Hearing your touch: A new acoustic side channel on smartphones

Ilia Shumailov Laurent Simon Jeff Yan Ross Anderson

Abstract

We present the first acoustic side-channel attack that recovers what users type on the virtual keyboard of their touch-screen smartphone or tablet. When a user taps the screen with a finger, the tap generates a sound wave that propagates on the screen surface and in the air. We found the device's microphone(s) can recover this wave and "hear" the finger's touch, and the wave's distortions are characteristic of the tap's location on the screen. Hence, by recording audio through the built-in microphone(s), a malicious app can infer text as the user enters it on their device. We evaluate the effectiveness of the attack with 45 participants in a real-world environment on an Android tablet and an Android smartphone. For the tablet, we recover 61% of 200 4-digit PIN-codes within 20 attempts, even if the model is not trained with the victim's data. For the smartphone, we recover 9 words of size 7–13 letters with 50 attempts in a common side-channel attack benchmark. Our results suggest that it not always sufficient to rely on isolation mechanisms such as TrustZone to protect user input. We propose and discuss hardware, operating-system and application-level mechanisms to block this attack more effectively. Mobile devices may need a richer capability model, a more user-friendly notification system for sensor usage and a more thorough evaluation of the information leaked by the underlying hardware.

1 Introduction

An extensive body of research has shown how to exploit built-in smartphone sensors to extracts users' personal information through side channel attacks; see [1, 2, 5, 6, 14, 25, 30, 38, 40, 43, 44, 51, 52] and [17].

A comprehensive survey of such attacks [16] pointed out that side-channel attacks using microphones are less well studied. Acoustic attacks against *physical* keyboards have received some attention; see [3, 23, 32, 50, 54, 55] and [19]. Each key has different physical characteristics or defects which enable classification. Acoustic attacks on *touchscreen soft* keyboard are inherently more difficult, since each finger tap happens on the same surface. Some active attacks were explored by [17]; the previous work on passive acoustic attack on virtual keyboards is by [43], who used the audio for tap detection, not for classification.

In this paper, we present the first fully passive acoustic side channel attack against *touch-screen virtual keyboards*. When a user enters text on the device's touchscreen, the taps generate a sound wave. The device's microphones can recover the tap and correlate it with the keystroke entered by a victim. The spying app may have been installed by the victim herself, or by someone else, or perhaps the attacker gave the device to the victim with the

app pre-installed – there are several companies offering such services, such as mSpy [13]. We also assume the app has microphone access. Many apps ask for this permission and most of us blindly accept the list of demanded permissions anyway [22].

The only previous work we could find on passive acoustic attacks against virtual keyboards is by Narain et al. [37], who claimed that stereo microphones are more effective than other sensors. From their evaluation, however, it is hard to infer the accuracy of their attack, and they assumed that the keyboard reports when a tap happens, which is not in general the case.

We improved on this early work by overcoming a number of challenges, including extraction of relevant features, reliable detection of finger taps in noisy environments, and re-synchronization of raw signal data (section 2.1). To make the attack stealthier and more practical, our spying app is trained offline; so there is no need to train the model with the victim's data. The only constraint is that the attacker to has train the model with the same smartphone or tablet. The whole attack pipeline is presented on figure 2 on page 4.

Through a study conducted with 45 participants, we show that an attacker can successfully perform PIN inference attacks (section 3 on page 8). We recover 31 out of 50 4-digit PINs in 20 attempts. We also show that keyboard inference attacks are feasible. For example, out of 27 passwords, we recover 7 words on the phone and 19 on the tablet within 10 attempts. This illustrates the hazards of reasoning about smartphone sandboxing given the complexity of modern platforms, as well as the need for a more realistic threat model for modern hardware.

We discuss possible countermeasures in section 4 on page 15. We propose a number of steps that app authors can take to protect against such attacks, even in the absence of better operating-system support. We then discuss more effective measures that could be built into the operating system or hardware.

We contribute the following:

- The first purely acoustic side channel attack against a smartphone *virtual keyboard*;
- Proof that one can perform accurate time-difference-of-arrival calculations between two microphones on the same phone, revealing where along the axes between two microphones the tap is happening;
- An extensive study of PIN and word inference with 45 people on tablets and smartphones;
- A comparison of our attack with previous work.

2 Background

2.1 Tap Detection

When a user taps the screen of a smartphone, sound waves start propagating not only through the air but also through the device itself. The screen behaves as a plate that is fixed on the sides to the rest of the smartphone body. Once the screen starts vibrating the vibration patterns can be picked up, and when observed with multiple microphones

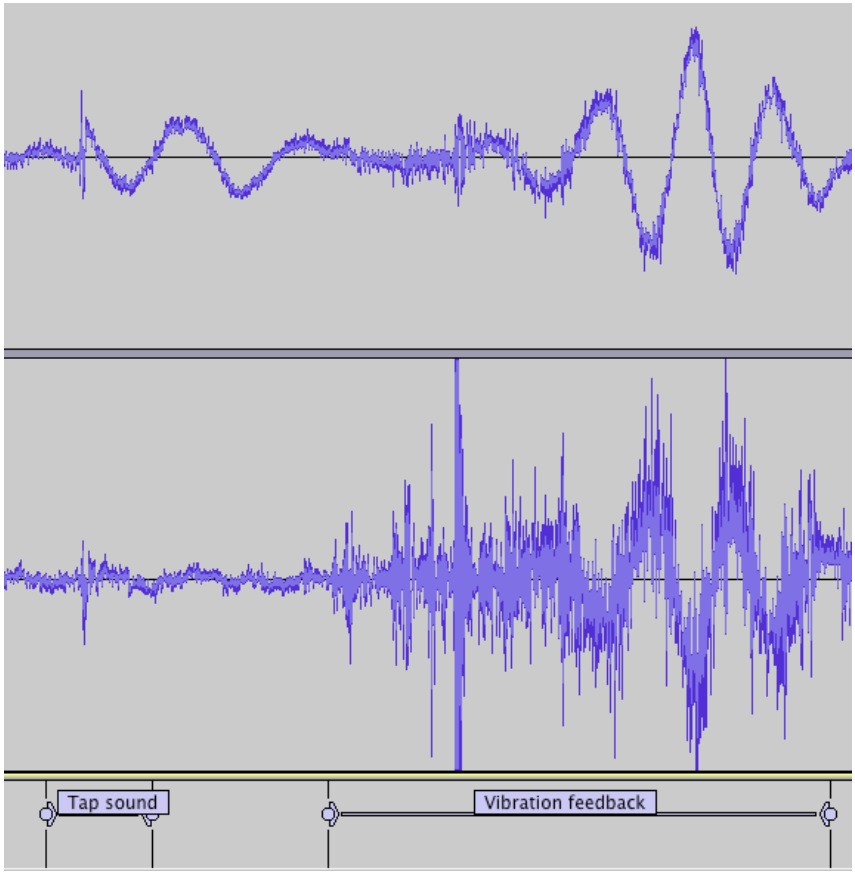


Figure 1: Press on character 'f' with a vibration feedback.

the patterns can in theory be distinguished, as they are unique (up the symmetry in case of two microphones).

In practice, however, the sampling rate of modern smartphones imposes limits. For example the Nexus 5's screen is made of Gorilla Glass 3 [20] in which the speed of sound is estimated at 4154.44 m/s; but the built-in microphones can only sample every 0.0226 ms, so they sample the energy propagating through the glass every 0.094 meters.

The acoustic data captured from a microphone is noisy. The original paper by Asonov and Agrawal that developed attacks against physical keyboards used FFT energy-based thresholding to deal with this [3], and a similar approach was adopted by many of the follow-up papers [9, 55, 54]. This worked quite well in a lab environment, especially since the keyboards they attacked were mechanical and made a lot of noise. However, the real world has a varying signal-to-noise ratio and energy thresholding simply causes a lot of false positives. Narain et al. used gyroscope data to detect key taps [37]; this is much better then energy thresholding, but still works rather poorly to detect the actual beginning of the tap, given the delays introduced by the OS. We have observed that the timestamps returned by the Android `elapsedRealtimeNanos` API on a Nexus 5 running Android 5.1.1 are anywhere around 8,000-15,000 samples away from the moment the tap actually happened. In a real-world attack on a target who enters the same PIN a lot of times with two hands, the time difference between taps is swamped by the OS latency. Simon and Anderson understood this, and used vibration feedback to locate the start of the tap [43]; but vibration can be suppressed by a banking app.

We therefore set out to solve the problem of purely acoustic tap detection, and observed that the taps have a very distinctive pattern of propagation through the screen and air –

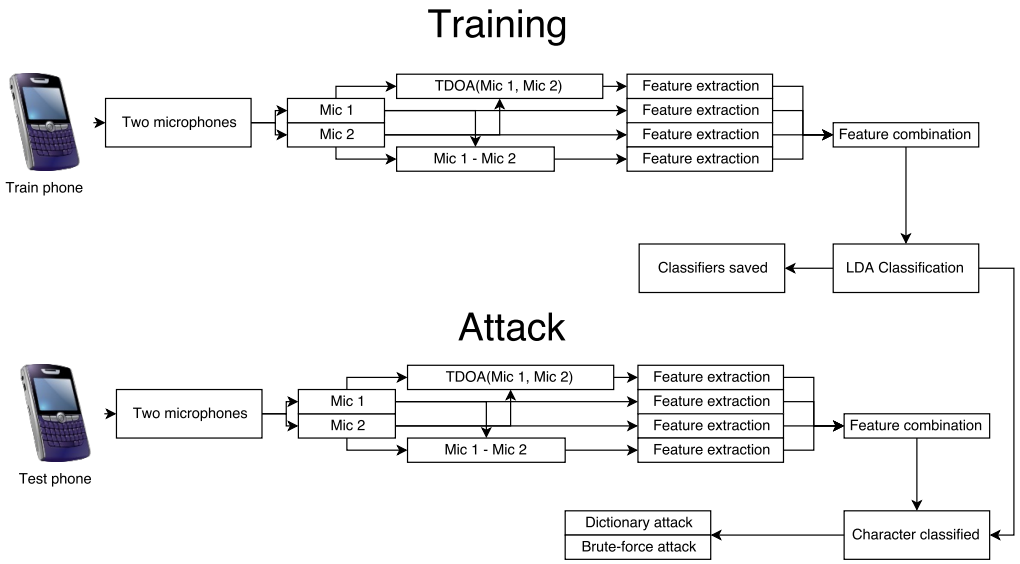


Figure 2: Attack pipeline.

one that is similar for different participants. In particular, we found that high frequencies are observed first for a few samples, followed by a long period of low frequencies. We still had to deal overlapping signals when the subject typed quickly, but found that both sound and vibration feedback are very distinctive, so it is possible to disambiguate them by subtracting the feedback when the cross-correlation is the highest with the model of the feedback sound.

2.2 Time Difference Of Arrival

When we have multiple microphones, we can calculate the time difference between the reception of the signal by the bottom one and the top one. The theoretical accuracy of time difference of arrival for a Nexus 5 smartphone sampling audio at 44.1 kHz with two microphones is shown in figure 6 on page 7 – the locations for the Nexus 5 are presented in figure 3 on the next page, and for the Nexus 9 in figure 5 on page 7. This corresponds to samples taken every 8 mm on the screen. The colours correspond to areas where the delay between signal reception by the bottom and the top microphone is constant. Theoretically, we should be able to distinguish where along the Y-axis the tap is happening when operating in portrait orientation and along the X-axis in landscape orientation.

There are many known methods to compute the time difference of arrival, such as CC, GCC-PHAT, GCC-SCOT, GCC-Roth, GCC-ML, GCC-HBII [7, 29], ASDF [27, 39], LMS [21, 45], AED [8, 26, 46], thresholding [53], ATDC [18] and combination techniques [34]. Each one of these was designed to process particular types of signal with a specific signal-to-noise ratio (SNR). We implemented all of them to see which works best.

We found that none of them worked reliably on their own. There are several reasons for this. First, the microphone hardware is not ideal and there is some sampling rate skew, so the samples can have a delay between them. Second, the environment is noisy and its noise can be difficult to model. Third, we observed that participants often blocked a microphone with their hands when typing.

After experimenting with several participants' data, we found that preprocessing the data

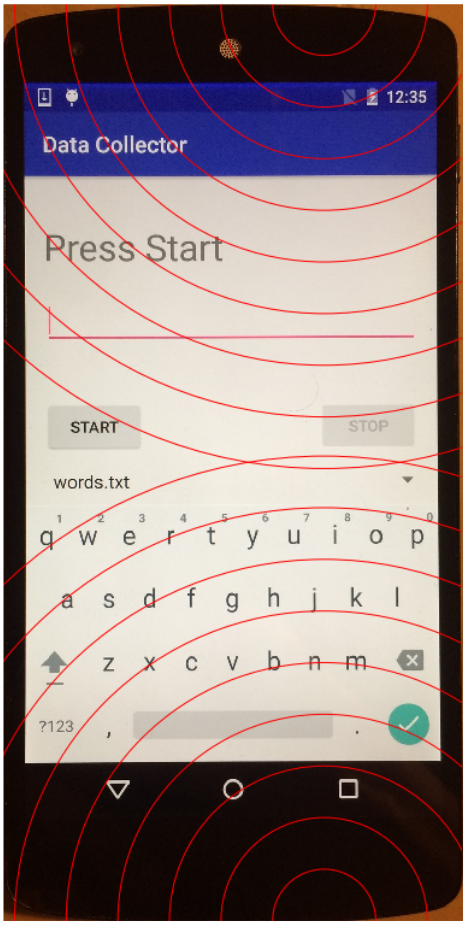


Figure 3: The location of the two Nexus 5 microphones, where the length between the subsequent radiuses is the distance traveled by a soundwave through air during one sampling period.

greatly improves the accuracy – we could use cross-correlation with a signal that is passed through a band-pass filter to achieve reliable results. We found that for each tap there is an energy peak at the frequency range of 1,300Hz – 1,700Hz for both our test devices. So, we decided to use a first-order bandpass Butterworth filter on the signals from both microphones before calculating the time delay. As the time difference is characteristic of the screen location of a user tap, we use it as a feature to input to a classifier. But we also use additional features, as explained next.

2.3 Acoustics of a tap

The tap itself is rather short, with an initial burst of about 60 samples at a frequency of about 8000–8500Hz, followed by 4000–4400 Hz, followed by a 60–70 Hz wave for about 1,500 samples (figure 4 on the next page). Having explored different combinations of features and data-set sizes we found that most of the tap information is actually contained in the first 128 samples – in contrast to physical keyboards, where most of the useful information was contained in the release movement of the button.

But with that many samples, we could not efficiently use conventional spectral audio features, such as spectral contrast [28]. We have attempted to over-sample the signal to get sub-sample values for the signal in the given bands, but that did not really help.

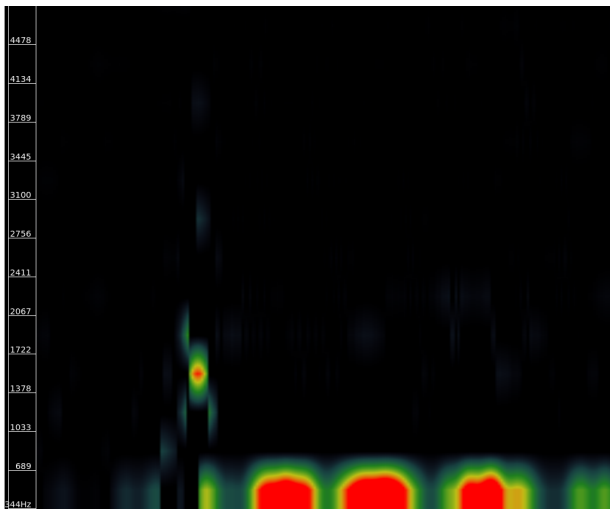


Figure 4: The short-time Fourier transform breakdown of an averaged tap sound.

Popular papers in the field of acoustics-based side-channel attacks usually use Mel Frequency Cepstrum Coefficients (MFCC) [36] for feature extraction, starting with one of the early attacks on a physical keyboard [55]. However, MFCC was designed to replicate the human ear, and humans are not good at distinguishing tapping sounds on a screen. So MFCC is likely not optimal for tap classification. Quefrency (also known as Cepstrum), on the other hand, makes use of all the frequencies by calculating the Inverse Fast Fourier transform (IFFT) of their logarithm, but does not weight them according to the human ear's characteristics.

So we use raw quefrency calculated on the first 128 samples of audio data (after the tap) as an additional feature to our classifier. In fact, we combined quefrencies for the signals coming from one and more microphones. Throughout this paper, we will refer to three different feature sets: 1) Top – refers to quefrency of a signal from the top microphone; 2) Botm – refers to quefrency of a signal from the bottom microphone; 3) Top+Botm – refers to Top features, concatenated to Botm features, concatenated to the estimated delay between the two signals and to quefrency of the difference between the two signals.

For classification, we use Linear Discriminant Analysis (LDA), more specifically the one based on Singular Value Decomposition (SVD), as it showed the best results in our tests.

In the evaluation section (section 3 on page 8) we will present F1 scores for our attack. These are a better reference of test accuracy than a simple percentage of correctly guessed samples. F1 takes into the account both the precision (number of correct positive results divided by the number of all positive results) and the recall (number of correct positive results divided by the number of positive results). It is calculated as

$$F1 = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.$$

The macro F1 presented in this paper takes the average of the precision and recall reported for different labels.

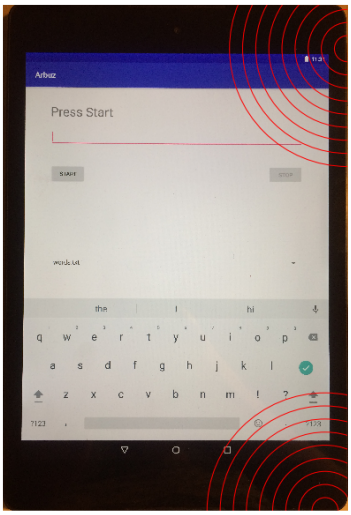


Figure 5: The location of the two Nexus 9 microphones, where the length between the subsequent radii is the distance travelled by a sound wave through air during one sampling period.

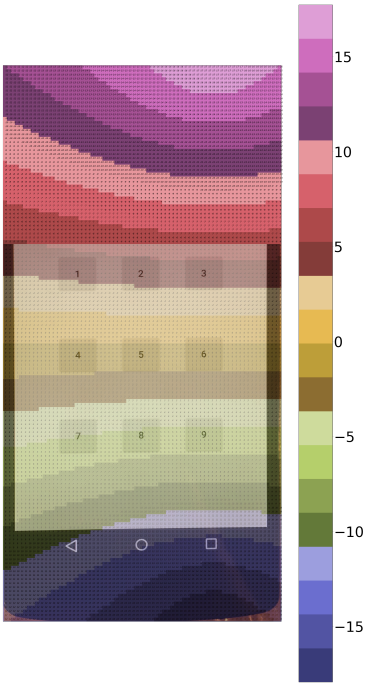


Figure 6: The location of the two Nexus 5 microphones, where the length between the radii around each microphone is the distance travelled by a soundwave during one sampling period.

3 Evaluation

3.1 Experimental design

We developed an Android application where participants entered letters and words into an input field, or digits into a standard PIN pad. In each case, the app displayed the text or digits to be entered by participants. The app worked in both landscape and portrait orientation; and had support for feedback mechanisms (sound, vibration). While the user performed the requested task, the application collected audio through the built-in microphones.

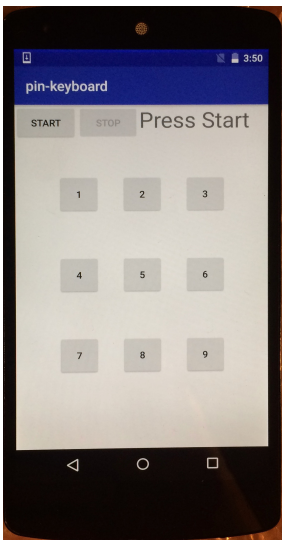
As Android 5.1.1 prohibits the use of feedback mechanisms for PIN input, we tested them only for text input. We ran PIN-entry experiments only on the smartphone since the PIN pad has the same size on both tablet and smartphone. The PIN-code entry experiments were conducted only in portrait orientation because smartphones do not allow PIN input in landscape orientation.

We conducted the experiments outside of the lab environment in order to simulate real-life conditions more closely, thereby improving the validity of the study. Participants performed the task at the university in three different places: *1)* in a common room, where people chat and occasionally a coffee machine makes some Latte Macchiato along with loud sounds; *2)* in a reading room where people either type on a computer or speak in a semi-loud voice; and *3)* in the library where people are silent, but there are a lot of clicking sounds from nearby laptops. All three places had windows open, letting in ambient noise.

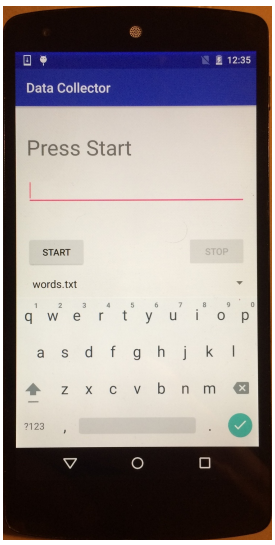
As the speed of sound in air depends on its temperature, the data were only collected indoors during the day with air temperature ranging from 22–25 degrees Celsius and with no other user-started processes running in the background. We used two LG Nexus 5 phones and a Nexus 9 tablet (all running Android 5.1.1). The Nexus 5 measurements are 137.84mm × 69.17mm × 8.59mm and the Nexus 9 228.2mm × 153.7mm × 8mm. Both are fairly slim devices, with all of the internal components located relatively close to each other. Both devices have two microphones that can be accessed using the Android SDK. The main Nexus 5 microphone is located at the bottom of the device, the secondary one at the top. The Nexus 9 tablet is different, with one microphone at the bottom, and another one on the right side.

We used two LG Nexus 5 phones and a Nexus 9 tablet, which have a standard audio sampling rate. They both ran Android Lollipop which represented 35% of devices at the time of the study and still represents 20% in 2019. We suspect devices with higher sampling rates and better microphones may improve the attack, but we leave that for future work.

Due to the novelty of the attack itself, and the lack of a standard way of benchmarking side-channel attacks, the interpretation of our results is not straightforward. We want the reader to take away from the evaluation the following: 1) the attack on a small virtual keyboard performs about as well as to attacks on big physical keyboards; 2) the attack yields similar performance to previous virtual keyboard attacks based on other sensors such as the accelerometers and gyros, despite the microphone being much noisier.



(a) PIN-entry application



(b) Text-entry application

Figure 7: Application developed for the experiments

3.2 Data Collection

We conducted four separate experiments. In the first, 10 participants were asked to press each of the nine digits (1 to 9) 10 times. The order of occurrence of the digits was randomised. In the second experiment, we asked 10 other participants to type 200 unique 4-digit PINs. In the third, we asked another group to type letters. The order of occurrence was also randomised. In the fourth experiment, participants were asked to type five-character words from the NPS chat corpus [11]. A summary of all experiments for letter and word input is presented in table 1. The applications developed for data collection are presented in figure 7.

Table 1: Description of the experiments conducted.

what is being typed-in	orientation	feedback
Nexus 5 smartphone		
200 most popular words (NPS)	portrait	no
200 most popular words (NPS)	landscape	no
100 most popular words (NPS)	landscape	sound
100 most popular words (NPS)	landscape	vibration
26 alphabet characters ×10 each	portrait	no
26 alphabet characters ×10 each	landscape	no
26 alphabet characters ×10 each	landscape	sound
26 alphabet characters ×10 each	landscape	vibration
9 digits ×10 each	portrait	no
4-digit PINS ×10 each	portrait	no
Nexus 9 tablet		
100 most popular words (NPS)	portrait	no
100 most popular words (NPS)	landscape	no
26 alphabet characters ×10 each	portrait	no
26 alphabet characters ×10 each	landscape	no

In total, we obtained data from 45 people, 21 female and 24 male. Two people refused to reveal their age; for the other 43 the mean age was 24.4 years and the range was between

21 and 32 years. Overall, we collected about 30 hours of audio.

For each audio recording, we extracted the taps as explained in section 2 on page 2. We then used two methods to test the accuracy of the attack. In the first, we used 70% of the data for training the model (the "train" set), and the remaining 30% for testing (the "test" set). Those datasets may contain data from the same users. In the second method, we performed the leave-one-subject-out (LOSO) evaluation. Basically, the model is trained on N-1 participants and tested on 1; the results are averaged over all participants. In our threat model, we made an assumption of availability of records of 100 taps per character/digit prior to the attack. In the evaluation, this was implemented through random under-sampling prior to training of the training set – we randomly dropped the samples from the original training dataset to contain 100 samples. In the cross-validation (LOSO on top of k-fold), classifiers were run five times each and the results were averaged. Every k-fold run was performed on newly under-sampled data with new classifiers. Similarly, in the word prediction tasks, the training data was under-sampled every time. However, we also regenerated the word sequences every time from each of the participants of the experiments.

3.3 Ethical considerations

This study went through the standard ethics review process in our institution and closely followed the departmental guidelines for data collection from human participants. The data were collected on a voluntary basis with no payment to participants. It was necessary to ensure that the participants were not harmed in any way and that no sensitive information was leaked. For these purposes, a number of measures were taken.

First, each of the volunteers was required to read and sign the consent form, which described the experiment. Consent was also obtained verbally; we explained the aim of the study to the participants to ensure they understood what type of data they were handing to us.

Second, no Personally Identifiable information (PII) were collected in our study, like name, date of birth or email address. The volunteers were only asked to type artificial data during the experiments which were randomly generated by us.

Third, we made sure that the data collected were impossible to trace back to each individual participant.

Fourth, the data were stored on a secured computer, to which only the researchers in charge of analysis have access.

3.4 Single digit inference

The train, test, and subject-independent (i.e. LOSO) accuracies for single digit classification on the first attempt are presented in table 2 on the facing page. With a single microphone, we correctly predict F1=34 $\pm$ 5 and F1=55 $\pm$ 5 of the PINs for the bottom (Botm) and the top microphones respectively – compared to $1/9 = 11\%$ (although not a valid comparison between percentages and F1, it should still give an intuition about performance) for a random guess. Using two microphones, the accuracy improves to F1=64 $\pm$ 6.

Table 2: Single digit classification accuracy on the Nexus 5 smartphone. Macro F1 is reported. Additional paper reported accuracies are presented.

Feature-set	train	test	LOSO
Botm	34 ± 4	34 ± 5	31 ± 22
Top	56 ± 8	55 ± 5	56 ± 30
Top + Botm	63 ± 8	$\mathbf{64 \pm 6}$	$\mathbf{66 \pm 37}$
Simon and Anderson [43]	–	26%	-
Sun et al. [49]	–	38%	-
Xu et al. [51]	–	40%	-
Cai et al. [15]	–	50%	-
Our best double	–	$\mathbf{65\% \pm 3}$	-

For the subject-independent case (LOSO), the standard deviation is higher. In the worst case, with the Botm microphone only, digits are predicted slightly better than a random guess; but in the best case almost half of the digits are correctly predicted on the first attempt. Results for the Top microphone are higher still, twice as good in the worst case and 85% in the best case scenario. With two microphones, we correctly predict 3 times better than a random guess in the worst case; and 100% of the digits in the best case.

When a digit is predicted incorrectly, the classifier is quick to rectify it (the guesses are ordered by the confidence value associated with a prediction yielded by the classifier). For example, with two microphones – as shown in figure 8 on the next page – 30% of incorrect predictions are fixed after the first attempt with a further 20% fixed at the second attempt. The rate of fixing then flattens out. This means that, in three attempts, the classifier correctly predicts more than 80% of the digits.

3.5 PIN inference

The results for 4-digit PIN predictions are shown in figure 9 on page 13. We consider 150 random PINs from 20 people as explained in section 3.2 on page 9. With a single microphone, we can recover 54% of PINs after 10 attempts. We compare our attack with previous work in table 3 on page 14. It performs at least as well as some of them, and often better. For example, with two microphones, we recover 91/150 4-digit PINs in just 20 attempts – without knowing anything about PIN distribution or the timing between subsequent taps [12, 10].

3.6 Letter & Word inference

In this section, we present the prediction results for the 26 letters of the English alphabet for both phone and tablet (table 4 on page 14). On the phone, we outperform a random guess by a factor of three if we access a single microphone for both LOSO and test. Audio from the top microphone yields better results than audio from the bottom microphone. We are not sure why this happens, but we suspect it may have to do with the way people hold the phone. Surprisingly, we found that portrait mode yields better results than landscape mode for the smartphone, while for the tablet, there is no noticeable difference between portrait and landscape modes. We believe this can be explained by the way

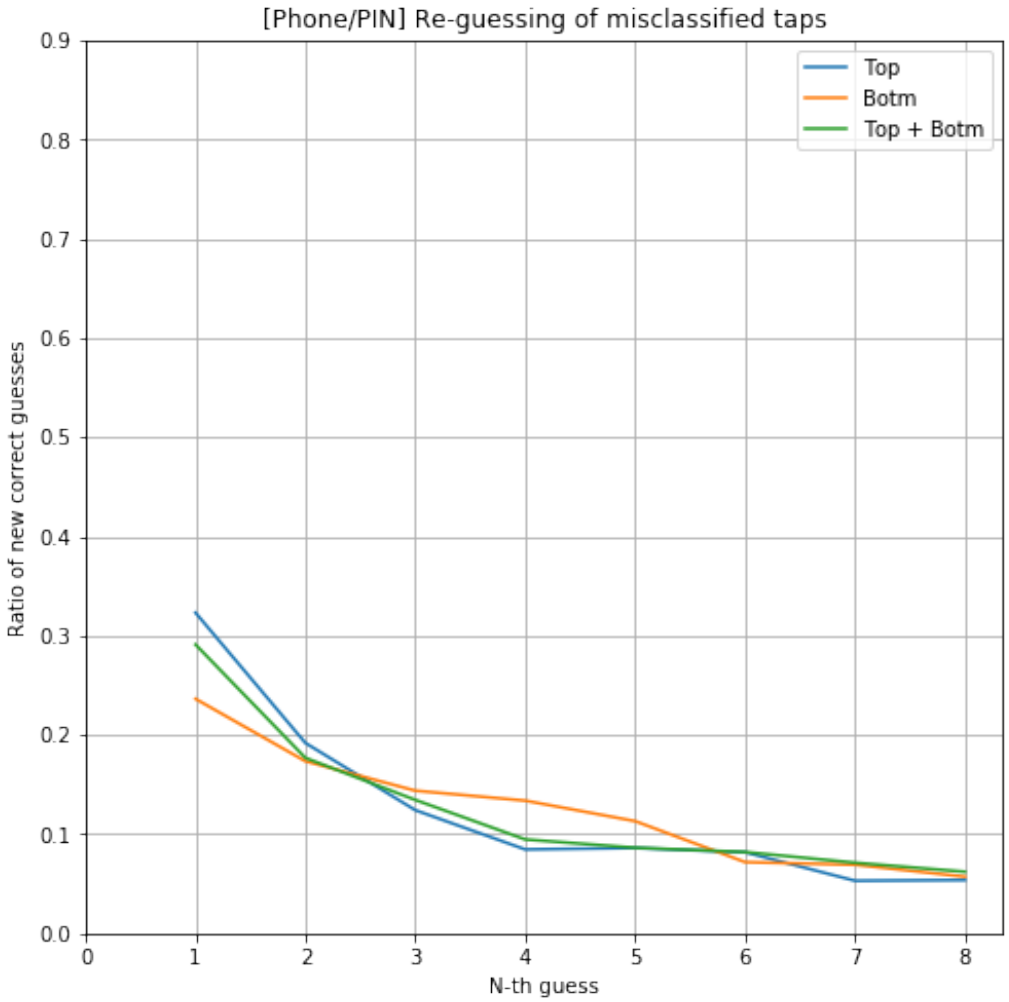


Figure 8: The percentage of fixed errors with every new attempt.

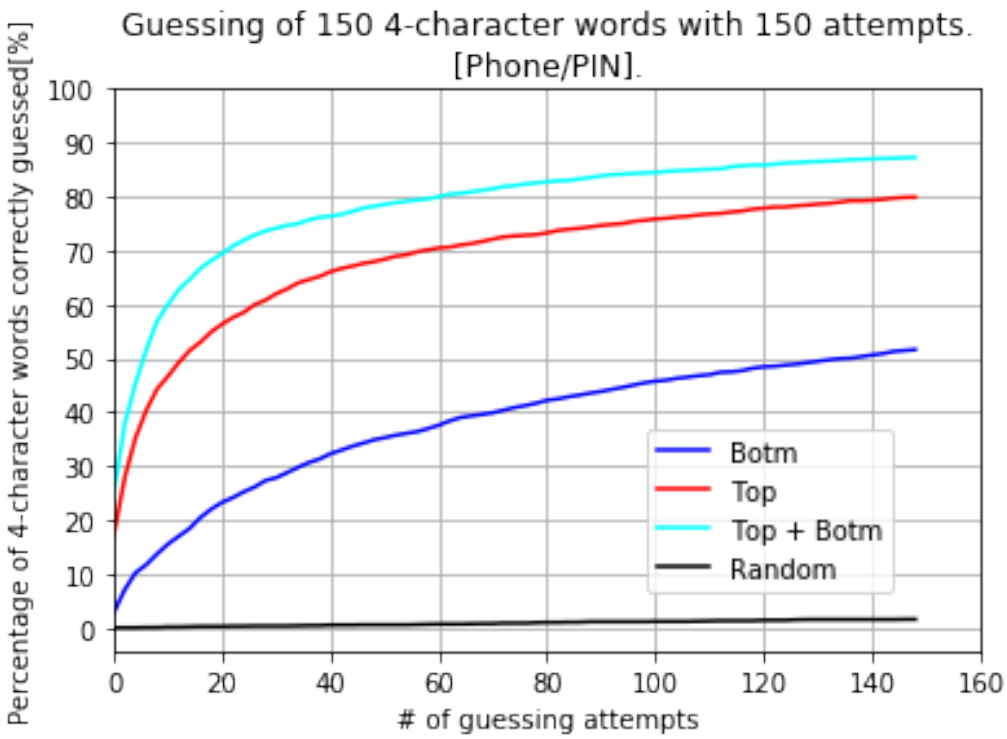


Figure 9: Performance of the classifier trying to predict 200 random PIN-codes of size 4 collected from the Nexus 5 smartphone in landscape orientation based on the taps obtained from 20 people. The results are shown from 30 runs with the average presented, each based on the mid-performing classifier.

we type on these devices: people usually put the tablet on their lap or a table in front of them, whereas they hold the phone in their hand and this may partially block the bottom microphone. When using two microphones, we obtain the best results, with an improvement of a factor of 1.5 over a single microphone.

To compare our results to previous work, we followed the same evaluation methodology presented in [33], [9], [49], [35]: we randomly selected 27 words of length 7-13 from the corn-cob dataset and attempted to classify them. There are certain differences between our work and these papers though, for example, the training set size, the number of participants and the training procedures may vary slightly. Two of the papers evaluated attacks on physical keyboards which have higher accuracy in general. Although direct comparison is hard, this still provides a valuable baseline for our work. We present the comparison in table 5 on page 15.

Simon and Anderson's video side channel outperforms our attacks for a single microphone on the phone but shows similar results for the tablet. This suggests it is harder to perform acoustic attacks on a smartphone virtual keyboard as the distance between "buttons" is very short. For the tablet and when using two microphones, we obtain similar results as for attacks on physical keyboards.

3.7 Word inference in real-world use

The evaluation we presented in the previous section was motivated by previous papers. However, in the real world it is much harder to actually perform attacks for a number of reasons, one of which is the lack of rich enough dictionaries and text corpora.

Table 3: PIN Attack performance comparison. We report the best performing classifiers in single and double configurations.

Attack by	set size	10^{th} try	20^{th} try
Our best single	50	42%	50%
Aviv et al. [6]	50	55%	-
Our best double	50	**55**%	61%
Simon and Anderson [43]	50	61%	84%
Spreitzer [48]	50	79%	-
Shukla [42]	50	94%	-
Our best single	100	41%	49%
Simon and Anderson [43]	100	48%	58%
Our best double	100	**51**%	59%
Our best single	150	40%	48%
Simon and Anderson [43]	150	44%	53%
Our best double	150	**52**%	61%
Simon and Anderson [43]	200	40%	53%
Our best single	200	43%	48%
Our best double	200	**53**%	61%

Table 4: Single character classification accuracy for 26 classes. Macro F1 value is presented.

Feature-set	Phone			Tablet		
	Train	Test	LOSO	Train	Test	LOSO
Portrait; Botm mic	10 ± 3	11 ± 1	6 ± 7	26 ± 3	28 ± 2	22 ± 16
Portrait; Top mic	16 ± 3	17 ± 2	10 ± 9	26 ± 5	25 ± 2	20 ± 14
Portrait; both mics	23 ± 4	$\mathbf{23 \pm 3}$	$\mathbf{12 \pm 15}$	44 ± 3	$\mathbf{45 \pm 3}$	$\mathbf{37 \pm 24}$
Landscape; Botm mic	10 ± 3	11 ± 2	7 ± 8	26 ± 3	26 ± 2	21 ± 19
Landscape; Top mic	13 ± 2	16 ± 3	9 ± 14	24 ± 3	24 ± 2	19 ± 12
Landscape; both mics	20 ± 4	$\mathbf{23 \pm 3}$	$\mathbf{13 \pm 20}$	43 ± 3	$\mathbf{44 \pm 1}$	$\mathbf{37 \pm 27}$
Random attack	3 ± 1	3 ± 1	3 ± 1	3 ± 1	3 ± 1	3 ± 1

This paper, however, suggests that one can perform inference attacks in such a situation. When mistakes are made, most of them get fixed quickly, suggesting that instead of dictionary search one can focus on a subset of predictions in which the classifier is the most confident, allowing some word to be guessed correctly.

Figure 10 shows the performance of word guessing based on a different prediction policy, with guesses going to depth 5 and 10. As can be seen, the overall accuracy stayed the same, despite a massively reduced prediction space. The space was reduced from 26^4 to 10^4 and 5^4 or by 45 and 731 times respectively.

Finally, in the real world most of the words in the dictionaries are extremely rare. To reflect this in our evaluation, we used the NPS chat corpus accessed through the NLTK framework [24, 11], a real-world chat dataset. We then evaluated our attack on the most common 100 words of this dataset. To do that, we followed the approach of [43] and synthesised words randomly from the collected single tap sounds. In addition to the classifier presented in section 2 on page 2, we also trained an N-gram model using the

Table 5: 27 corn-cob words of size 7-13 benchmark. We report the best performing classifiers in single and double configurations.

Attack by	10-attempts	50-attempts
Phone/best single	21%	30%
Phone/best double	25%	34%
Marquardt et al. [35]	43%	56%
Berger et al. [9]	43%	73%
Tablet/best single	43%	55%
Liu et al. [33]	63%	82%
Sun et al. [49]	63%	93%
Tablet/best double	70%	80%

Brown text corpus available through the NLTK framework [31, 11]. More precisely, to incorporate the information from the N-gram model into the predictions, we used the following function

$$P_{combined}(pred) = P_{word}(pred) * P_{n-gram}(pred),$$

where

$$P_{word}(pred) = \prod_{l=1}^{n} p(pred_l)$$

and

$$P_{n-gram}(pred) = \prod_{l=1}^{(n-k)} p(pred_{l:l+k}).$$

Here $pred_{l:l+k}$ is the subword from character on index l to character on index $l+k$, and p is the n-gram provided probability.

The results are presented in figure 11 on the following page. Without the language model, about 6% of 4-letter words are correctly brute-forced in 20 attempts. The use of a 3-gram model [47, 41] improves those results to 10%. This is encouraging because it suggests that for real sentences, language models will help even more.

4 Countermeasures

There are a number of ways to mitigate the acoustic attack presented in this paper. These can be implemented at different levels of the software and hardware stack.

At the hardware level, a radical solution is to remove the microphone from the handset or tablet. If projects like the Google Ara [4] gain traction, this may become viable for certain users. But this will never be acceptable for everyone, as we rely more and more on voice to interact with smart devices.

One could introduce physical switches to phones, so users could turn the microphone off. But this would introduce usability issues for most of us: imagine having to physically enable the microphone every time you answer a voice call! Press-to-talk radios were progressively replaced by voice-operated microphones since the technology became available fifty years ago, and the trend in phones has been to have fewer buttons over time.

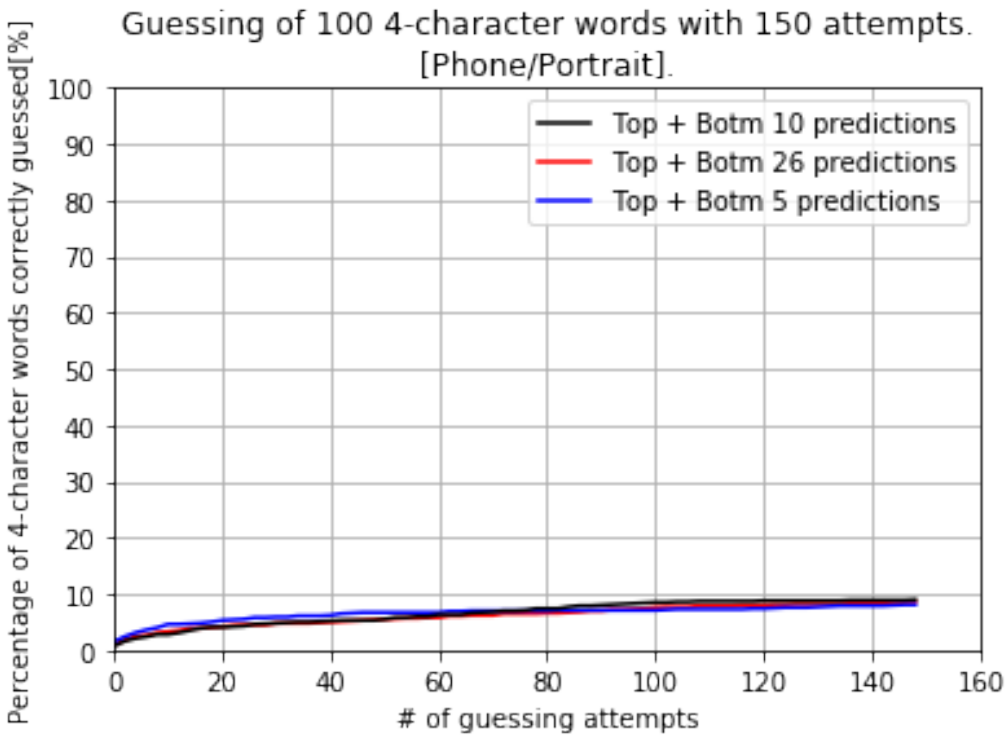


Figure 10: Performance of the classifier trying to predict 100 most common words of size 4 collected from the Nexus 5 smartphone in portrait orientation with different combination depths based on the taps obtained from 20 people. The results are shown from 30 runs with the average presented, each based on the mid-performing classifier.

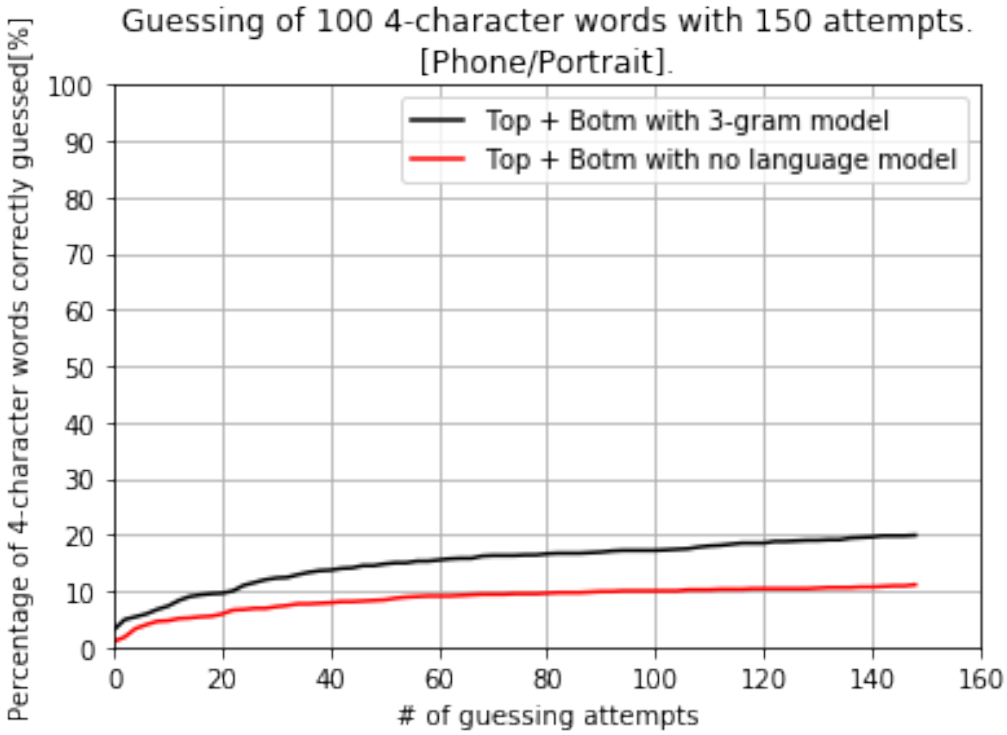


Figure 11: Performance of the classifier trying to predict 100 most popular words of size 4 collected from the Nexus 5 smartphone in portrait orientation based on the taps obtained from 20 people. The results are shown from 30 runs with the average presented, each based on the mid-performing classifier.

Many other possibilities go too strongly against the grain of commercial competition. One could lower the microphone sampling frequency, but as vendors compete for the best video recording, this is unlikely to be acceptable. An additional glass layer on top of the screen could absorb most of the finger impact, but would not appeal to vendors fighting to make devices slimmer and lighter.

The most usable approach at the level of hardware design might be to make live microphones obvious, perhaps by having a diode connected to the energy rails of the microphone itself, lighting up whenever it's in use, and forming an icon that represents a microphone. The obvious objection is that a live-mike icon would often be ignored, like most security warnings. However, it would give some value to some users, and although it would carry a small cost in hardware, something similar could be implemented in the OS for zero marginal cost.

At the operating-system level, one might also try to prohibit audio recording during data entry. But this would break many apps, for example when users type during a voice call.

A more sophisticated approach would be to offer a properly-engineered PIN entry facility, which when called by one application would temporarily blank, and/or introduce noise into, the microphone channel seen by other apps. This approach should logically be extended to other sensors that can be used to harvest PINs via side-channels such as the accelerometer, gyro and camera. If restricted explicitly to PIN entry, it might be used sufficiently sparingly that its interference with other apps might well be tolerable.

As the next step, a "secure text entry" facility could be provided for apps to solicit passwords and other authentication information. But thought would need to be given to possible interactions, for example between banking apps that insist on secure password entry and navigation apps that insist on continuous access to accelerometers and gyros. If secure text entry came to be the default for messaging apps that offer end-to-end encryption, the scale of use might start to cause problems elsewhere in the app ecosystem.

More subtle degradation should be explored, such as for the OS to introduce timing jitter, or decoy tap sounds, into the microphone data stream. This way an attacker could not reliably identify tap locations. As the taps themselves are pretty unnoticeable for humans, this should not disturb disturb applications that run in the background and collect audio with user consent.

At the application level, an app might itself introduce false tap sounds into the device, in order to jam and confuse any hostile apps that happen to be listening. Such tactical jamming is a low-cost countermeasure and as far as we can see may be the most feasible mitigation for a developer of (say) a banking app to deploy today.

Overall, we favour two approaches: 1) reporting to the user which sensors are on; 2) introducing a secure attention sequence – a new high-priority mode in the phone interaction pipeline for passwords or other sensitive text entry, in which all sensors except the touch screen are blocked. The first is fairly straightforward but raises issues of usability, liability and compliance, while the second requires more engineering but might provide higher assurance. Until these (or other mitigations) are implemented in the platform, app developers should consider the use of tactical jamming if PIN theft via side channels is ever deployed at scale.

5 Limitations

In our threat model, we used only the microphones to classify and detect finger taps, and we limited ourselves to 26 English characters and 9 digits. Once we move from password capture to the inference of more general text input, there are other characters to consider, such as the backspace, the space buttons, etc. These complicate the machine-learning models and will lead to lower prediction accuracy. But attackers could use other sensors too, such as the accelerometers and gyroscopes. So it would be of interest to evaluate the attacks that are possible using multi-sensor side channels, and measure the performance of different classifiers with different alphabets and in different SNR settings.

In our attack, we managed to recover letters in the subject-independent model). It is unclear why the attack generalises well; presumably people's fingers are more similar than their voices are. More research on this might be useful. Future research could also investigate the transferability of the attack to different types of device. However, the trend towards more and better-quality microphones suggests that attacks will only get better.

6 Related Work

This paper fits into the area of side-channel attacks on smartphones. In particular, we present a novel acoustic side-channel. The microphone is not typically considered to leak information about the locations of taps on the screen, and acoustic attacks are relatively unexplored [16].

Attacks on physical keyboards are not new. Asonov and Agrawal were among the first to demonstrate acoustic attacks on mechanical keyboards [3]. Their work was further improved by Zhuang et al. [55] who showed that an attacker can deduce text even in unsupervised operation, and by Compagno et al. who showed how to infer text based on keyboard acoustics acquired through Skype [19].

But what about virtual keyboards on the screens of smartphones or tablets? Cai and Chen showed that an attacker can use motion sensors to recover PIN codes entered by a user with a 70% recovery rate [15]. Their threat model is very similar to ours – they assume a malicious app residing on the victim's phone with sensor access.

So is it enough for a security-conscious user to stop apps using the motion sensors? Simon and Anderson reported an attack using camera movement to measure handset orientation, instead of the accelerometers and gyros, and using the microphone for the tap detection [43]. This let them recover PINs with reasonable probability.

Our attack continues this line of research. It's not enough to deny apps permission to use motion sensors and the camera; to stop PIN leakage completely you have to deny access to the microphones too. That will be infeasible for most users and many apps.

Our attack exploits the double microphones found in many modern smartphones. These have been used previously to perform acoustic attacks, but not on the smartphone itself. For example, Liu et al. presented a system for keyboard press inference using the multiple microphones of a nearby smartphone [32]. Similarly, Zhu et al. present several context-free attacks on keyboard emanations using only time difference of arrival information [54], and show that multiple smartphones can increase the accuracy of the results they obtain.

7 Conclusion

We present a new acoustic side-channel attack on touch-screen devices such as smartphones and tablet computers. We use the device's own microphones to infer text entry from the sounds made by finger taps on the screen.

We showed that the attack can successfully recover PIN codes, individual letters and whole words. On a Nexus 5 smartphone in portrait orientation, the attack was able to recover most PINs and some words; specifically, 146 of 200 4-digit PINs after 10 attempts, and 8 out of 27 random words from the corn-cob dataset of size 7-13 after 20 attempts.

We also showed that it is possible to achieve high accuracy in a subject-independent setting, so one can attack users' smartphones remotely with a classifier pre-trained on other peoples' devices.

Our attack achieved an even better accuracy on tablets than on smartphones. For example, on a tablet, we recovered not just some words but most words; specifically, 19 words out of 27 after 10 attempts.

To assess the practical threat, when guessing words from a set of words commonly used in text messaging, we were able to recover 80 words out of 200 after 20 attempts. Furthermore, insights into the way mistakes are made allowed us to develop a model for an effective attack that would use a language model on both letter and word levels.

In conclusion, we have shown a new acoustic side-channel attack on smartphones and tablets, and described how to exploit it effectively.

Finally, we have discussed what mitigations are possible, both for application writers now, and for future versions of mobile phone operating systems.

References

[1] Ahmed Al-Haiqi, Mahamod Ismail, and Rosdiadee Nordin. On the best sensor for keystrokes inference attack on android. *Procedia Technology*, 11:pp. 989 – 995, 2013. 4th International Conference on Electrical Engineering and Informatics, ICEEI 2013.

[2] Panagiotis Andriotis, Theo Tryfonas, George Oikonomou, and Can Yildiz. A pilot study on the security of pattern screen-lock methods and soft side channel attacks. In *Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '13, pages 1–6, New York, NY, USA, 2013. ACM.

[3] Dmitri Asonov and Rakesh Agrawal. Keyboard acoustic emanations. In *Security and Privacy, 2004. Proceedings. 2004 IEEE Symposium on*, pages 3–11. IEEE, 2004.

[4] Google ATAP. Ara, 2014.

[5] Adam J Aviv. *Side channels enabled by smartphone interaction*. PhD thesis, Pennsylvania State Univ., University Park, PA, USA, 2012.

[6] Adam J. Aviv, Benjamin Sapp, Matt Blaze, and Jonathan M. Smith. Practicality of accelerometer side channels on smartphones. In *Proceedings of the 28th Annual Computer Security Applications Conference*, ACSAC '12, pages 41–50, New York, NY, USA, 2012. ACM.

[7] L. Bedini, S. Fossi, and R. Reggiannini. Generalised crosscorrelator with data-estimated weighting function: a simulation analysis. *Communications, Radar and Signal Processing, IEE Proceedings F*, 133(2):pp. 195–198, April 1986.

[8] Jacob Benesty. Adaptive eigenvalue decomposition algorithm for passive acoustic source localization. *The Journal of the Acoustical Society of America*, 107(1):pp. 384–391, 2000.

[9] Yigael Berger, Avishai Wool, and Arie Yeredor. Dictionary attacks using keyboard acoustic emanations. In *Proceedings of the 13th ACM Conference on Computer and Communications Security*, CCS '06, pages 245–254, New York, NY, USA, 2006. ACM.

[10] Nick Berry. http://www.datagenetics.com/blog/september32012/, 2012. Datagenetics blog. Accessed: 01/06/2017.

[11] Steven Bird, Ewan Klein, and Edward Loper. *Natural Language Processing with Python*. O'Reilly Media, Inc., 1st edition, 2009.

[12] Joseph Bonneau, Sren Preibusch, and Ross Anderson. A birthday present every eleven wallets? the security of customer-chosen banking pins. In *FC 12: The 16 th International Conference on Financial Cryptography and Data Security*, 2012.

[13] Thomas Brewster. This software company may be helping people illegally spy on their spouses. *Forbes*, Feb 22 2017.

[14] Liang Cai and Hao Chen. Touchlogger: Inferring keystrokes on touch screen from smartphone motion. In *Proceedings of the 6th USENIX Conference on Hot Topics in Security*, HotSec'11, pages 9–9, Berkeley, CA, USA, 2011. USENIX Association.

[15] Liang Cai and Hao Chen. On the practicality of motion based keystroke inference attack. In *Proceedings of the 5th International Conference on Trust and Trustworthy Computing*, TRUST'12, pages 273–290, Berlin, Heidelberg, 2012. Springer-Verlag.

[16] Liang Cai, Sridhar Machiraju, and Hao Chen. Defending against sensor-sniffing attacks on mobile phones. In *Proceedings of the 1st ACM Workshop on Networking, Systems, and Applications for Mobile Handhelds*, MobiHeld '09, pages 31–36, New York, NY, USA, 2009. ACM.

[17] Peng Cheng, Ibrahim Ethem Bagci, Utz Roedig, and Jeff Yan. Sonarsnoop: Active acoustic side-channel attacks. *CoRR*, abs/1808.10250, 2018.

[18] H. H. Chiang and C. L. Nikias. A new method for adaptive time delay estimation for non-gaussian signals. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 38(2):pp. 209–219, Feb 1990.

[19] Alberto Compagno, Mauro Conti, Daniele Lain, and Gene Tsudik. Don't skype & type!: Acoustic eavesdropping in voice-over-ip. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security*, ASIA CCS '17, pages 703–715, New York, NY, USA, 2017. ACM.

[20] Incorporated Corning. Specification: Corning gorilla glass 3 with native damage resistance, 2015.

[21] P. Feintuch, N. Bershad, and F. Reed. Time delay estimation using the lms adaptive filter–dynamic behavior. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 29(3):pp. 571–576, Jun 1981.

[22] Adrienne Porter Felt, Elizabeth Ha, Serge Egelman, Ariel Haney, Erika Chin, and David Wagner. Android permissions: User attention, comprehension, and behavior. In *Proceedings of the Eighth Symposium on Usable Privacy and Security*, SOUPS '12, pages 1–14, New York, NY, USA, 2012. ACM.

[23] Au Hiu Yan Fiona. Keyboard acoustic triangulation attack. *The Chinese University Of Hong Kong: bachelors thesis*, 2006.

[24] Eric Forsyth, Jane Lin, and Craig Martell. Nps internet chatroom conversations, release 1.0, 2010.

[25] Gabriel Goller and Georg Sigl. Side channel attacks on smartphones and embedded devices using standard radio equipment. In Stefan Mangard and Axel Y. Poschmann, editors, *Constructive Side-Channel Analysis and Secure Design: 6th International Workshop, COSADE 2015, Berlin, Germany, April 13-14, 2015. Revised Selected Papers*, pages 255–270. 2015.

[26] Y. Huang, J. Benesty, and G. W. Elko. Adaptive eigenvalue decomposition algorithm for real time acoustic source localization system. In *1999 IEEE International Conference on Acoustics, Speech, and Signal Processing. Proceedings. ICASSP99 (Cat. No.99CH36258)*, volume 2, pages 937–940, Mar 1999.

[27] G. Jacovitti and G. Scarano. Discrete time techniques for time delay estimation. *IEEE Transactions on Signal Processing*, 41(2):pp. 525–533, Feb 1993.

[28] Dan-Ning Jiang, Lie Lu, Hong-Jiang Zhang, Jian-Hua Tao, and Lian-Hong Cai. Music type classification by spectral contrast feature. In *Proceedings. IEEE International Conference on Multimedia and Expo*, volume 1, pages 113–116 vol.1, 2002.

[29] C. Knapp and G. Carter. The generalized correlation method for estimation of time delay. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 24(4):pp. 320–327, Aug 1976.

[30] Mengyuan Li, Yan Meng, Junyi Liu, Haojin Zhu, Xiaohui Liang, Yao Liu, and Na Ruan. When csi meets public wifi: Inferring your mobile phone password via wifi signals. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, pages 1068–1079, New York, NY, USA, 2016. ACM.

[31] Anne Lindebjerg. Brown dataset: list of samples, 1997.

[32] Jian Liu, Yan Wang, Gorkem Kar, Yingying Chen, Jie Yang, and Marco Gruteser. Snooping keystrokes with mm-level audio ranging on a single phone. In *Proceedings of the 21st Annual International Conference on Mobile Computing and Networking*, MobiCom '15, pages 142–154, New York, NY, USA, 2015. ACM.

[33] Xiangyu Liu, Zhe Zhou, Wenrui Diao, Zhou Li, and Kehuan Zhang. When good becomes evil: Keystroke inference with smartwatch. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1273–1285. ACM, 2015.

[34] Radu-Sebastian Marinescu, Andi Buzo, Horia Cucu, and Corneliu Burileanu. Applying the accumulation of cross-power spectrum technique for traditional generalized cross-correlation time delay estimation. *International Journal On Advances in Telecommunications*, 6(3):pp. 98–108, 2013.

[35] Philip Marquardt, Arunabh Verma, Henry Carter, and Patrick Traynor. (sp)iphone: Decoding vibrations from nearby keyboards using mobile phone accelerometers. In *Proceedings of the 18th ACM Conference on Computer and Communications Security*, CCS '11, pages 551–562, New York, NY, USA, 2011. ACM.

[36] Paul Mermelstein. Distance measures for speech recognition, psychological and instrumental. *Pattern recognition and artificial intelligence*, 116:pp. 374–388.

[37] Sashank Narain, Amirali Sanatinia, and Guevara Noubir. Single-stroke language-agnostic keylogging using stereo-microphones and domain specific machine learning. In *Proceedings of the 2014 ACM conference on Security and privacy in wireless & mobile networks*, pages 201–212. ACM, 2014.

[38] Matei Negulescu and Joanna McGrenere. Grip change as an information side channel for mobile touch interaction. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 1519–1522, New York, NY, USA, 2015. ACM.

[39] M. Ross, H. Shaffer, A. Cohen, R. Freudberg, and H. Manley. Average magnitude difference function pitch extractor. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 22(5):pp. 353–362, Oct 1974.

[40] A. Sarkisyan, R. Debbiny, and A. Nahapetian. Wristsnoop: Smartphone pins prediction using smartwatch motion sensors. In *2015 IEEE International Workshop on Information Forensics and Security (WIFS)*, pages 1–6, Nov 2015.

[41] C. E. Shannon. Prediction and entropy of printed english. *The Bell System Technical Journal*, 30(1):pp. 50–64, Jan 1951.

[42] Diksha Shukla, Rajesh Kumar, Abdul Serwadda, and Vir V. Phoha. Beware, your hands reveal your secrets! In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 904–917, New York, NY, USA, 2014. ACM.

[43] Laurent Simon and Ross Anderson. Pin skimmer: Inferring pins through the camera and microphone. In *Proceedings of the Third ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*, SPSM '13, pages 67–78, New York, NY, USA, 2013. ACM.

[44] Laurent Simon, Wenduan Xu, and Ross Anderson. Don't interrupt me while i type: Inferring text entered through gesture typing on android keyboards. *PoPETs*, 2016(3):pp. 136–154, 2016.

[45] H. C. So and P. C. Ching. Comparative study of five lms-based adaptive time delay estimators. *IEE Proceedings - Radar, Sonar and Navigation*, 148(1):pp. 9–15, Feb 2001.

[46] Victor Solo and Xuan Kong. *Adaptive Signal Processing Algorithms: Stability and Performance*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1995.

[47] Raymond J Solomonoff. An inductive inference machine. In *IRE Convention Record, Section on Information Theory*, volume 2, pages 56–62, 1957.

[48] Raphael Spreitzer. Pin skimming: Exploiting the ambient-light sensor in mobile devices. In *Proceedings of the 4th ACM Workshop on Security and Privacy in Smartphones & Mobile Devices*, SPSM '14, pages 51–62, New York, NY, USA, 2014. ACM.

[49] Jingchao Sun, Xiaocong Jin, Yimin Chen, Jinxue Zhang, Yanchao Zhang, and Rui Zhang. Visible: Video-assisted keystroke inference from tablet backside motion. 2016.

[50] Ranade Vinayak, Smith Jeremy, and Switala Ben. Acoustic side channel attack on atm keypads. 2009.

[51] Zhi Xu, Kun Bai, and Sencun Zhu. Taplogger: Inferring user inputs on smartphone touchscreens using on-board motion sensors. In *Proceedings of the Fifth ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WISEC '12, pages 113–124, New York, NY, USA, 2012. ACM.

[52] Lin Yan, Yao Guo, Xiangqun Chen, and Hong Mei. A study on power side channels on mobile devices. In *Proceedings of the 7th Asia-Pacific Symposium on Internetware*, Internetware '15, pages 30–38, New York, NY, USA, 2015. ACM.

[53] D. Ye, J. Y. Lu, X. J. Zhu, and H. Lin. Generalized cross correlation time delay estimation based on improved wavelet threshold function. In *2016 Sixth International Conference on Instrumentation Measurement, Computer, Communication and Control (IMCCC)*, pages 629–633, July 2016.

[54] Tong Zhu, Qiang Ma, Shanfeng Zhang, and Yunhao Liu. Context-free attacks using keyboard acoustic emanations. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, pages 453–464, New York, NY, USA, 2014. ACM.

[55] Li Zhuang, Feng Zhou, and J. D. Tygar. Keyboard acoustic emanations revisited. *ACM Trans. Inf. Syst. Secur.*, 13(1):pp. 1–26, November 2009.