal produced for a key press is strongly correlated to the energy imparted by the user, and therefore makes a good surrogate for keystroke energy. This feature is derived by summing the squared signal values in a small region surrounding the *hit* peak.

3) Fourier Coefficients: If the total keystroke energy is a good feature for distinguishing between users, then the energy in different frequency bands is also worth exploring. The Fourier coefficients feature computes the energy component in equally spaced filter banks over the 11kHz sample space. The features are then normalize by the total energy to compensate for background noise effects.

4) Mel-Frequency Cepstral Coefficients (MFCC): Research in digital signal processing of speech signals has long identified MFCCs as critical for automated speech recognition (see Deller et al. [5]), and are now used extensively in speech and audio processing system. The Mel-frequency scale provides a mapping between real frequency (measured in Hz) and perceived frequency, and was originally used to describe the human auditory perception. There are a number of "standard" scales cited in the literature. The scale used in this work is defined as,

$$f_{mel} = 1127 \ln\left(1 + \frac{f_{Hz}}{700}\right)$$

The MFCCs are then computed as,

$$mfcc = \mathfrak{F}^{-1}\left\{\sum_{j=0}^{N-1} H_{i,j} \log \|S_j\|\right\}_{i=1}^{n}$$

where S is the Fourier transform of the waveform, and H_i is the i-th Mel-frequency filter bank.

V. Probabilistic Models

A. Naive Bayes Model

Let each training example be defined by a matrix consisting of a sequence of $m^{(i)}$ feature vectors, $X^{(i)} = \begin{bmatrix} \tilde{x}_1^{(i)} & \tilde{x}_2^{(i)} & \dots & \tilde{x}_{m^{(i)}}^{(i)} \end{bmatrix}$, each feature vector representing a single keystorke. The probability of a particular user, k, given the data is then,

$$p\left(y^{(i)} = k | X^{(i)}\right) = \frac{1}{Z} p\left(X^{(i)} | y^{(i)} = k\right)$$

where we have assumed equal priors on the $y^{(i)}$, and Z is the partition function. Now, under the naive Bayes assumption that each keystroke is independent, we have,

$$p\left(y^{(i)} = k | X^{(i)}\right) = \frac{1}{Z} \prod_{j=1}^{m^{(i)}} p\left(\tilde{x}_j^{(i)} | y^{(i)} = k\right)$$

The probability of each feature vector given the user was modeled as a multivariate Gaussian,

$$p\left(\tilde{x}_j^{(i)} | y^{(i)} = k\right) = \frac{1}{(2\pi)^{\frac{n}{2}} |\Sigma_k|^{\frac{1}{2}}} e^{-\frac{1}{2}\left(\tilde{x}_j^{(i)} - \mu_k\right)^T \Sigma_k^{-1} \left(\tilde{x}_j^{(i)} - \mu_k\right)}$$

[2]By either missing actual keystrokes or falsely detecting non-existent keystrokes.

[3]In fact, even the system proposed by Lau which measures key hit times directly suffers from inaccuracies caused by poor resolution of the software timers provided by the operating system and latency intorduced by intermediate software layers. The best approach would be to replace the keyboard devcie driver with a custom module to record key times precisely.

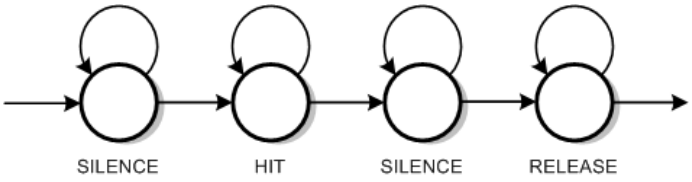

Fig. 4. Hidden Markov Model for single keystroke acoustics incorporating both *hit* and *release* peaks.

B. Hidden Markov Model

As discussed above, the keystroke detection algorithm will sometimes fail to detect actual keystrokes and ocassionally produces erroneous keystrokes. In an attempt to remove the keystroke detection algorithm from the classification procedure, a left-to-right hidden Markov model (HMM) with states for silence, *hit* peak, silence, and *release* peak was designed. The HMM can then automatically detect keystrokes given a sequence of feature vectors from the entire acoustic waveform. The HMM has the added benefit of modeling temporal information (such as inter-key timing).

Each state of the HMM contained a single multivariate Gaussian to model the observation probability distribution given that state. The *hit* and *release* states were initialized with parameter estimates derived from feature vectors at the estimated position of the keystrokes, while the silence states were initialized with feature vectors extracted from the midpoint between two keystrokes. One HMM per user was then trained using the entire sequence of feature vectors (each generated from windows 5ms apart) using the Baum-Welch re-estimation algorithm.

VI. Results

A. Evaluating Classification Performance

Classification performance was evaluated using leave-one-out cross-validation (LOOCV) on the entire training set of 150 samples. A separate model was trained for each user, and the test sample applied to each model. The predicted user label was taken to be that of the model with highest posterior, in the usual way,

$$\arg\max_k p\left(\hat{y}^{(test)} = k | X^{(test)}\right)$$

The confusion matrices for the naive Bayes and hidden Markov model classifiers using feature vector consisting of 10 MFCCs are shown below. The naive Bayes classifier achieves 96% accuracy while the HMM attains 98%. Results for other features combinations are summarized in table IV.

B. Evaluating Authentication Performance

The user authentication protocol evaluated works as follows. A username/passowrd combination is authenticated as being typed by the real user by comparing a user-specific statistical model of the keystroke biometrics with a population model (where the models are either naive Bayes or HMMs as described earlier). Although in this work, a separate population model was trained for each real user, a single population model could be envisioned for all users in a large system.

TABLE II
Naive Bayes Gaussian Discriminant Classifier (Using 50ms Window and 10 MFCCs) LOOCV Confusion Matrix

	C	Ke	Ko	N	R	S
C	20	-	-	-	-	-
Ke	-	20	-	-	-	-
Ko	-	-	20	-	-	-
N	1	-	1	28	-	-
R	-	1	1	-	18	-
S	-	-	-	1	-	39

TABLE III
Hidden Markov Model LOOCV Confusion Matrix

	C	Ke	Ko	N	R	S
C	20	-	-	-	-	-
Ke	-	20	-	-	-	-
Ko	-	-	20	-	-	-
N	-	-	-	30	-	-
R	-	-	3	-	17	-
S	-	-	-	-	-	40

Authentication performance was evaluated by assigning each of the six users in the training set the role of imposter, $y_{imposter}$, in turn. Then, for each user remaining, $y_{real} \neq y_{imposter}$, train a user model on $\left\{X^{(i)}; y^{(i)} = y_{real}\right\}$ and a population model on $\left\{X^{(i)}; y^{(i)} \neq y_{real}, y_{imposter}\right\}$. On each iteration we keep track of the number of true/false positives/negatives,

$$n_{tp} = \sum_{i:y^{(i)}=y_{real}} 1\left\{\frac{p\left(X^{(i)};\theta_u\right)}{p\left(X^{(i)};\theta_p\right)} > t\right\}$$
$$n_{fp} = \sum_{i:y^{(i)}=y_{imposter}} 1\left\{\frac{p\left(X^{(i)};\theta_u\right)}{p\left(X^{(i)};\theta_p\right)} > t\right\}$$
$$n_{tn} = \sum_{i:y^{(i)}=y_{imposter}} 1\left\{\frac{p\left(X^{(i)};\theta_u\right)}{p\left(X^{(i)};\theta_p\right)} \leq t\right\}$$
$$n_{fn} = \sum_{i:y^{(i)}=y_{real}} 1\left\{\frac{p\left(X^{(i)};\theta_u\right)}{p\left(X^{(i)};\theta_p\right)} \leq t\right\}$$

where t, the acceptance threshold, can be adjusted to trade-off between FAR and FRR, and θ are the appropriate model parameters.

The false acceptance rate (FAR) and false rejection rate (FRR) are then computed as,

$$FAR = \frac{n_{fp}}{n_{fp}+n_{tn}} \quad \text{and} \quad FRR = \frac{n_{fn}}{n_{fn}+n_{tp}}$$

In most authentication schemes, it is desirable to keep the FRR as low as possible to prevent denying access to legitimate users, while maintaining a sufficiently small FRR to be of practical benefit. Results for FAR and FRR for various feature combinations are shown in table IV.

C. Summary of Results

The following table summarizes the results of a number of experiments varying the window size and feature selection. Due to the prohibitively expensive cross-validation process involved in evaluating the HMM performance, most of the

experiments were conducted using the naive Bayes Gaussian discriminant classifier, and only the most promising feature combinations used with the HMM classifier.

TABLE IV
SUMMARY OF RESULTS

Model	Classification Rate (%)	Authentication (% FAR / FRR)
Naive Bayes (inter-key timing)	32.7	13.1 / 1.6
Naive Bayes (key energy, inter-key timing)	41.1	13.1 / 1.6
Naive Bayes (20ms window normalized Fourier)	69.3	25.1 / 8.0
Naive Bayes (20ms window MFCCs)	90.7	16.0 / 2.8
Naive Bayes (50ms window MFCCs)	96.7	16.3 / 1.3
Naive Bayes (20ms window MFCCs, inter-key timing)	94.0	12.9 / 1.6
HMM (50ms window MFCCs, 5ms increments)	98.0	21.1 / 7.9

It should be noted that since there were six users in the training set, a random algorithm would achieve an accuracy of 16.7% in the identification task, and FAR and FRR of 50% in the authentication task. The performance numbers of all classifiers discussed in this report are significantly better than random.

VII. Discussion and Further Work

The results above clearly show that biometric information can be extracted from keyboard acoustic emanations and used to identify users with very high accuracy. The Mel-frequency cepstral coefficients provide robust features for classification with both naive Bayes and HMM classifiers, the naive Bayes classifiers being much easier to implement and quicker to train. The authentication results suffer from unacceptable large FAR to be used in any practical application, but are still significantly better than random.

Keystroke detection, while error prone, did not seem affect overall performance of the classification task. This is most likely due to the user prediction being determined by as the product of many individual feature vector posteriors, and therefore tolerant to a small number of erroneous data points. Further work would still be beneficial in improving keystroke detection and determining its affect on the authentication task.

One method for improved keystroke detection can include a trained HMM. The optimal state sequence when given a series of feature vectors can then be used to determine which feature vectors pertain to *hit* and *release* peaks, and which are background noise.

Other work that follows from this project includes analyzing the effect of increasing the size of the user space as well as identification from free-form typing instead of fixed username and password. Furthermore, the incorporation of other biometrics such as mouse movement patterns can be studied.

The work of Zhuang et al. [2] would also benefit from being able to model artifacts introduced by a user's typing behavior versus those necessary for reconstruction of text. In deed, monitoring a user's typing behavior may lead to yet other novel applications such as context-sensitive assistance.

Finally, although this research has shown that it is possible to identify a user based on keyboard acoustic eminations, there are still a lot of practical issues that would need to be resolved in order to actually deploy a robust system. Such issues include filtering out background noise, dealing with differences between keyboards, and tracking temporary and permanent changes in typing patterns (for example when a user injures a hand or finger). That said, this work has highlighted what is possible and provides some insights that can be applied in a range of similar research areas involving biometrics and emanation security.

Acknowledgment

The authors would like to thank the members of the CS229 class who volunteered the biometric typing samples for used in this study. We are also grateful to Dan Boneh in discussions regarding this subject matter.

References

[1] D. Asonov and R. Agrawal, "Keyboard acoustic emanations." in *IEEE Symposium on Security and Privacy*, 2004, pp. 3–11.

[2] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," *To appear in Proceedings of the 12th ACM Conference on Computer and Communications Security*, November 2005. [Online]. Available: http://trust.eecs.berkeley.edu/pubs/3.html

[3] F. Monrose, M. K. Reiter, and S. Wetzel, "Password hardening based on keystroke dynamics," in *CCS '99: Proceedings of the 6th ACM conference on Computer and communications security*. New York, NY, USA: ACM Press, 1999, pp. 73–82.

[4] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Trans. Inf. Syst. Secur.*, vol. 5, no. 4, pp. 367–397, 2002.

[5] J. John R. Deller, J. G. Proakis, and J. H. Hansen, *Discrete Time Processing of Speech Signals*. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1993.

[6] e. a. Lau, E., "Enhanced user authentication through keystroke biometrics," *6.857 Computer and Network Security final project report*, 2004.