Verification of Keyboard Acoustics Authentication on Laptops and Smartphones Using WebRTC

Christian Tietz, Eric Klieme, Lukas Behrendt, Pawel Böning, Leonard Marschke and Christoph Meinel
Hasso Plattner Institute
University of Potsdam
Germany
Email: {christian.tietz, eric.klieme, christoph.meinel}@hpi.de
{lukas.behrendt, pawel.boening, leonard.marschke}@student.hpi.de

***Abstract*—This paper evaluates authentication based on sounds emitted while typing on a laptop's keyboard only with the help of the integrated microphones and a smartphone placed nearby. The sound samples of 26 individuals were recorded in a novel real-world scenario using only web technologies. The data of laptop and the smartphone was compared based on existing approaches. They can reach Equal-Error-Rates (EERs) of 11.2% on the laptop data and 9.3% on the smartphone data.**

***Index Terms*—Smartphone, Behavioral Authentication, Keyboard Dynamics, Keyboard Acoustics**

I. Introduction

Nowadays, authentication is critical for many services. Without reliable authentication, it is impossible to offer services like online banking or remote desktop applications. Common authentication schemes like passwords, however, are not sufficient due to the fact that they often only authenticate the user at the beginning of a session. For instance, following a password authentication, unauthorized access is still possible if the user changes during the use of the service [1], [2].

In biometric authentication a user's physical characteristics are analyzed to verify whether they are who they claim to be. Measurements of physiological characteristics like fingerprints or face geometry, but also behavioral attributes, can be used to determine a unique set of features to distinguish between different individuals. These measurements can be acquired continuously and in parallel to other activities, which is useful for services like continuous desktop authentication [3].

In the field of mobile devices, there already exist authentication schemes that leverage behavioral characteristics. These schemes mostly concentrate on features retrieved by sensors such as accelerometers or magnetometers, which recognize users by their movement. Such movements include walking with the device in one's pocket [4], [5] and handling the device during a phone call [5], [6] or while typing a text message [5], [7], [8].

While these techniques are promising if the user is actively moving and using the device, they do not ensure continuous authentication during stationary activities, especially when the device is not being used. One such distinct scenario is working on a computer while the mobile device like the smartphone is positioned next to the keyboard [9].

Applying the typing sound authentication to the smartphone gives the phone the possibility to continuously authenticate their owner while he is typing. This can reduce the number of unlock which is a nice feature because a smartphone is unlocked multiple times a day [10]. Approaches that use typing sounds already exists like the work of Roth et al. [11]. They showed that it is possible to recognize users based on their typing sound with an Equal-Error-Rate (EER) of 11% in a lab environment using a web cam connected to a computer.

This paper investigates the authentication performance of smartphone microphones and compares them against the microphones that are integrated into a laptop using the algorithms of Roth et al. as basis.

In summary, this paper's main contributions are the following:

- We record typing data of 26 individuals in a static and dynamic text setting in an unsupervised real-world scenario including only their own notebook and own smartphone. We are first to show that standard WebRTC implementations of different browsers can be used to record audio data but several challenges still remain (see section III).
- We provide a normalization process and present that the influence of different environments and differences in the specifications of microphones can be reduced (see subsection IV-A).
- We enhance the authentication procedure of Roth et al. [11] by a majority voting (see section IV)
- We compare the collected laptop and smartphone data using the authentication approach and an attacker performing random attacks. The results show an EER of 11.2% for laptop data and also show that smartphones can also achieve similar results with an EER of 9.3% (see V).

II. Prior Work

This section presents related work for authentication via keystroke sounds and there are three different areas of prior approaches are relevant for our research: *keyboard dynamics*,

978-1-7281-3949-4/19/$31.00 ©2019 IEEE

keyboard acoustics, and the recent effort to combine these two areas.

A. Keyboard Dynamics

Keyboard dynamics are researched for over 30 years with the first research known to us being conducted by Umphress and Williams [12] in 1985. They introduced digraph latency (the time difference between two keystrokes of two specific letters) as a feature. In their experiment, they measure the digraph latencies by logging every keystroke. They then authenticate the user by comparing the mean latency for each digraph with a profile, achieving a *false acceptance rate (FAR)* of 6% and a *false rejection rate (FRR)* of 12%.

Up until today, most research in this field is based on digraph latency, aiming to improve upon it. This is done by deriving additional features from the digraph latency and by evaluating new classification algorithms.

In their survey paper Banerjee and Woodard [13] distinguish between four categories of classification algorithms that have been used in this area of research:

- Statistical algorithms, e.g. t-tests [12]
- Neural networks, e.g. Fuzzy ARTMAP [14]
- Pattern recognition, e.g. support vector machines [15]
- Search heuristics and combination of algorithms, e.g. genetic algorithms [16]

Bergadano et al. [17] first attempted to use trigraph latencies, which describe the time between the first and the last of three keystrokes, as well as 4-graphs and 6-graphs. They found out that trigraphs have more discriminatory power than digraphs, while longer n-graphs are less stable and repeated too seldom, leading to lower accuracies. Multiple other teams have since used trigraphs in their research [18]–[20].

In 1997 Obaidat and Sadoun [14] separate keystroke latency into the time between pressing and releasing the first key, and the time between releasing the first key and pressing the second key. In recent literature, these times are often called *dwell time* and *flight time*, respectively, and can be used as separate features [13]. Robinson et al. [21] found out that classifiers trained on dwell time alone perform better than those trained on latency alone.

While keyboard dynamics has not entered the mainstream as of yet, there are commercial solutions such as *TypingDNA*[1] and *KeyTrac*[2] that use it as an authentication method.

B. Keyboard Acoustics

This field deals with attacking user input to break user/password schemes and with eavesdropping on private user communications. The acoustic emanations of keyboards are used to infer the original text typed by the user. The majority of methods employed in current research assume that different keys emit distinct sounds while being pressed and released, thus it is believed to be possible to reconstruct the typed input from the sound of the typing alone. Problems that current approaches still have are the presence of keystroke overlap (touch typing) and background noise in practice [22].

Asonov and Agrawal [23] were the first to conduct an attack using a neural network trained to distinguish different single keystroke sounds and briefly discuss the issues of recording distance, typing style and different keyboards. They achieve a recognition rate of 79%, although that rate drops significantly below 50% when the keyboard and/or the typist change. Zhuang et al. [24] expand on that approach by additionally including language models and reach accuracies of up to 92%.

Different methods for feature extraction are used throughout the research literature. Plain audio is used by Kelly [22] to directly represent the key samples, leading to a very large number of features which require a dimensionality reduction to be feasible. Furthermore, the fast Fourier transform (FFT) and Mel-frequency cepstral coefficients (MFCC) of the audio samples infer a feature set to be used for further processing, also discussed by Kelly [22], Zhuang et al. [24], Roth et al. [11] and Compagno et al. [25]

C. Authentication via Keystroke Sound

As the most recent of the three fields, authentication via keystroke sound combines the two previously mentioned fields and aims to assess user identities via sounds emitted by keyboards. Roth et al. [11], [26] are to our knowledge the first to explore this approach. They present a three-step algorithm that consists of temporal segmentation/feature extraction, keystroke clustering and scoring. The segmentation takes place by detecting keystroke windows in the FFT of the audio sample, which later get transformed into their respective MFCCs. The resulting feature sets are then clustered into a virtual alphabet. For each detected keystroke the virtual label, detection time and MFCC feature set are transformed to form four scores: digraph statistic, histogram of digraphs, histogram of virtual letters and intra-letter distance. A fusion function is lastly used to form a similarity score for a given set of subjects. This approach achieved an Equal Error Rate (EER) of 11% authenticating a user from a database of 50 subjects [11].

Even more recently, based on the work of Roth et al., the works of Pleva et al. [27]–[29] show a slightly different approach which combines audio data with conventional key timing data and uses Hidden Markov Models. Their best acoustic model achieved an EER of 8.99% on a database of 50 subjects [29].

Previous methods employ static texts and homogeneous lab environments for training and testing which may be subject to instabilities in practice. While Roth et al.'s approach performed quite well in the environment of only one microphone and one keyboard, we decided to verify and extend the approach of Roth et al. in a real-world setting by:

- applying the authentication approach to mobile devices
- allowing multiple keyboard types
- allowing multiple microphone types
- allowing a level of background noise during recordings

[1]https://www.typingdna.com
[2]https://www.keytrac.net

- using built in APIs of modern web browsers and thus require no specific hardware / software for recording typing data

III. Data Collection

Comparing typing audio from laptop and smartphones requires data from both sources. To our knowledge, there are no publicly available datasets which fits our needs. Therefore, we built a web application to collect the data in two steps.

The first step was a supervised study to get knowledge about how users behave during the study. With this information, we optimized the study for the second, unsupervised stage in order to provide a more intuitive user experience and to have a ground truth e.g. regarding phone positioning. The second stage is the source of data that is used for the comparison in the evaluation. We collected data of 26 different users and used the same web application for both recording stages.

This section gives information about the web application architecture, presents the study procedure and the results of the supervised and unsupervised instantiations of the study.

A. Web Application Architecture

The general web application architecture makes use of the frontend/backend paradigm in combination with a RESTful API. Our architecture is depicted in Figure 1. The web application only requires WebRTC[3] which is supported by all major browsers.

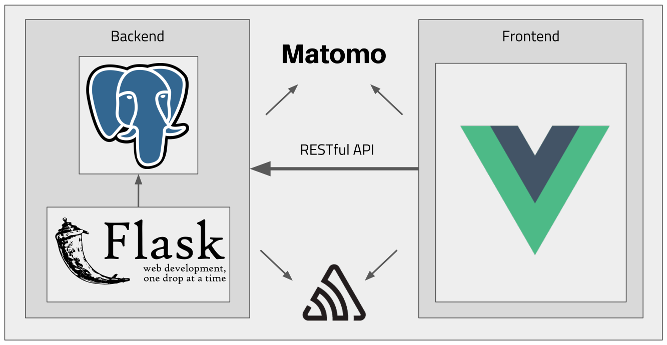


Fig. 1: Backend architecture: We use HTML5, JavaScript and Vue.js 2 for the frontend and Flask and PostgreSQL for the backend. Sentry is used to capture errors during development and during the execution of the study. Matomo keeps track of the time users need for different steps.

During the experiment, the following data is collected:

- Sound stream for computer microphone
- Sound stream for smartphone microphone
- Keystroke timings
- Transition timings between experiment stages

The audio data is sent to the server in one-second chunks as soon as it is recorded. By uploading data during the study, the time at the end of the study for uploading was significantly reduced. This approach also minimizes the client's RAM usage, as audio data that has been uploaded can be safely deleted. Sound files are encoded to the lossless WAV format and are sent as raw data to the REST API. Data on timings and study completion is transmitted as a JSON object.

We decided against compressing our raw audio files in order to avoid lossy compression which could potentially reduce the discriminating power of the audio features. A lossless compression was not possible because not all browsers support the same lossless codecs.

B. Study Procedure

To collect data, we used the presented web application to record typing sounds on laptops and smartphones. The application was available over a simple URL and, therefore, users do not need to install programs on their devices.

Each user who wants to participate in the study has to go through four phases: user agreement, pairing, static text and dynamic text.

When participating in the study, the user has to read and accept the user agreement first. In this agreement, we inform the users about the purpose of the study, what data we will collect and how we will use the data and that it will be stored and processed anonymously.

The second step checks the availability of the built-in microphone of the laptop and sets up the pairing with the smartphone. For this, the user opens the web app on his smartphone and enters the code which he sees in the web application on his laptop. We use a simple token-based synchronization mechanism to implement this. We asked the users to put their phone on the side of their laptop as shown in Figure 2. We observed in our daily life that many people put their

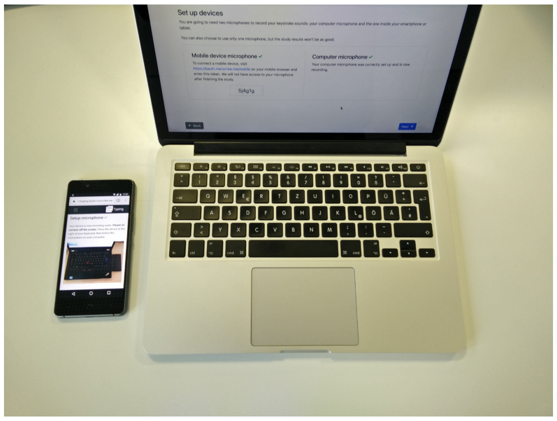


Fig. 2: The optimal recording setup with a laptop and a smartphone on one side

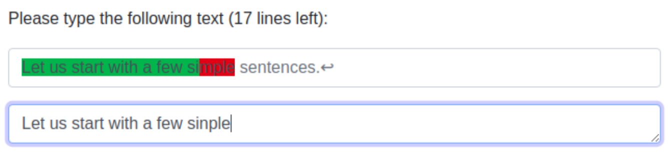
smartphones close to they laptops and keyboards. So, they can immediatly see when they got a call or message and from who it is. That is, why we asked them to put their smartphone next to the laptop.

The pairing with a smartphone can be skipped. Then only the audio from the laptop is recorded.

The last two steps are doing the actual recording. In the third step, the web application presents a series of lines of static

[3]https://webrtc.org

text to the user that he is supposed to type in an input field. We enforce the correctness of the typed text: if a user makes a typing error, he need to correct it before he can proceed (see Figure 3). The static text consists of normal sentences, pangrams (sentences that contain every letter of the alphabet) and simple mathematical equations.


Please type the following text (17 lines left):

Let us start with a few simple sentences.↩

Let us start with a few sinple

Fig. 3: The static text UI. The text is displayed above and has to be entered into the input field below. Correctly entered text is highlighted in green and the text after an error occurred is highlighted in red. When all text is highlighted in green, pressing the enter button proceeds with the next line.

In the fourth and last step, the user is required to write a dynamic text of at least 500 characters. Users are free to type any text they want, with the instruction text suggesting "Describe your current environment" as an example prompt. We also do not enforce the correctness anymore.

The idea to make a static text and free text part is taken from the work of Roth et al. [11]. We use the static part to make sure that each user has typed every letter in the alphabet and the numbers at least once. The free text is used to record the user specific typing style.

C. Supervised Pilot Study

After developing the study, it was tested with five participants. The main goal was to improve the usability of the user interface as well as to find unknown bugs caused by different browsers and operating systems. During these tests we encountered an unexpectedly high number of problems regarding the audio recording:

- Implementations of the Web Audio API vary across different browsers. We, therefore, had to implement multiple user agent checks and different methods to check for recording errors.
- A bug in the Chrome browser[4] leads to distorted audio recordings. We were able to mitigate this problem in our implementation so that it is only an issue when using very low-performance devices.
- When recording audio with a smartphone, it would often go into standby, which would prevent the audio recording callbacks from being executed and eventually lead to a crash. We, therefore, adopted the library *NoSleep.js*[5], which keeps smartphones from going into standby by simulating the playback of a video.

[4]https://bugs.chromium.org/p/chromium/issues/detail?id=327649
[5]https://github.com/richtr/NoSleep.js

D. Unsupervised Experiment Study

The second study is an unsupervised variant of the first one and provided the data for the following evaluation. We advertised the study through our social networks and provide the link to the web application. The participants went through the study by themselves only following the textual instructions. The time to finish the study for each participant was 8 to 10 minutes. At the end of each study, we collected some meta data of the person including age, gender, experience with the keyboard, environment, writing hours and keyboard types through the web application.

After finishing the data collection, we needed to filter the resulting data for potential misuse and outliers because of the unsupervised nature of the study. This was done manually. In the end, we collected usable data from 26 participants where eight users only provide laptop audio and the other 18 both laptop and smartphone audio. From the meta data, we get the following information about our 26 participants:

- Age: 17 between 20 and 30 years, 7 between 30 and 40 years and 2 over 40 years.
- Gender: 22 males and 4 females
- keyboard types: 17 chiclet keyboards, 5 mechanical keyboards and 4 rubber dome keyboards.

IV. Authentication Procedure

This section describes the authentication algorithm that comprises three steps: audio normalization, preprocessing (extracting keystrokes) and the authentication (scoring) algorithm.

A. Audio Normalization

When analyzing the audio files recorded in our public experiment, we noticed that different environment parameters had a strong impact on the audio levels in the recording. These parameters include the noise level, the loudness of the keystrokes, the used microphone and the software settings of the browser and operating system.

This audio normalization was designed to deal with the following problems:

Comparable gain: Due to different browser and operating system settings, audio files had different gain levels varying from very low to very high.

Automatic gain control: We experienced situations where automatic gain control built into the user's operating system changed the gain level while the user completed the study, which cannot be turned off by browser APIs. We, therefore, created a window-based gain control which flattens these changes in the resulting audio file. To suppress high background noise in silent windows, we built another window function that suppresses high volume gains for a window if nearby windows are similarly silent.

Unified codomain of audio data: For an easier analysis process we convert the codomain of the single audio amplitudes to a codomain of $[-1;1]$.

To address these problems, we developed a process to normalize the audio files. At first, a gain per chunk gc is calculated (see Equation 1). This is a factor to multiply the real value with. The gc is computed on every chunk n of the size cs, where $value_x$ is the raw data value at one sample point at position x. Also this converts the codomain of the real data to $[-1; 1]$ for every chunk. Furthermore, this gain would amplify each chunk to have a maximum amplitude of 1 and/or -1.

$$gc_n = \frac{1}{max_{i=cs\cdot n}^{cs\cdot(n+1)} value_i} \tag{1}$$

In the next step, an arithmetic mean of nearby chunks is computed (see Equation 2). The window around a chunk is given by size p where p is the extent of the window to the right (into the "future") and to the left (into the "past") around the current chunk. So, every chunk n gets an own arithmetic mean value.

$$gm_n = \frac{\sum_{m=n-p}^{n+p} gc_m}{2\cdot p + 1} \tag{2}$$

Afterwards, we calculate the harmonized gain gh (see Equation 3) by checking, whether the chunk gain gc is smaller than the mean gain gm multiplied by a static amplify factor af, which is set to $af = 1.3$ in our case.

$$gh_n = \begin{cases} gc_n & \text{if } gm(n)\cdot af < gc(n) \\ gm_n & \text{otherwise} \end{cases} \tag{3}$$

To prevent hard gain jumps between chunks, we adapt the gain of consecutive chunks. To apply the adaption, a new gain modification function m is introduced (see Equation 6). When one chunk has a higher gain than the other one, the higher one gets modified to create a smooth transition between different gain levels.

$$m_1(n, cs, i) = \cos(\frac{2\cdot i}{cs\cdot\pi})\cdot(gh_n - gh_{n-1}) \tag{4}$$

$$m_2(n, cs, i) = \cos(\frac{2\cdot i}{cs\cdot\pi} + \pi)\cdot(gh_n - gh_{n+1}) \tag{5}$$

$$m(n, cs, i) = \begin{cases} m_1(n, cs, i) & \text{if } |gh_{n-1}| < |gh_n| \wedge i < \frac{cs}{2} \\ m_2(n, cs, i) & \text{if } |gh_{n+1}| < |gh_n| \wedge i > \frac{cs}{2} \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

Lastly, the final gain gf is calculated (see Equation 7), which is applied as a factor to the raw value of the sound file.

$$gf(n, cs, i) = gh_n \cdot m(n, cs, i) \tag{7}$$

Although most audio files were recorded as stereo files, we use only one channel because the differences between the channels were marginal.

B. Preprocessing

After the normalization, the preprocessing takes places: the keystroke extraction. Zhuang et al. [24] observed that the energy of keystroke sounds is concentrated in the range between 400 Hz and 12 kHz. Roth et al. [11] used this to extract the keystrokes from the sound data and we follow this approach. They do it by applying a sliding window to the sound stream, computing a FFT on theses windows and summarizing the magnitudes in the ranges of 400 Hz to 12k Hz. In this new signal, timestamps bigger than a threshold θ give the starting points of keystrokes. For each detected keystroke (starting point + 40 ms average keystroke duration), the MFCC (Mel-Filter Cepstral Coefficients) are computed. The result is a 256-dimensional feature vector for each keystroke [11] [24].

C. Authentication Algorithm

As authentication algorithm, we also use the approach of Roth et al. [11] and summarize the main parts.

The authentication algorithm computes a score between two audio signals and if the score is above a threshold these two audio signals are considered to belong to the same user (see Algorithm 1). To compute the similarity score for two audio signals, four metrics are computed first: digraph statistic, histogram of digraphs, histogram of virtual letters and intra-letter distance. Then, a score between the same metric of both signals is computed which results in four scores, one for each metric comparison. The computation of the similarity score is a fusion (projection using linear discriminate analysis (LDA)) of these four previous scores.

To get the digraphs and letters, a virtual alphabet is used that is computed in the training phase. The keystrokes from the training data are clustered using the K-Means algorithm. The number of clusters (K) represents the size of the virtual alphabet. For the digraphs, each detected keystroke is assigned a virtual letter from the alphabet and two successive letters form a digraph. From all possible digraphs in the training data, only the top D percent are used. In the evaluation, we test different values for K and D to find out which combination works best for our data (see section V).

Algorithm 1 Authentication Algorithm of Roth et al.

```
Require: Trained virtual alphabet and threshold T
  function AUTH($S$, $S'$)
      $score \leftarrow computeSimilarityScore(S, S')$
      if $score >= T$ then
          return 1
      else
          return 0
      end if
  end function
```

We extended the authentication algorithm to support majority voting based on the number of reference samples. Each user has multiple samples of typing sounds. Roth et al. chose one sample as the reference sample and the rest as the test samples. In the majority voting approach, we can use multiple reference samples per user. Each test sample is compared to all reference samples. In our experiments, we used three and five reference samples and, therefore, the number scores that needs to be higher than a threshold are two or three, respectively. The algorithm is shown in Algorithm 2.

Algorithm 2 Authentication Algorithm with Majority Voting

```
function AUTH2($LS$, $S'$)    ▷ LS is list of reference samples
    $count \leftarrow 0$
    for all S in LS do
        $count \leftarrow count + AUTH(S, S')$
    end for
    if $count > lengthOf(LS)/2$ then
        return 1
    else
        return 0
    end if
end function
```

V. Evaluation

In this section, we discuss the evaluation of our collected data from page 3 and consider audio from laptop and smartphone as two separated datasets. The evaluation procedure for each dataset is as follows:

- Data was split into chunks of 10, 30 and 60 seconds
- Configurations for each chunk size were generated with different virtual alphabet size (K) and different percentage of top digraphs (D)
- A train test split of 30-70 was applied
- For each configuration, a virtual alphabet and the projection parameters of the LDA were trained on the train data
- Pairwise scores were computed on the test data
- EERs on the scores were computed using no majority voting, majority of three and majority of five

For testing and computing the EER we assume an attacker performing random attacks [30] by using all using all the other samples from the test set as attacker samples. For the evaluation, we look at the results of the laptop data first, followed by the result of the smartphone and we finish with a comparison.

A. Laptop Data

The best result on the laptop dataset is achieved with a chunk size of 60 seconds, a virtual alphabet size of 60, using 90% of the top digraphs from the training phase and a majority voting of five. The EER for this configuration is 11.2% as shown in Table 3c. We can also observe that the authentication performance increases with larger chunk sizes. The best EER for the chunk sizes 10, 20 and 30 are 11.7%, 11.5% and 11.2%, respectively (see Tables 1b, 2c and 3c). The next observation is that the best results for each configuration are achieved for a virtual alphabet size of 60 which corresponds to Roth et al.'s findings [11]. The D parameter varies. We can also see that majority voting improves the result. For chunk size of 10 seconds, without majority voting we have a best EER of 17% and with majority voting 11.7% (see Tables 1a and 1b). When comparing both, the majority of three and majority of five, we can observe that majority of five is often the better option (see Tables 2b, 2c and Tables 3b, 3c). For the chunk size of 10 seconds, the EER does not improve and worsens from 11.7% to 12.6% when using majority voting of five (see Table 1).

B. Smartphone Data

The best result on the smartphone dataset is achieved with a chunk size 60 seconds, a virtual alphabet size of 60, using 70% of the top digraphs from the training phase and a majority voting of five. The EER for this configuration is 9.3% as shown in Table 6c. The results improve with larger chunk sizes reaching EERs of 19.1%, 14.7% and 9.3% for chunk sizes 10, 30 and 60 seconds, respectively (see Tables 4c, 5c and 6c). The Tables 5 and 6 show that the smartphone data achieves best results using a larger virtual alphabet of 60 letters. For the chunk size of 10 seconds, a virtual alphabet of 20 letters achieved the best results (see table 4) The results for each chunk size get better when majority voting is used. In the Tables 4, 5 and 6 the EERs improves with using majority voting from 21.8% to 19.1%, 17.8% to 14.7% and 14.7% to 9.3, respectively.

C. Comparison

Overall, there are some general points that are similar for both, laptop and smartphone data. First, the results get better with larger chunk sizes. Second, majority voting improves the result and majority of five gives better results than majority of three. Lastly, the best results are achieved using the larger virtual alphabet size of 60.

When comparing the results of laptop and smartphone, we also can see differences. Laptop data results are far better than the results on the smartphone data for smaller chunk sizes of 10 and 30 seconds. While the laptop data achieves EERs of 11.7% and 11.4%, the EERs of the smartphone data are only at 19.1% and 14.7% (see Tables 1, 2, 4 and 5). If we look at the results of the 60 seconds chunk sizes, the smartphone data can achieve similar and even better EERs than the laptop data, e.g. 9.3% for smartphone and 11.2% for laptop with their best configuration.

VI. Conclusion and Future Work

In this paper, we verified behavioral authentication based on sounds emitted by keyboards in different environment conditions. These environment conditions are defined by a user's hardware (keyboard and microphone) and the situational ambient noise. For that purpose, we are the first to build a web application and record typing audio from the laptop keyboards using the built-in microphone and a smartphone that is positioned next to the laptop. Using the application, we collected data from 26 participants.

We analyzed the data using different chunk sizes and configuration parameters. We also applied the approach of

Tab. 1: EERs for chunk size 10 seconds of laptop data using

D \ K	20	30	45	60
10%	18.9	17.7	17.3	**17.0**
20%	18.9	17.7	17.3	**17.0**
50%	18.8	17.7	17.4	17.1
70%	18.8	17.8	17.4	17.1
90%	18.8	17.8	17.4	17.1

(a) no majority voting

D \ K	20	30	45	60
10%	14.8	13.7	12.9	11.8
20%	14.8	13.7	12.9	**11.7**
50%	14.4	13.5	12.9	**11.7**
70%	14.2	13.6	12.9	11.8
90%	14.4	13.7	12.9	11.9

(b) majority voting of 3

D \ K	20	30	45	60
10%	14.9	13.6	12.9	12.7
20%	14.9	13.7	13.0	12.7
50%	14.7	13.7	13.1	**12.6**
70%	14.7	13.6	13.1	12.7
90%	14.7	13.7	13.1	12.7

(c) majority voting of 5

Tab. 2: EERs for chunk size 30 seonds of laptop data using

D \ K	20	30	45	60
10%	17.2	16.8	15.1	15.4
20%	16.7	16.5	15.5	**14.4**
50%	16.5	16.3	14.7	**14.4**
70%	16.6	16.5	15.3	14.9
90%	16.6	16.5	14.6	14.8

(a) no majority voting

D \ K	20	30	45	60
10%	13.9	14.3	12.5	14.3
20%	13.6	14.3	13.3	12.5
50%	13.4	13.3	11.8	**11.5**
70%	13.1	13.9	12.9	13.8
90%	13.2	13.7	12.1	12.2

(b) majority voting of 3

D \ K	20	30	45	60
10%	14.8	15.6	13.2	14.3
20%	14.5	15.4	14.4	12.7
50%	14.8	13.9	13.0	**11.4**
70%	13.9	14.6	13.9	13.8
90%	13.9	14.3	12.8	12.0

(c) majority voting of 5

Tab. 3: EERs for chunk size 60 seconds of laptop data using

D \ K	20	30	45	60
10%	15.4	14.5	14.3	13.1
20%	15.4	14.7	14.9	13.8
50%	14.9	15.6	13.6	13.2
70%	15.0	14.9	13.5	13.6
90%	14.9	15.9	13.5	**12.8**

(a) no majority voting

D \ K	20	30	45	60
10%	15.4	14.5	14.3	13.1
20%	15.9	15.3	14.6	12.8
50%	14.2	15.8	13.4	13.3
70%	14.1	13.8	12.9	**12.4**
90%	14.0	14.8	13.5	12.6

(b) majority voting of 3

D \ K	20	30	45	60
10%	15.4	14.3	13.7	13.1
20%	15.8	15.7	14.0	12.3
50%	15.4	15.6	13.4	12.5
70%	14.9	13.4	12.2	12.4
90%	14.9	14.8	13.1	**11.2**

(c) majority voting of 5

Tab. 4: EERs for chunk size 10 seconds of smartphone data using

D \ K	20	30	45	60
10%	**21.8**	23.3	22.7	22.3
20%	21.9	23.3	22.7	22.1
50%	21.9	23.4	22.7	22.4
70%	21.9	25.8	23.9	22.4
90%	23.3	23.9	24.2	22.5

(a) no majority voting

D \ K	20	30	45	60
10%	21.3	22.6	21.3	24.2
20%	21.3	22.9	21.2	24.4
50%	**21.1**	23.0	21.4	24.5
70%	21.4	26.2	24.3	24.7
90%	23.1	23.0	24.9	24.8

(b) majority voting of 3

D \ K	20	30	45	60
10%	19.2	21.2	22.5	22.0
20%	19.2	21.1	22.6	21.9
50%	**19.1**	21.4	22.1	21.5
70%	19.4	23.1	23.2	22.0
90%	19.2	21.1	23.4	21.9

(c) majority voting of 5

Tab. 5: EERs for chunk size 30 seconds of smartphone data using

D \ K	20	30	45	60
10%	19.5	21.7	19.7	**17.8**
20%	22.5	20.7	18.6	18.7
50%	20.7	22.4	19.1	17.9
70%	22.3	24.9	22.1	18.9
90%	25.0	23.4	21.4	22.9

(a) no majority voting

D \ K	20	30	45	60
10%	17.7	19.8	18.9	17.3
20%	21.2	18.9	17.4	**16.7**
50%	19.5	20.4	17.4	18.7
70%	19.9	25.0	23.0	18.4
90%	25.0	23.4	22.6	26.2

(b) majority voting of 3

D \ K	20	30	45	60
10%	16.6	19.5	18.1	16.5
20%	18.9	17.6	16.7	**14.7**
50%	17.8	19.6	17.2	17.1
70%	18.6	23.6	23.6	15.0
90%	23.6	23.6	22.1	26.0

(c) majority voting of 5

Tab. 6: EERs for chunk size 60 seconds of smartphone data using

D \ K	20	30	45	60
10%	18.8	18.8	16.7	15.9
20%	18.9	18.6	18.9	15.8
50%	19.8	22.9	15.9	**14.7**
70%	19.5	16.3	16.9	15.3
90%	20.1	17.7	15.5	16.8

(a) no majority voting

D \ K	20	30	45	60
10%	18.7	18.9	12.5	11.5
20%	17.9	17.9	16.3	12.6
50%	18.4	23.2	14.1	11.7
70%	18.7	14.9	11.9	**10.5**
90%	20.5	13.9	12.8	15.8

(b) majority voting of 3

D \ K	20	30	45	60
10%	15.4	17.6	11.3	11.9
20%	16.6	17.4	15.5	11.2
50%	19.9	21.9	13.0	11.1
70%	19.5	14.8	11.1	**9.3**
90%	18.8	12.9	12.5	12.0

(c) majority voting of 5

Roth et al.'s work and extend the authentication algorithm with majority voting. The best EER for laptop and smartphone data is 11.2% and 9.3%, respectively.

The results show that the smartphone can be used to record typing sounds and recognize the smartphone owner based on the typing sound. They also show that devices can use a model (e.g. virtual alphabet) that was trained in advance and works without knowledge about the underlying keyboard and microphone hardware.

Although the results are promising, they are not sufficient enough to provide a very secure user authentication system. Thus, there are a few things to note which will be done in future work. First, typing sound should not be used as a single method in a continuous authentication scheme but only in combination with other methods for a reliable result, especially if there is not much involvement of a keyboard. Second, we will look into other methods for keystroke detection, e.g. using deep learning approaches because keystroke detection is a very important part in this whole procedure. The third point is to implement the continuous typing sound authentication as streaming process on smartphones and optimize performance and battery usage. A fourth point is to analyze different positions of the smartphone (distance to the keyboard) and how this will increase or decrease the authentication result. The fifth and last point, we want to address in the future, is to improve the keyboard acoustic authentication against an attacker that can execute more complex attacks than random ones.

References

[1] K. Niinuma and A. K. Jain, "Continuous user authentication using temporal information," in *Biometric Technology for Human Identification VII*, vol. 7667. International Society for Optics and Photonics, 2010, p. 76670L.

[2] S. Shepherd, "Continuous authentication by analysis of keyboard typing characteristics," 1995.

[3] J. Wayman, A. Jain, D. Maltoni, and D. Maio, "An introduction to biometric authentication systems," in *Biometric Systems*. Springer, 2005, pp. 1–20.

[4] M. O. Derawi, C. Nickel, P. Bours, and C. Busch, "Unobtrusive user-authentication on mobile phones using biometric gait recognition," in *Intelligent Information Hiding and Multimedia Signal Processing (IIH-MSP), 2010 Sixth International Conference on*. IEEE, 2010, pp. 306–311.

[5] E. Klieme, C. Tietz, and C. Meinel, "Beware of smombies: Verification of users based on activities while walking," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*. IEEE, 2018, pp. 651–660.

[6] A. Buriro, B. Crispo, F. Del Frari, J. Klardie, and K. Wrona, "Itsme: Multi-modal and unobtrusive behavioural user authentication for smartphones," in *International Conference on Passwords*. Springer, 2015, pp. 45–61.

[7] H. Saevanee, N. L. Clarke, and S. M. Furnell, "Multi-modal behavioural biometric authentication for mobile devices," in *IFIP International Information Security Conference*. Springer, 2012, pp. 465–474.

[8] I. C. Stylios, O. Thanou, I. Androulidakis, and E. Zaitseva, "A review of continuous authentication using behavioral biometrics," in *Proceedings of the SouthEast European Design Automation, Computer Engineering, Computer Networks and Social Media Conference*. ACM, 2016, pp. 72–79.

[9] G. N. Healy, E. G. Eakin, A. D. LaMontagne, N. Owen, E. A. Winkler, G. Wiesner, L. Gunning, M. Neuhaus, S. Lawler, B. S. Fjeldsoe *et al.*, "Reducing sitting time in office workers: short-term efficacy of a multicomponent intervention," *Preventive medicine*, vol. 57, no. 1, pp. 43–48, 2013.

[10] A. K. Karlson, B. R. Meyers, A. Jacobs, P. Johns, and S. K. Kane, "Working overtime: Patterns of smartphone and pc usage in the day of an information worker," in *International Conference on Pervasive Computing*. Springer, 2009, pp. 398–405.

[11] J. Roth, X. Liu, A. Ross, and D. Metaxas, "Investigating the discriminative power of keystroke sound," *IEEE Transactions on Information Forensics and Security*, vol. 10, no. 2, pp. 333–345, 2015.

[12] D. Umphress and G. Williams, "Identity verification through keyboard characteristics," *International journal of man-machine studies*, vol. 23, no. 3, pp. 263–273, 1985.

[13] S. P. Banerjee and D. L. Woodard, "Biometric authentication and identification using keystroke dynamics: A survey," *Journal of Pattern Recognition Research*, vol. 7, no. 1, pp. 116–139, 2012.

[14] M. S. Obaidat and B. Sadoun, "Verification of computer users using keystroke dynamics," *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, vol. 27, no. 2, pp. 261–269, 1997.

[15] E. Yu and S. Cho, "Keystroke dynamics identity verification—its problems and practical solutions," *Computers & Security*, vol. 23, no. 5, pp. 428–440, 2004.

[16] G. L. Azevedo, G. D. Cavalcanti, and E. C. Carvalho Filho, "Hybrid solution for the feature selection in personal identification problems through keystroke dynamics," in *Neural Networks, 2007. IJCNN 2007. International Joint Conference on*. IEEE, 2007, pp. 1947–1952.

[17] F. Bergadano, D. Gunetti, and C. Picardi, "User authentication through keystroke dynamics," *ACM Transactions on Information and System Security (TISSEC)*, vol. 5, no. 4, pp. 367–397, 2002.

[18] P. S. Dowland and S. M. Furnell, "A long-term trial of keystroke profiling using digraph, trigraph and keyword latencies," in *Security and Protection in Information Processing Systems*. Springer, 2004, pp. 275–289.

[19] M. Choraś and P. Mroczkowski, "Recognizing individual typing patterns," in *Iberian Conference on Pattern Recognition and Image Analysis*. Springer, 2007, pp. 323–330.

[20] K. Revett, F. Gorunescu, M. Gorunescu, M. Ene, P. S. T. Magalhães, and H. D. d. Santos, "A machine learning approach to keystroke dynamics based user authentication," *International Journal of Electronic Security and Digital Forensics*, vol. 1, no. 1, pp. 55–70, 2007.

[21] J. A. Robinson, V. Liang, J. M. Chambers, and C. L. MacKenzie, "Computer user verification using login string keystroke dynamics," *IEEE transactions on systems, man, and cybernetics-part a: systems and humans*, vol. 28, no. 2, pp. 236–241, 1998.

[22] A. Kelly, "Cracking passwords using keyboard acoustics and language modeling," *University of Edinburgh*, 2010.

[23] D. Asonov and R. Agrawal, "Keyboard acoustic emanations," in *null*. IEEE, 2004, p. 3.

[24] L. Zhuang, F. Zhou, and J. D. Tygar, "Keyboard acoustic emanations revisited," *ACM Transactions on Information and System Security (TISSEC)*, vol. 13, no. 1, p. 3, 2009.

[25] A. Compagno, M. Conti, D. Lain, and G. 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*. ACM, 2017, pp. 703–715.

[26] J. Roth, X. Liu, A. Ross, and D. Metaxas, "Biometric authentication via keystroke sound," in *Biometrics (ICB), 2013 International Conference on*. IEEE, 2013, pp. 1–8.

[27] M. Pleva, E. Kiktova, J. Juhar, and P. Bours, "Acoustical user identification based on mfcc analysis of keystrokes," *Advances in Electrical and Electronic Engineering*, vol. 13, no. 4, pp. 309–313, 2015.

[28] M. Pleva, E. Kiktova, P. Viszlay, and P. Bours, "Acoustical keystroke analysis for user identification and authentication," in *Radioelektronika (RADIOELEKTRONIKA), 2016 26th International Conference*. IEEE, 2016, pp. 386–389.

[29] M. Pleva, P. Bours, S. Ondáš, and J. Juhár, "Improving static audio keystroke analysis by score fusion of acoustic and timing data," *Multimedia Tools and Applications*, vol. 76, no. 24, pp. 25 749–25 766, 2017.

[30] A. Buriro, Z. Akhtar, B. Crispo, and S. Gupta, "Mobile biometrics: Towards a comprehensive evaluation methodology," in *2017 International Carnahan Conference on Security Technology (ICCST)*. IEEE, 2017, pp. 1–6.