(sp)iPhone: Decoding Vibrations From Nearby Keyboards Using Mobile Phone Accelerometers

Philip Marquardt*
MIT Lincoln Laboratory
244 Wood Street, Lexington, MA USA
philip.marquardt@ll.mit.edu

Arunabh Verma, Henry Carter and Patrick Traynor
Georgia Institute of Technology
{arunabh.verma@, carterh@, traynor@cc.}gatech.edu

ABSTRACT

Mobile phones are increasingly equipped with a range of highly responsive sensors. From cameras and GPS receivers to three-axis accelerometers, applications running on these devices are able to experience rich interactions with their environment. Unfortunately, some applications may be able to use such sensors to monitor their surroundings in unintended ways. In this paper, we demonstrate that an application with access to accelerometer readings on a modern mobile phone can use such information to recover text entered on a nearby keyboard. Note that unlike previous emanation recovery papers, the accelerometers on such devices sample at near the Nyquist rate, making previous techniques unworkable. Our application instead detects and decodes keystrokes by measuring the relative physical position and distance between each vibration. We then match abstracted words against candidate dictionaries and record word recovery rates as high as 80%. In so doing, we demonstrate the potential to recover significant information from the vicinity of a mobile device without gaining access to resources generally considered to be the most likely sources of leakage (e.g., microphone, camera).

Categories and Subject Descriptors

K.6.5 [**Management of Computing and Information Systems**]: Security and Protection—*invasive software*

General Terms

Security

Keywords

mobile phones, information leakage, accelerometer

1. INTRODUCTION

Mobile phones are becoming increasingly powerful devices. In addition to being able to run applications ranging from email clients to web browsers, a progressively sophisticated set of sensors are enabling these devices to more actively interface with the world around them. From gestures captured by accelerometers for games to augmented reality applications displaying metadata tags on video from the camera in real-time, mobile phones are becoming adept at capturing and harnessing rich features and data from their surroundings.

Unfortunately, the array of sensors in these devices can also be used in unintended ways. As many have suggested, malware could potentially gain access to a mobile phone's camera to take photos or video of the owner and their surroundings [16, 37]. In cases where the camera may not be pointed at an interesting target, a malicious application could instead attempt to activate the device's microphone and record ambient sounds or syphon GPS data to track the target's location [17, 13, 23, 21]. Recognizing the potential for the leakage of sensitive information, many modern mobile phone operating systems provide mechanisms by which access to such sensors is protected by explicit permission from the user. For instance, the manifest files included with every Android application provide an unambiguous list of all of permissions an application may ever request at installation time. Should an application not request access to these resources[1] or the user deny the application such rights, the application will not be able to potentially abuse these resources. However, access to all of the sensors contained in these devices are not so tightly controlled.

In this paper, we demonstrate that unfettered access to the accelerometers available on many mobile phones can allow for significant unintended leakage of information from a user's environment. We show that a malicious application with access to the accelerometer feed can record and reconstruct the keypresses made on a nearby keyboard based solely on the observed vibrations. We develop profiles for pairs of keypress events using a neural network, which creates an abstract representation of the relationship between consecutive events. We then recover the typed content by translating from our intermediary form to English words using a number of different dictionaries. Such tasks are not a trivial application of standard techniques, especially when compared to previous efforts in this space. Specifically, we must overcome much lower sensor sampling rates than has been experienced in the related acoustic and electromagnetic-based eavesdropping attacks, which makes deciphering individual keypresses extremely difficult.

Through this, we make the following contributions in this paper:

- **Recover keystrokes with far less precise monitoring equipment:** We note that while a number of related efforts analyze

*Work conducted while at Georgia Institute of Technology

[1]Specifically, the manifest file would need to contain `RECORD_AUDIO`, `CAMERA` and `ACCESS_FINE_LOCATION` [2].

CCS'11, October 17–21, 2011, Chicago, Illinois, USA.

keyboard emanations, our techniques overcome a noteworthy disadvantage. Most critically, the sampling rates of the devices used to recover acoustic and electromagnetic emanations are six (2.5 GHz) [40] and two (44.1 kHz) [3] orders of magnitude *greater* than that of the accelerometers running in a modern mobile phone (100 Hz).

- **Develop an infrastructure for characterizing keypress vibrations:** We capture, analyze and develop profiles of pairs of keypress events on a nearby keyboard based on the vibrations created when they are pressed. Our inputs are then processed using a neural network to create an intermediary representation of the relative position and distance between keypress pairs, which is combined with candidate dictionaries to successfully recover words at rates as high as 80%.

- **Provide effective mitigation techniques:** In spite of a rich resource management options, we demonstrate that mobile phones lack sufficient policies to prevent such attacks. We offer suggestions to dramatically reduce the effectiveness of these kinds of attacks without appreciably affecting other applications requiring legitimate access to the accelerometers.

Note that for all its apparent obstacles, our approach has a significant advantage over previous work. Attacks designed to compromise keystrokes using electromagnetic and acoustic emanations have thus far required that an adversary gain undetected physical access to the space occupied by their target. Our approach eliminates this requirement by allowing our malicious application to run on a device most users are likely to already be carrying with them.

The remainder of this paper is organized as follows: Section 2 discusses important related work and provides context for our contributions; Section 3 describes our attack and threat model in detail; Section 4 attempts to apply previously used methods and demonstrates why they do not work in this setting; Section 5 describes our model and experimental setup in detail; Section 6 offers experimental results; Section 7 discusses a number of related issues and mitigation strategies; Section 8 provides concluding remarks.

2. RELATED WORK

The emanations of electrical and mechanical devices have long been known to expose information about their users' activities. As early as the 1940s, scientists working on the TEMPEST project demonstrated the ability to capture the electromagnetic signals generated by the Bell 131-B2 teletype terminal, which was used to encrypt communications by the US military [29]. These techniques were improved and used during the Vietnam War to detect Viet Cong trucks at distances of up to ten miles [28]. More recently, researchers applied similar techniques to CRT [39] and LCD [27] monitors and demonstrated the ability to recover their contents both at great distance and through significant obstacles (e.g., brick walls). Similar attacks on electromagnetic emanations have since been demonstrated against Smart Cards [31], CMOS chips [1], serial port cables [34] and keyboards [40]. While such an attack vector was certainly within the capabilities of sophisticated adversaries, its use by the general public has thus far been fairly limited.

Optical emanations and reflections also offer a potential source of information. Kuhn [26] demonstrated that temporal variations of the diffuse reflections of CRT monitors could be processed to reveal the contents displayed on the screen. Backus et al. [7] were able to capture the content from any monitor based on mirroring by nearby objects with high reflectiveness, including soda bottles, tea pots and eye glasses. This work was later extended to capture reflections

Figure 1: Our experimental placement of a mobile phone running a malicious application attempting to recover text entered using the nearby keyboard.

directly from a target's eye [5]. While these attacks are effective at distances of up to 30 meters, they are difficult to execute without a constant and unobstructed line of sight, which may potentially expose the presence of the adversary.

The acoustic emanations made by a range of devices are more easily captured by less capable adversaries. For instance, using a simple consumer-grade microphone, the contents of printouts made by a dot-matrix printer can be recognized with high accuracy [11, 6]. Recognizing this, the research community has spent significant effort recreating key presses on computer keyboards through such auditory channels. Specifically, previous work has shown the ability to recover greater than 80% of keyboard presses given substantial training [3], without training [42] and based on acoustic dictionaries [10]. While such approaches are certainly more within the reach of the adversary, they are extremely difficult to scale to large deployments as they require that a microphone is physically located near all potential targets at all times.

The vibrations caused by physical activities can also be exploited to surreptitiously leak information. For instance, laser vibration systems focused on flat surfaces such as windows can recover both voiced conversations and typing [8]. As such systems are becoming more readily available to the public [22], their use becomes more practical. However, like the above acoustic case, the scalability of their deployment makes large-scale attacks using such equipment difficult. Through the use of compromised mobile phones, our work focuses on taking advantage of such side channels while making such attacks significantly more scalable.

3. ATTACK DESCRIPTION AND THREAT MODEL

Many previous emanation papers have been successful because they were able to deploy sensitive equipment near their intended target. From microphones to an antenna, previous attacks require an adversary to violate the physical security of the area near their target. This observation does not make such attacks impossible, but certainly increases both difficulty and likelihood of detection. Our attack instead attempts to recover emanations with a device that the target themselves will bring near the keyboard.

Mobile phones perfectly match this description. These devices contain a range of sensors (e.g., camera, microphone, GPS) through which a malicious program could potentially eavesdrop upon a target [32, 38]. Such attacks have long been expected and can easily be prevented by denying access to such resources [24, 9]. Access to information generated by accelerometers, however, is not protected in any current mobile phone operating system. Accordingly, we investigate whether or not this sensor can record nearby phenomenon with sufficient accuracy to recover sensitive information.

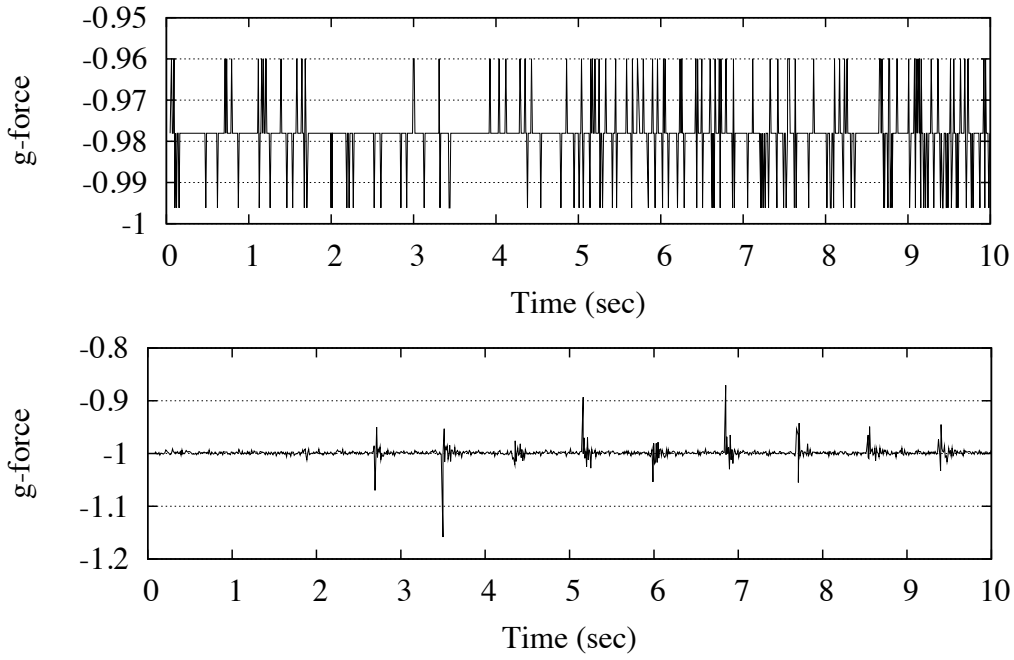

Figure 2: A comparison of accelerometer readings from an iPhone 3GS (top) and an iPhone 4 (bottom) during the spaced and repeated pressing of a single key. Note that keypress events are easily recognizable on the iPhone4, whereas they are indistinguishable on the iPhone 3GS.

Our attack is based on a simple observation. Specifically, many users place their mobile devices on their desk when they are working so that their are easily accessible. Figure 1 shows our experimental setup. An adversary attempting to recover sensitive information entered by a user on a desktop computer would operate as follows: the adversary would install a malicious application on the mobile phone belonging to their target. This step could be by gaining physical access to the device or through social engineering (e.g., suggesting an innocuous and seemingly useful application to a colleague, spam, etc). Secondly, the application will record accelerometer data. This operation need not occur constantly, but could instead occur by explicit wake up (e.g., text message trigger) or by periodic sampling for activity. Finally, the malicious application has some means of exfiltrating the observed data. This channel could via a direct connection to the Internet or potentially indirectly through another application. We do not assume access to any additional resources.

4. APPLICATION OF PREVIOUS TECHNIQUES

A number of papers have recently been published on the decoding of keyboard emanations. Two questions naturally arise. First, given that the accelerometers found in current mobile phones sample at rates that are orders of magnitude smaller than previous acoustic and electromagnetic attacks, *can keypresses even be detected by these sensors?* Secondly, if such events can be observed, *can previously developed methodology (i.e., identifying individual keys using neural networks) be blindly applied to identify keystrokes?*

We answer the first question by comparing the capabilities of two different phones - the iPhone 3GS and the iPhone 4. Figure 2 shows the raw amplitude readings from the z-axis accelerometers on these devices when placed two inches from the left side of a QWERTY-style keyboard. Most noticeably, generic keypress events are unidentifiable on the iPhone 3GS. The relatively noisy output generated by the accelerometers on the iPhone 3GS provide very few differentiable features over which classification could occur. The presence of a better accelerometer and a gyroscope on the iPhone 4 provides significantly improved separation of keypresses and shows 9 distinct events. This observation leads us to assert that accelerometer-based eavesdropping could potentially be implemented on some phones.

Having satisfactorily determined that keypress events are observable using the accelerometers included on a currently available mobile phone, we attempt to determine whether such events are measurably different enough to allow for classification. Asonov et al. [3] showed accuracies of 79% by processing acoustic emanations using microphone signals digitized with a standard PC sound card with 44.1 kHz sampling rate. In their experiments, audio samples were captured and feature vectors were created using FFT values for each sample. These feature vectors were used to train a neural network for classification.

Our initial experiments aimed to recreate the methods used in the acoustic work. We gathered samples using an iPhone 4 on a wooden surface located 2 inches from a wireless Apple Bluetooth keyboard. Each alphabetic key was sampled 150 times by gathering x, y, z values from the raw accelerometer output. Due to the constraints on the accelerometer hardware, our sample rates were limited to 100 Hz, 441 times less than Asonov et al. The raw output was then processed through several algorithms for creating feature vectors, including mean, min, max, FFT, MFCC and other time window values. We trained and tested a neural network using a 70%/30% data split. As expected, our overall accuracies (25.89%) were significantly lower than Asonov et al. (78.85%). Specifically, only 3 letters out performed the acoustic results, 'a', 'e', and 'o'. However, overall the accuracies of our initial results

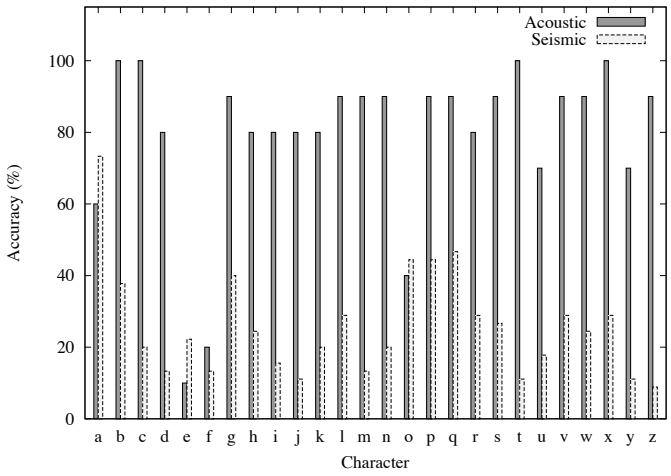

Figure 3: Comparison of the per-letter accuracies for acoustic (theirs) and seismic (ours) emanations using the methodology by Asonov et al. [3]. Note that our proposed use of accelerometer provides significantly less accurate results (25.89% vs 78.85%) based on this approach. Accordingly, a different approach is necessary.

were well below the results gathered in the acoustic experiments. Figure 3 shows comparative results for each alphabetic key.

These initial results tell us that while keypress events are detectable, the direct application of previous techniques does not provide an effective means of recovering text entered on a nearby keyboard. We therefore investigate an approach more suited to our limited sampling abilities.

5. MODELING KEYPRESS EVENTS

In this section, we present our model for identifying keypress events. We then discuss our experimental setup and processing architecture.

5.1 Keypress Event Modeling

Because of the extremely low sampling rates available to accelerometers and our previously demonstrated difficulty in recognizing individual keys, we instead attempt to characterize pairs of keypresses and their relation to each other. Let P_i, P_j equal sequential keypress events. We characterize the relation between two successive events, $rel(P_i, P_j)$ using the following two features:

- **Horizontal Orientation:** The location $loc(P_i)$ of each keypress event relative to a 'central-line' dividing the keyboard into *left* and *right* partitions.
- **Distance Between Consecutive Keypresses:** For a threshold distance in keys α, we define the distance $dist(P_i, P_j)$ of consecutive keypress events as being either *near* or *far*, where $near < \alpha$ and $far \geq \alpha$.

We define consecutive keypress events as $rel(P_i, P_j) = loc(P_i) ||loc(P_j)||dist(P_i, P_j)$, where $||$ represents feature concatenation. A word is thus composed of a series of concatenated event-pairs, referred to as its 'word-profile'. As an example, assuming $\alpha = 3$ and a central-line marking all keys including and to the left of 't', 'g' and 'b' on a standard QWERTY keyboard as *left* and all keys including and to the right of 'y', 'h' and 'n' as *right*, the word "canoe" would be represented as:

LLN.LRF.RRF.RLF.

Note that for the five letter word "canoe", there are four pairs of consecutive keypresses: "ca", "an", "no", and "oe". Accordingly, all words of length n will be represented as abstract strings of length $n - 1$ in our system. These abstract words can then be processed for content using dictionaries as discussed below. We note that as there are only two one letter words in English (e.g., 'a' and 'I'), that we can identify such words as a special case in our architecture.

5.2 Framework Overview

To convert raw accelerometer data into dictionary words, our framework consists of a two phase process: the *learning* and *attack* phases. Figure 4 provides an overview of this process.

5.2.1 Learning Phase

Before we begin training our learner, our framework defines a preprocessing step needed to build the training data. This step consists of determining parameters for where the keyboard will be split for *left-right* and *near-far* labeling (α and central-line) . Once these parameters are defined, we build training data and word-profiles from different predefined English dictionaries for use during the learning and attack phases. To avoid over-fitting in the learners, the preprocessing step also ensures that the training data has an equal distribution of features.

After preprocessing, our framework trains the model through data collection, feature extraction, word labeling, and supervised learning processes.

Data Collection: When a key is pressed, the iPhone's accelerometer senses the surface vibrations and gives outputs values of the instantaneous acceleration on the x, y, and z axes. Our 'Data-Collector' application running on the iPhone records these acceleration values at a variable sampling rate (100Hz on average) along

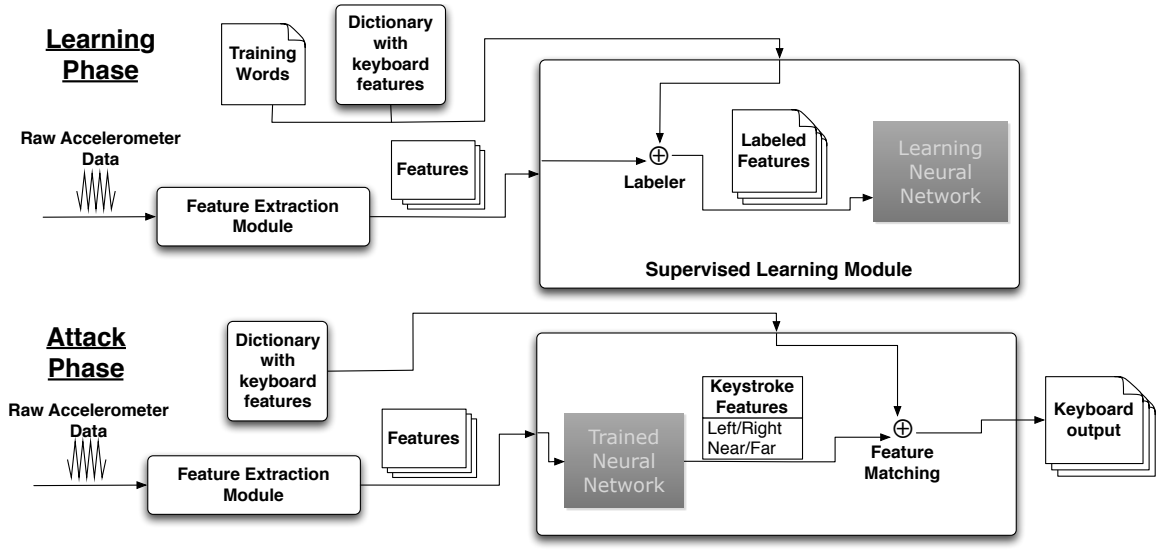

Figure 4: The Data Processing Architecture. This diagram provides a high level overview of the architecture used for training the neural network and for analyzing keyboard input during an attack.

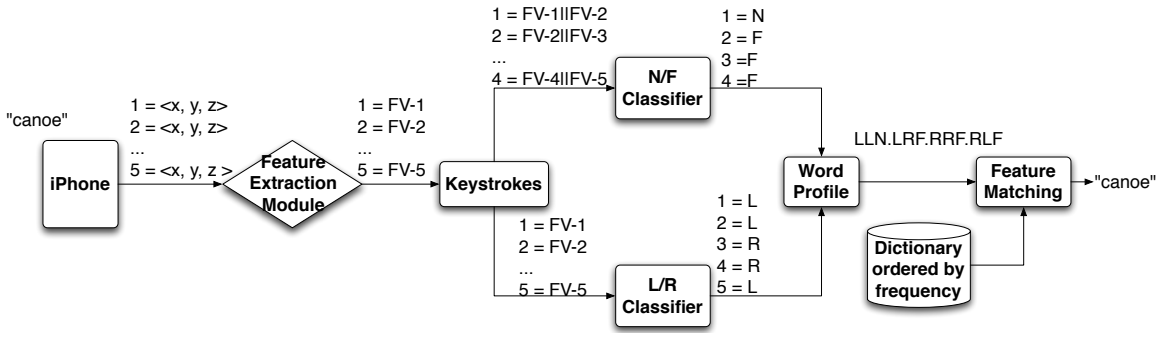

Figure 5: An example of identifying a single word "canoe". As the victim types, raw x, y, and z values are captured and converted to feature vectors. These feature vectors are analyzed by two classifiers, creating a word profile that is used to select the best possible match from a dictionary of words.

with the key that was pressed. At the start of the learning phase, we type all the letters in the English alphabet (a through z) 150 times each (with no fixed ordering or timing) on the target keyboard. The 'Data-Collector' application continuously records the seismic emanations produced by these key-presses resulting in the raw-acceleration dump for learning phase D_L of 3900 distinct key-press events corresponding to all the alphabets (150 times).

Feature Extraction: The raw acceleration values collected are then processed to extract relevant features to be used to train the model. Due to the iPhone accelerometer's low sampling rate, we were unable to obtain satisfactory classifier performance by only using features gathered in previous emanations work [3]. Therefore, we use a combination of time-domain, frequency-domain and cepstral features to construct our 'feature-vector'. Asonov et al. [3] determined a key-press is approximately 100 ms long guiding us to use the same time window for our feature extraction calculations. From the raw acceleration (x, y, z) values stored in D_L, we calculate time-domain features including root mean square (rms), skewness, variance, and kurtosis and combine those with spectral and cepstral features such as fast Fourier transform (FFT) and mel-frequency cepstrum coefficients (MFCCs) respectively. The final feature-vector corresponding to the x, y, z accelerations of a key-press (P_i) is denoted as: $FV(P_i) =<$ mean, kurtosis, variance, min, max, energy, rms, mfccs, ffts $>$. At the end of this step, we have a set (SF) containing 150 feature-vectors (FVs) corresponding to each English letter. i.e., $SF = \{FV(P_i) \mid \forall P_i \in D_L\}$

Word Labeling: Once the feature vectors are extracted from the raw accelerometer data, the framework creates a training set by labeling individual key and key-pair samples based on the defined dictionary. To prevent over-training, a preprocessing step is performed which determines the number of samples to take per dictionary character by finding an even distribution of *left* (L) and *right* (R) and *near* (N) and *far* (F) labels.

Each word in the training dictionary is broken down into its constituent characters and character-pairs. As described earlier, a word of length n letters would be broken into n characters and $n-1$ character-pairs. For each character, we randomly select 100 feature vectors that correspond to that character from the Feature Extraction step and label each of these feature vectors as *left* (L) or *right* (R) based on the Word Labeling step. In a similar manner, for each character-pair we construct 100 composite feature vectors by concatenating 100 random feature vectors corresponding to the first character and 100 random feature vectors corresponding to the second character. Each of these composite feature vectors is labeled

near (N) or *far* (F), again using the labels from the Word Labeling step. Once this process is completed for every character and character-pair in the dictionary, the labeled feature vectors can be used to train the neural networks in the last step.

Supervised Learning: The final step of the learning phase consists of training a model that will be used during the attack phase to classify each key and key-pair accordingly. We built two separate models by training a *left-right* neural network and a *near-far* neural network for classifying *left-right* and *near-far* feature-vectors respectively. Both neural networks where trained using 500 training cycles and a learning rate of 0.3.

5.2.2 Attack Phase

Our framework defines process for the attack phase much in the same way the learning phase was defined by attempting to recover words through data collection, feature extraction, key-press classification, and word matching.

Data Collection: When the 'Data-Collector' application becomes malicious, the same properties and procedures apply to the attack phase that existed in the learning phase with the exception of the ability to tag an accelerometer with the appropriate key-press. Under these circumstances we're provided with a raw-acceleration dump for attack phase D_A of all the key-presses entered by the victim minus the character pressed.

Feature Extraction: After the raw-acceleration data is collected by the malicious application, the attack phase then calculates the same features that were derived during the learning phase. The feature-vectors are then used to create two sets of data, one for classifying *left* vs. *right* and one for classifying *near* vs. *far*.

Key-press Classification: Once the data has been processed, our framework attempts to classify each vector. The model produced by the *left-right* neural network is used to predict the L/R label for each individual key-press while the model produced by the *near-far* neural network is use to predict the N/F label for each pair of key-presses. Each word is then labeled with its appropriate word profile using the predicted labels and sent to the word matcher.

Word Matching: The word matcher takes each predicted word profile of length $n-1$ and assigns a score against each word in the dictionary with length n. Algorithm 1 defines the scoring algorithm. The scored dictionary words are then sorted and the top two scores are presented as candidate predictions for the given predicted profile.

Algorithm 1 Word matcher scoring

```
curScore = 0
for all words in dic of len(n) do
    for i = 0 to 2 do
        if dic.word.profile[i] = prediction.profile[i] then
            if dic.word.profile[i] = L or dic.word.profile[i] = R then
                curScore++
            else if dic.word.profile[i] = N or dic.word.profile[i] = F then
                curScore++
            end if
        end if
    end for
end for
return curScore
```

5.3 Experimental Setup

In all of our experiments, we set the target desktop computer on a wooden desk. We then placed the iPhone used to collect keyboard vibrations beside the target computer's keyboard. To maintain consistency, we always placed the phone at a distance of two inches from the left hand side of the keyboard. *It is important to note that there is nothing inherently desirable about the orientation of the device used in our experiments*. The attack will work if the device is trained in any orientation. All raw accelerometer readings were collected by the phone, then uploaded to a remote server. This server ran the data processing and classification algorithms used to analyze the collected accelerometer readings.

The hardware and software components used to perform the experiments are described below:

1. Keyboard: We used Apple A1255 Wireless Bluetooth keyboard. Bluetooth was used so that we could send the typed characters to the data-collector application running on the iPhone while collecting accelerometer readings. We did this to automatically and accurately label the collected data during the Learning phase and to match the classifier's predicted word-profile to the actual word-profile in the Attack phase.

2. Phone and Accelerometer: We used the iPhone 4 (running firmware no. 03.10.01). The phone contains an ST-LIS331DLH accelerometer; operating with sensitivity of 1 mg/digit, g-range of $\pm 2g$ and Acceleration noise density of 218 $\frac{\mu g}{\sqrt{Hz}}$ [41]. The maximum accelerometer sampling rate achievable in our experiments was 100Hz.

3. Signal Processing: For calculating the FFT coefficients, we used MATLAB's Discrete Fourier transform function (FFT).The mel-frequency cepstral coefficients values were calculated using its Voicebox toolbox [12]. For MFCC calculations, we set the number of channels in the Mel Scale Filter Bank to 32 and use 3 MFCCs computed for the frame-duration of 0.025s and frame step of 0.01s.

4. RapidMiner / Machine learning: We used the RapidMiner (version 5.0.010) data mining application and libraries for training and testing our neural network models and developing our framework. This choice was motivated by RapidMiner's extensibility and integration in Java-based applications, the core of our framework.

6. EXPERIMENTAL RESULTS

To test the accuracy of our attack, we performed four experiments, varying the test sentences and the target dictionary used to match word profiles. In our first experiment, we demonstrate the accuracies that the L/R and N/F classifiers are capable of achieving by analyzing a single test sentence from the Harvard Sentences [30]. We chose this list of sentences because it is phonetically-balanced, i.e. it uses specific English phonemes with a uniform distribution. In our second experiment, we examine the system's potential for text recovery using a small dictionary of the first ten Harvard sentences and attempting to recover the same ten Harvard sentences as test data. In our third experiment, we calibrate our work against previous research in this space by re-creating the experiment used by Berger et al. [10] in order to gauge the relative accuracy of our approach against very large dictionaries. Finally, we simulate a real-world attack on a piece of written media selected from the USAToday [14]. We construct a dictionary based on the context of the target article using related news stories, then type the article as test data.

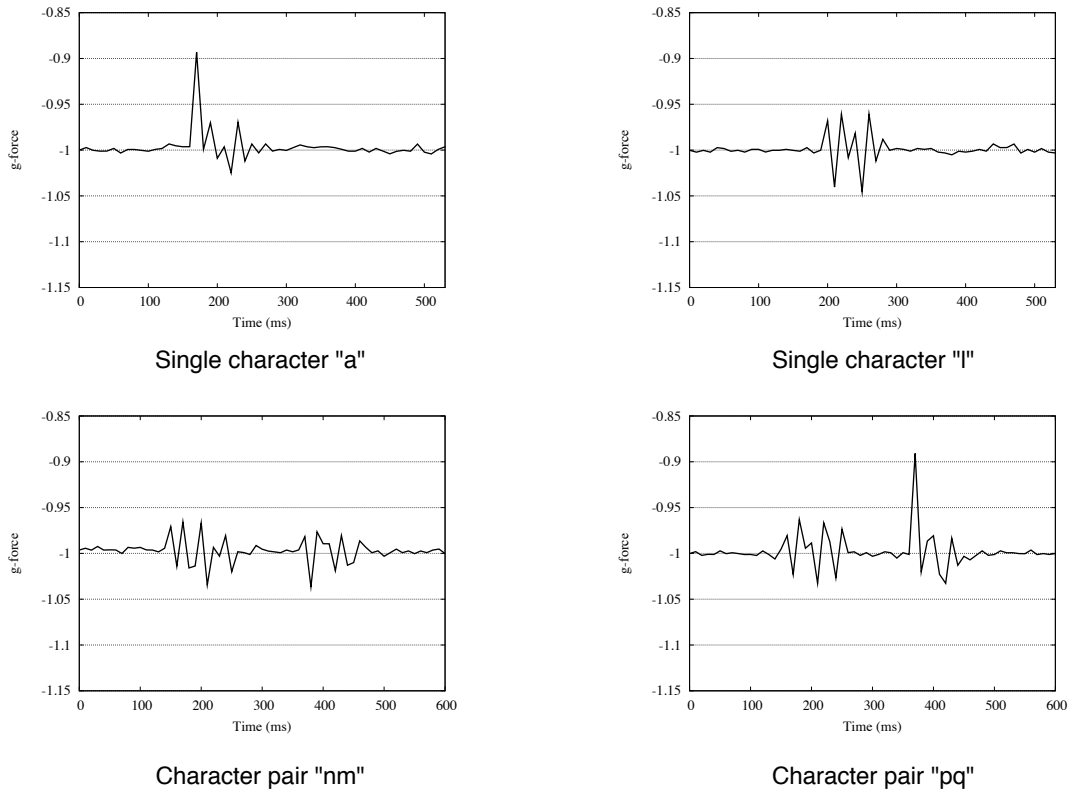

Figure 6: Examples of g-force measurements for L/R keypresses and N/F pairs. Note the distinct difference between left and right characters “a" and “l", as well as the varying level of contrast between near and far pairs “nm" and “pq". Also, left and right characters are even distinguishable across graphs, with “a" and “q" creating similar left side vibrations while “l", “n", “m", and “p" create similar right side vibrations.

6.1 Feature Accuracy

Our first analysis of these three experiments examined the base accuracy of both classifiers in correctly distinguishing letters as L/R and pairs as N/F. Figure 6 shows examples of the g-force measured by the accelerometer for two individual keypresses and two pairs. Even viewing with the naked eye, the difference between keypresses on the left end of the keyboard (“a”) and the right end (“l”) is significant. When visually examining concatenated pairs, the difference between two adjacent keypresses (“nm”) is minimal when contrasted to the difference in two distant keypresses (“pq”). Our results demonstrate that for a majority of keypresses and keypress pairs, these differences are distinguishing enough to correctly identify the approximate location of a keypress on the keyboard. Moreover, these results reinforce the results shown in Figure 3. The keypresses of letters in close proximity, such as “nm”, create very similar vibrations, making it incredibly difficult to identify the specific key pressed. Thus, our method of identifying a region rather than a specific key provides a more feasible means of differentiating between keypresses.

In our first experiment, the L/R classifier was able to correctly identify 91% of the individual keypresses as right or left, and 70% of the keypress pairs as near or far. As the size of our test data grew, these percentages declined. For the second experiment, our L/R accuracy was 84% and our N/F accuracy 65%. This drop in accuracy is to be expected with more keypresses, as the target will exhibit minor inconsistencies in striking the same key over time. Considering that many emails sent are no more than one or two sentences, these experiments represent a significant leakage of information.

6.2 Recovering Text

Our next analysis examined the percentage of text recovered by the matcher given the word profiles produced by the neural networks. When examining the preliminary results of our experiment, we noted that the overall percentages of words recovered noticeably dropped due to the frequency of two and three letter words in the analyzed text. Since these words are generally articles and conjunctions (e.g. an, the, and, or), they can be easily interpolated by examining the semantics of the recovered text. In our experiments, we opted to consider only “long" words of four letters or more in all final percentages of recovered words. In each figure, the “short” words will be denoted with asterisks.

In the first two experiments, the dictionary contained a small set of words containing the test data exactly. Our first experiment took vibrations from typing a single sentence and attempted to analyze the text. In Figure 7, we show the results of this experiment and compare the recovered sentence to the actual typed sentence. For each word profile, the matcher returned a list of possible results for that profile weighted by frequency of use. For the first Harvard sentence, 80% of the words were correctly identified as the first choice for their interpreted word profile. In the second experiment, we expanded the test data to the first ten Harvard sentences. For this larger analysis, 46% of the words were correctly identified as the first choice for their interpreted word profile. However, if we

1st Choice Correct = 80%
L/R Accuracy = 91.07%
N/F Accuracy = 70.15%

Typed Text: The birch canoe slid on the smooth planks
Recovered Text: *** punch canoe slid ** *** smooth planks

Figure 7: Experiment 1. In this experiment, we attempted to interpret a single sentence, shown here. Our dictionary was composed of words from the first ten Harvard sentences. The first line shows the sentence as it was typed, and the second line shows the sentence as our program analyzed it. A string of asterisks represents a word of three letters or fewer, all of which were omitted.

1st or 2nd Choice Correct = 72.92%
L/R Accuracy =83.95%
N/F Accuracy = 64.88%

Typed Text: Glue the sheet to the dark blue background
Recovered Text: Glue *** sheet ** *** well hogs background
blue

Typed Text: These days a chicken leg is a rare dish
Recovered Text: These days * chicken *** ** * rare dish

Figure 8: Experiment 2. Again using the first ten Harvard sentences as a dictionary, we attempted to interpret all ten sentences and selected two to display here. We again show the text as it was typed and as our program analyzed it. If the correct word was weighted second by the matcher, it is listed below the first choice word, which is written in line. A string of asterisks represents a word of three letters or fewer, all of which were omitted from the analysis.

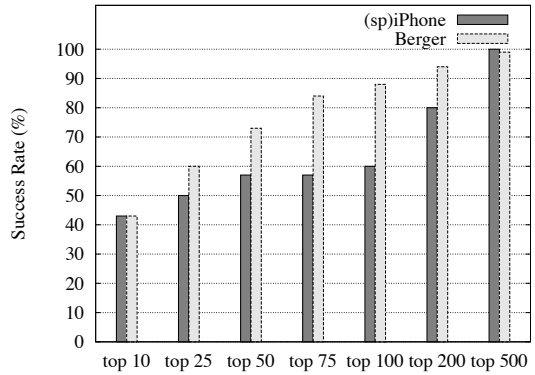

Figure 9: A comparison of accuracy of our ((sp)iPhone) and the acoustic dictionary techniques proposed by Berger et al [10]. Note that in spite of our dramatically reduce sampling rate, our results are roughly comparable for a very large dictionary (i.e., 57,000+ words).

also consider the words ranked second most likely by the matcher, this accuracy rises to 73%, as shown in Figure 8. By examining the semantics of a captured sentence, we can easily choose the correct choice between the top two choices for a word profile.

6.3 Comparison to Previous Work

In an attempt to calibrate this work against previous efforts in this space, we re-created the experiment from the work by Berger et al. [10]. This work cites a "corn-cob" dictionary of approximately 58,000 words and attempts to use the acoustic emanations of a keyboard to recover 27 test words ranging in length from 7-13 characters. In our experiment, we used a portion of the same dictionary, containing approximately 57,500 words (all the words with 4 or more characters). Rather than using the same 7-13 character word list, we selected 30 words from a randomly chosen USAToday article [14] (announcing a court ruling to keep then candidate Rahm Emanuel on the ballot for the February 2011 Chicago mayoral election), ranging in length from 4-9 characters. This test word set better demonstrates the practical accuracy of our technique by using a test sample from written media rather than an arbitrary selection of long words. The comparison is shown in Figure 9.

After running our list of test words through the data processing architecture, our techniques demonstrated comparable accuracies to that of Berger et al. [10]. Our technique placed the correct guess in the top 10 potential words selected for each test word 43% of the time, matching the work of Berger. When examining the number of correct terms in the top 50 potential words selected, our accuracy trailed somewhat at 56%, compared to 72% for the acoustic work. However, it is important to notice two distinct challenges faced by our technique. First, the frequency which we sample vibrations is two full orders of magnitude smaller than in the acoustic emanation work. With this extremely limited bandwidth, we are still able to match the acoustic accuracy in the top 10 potential words selected. Second, our word list uses words of 4-9 characters in length, as opposed to the 7-13 character words used in the previous work. The dictionary attacks used in both works allow longer words to be identified more easily since there are fewer potential matches in the dictionary of terms.

6.4 A More Realistic Attack

To demonstrate the feasibility of our technique in a real-world

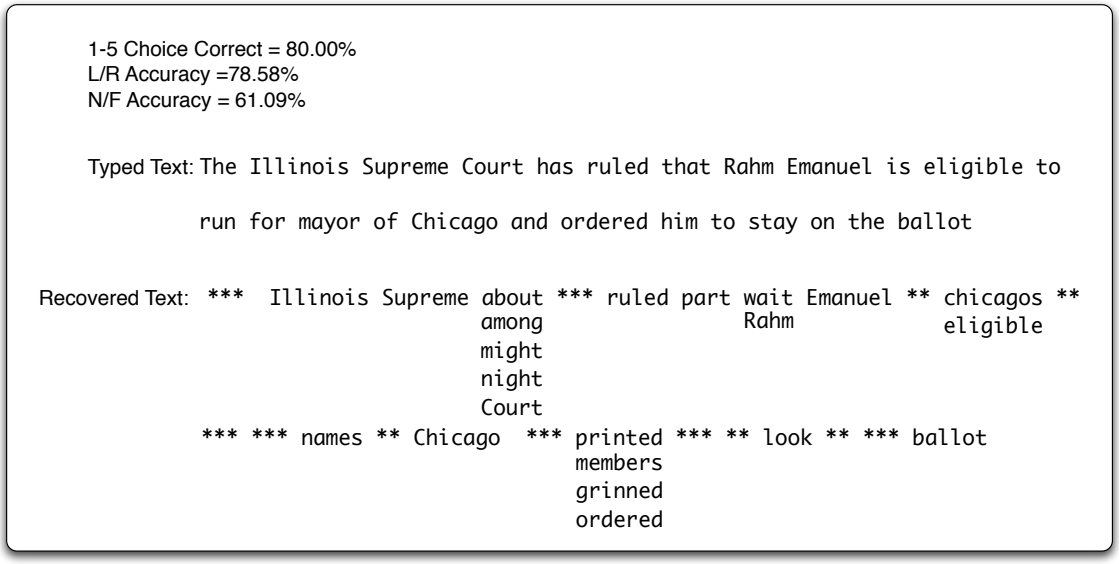

Figure 10: Experiment 4. Using news articles as a dictionary, we analyzed the typed contents of a USAToday article. Here we show an excerpt as it was typed and as our program analyzed it. If the correct word was weighted between second and fifth choice by the matcher, it is listed below the words that were weighted more heavily. A string of asterisks represents a word of three letters or fewer, all of which were omitted from the analysis.

attack, we examined the following scenario. Since an attacker may know the general context of a target's message, it is reasonable to construct a dictionary based upon terms that are likely to appear in the target's writing. So we constructed a dictionary using seven news articles related to Rahm Emanuel (from the New York Times [20, 18, 19] and USAToday [36, 15, 25, 35]) that were published during the week before the target text. This simulates eavesdropping on the reporter typing the USAToday piece. We again apply the methodology by Berger et al [10] to measure our results.

We again examined the same words of our test sample (from Experiment 3), this time using our 799 word context-aware dictionary. In 40% of the tests, our technique selected the correct word as first choice, and 53% fell in the first or second choice, as shown in Figure 10. The correct word fell in top 5 predictions for 80% of the tests. Based on these results, it is apparent that an attacker could easily recover a dangerous amount of text captured using our technique, simply by knowing some context for the target's writing.

7. DISCUSSION

7.1 Recognition versus Distance

Our experiments thus far have been conducted with the mobile phone located within two inches of the targeted keyboard. We believe that this is reasonable given the dimensions of most desks and the assumption that many users will wish to have their mobile phone with reach while working at their computer. However, our analysis would not be complete without an understanding of how increasing the distance between the mobile phone and its target impact recovery.

As expected, even small increases in distance dramatically reduce the effectiveness of this attack. Between absorption and attenuation of signal based on the Inverse Square Law, our accuracy drops rapidly at distance of one foot. Distinguishing keystrokes from noise at two feet is extremely difficult, effectively disrupting the attack. Beyond this range, our mechanism quickly approaches random guessing as the received signals are simply too small to meaningfully distinguish between. As we discuss in one of the following subsections, this simple observation functions as a powerful mitigation to these attacks.

Due to the low sensitivity of the accelerometer, the distance limitation is a difficult problem to overcome. Environmental factors, such as the surface characteristics of the desk (discussed further in the following section), add more complexity to this challenge. In the wrong conditions, even a small increase in distance could completely inhibit the attack, while good conditions could facilitate keystroke recognition from well over a distance of one foot. Because of variable environmental conditions, we can only guarantee proper functionality within one foot.

7.2 Challenges and Limitations

We examined three additional challenges that could be encountered with this application. Our first potential challenge relates to the orientation of the monitoring device. In all of our experiments, we positioned the mobile device vertically to the left of the keyboard, as shown in Figure 1. However, if this orientation were to change, the vibrations measured for the same keystrokes would be captured differently on the device's x and y axes. On first review, this would seem to cause errors in accuracy. The first strategy for compensating for a change in orientation would be to simply re-train the neural network to identify keystrokes coming from a device in the new orientation. However, according to the work of Zhuang et al. [42], the neural network *does not have to be re-trained* in this manner. Instead, the keystrokes could potentially be identified based on measuring their frequency. By using this technique to analyze vibrations captured by the mobile device in *any* orientation, we believe that we can produce similar results regardless of the orientation of the mobile device. We plan to explore this issue in future work.

The second potential challenge we considered was ambient vibrations. There is a plethora of possible scenarios where vibrations

in the environment around the device could potentially garble the keystroke information being collected. Some of these vibrations could be subtle or consistent enough that our application would be able to distinguish keystroke information from the ambient noise. For example, if an office is adjacent to an air conditioning unit, a consistent vibration could potentially be detected and filtered out. However, other more obtrusive possibilities also exist. In skyscrapers, the movement of the building itself could be detected, causing periodic interference with keystroke detection. Users who bounce their knee or habitually tap on their desk would also send significant vibrations into the device, again causing a loss of pertinent data. This problem, although not common in most work environments, still merits further consideration with regards to the overall accuracy of this attack.

Typing speed is also likely to pose a problem to some recovery scenarios. Like previous studies [3], we limited the rate at which we typed so that easily distinguishable characters could be recorded and a proof of concept implementation of the attack created. However, some users that type very fast are likely to cause problems to the current Data-Collector. These problems arising from such behavior are not likely to be a result of keypresses overlapping - each keypress lasts roughly 100 ms and a user typing a key every 100 ms would be able to type approximately 120 words per minute (i.e., the extreme high range for professional typists [4]). Instead, the rapid movement of hands on the surfact is likely to cause addition noise, potentially making recognition more difficult. We intend to study this issue in greater depth in our future work.

The final challenge we considered was desk surface characteristics. The capability of an accelerometer to detect vibrations in a desk is directly dependent on the surface's ability to amplify or dampen these vibrations. In all of our experiments, we recorded keystrokes on the most common desk surface, wood. To begin to answer the question of the impact of surface characteristics on accuracy, we performed additional experiments on a ceramic tile surface. As would be expected from a rigid surface, keystroke vibrations on ceramic tile were not carried to the device at all, completely inhibiting the use of our application. The extent of this limitation to other surfaces, such as metal or plastic, merits some further consideration. However, we consider vibration-inhibiting surfaces such as tile to be minor special cases when evaluating the overall usefulness of our application.

7.3 Mitigation Strategies

The attack discussed in this paper demonstrates the need for careful thinking about how the array of sensors now available to mobile phones can be accessed. While access to the most obvious candidates for information leakage (e.g., camera, microphone, GPS) is increasingly being protected, we have shown that access to seemingly innocuous sensor data can result in the accurate recovery of potentially sensitive data. As a result, we now offer a number of short and long term mitigation strategies and mechanisms to address these problems.

The simplest mechanism is in preventing mobile phones from coming too close to keyboards. Some businesses and many government buildings already forbid their employees from carrying such devices on the premises. However, such an approach may be too restrictive for most corporate and home environments, especially given the common and legitimate use of mobile phones in these settings. Alternatively, a party concerned about such eavesdropping can place their mobile phone in a briefcase, backpack or handbag which they regularly carry. Finally, a user may simply place their mobile device on a separate surface. As our experiments have demonstrated, the accelerometers contained in mobile phones at the time of this writing are not nearly sensitive enough to detect and uniquely identify keypress generated vibrations.[2]

In the long term, mobile device operating systems should provide more finer-grained control to their resources. It is likely that attacks using other unregulated sensors in unintended ways will be possible on these platforms. Such access need not simply be Boolean in this particular case. Instead, we can limit the sampling rate to take advantage of the information theoretic lower bounds for avoiding aliasing based on the Nyquist-Shannon sampling theorem [33]. Specifically, an observed signal must be sampled at the Nyquist rate, at least two times the rate of the highest frequency, to prevent signals from becoming indistinguishable from each other. From our experiments, we determined that the highest observed frequency in our dataset was approximately 15 Hz. Accordingly, by providing all applications with less than 30 Hz resolution to accelerometer data by default, such attacks become theoretically impossible. This approach will be particularly successful for applications such as text editors which are likely to only use accelerometer data to rotate the contents of the screen based on the user turning their phone. However, the use of a decreased sampling rate may not be appropriate for applications such as games which may require more accurate measurements of movement (e.g., driving games in which the car is steered by rotating the phone). These applications should instead be given explicit access to a high-sampling permission, much as application manifests for Android can be written for fine and coarse-grained GPS access [2].

8. CONCLUSION

Mobile phones contain an array of powerful sensors. While access to many of the most obvious sources of information is generally restricted, the use of a number of a number of other seemingly innocuous sensors is not. In this paper, we demonstrate that unfettered access to accelerometer data allows a malicious application to recover and decode the vibrations caused by keypresses on a nearby keyboard. By characterizing consecutive pairs of keypress events, we demonstrate the ability to recover as much as 80% of typed content. We then provide a number of short and long term mitigation strategies. In so doing, we demonstrate that access to increasingly capable sensors by applications running on mobile phones must be more carefully regulated.

Acknowledgments

This work was supported in part by the US National Science Foundation (CNS-0916047, CAREER CNS-0952959). Any opinions, findings, conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the National Science Foundation.

9. REFERENCES

[1] D. Agrawal, J. Rao, and P. Rohatgi. The EM side-channels. In *Proceedings of the Workshop on Cryptographic Hardware and Embedded Systems (CHES)*, 2002.

[2] Android Developers. Manifest.permission. `http://developer.android.com/reference/android/Manifest.permission.html`, 2010.

[3] D. Asonov and R. Agrawal. Keyboard Acoustic Emanations. In *Proceedings of the IEEE Symposium on Security and Privacy*, 2004.

[2]Assuming that the device is not placed on a highly amplifying surface.

[4] R. Ayres and K. Martinas. 120 WPM for Very Skilled Typist. *On the Reappraisal of Microeconomics: Economic Growth and Change in a Material World*, page 41, 2005.

[5] M. Backes, T. Chen, M. Durmuth, H. P. A. Lensch, and M. Welk. Tempest in a Teapot: Compromising Reflections Revisited. In *Proceedings of the IEEE Symposium on Security and Privacy (OAKLAND)*, 2009.

[6] M. Backes, M. Duermuth, S. Gerling, M. Pinkal, and C. Sporleder. Acoustic Side-Channel Attacks on Printers. In *Proceedings of the USENIX Security Symposium (SECURITY)*, 2010.

[7] M. Backes, M. Durmuth, and D. Unruh. Compromising Reflections – or – How to Read LCD Monitors Around the Corner. In *Proceedings of the IEEE Symposium on Security and Privacy (OAKLAND)*, 2008.

[8] A. Barisani and D. Bianco. Sniffing Keystrokes With Lasers and Voltmeters. In *Proceedings of Black Hat USA*, 2009.

[9] D. Barrera, H. Kayacik, P. van Oorschot, and A. Somayaji. A Methodology for Empirical Analysis of the Permission-Based Security Models and its Application to Android. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2010.

[10] Y. Berger, A. Wool, and A. Yeredor. Dictionary Attacks Using Keyboard Acoustic Emanations. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2006.

[11] R. Briol. Emanation: How to keep your data confidential. In *Symposium on Electromagnetic Security For Information Protection*, 1991.

[12] M. Brookes. Voicebox: Speech processing toolbox for matlab. `http://www.ee.ic.ac.uk/hp/staff/dmb/voicebox/voicebox.html`.

[13] L. Cai, S. Machiraju, and H. Chen. Defending Against Sensor-Sniffing Attacks on Mobile Phones. In *Proceedings of ACM SIGCOMM Workshop on Networking, Systems, Applications on Mobile Handhelds (MobiHeld)*, 2009.

[14] C. Camia. Rahm emanuel can run for chicago mayor. `http://content.usatoday.com/communities/onpolitics/post/2011/01/chicago-mayor-rahm-emanuel-/1`, January 2011.

[15] C. Camia. Rahm emanuel to fight to get onto chicago ballot. `http://content.usatoday.com/communities/onpolitics/post/2011/01/rahm-emanuel-chicago-mayor-court-ruling-/1`, January 2011.

[16] Z. Cheng. Mobile Malware: Threats and Prevention. `http://www.mcafee.com/us/local_content/white_papers/threat_center/wp__malware_r2_en.pdf`, 2007.

[17] D. Dagon, T. Martin, and T. Starner. Mobile Phones as Computing Devices: The Viruses are Coming! *IEEE Pervasive Computing*, 3(4):11–15, October - December 2004.

[18] M. Davey. Emanuel keeps campaigning, as he fights to get back on the ballot. `http://thecaucus.blogs.nytimes.com/2011/01/25/emanuel-keeps-campaigning-as-he-fights-to-get-back-on-the-ballot/?scp=8&sq=Rahm+Emanuel&st=nyt`, January 2011.

[19] M. Davey. Emanuel raises $10 million for chicago mayor's race. `http://query.nytimes.com/gst/fullpage.html?res=9B00E0DC1E3EF932A15752C0A9679D8B63&scp=10&sq=Rahm+Emanuel&st=nyt`, January 2011.

[20] M. Davey and J. Schwartz. Emanuel back on ballot; court will hear case. `https://www.nytimes.com/2011/01/26/us/politics/26rahm.html?scp=9&sq=Rahm+Emanuel&st=nyt`, January 2011.

[21] M. Egele, C. Kruegel, E. Kirda, and G. Vigna. PiOS: Detecting Privacy Leaks in iOS Applications. In *Proceedings of the ISOC Network and Distributed Systems Security (NDSS) Symposium*, 2011.

[22] Electromax International, Inc. Spy Surveillance: Laser Listening Systems. `http://www.electromax.com/laser.html`, 1998.

[23] W. Enck, P. Gilbert, B.-G. Chun, L. P. Cox, J. Jung, P. McDaniel, and A. N. Sheth. TaintDroid: An Information-Flow Tracking System for Realtime Privacy Monitoring on Smartphones. In *Proceedings of the USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, 2010.

[24] W. Enck, M. Ongtang, and P. McDaniel. On Lightweight Mobile Phone Application Certification. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2009.

[25] J. Keen. Rahm emanuel's mayoral bid hits pothole. `http://www.usatoday.com/news/politics/2011-01-25-emanuel25_ST_N.htm`, January 2011.

[26] M. G. Kuhn. Optical time-domain eavesdropping risks of CRT displays. In *Proceedings of the IEEE Symposium on Security and Privacy (OAKLAND)*, 2002.

[27] M. G. Kuhn and R. J. Anderson. Soft Tempest: Hidden Data Transmission Using Electromagnetic Emanations. *Information Hiding*, Lecture Notes in Computer Science 1525:124–142, 1998.

[28] B. C. Nalty. The War Against Trucks Aerial Interdiction in Southern Laos 1968-1972. Office of Air Force History, Washington, DC, 2005.

[29] National Security Agency. TEMPEST. `http://www.nsa.gov/public_info/files/cryptologic_spectrum/tempest.pdf`, 2007.

[30] I. S. on Subjective Measurements. Ieee recommended practices for speech quality measurements. *IEEE Transactions on Audio and Electroacoustics*, 17:227–46, 1969.

[31] J.-J. Quisquater and D. Samyde. ElectroMagnetic Analysis (EMA): Measures and Counter-Measures for Smart Cards. In *Proceedings of the International Conference on Research in Smart Cards (E-SMART)*, 2001.

[32] R. Schlegel, K. Zhang, X. Zhou, M. Intwala, A. Kapadia, and X. Wang. Soundcomber: A Stealthy and Context-Aware Sound Trojan for Smartphones. In *Proceedings of the ISOC Network and Distributed Systems Security (NDSS) Symposium*, 2011.

[33] S. W. Smith. *The Scientist and Engineer's Guide to Digital Signal Processing*. Elsevier/Newnes Publishing, 2002.

[34] P. Smulders. The Threat of Information Theft by Reception of Electromagnetic Radiation from RS-232 Cables. *Computers and Security*, 9(1):53–58, 1990.

[35] D. Stanglin. Court puts rahm emanuel back on ballot, agrees to hear case. `http://content.usatoday.com/communities/ondeadline/post/2011/01/`

rahm-emanuel-back-on-the-ballot-pending-possible-court-hearing/1, January 2011.

[36] D. Stanglin. Rahm emanuel to fight ruling booting him from mayoral ballot. http://content.usatoday.com/communities/ondeadline/post/2011/01/court-says-rahm-emanuel-ineligible-to-run-for-chicago-mayor/1, January 2011.

[37] T. Thomas. Malware on the Move. http://mobile-security-software-review.toptenreviews.com/malware-on-the-move.html, 2008.

[38] P. Traynor, C. Amrutkar, V. Rao, T. Jaeger, P. McDaniel, and T. La Porta. From Mobile Phones to Responsible Devices. *Journal of Security and Communication Networks (SCN)*, To Appear 2011.

[39] W. Van Eck. Electromagnetic Radiation from Video Display Units: An Eavesdropping Risk? *Computers and Security*, 4:269–286, 1985.

[40] M. Vuagnoux and S. Pasini. Compromising Electromagnetic Emanations from Wired and Wireless Keyboards. In *Proceedings of the USENIX Security Symposium (SECURITY)*, 2009.

[41] www.st.com. Lis331dlh. http://www.st.com/stonline/products/literature/ds/15094.pdf, July 2009.

[42] L. Zhuang, F. Zhou, and J. D. Tygar. Keyboard Acoustic Emanations Revisited. In *Proceedings of the ACM Conference on Computer and Communications Security (CCS)*, 2005.