Acta Univ. Sapientiae Informatica **15,** 1 (2023) 10-24

DOI: 10.2478/ausi-2023-0002

# SapiPin: Observations on PIN-code typing dynamics

Margit ANTAL
Sapientia Hungarian University of Transylvania, Cluj-Napoca, Romania
email: manyi@ms.sapientia.ro
ORCID:0000-0003-3596-1365

Krisztián BUZA
Jozef Stefan Institute
Ljubljana, Slovenia
email: chrisbuza@yahoo.com
ORCID:0000-0002-7111-6452

**Abstract.** In this paper, we report on PIN-code typing behaviour on touchscreen devices of 112 subjects. Detailed statistical analysis revealed that the major difference between subjects is in inter-key latency. Key-press duration variations are insignificant compared to inter-key latency variations. Subjects were grouped into meaningful clusters using clustering. The resulting clusters were of slow, medium, and fast typists. The dataset was split randomly into two equal size subsets. The first subset was used to train different synthetic data generators, while the second subset was used to evaluate an authentication attack using the generated synthetic data. The evaluation showed that slow typists were the hardest to attack. Both the dataset and the software are publicly available at https://github.com/margitantal68/sapipin_paper.

## 1 Introduction

While the dynamics of typing a text on a usual keyboard has been shown to be characteristic to the user, see e.g. [3, 14] and the references therein, the dynamics of typing PIN codes on numeric keyboards is somewhat understudied even though most of the wide-spread user authentication techniques are based

**Key words and phrases:** keystroke dynamics, PIN-code, synthetic attack, anomaly detection

on (or include the use of) PIN-codes: we may use PIN-codes to unlock our smartphone and One-Time PIN-codes (OTP) in online banking applications. Therefore, it is important to examine the typing characteristics of people when typing such short PIN-codes. PIN-codes are usually entered using a numeric keypad, which may significantly affect the typing rhythm.

Unlike previous research where all participants had to type the same PIN-code [6, 13], in our research all participants had to type a randomly generated PIN-code 20 times without error. This made it possible to observe similarities when typing different PIN-codes, and also made it possible to observe how a stable typing rhythm develops for each individual.

Our main contributions are as follows:

- We present *SapiPin*, a new dataset, which contains PIN-code typing data collected on mobile devices from 112 users.
- Exploratory data analysis through clustering revealed three meaningful typist groups.
- An attack on typing rhythm was performed using several types of synthetic data. We show the attack effectiveness for different typist groups.
- In order to assist reproduction of the results, we published our code in a public GitHub repository[1].

The rest of the paper is organized as follows. Section 2 provides a concise overview of related works. Section 3 presents the *SapiPin* dataset with the collection protocol and basic information about the subjects. Section 4 is devoted to exploratory data analysis. We begin Section 5 with a brief description of methods used to synthesize data and to detect anomalies. This is followed by the details of our evaluation protocol and our observations. Finally, we conclude in Section 6.

## 2 Related work

Many studies have already examined the dynamics of entering passwords. Earlier studies performed the experiments using a classic keyboard [11, 12], while the more recent ones used touch-screens of mobile phones [1, 3, 4, 2]. Moreover, Gunetti and Picardi [10] studied the usage of free text typing for continuous authentication.

[1]https://github.com/margitantal68/sapipin_paper

The dynamics of typing may be affected be various properties of the keyboard, such as its size and the placement of keys, especially whether the keyboard is alphanumeric (QWERTY) or numeric. Therefore, additionally to the research considering alphanumeric keyboards, it is important to study keystroke dynamics on numerical keyboards too.

Out of the works related to numeric keyboards, we point of the experiments of Clarke et al. [6] in which 16 subjects entered 10-digit telephone numbers and 4-digit PIN-codes. 30 samples were collected from which 20 were used for template creation and the remaining 10 for validation. They report 11.3% Equal Error Rate (EER) in an user authentication study. Better performance (8.60 % EER) was reported by Maxion and Killourhy [13] in an authentication experiment using 10-digit numbers. 28 subjects were involved in the experiment who donated 200 samples of the same 10-digit number in 4 consecutive sessions. Bours and Masoudian [5] investigated the usage of 6-digit one-time PIN- codes for user authentication. In an experiment conducted with 30 participants, they found that in the case of one-time PIN-codes the EER of authentication was 26%.

Some studies focused on the distribution of timings of keystroke dynamics. Dhakal et al. [8] conducted a study with 168,000 participants in order to find keystroke patterns linked to typing performance. They found huge differences in inter-key times between slow and fast typists, however, keypress times were very similar across these groups. Gonzales et al. [9] compared several distributions in order to rank them according to their similarity to timing histograms in free text keystroke dynamics. They found that log-logistic distributions are excellent choices for modelling the shape of timing histograms.

Timing distributions are very important when the task is to generate synthetic forgeries. Deian et al. [7] examined the robustness of keystroke-dynamics based biometrics against synthetic forgeries. Their bots generate both the keystroke duration and inter-key latency using Gaussian distributions. They reported high True Positive and low False Positive rates on a small dataset containing the data of 20 users. The good result is probably due to the weak attack, because the inter-key latency cannot be considered normally distributed.

None of the aforementioned works examined the distribution of keystroke timing in case of a numeric keyboard. This study aims to fill this gap.

| Personal attribute | Amount |
|---|---|
| Age | |
| 21–30 years | 103 |
| 40–60 years | 9 |
| Gender | |
| Male | 85 |
| Female | 27 |

Table 1: Personal information of the 112 subjects of the *SapiPin* dataset.

## 3 The *SapiPin* dataset

The *SapiPin*[2] dataset was collected in 2021 and contains 6-digit PIN typing of 112 voluntary participants. Each subject typed a randomly generated 6-digit PIN-code 20 times on his/her own mobile device. Only error-free typing was saved.

We used a web application only available on touchscreen mobile devices to collect data[3]. The application is implemented in PHP and JavaScript. Timestamps are recorded by the Date.now() JavaScript function. Subjects agree to have their data used for scientific experiments by sending the collected data to the email address provided for this purpose.

The keyboard layout of this data collector is shown in Fig. 1. Detailed personal information of the subjects is presented in Table 1.

## 4 Preprocessing and Data Analysis

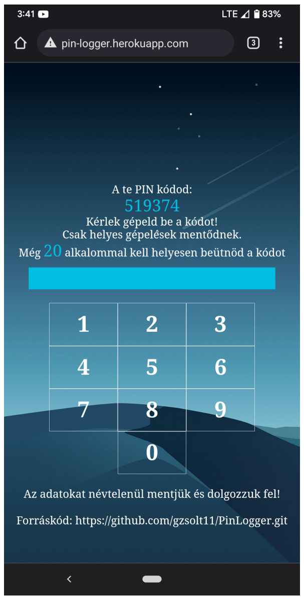
### 4.1 Preprocessing

The collected dataset was preprocessed to remove corrupted items. We excluded the data of a single person, the user with ID number 96, in case of whom it happened several times that the timestamps of pressing were identical to the timestamps of release which indicates an error with recording the data. As a result, we work with the data of 111 users from now on (The database published in 2022 still contains the data of user with ID 96).

In the next step, we computed two types of features: (i) keypress duration, a.k.a. hold time, and (ii) inter-key latency, i.e., the duration between releasing

[2] `https://github.com/margitantal68/sapipin`

[3] source code: https://github.com/gzsolt11/PinLogger.git



A te PIN kódod:
519374
Kérlek gépeld be a kódot!
Csak helyes gépelések mentődnek.
Még 20 alkalommal kell helyesen beütnöd a kódot

Az adatokat névtelenül mentjük és dolgozzuk fel!

Forráskód: https://github.com/gzsolt11/PinLogger.git

Figure 1: PIN logger.

a key and pressing the next key. Therefore, in the case of a 6-digit PIN-code, the following features were computed: $\mathsf{HT}_1$, $\mathsf{HT}_2$, $\mathsf{HT}_3$, $\mathsf{HT}_4$, $\mathsf{HT}_5$, $\mathsf{HT}_6$ (hold times), and $\mathsf{RP}_1$, $\mathsf{RP}_2$, $\mathsf{RP}_3$, $\mathsf{RP}_4$, $\mathsf{RP}_5$ (release-press times).

The next step in preprocessing was outlier detection. We found no outliers regarding keypress duration. In contrast, there were quite a few outliers for inter-key intervals. In such cases, users appeared to pause while typing. We declared a value of a feature as an outlier if it was greater than $\mu + 10\sigma$, where $\mu$ and $\sigma$ are the average and the standard deviation of the feature. For the subsequent analysis, outliers were always replaced by the average of the corresponding feature of the given user. In total, 13 outliers were replaced. These outliers belonged to the following 11 distinct users: 2, 11, 20, 21, 25, 48, 53, 58, 73, 77, 93. In most cases, each user exhibited a single instance of an outlier feature within their respective samples, with the exception of users 25 and 77, who each presented two instances. Of the 13 detected cases, 70% involved the user's first sample as the source of the outlier feature.

The final dataset contains 13320 keystrokes from 111 participants (120 keystrokes per participant). Boxplots of keypress duration and inter-key latency are shown in Fig. 2.

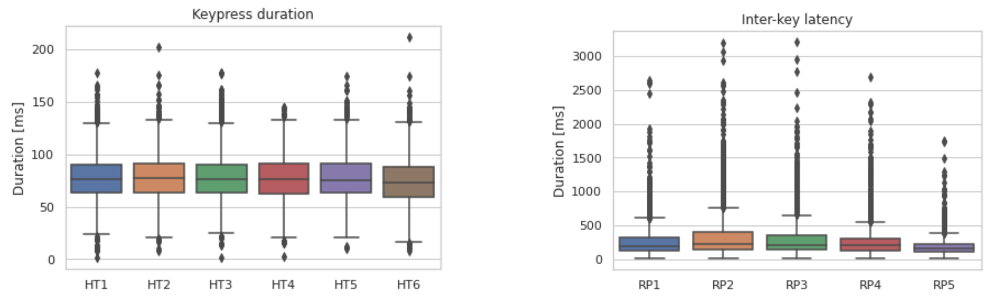


Figure 2: Keypress duration and inter-key latency boxplots

## 4.2 Data analysis

Outlier correction was followed by descriptive statistics. Keypress duration and inter-key intervals were analyzed separately. Fig. 3 shows the histograms of keypress duration (HT hold times) and inter-key interval (release-press times). The average keypress duration is 76.89 ms with a standard deviation of 22.17. The skewness of the distribution is 0.40, while its kurtosis is 0.64. In contrast, the average inter-key interval is 269.56 ms with standard deviation, skewness and kurtosis of 266.67, 3.78 and 21.36 respectively. It can be observed that the inter-key interval average is more than three times higher than keypress duration average. The high positive skewness of the inter-key interval distribution is inline with the observations in [8]. However, we observed an even higher skewness than in case of usual keyboards.

In a previous study [4], we observed that the average keypress duration, as well as the average inter-key interval is a discriminative feature for user authentication based on keystroke dynamics. Therefore, we computed three new features of PIN-code typing for each user: (i) $\mathsf{AVG_{HT}}$, the average of keypress duration, (ii) $\mathsf{AVG_{RP}}$, the average of inter-key interval, and (iii) $\mathsf{TOTAL_{TIME}}$, the duration of typing the entire PIN code. In the next step, correlations were computed between these features: while the correlations between $\mathsf{AVG_{HT}}$ and the two other features are close to zero ($\mathsf{corr}(\mathsf{AVG_{HT}}, \mathsf{AVG_{RP}}) = -0.04$, $\mathsf{corr}(\mathsf{AVG_{HT}}, \mathsf{TOTAL_{TIME}}) = 0.08$), $\mathsf{AVG_{RP}}$ strongly correlates with $\mathsf{TOTAL_{TIME}}$ ($\mathsf{corr}(\mathsf{AVG_{RP}}, \mathsf{TOTAL_{TIME}}) = 0.99$). We can state that the typing speed clearly depends only on the inter-key latency. Fig. 4 shows the uncorrelated nature of the $\mathsf{AVG_{HT}}$ and $\mathsf{AVG_{RP}}$ features.

We investigated how the typing rhythm of PIN code of each user evolves over time. To this end, we computed the ratio of $\mathsf{AVG_{RP}}$ to $\mathsf{AVG_{HT}}$ for each PIN

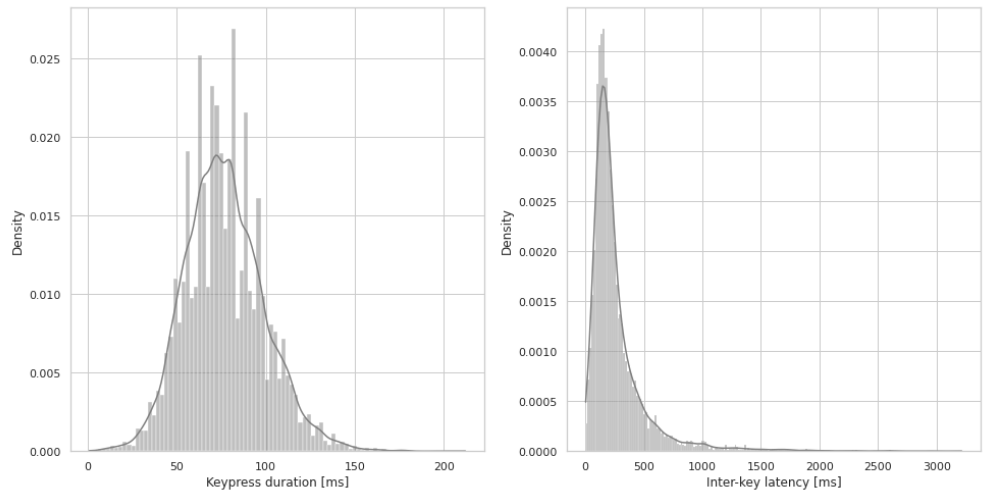


Figure 3: Keypress duration and inter-key latency histograms

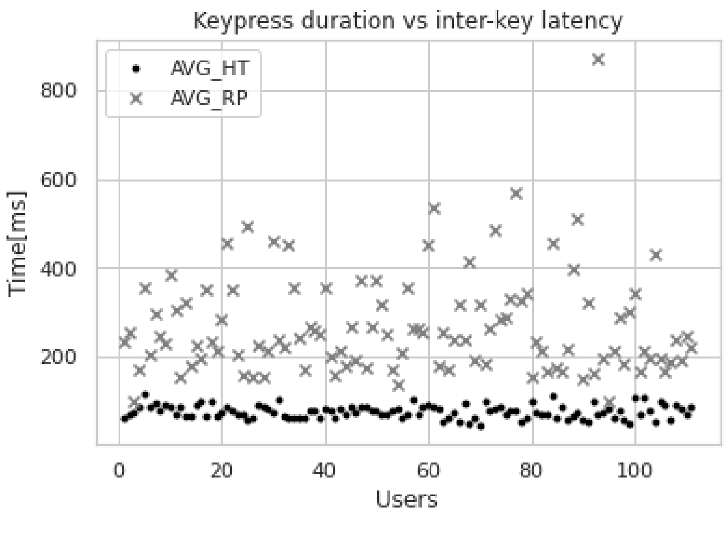


Figure 4: Users $\mathsf{AVG}_{\mathsf{HT}}$ and $\mathsf{AVG}_{\mathsf{RP}}$

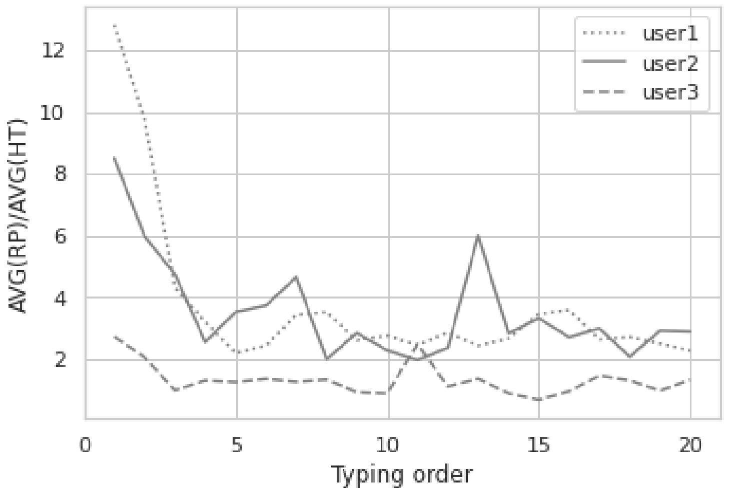
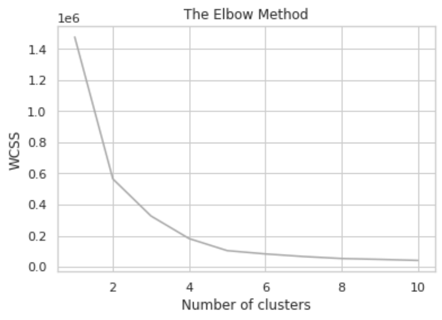


Figure 5: Rhythm of typing stabilises over time

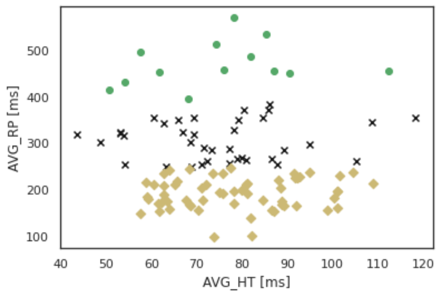
typing. Our analysis revealed that the first two samples typically exhibited significant differences from subsequent samples. However, once these initial samples were completed, a more consistent rhythm emerged with relatively smaller variations, as depicted in Figure 5.

## 4.3 Clustering

The objective of clustering is to find meaningful groups within a dataset. Clustering belongs to unsupervised learning: unlabelled instances are grouped (or the labels of instances are not used when the groups are determined). In our case, we expect to see clusters of users having similar typing characteristics. The original dataset was converted to a reduced dataset, where each user is represented by a single aggregated instance, which contains only two features $\mathsf{AVG}_{\mathsf{HT}}$ and $\mathsf{AVG}_{\mathsf{RP}}$ corresponding to the average hold time and average inter-key latency of the user. K-means was used as the clustering algorithm. In order to find the most suitable number of clusters, we used the elbow method. Fig. 6a shows the result of the elbow method, which indicates that this dataset contains four meaningful groups. We have one group containing only one user (userid=93), who has extremely large inter-key latency (this can be seen in Fig.4 too). We excluded this user from further analysis. We denote the three other groups as slow (13 users, green dots), medium (37 users, black x marker) and fast (60 users, yellow diamond marker) typists. It can be seen in Fig. 6b

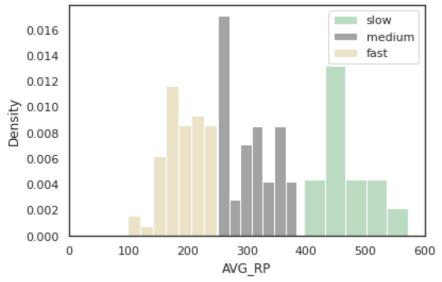


(a) Elbow method

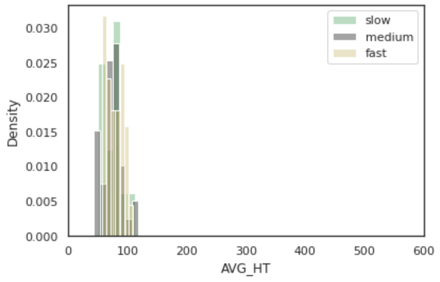


(b) User groups

Figure 6: K-means clustering



(a) Clusters' $\mathsf{AVG}_{\mathsf{RP}}$ histogram.

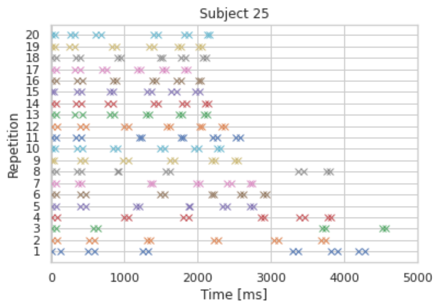


(b) Clusters' $\mathsf{AVG}_{\mathsf{HT}}$ histogram.

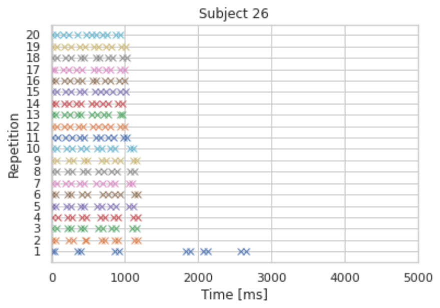
Figure 7: K-means clustering

that these three groups are separated by the $\mathsf{AVG}_{\mathsf{RP}}$ feature. The typing patterns of a slow typist and a fast typist are illustrated in Figure 8.

During the preprocessing phase of the data analysis, a total of 13 outliers were identified and attributed to 11 unique users. Specifically, one of the users (user ID=93) exhibited a significantly slow typing speed. The remaining 10 users were distributed across three typing categories, with four users categorized as slow typists (IDs 21, 25, 73, and 74), four users classified as medium typists (IDs 2, 11, 20, and 58), and two users classified as fast typists (IDs 48 and 53). These findings indicate that outliers were present across all levels of typing proficiency. As noted in the data analysis chapter, it was observed that outliers were more frequently detected within the initial typing pattern of each user.



(a) Slow typist.



(b) Fast typist.

Figure 8: Typing patterns

## 5 Attack framework

### 5.1 Synthetic data generation

The SDV package was used to generate synthetic data[4]. We used three types of models: Gaussian Copula, Conditionally Time Generative Adversarial Network (CTGAN), and Time Variational Autoencoder (TVAE). The CTGAN and TVAE models were proposed by Lei Xu et al. in a paper presented in 2019 at the NeurIPS conference [15].

A copula in mathematical terms is a distribution over $[0,1]^d$ unit cube, created by using the probability integral transform from a multivariate normal distribution over $R^d$. Essentially, a copula is a mathematical function that describes the joint distribution of multiple random variables by examining the dependence between their marginal distributions.

CTGAN is a method that uses GANs to model the distribution of tabular data and generate samples from it. It overcomes the issues of non-Gaussian and multimodal distributions with mode-specific normalization. CTGAN employs a conditional generator and training-by-sampling. High-quality models are trained with fully-connected networks and several recent techniques [15].

The third synthetic data generation method is a special variational autoencoder, named TVAE, as this is an adaptation of variational autoencoders to tabular data.

Synthetic data generators must be trained and the data used for their training should not be used in further evaluations. We solved this issue by dividing the *SapiPin* dataset into two subsets: we used the data of 55 randomly selected

[4]https://sdv.dev/

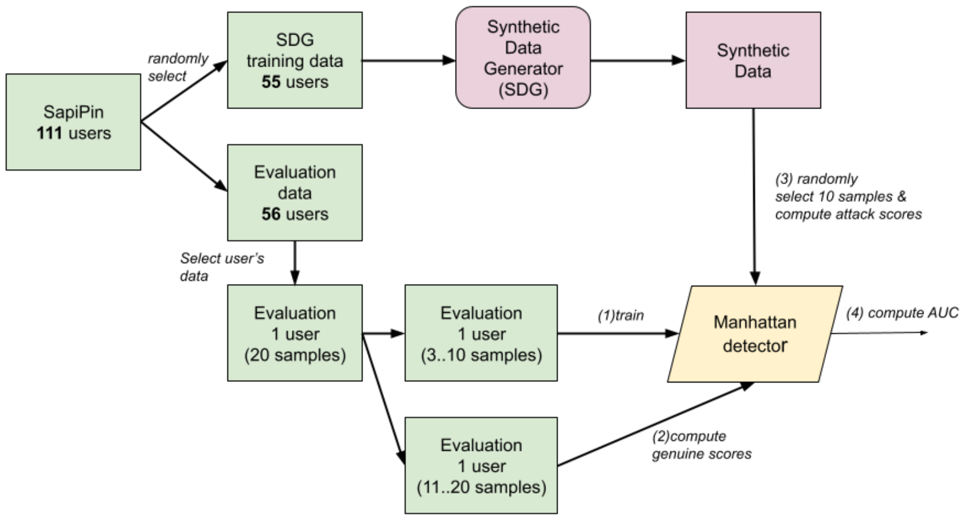

SapiPin
**111** users
randomly select
SDG training data
**55** users
Synthetic Data Generator (SDG)
Synthetic Data
Evaluation data
**56** users
Select user's data
Evaluation 1 user (20 samples)
Evaluation 1 user (3..10 samples)
Evaluation 1 user (11..20 samples)
(1)train
(2)compute genuine scores
(3) randomly select 10 samples & compute attack scores
Manhattan detector
(4) compute AUC

Figure 9: Evaluation of PIN-code attack framework

users to train the synthetic data generator, and we evaluated our attack framework on the data of the remaining 56 users (see the inital splitting of the data in the top-left of Figure 9).

## 5.2 Anomaly detector

We evaluate synthetic data quality via anomaly detector. High-quality synthetic data is difficult to detect. There are many different types of anomaly detectors. In the case of keystroke dynamics-based user authentication, one of the well-established detectors is the scaled Manhattan [12].

We trained a separate model for each user of the dataset using only real user data, and then evaluated it on both real and synthetic data. Based on the real and synthetic score values obtained from the evaluation results, we calculated the area under the curve (AUC). Average and standard deviation of these AUCs are reported for the 56 users in the evaluation dataset (see 4th step in Figure 9).

High quality synthetic data is difficult to detect by the anomaly detector, hence a lower AUC will be obtained in the case of better synthetic data.

## 5.3 Evaluation protocol and results

Each user in the dataset has 20 typing examples. As we observed that the first two examples are significantly different from the remaining ones, see Fig. 5, we excluded these two from anomaly detector training. Therefore, the first two examples were dropped, and only 8 genuine examples were used for training the anomaly detector. The remaining 10 examples were used for testing the anomaly detector, obtaining 10 positive (genuine) scores. We randomly selected 10 examples from the synthetic data as fraudulent examples, and presented them as an input to the anomaly detector. This resulted in 10 negative or attack scores. Based on the negative and positive score values, we calculated the AUC value for each user. The evaluation was repeated 10 times, each time using other randomly selected synthetic data for attack. Our synthetic data was generated based on training data which contains typing of different PIN-codes, therefore its samples are general typing rhythms of 6-digit PIN-codes. In our attack model we assume that the attacker knows the subject's PIN-code, but does not know its typing rhythm. Therefore, it uses a random sample from synthetic typing rhythms.

Table 2 reports the mean AUC and its standard deviation for each type of synthetic data. These results suggest that, compared with the synthetic data generated by CTGAN, both Gaussian Copula and TVAE generated data were highly similar to the genuine data, because in these cases it was more difficult to detect for the anomaly detector whether the data is real or synthetic, as indicated by the lower AUC value of Gaussian Copula and TVAE compared with that of CTGAN.

| Synthetic data generation method | AUC (std) |
|---|---|
| Gaussian Copula | 0.90 (0.10) |
| CTGAN | 0.98 (0.03) |
| TVAE | 0.89 (0.11) |

Table 2: Synthetic attacks on the *SapiPin* dataset.

The present study aimed to investigate the potential relationship between typing speed and AUC values. To this end, we analyzed the average typing time per participant, derived from eight training samples, in conjunction with the AUC values obtained from TVAE synthetic data (see Fig. 10). Each data point in the figure corresponds to a specific participant. Our findings reveal that

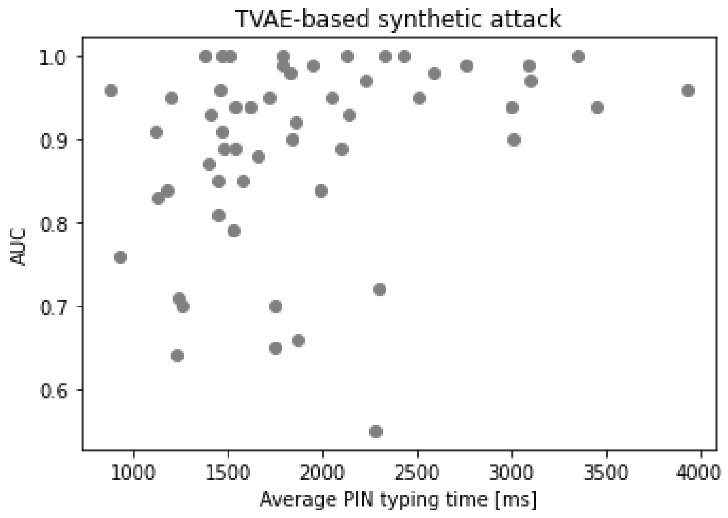


Figure 10: TVAE-based synthetic attack: users average typing time vs. AUC.

the AUC values attained by the slow typists were found to be comparatively elevated and exhibit a low degree of standard deviation. Although the majority of fast typists exhibit relatively high AUC values, there are several individuals in this group who demonstrate lower AUC values compared to the overall average.

### 5.4 Evaluation on typist groups

This section presents the outcomes of our investigation regarding three distinct typist groups, namely slow, medium, and fast typists. To this end, we applied the protocol described in the preceding section, with the evaluations carried out thrice, once for each typist group. Our findings, summarized in Table 3, highlight the highest degree of detection success among the slow typists, implying that they represent the most challenging target for potential attackers.

## 6 Conclusions

In this paper we presented *SapiPin*, a new dataset of 6-digit PIN-code typing on touchscreens. Data analysis revealed that the data can be divided into meaningful groups belonging to slow, medium, and fast typists. The major

| Synthetic data generation method | AUC (std) Slow | Medium | Fast |
|---|---|---|---|
| Gaussian Copula | 0.93 (0.08) | 0.89 (0.12) | 0.90 (0.09) |
| CTGAN | 0.98 (0.03) | 0.98 (0.04) | 0.98 (0.03) |
| TVAE | 0.95 (0.05) | 0.89 (0.12) | 0.88 (0.10) |

Table 3: Synthetic attacks on the typist groups.

difference between these groups is the inter-key latency. This observation confirms the results of a previous study [8] in which the same observations were reported for free text typing. Moreover, the reported timing distributions are similar to ours, even though only a 10-digit software keyboard was used in our study.

The dataset was split randomly into equally sized subsets. Half of the data was used to train different synthetic data generators. The best quality synthetic data (the most similar to the original one) was generated by the TVAE method. Using the synthetic data, we performed synthetic attacks on each user from the second half of the *SapiPin* dataset. Results show that slow typists are the hardest to attack. At the same time, in this group we can see the greatest dispersion among the typing samples of a user.

We acknowledge limitations regarding the generalizability of the results, as young, European participants (university students) are over-represented in the data.

## Acknowledgements

We would like to thank Zsolt Gergely for creating the data collection application and for his assistance with data collection.

## References

[1] J. Angulo, E. Wästlund. Exploring touch-screen biometrics for user identification on smart phones. In J. Camenisch, B. Crispo, S. Fischer-Hübner, R. Leenes, G. Russello, editors, *Privacy and Identity Management for Life*, pages 130–143, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. ⇒11

[2] M. Antal, L. Nemes. The mobikey keystroke dynamics password database: Benchmark results. In *Software Engineering Perspectives and Application in Intelligent Systems*, pages 35–46, Cham, 2016. Springer International Publishing. ⇒11

[3] M. Antal, L. Z. Szabo, I. Laszlo. Keystroke dynamics on android platform. *Procedia Technology*, **19** (2015) 820–826. 8th Int. Conf. Interdisciplinarity in Engineering, INTER-ENG 2014, 9–10 October 2014, Târgu Mureș, Romania. ⇒10, 11

[4] M. Antal, L. Z. Szabó. An evaluation of one-class and two-class classification algorithms for keystroke dynamics authentication on mobile devices. In *20th Int. Conf. on Control Systems and Computer Sci.*, pages 343–350, 2015. ⇒11, 15

[5] P. Bours, E. Masoudian. Applying keystroke dynamics on one-time pin codes. In *2nd Int. Workshop on Biometrics and Forensics*, pages 1–6, 2014. ⇒12

[6] N. Clarke, S. Furnell, B. Lines, P. Reynolds. Keystroke dynamics on a mobile handset: a feasibility study. *Information Management & Computer Security*, **11,** 4 (2003) 161–166. ⇒11, 12

[7] S. Deian, S. Xiaokui, D. Danfeng. Robustness of keystroke-dynamics based biometrics against synthetic forgeries. *Computers & Security*, **31,** 1 (2012) 109–121. ⇒12

[8] V. Dhakal, A. M. Feit, P. O. Kristensson, A. Oulasvirta. Observations on typing from 136 million keystrokes. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, page 1–12, New York, NY, USA, 2018. Association for Computing Machinery. ⇒12, 15, 23

[9] N. González, E. P. Calot, J. S. Ierache, W. Hasperué. On the shape of timings distributions in free-text keystroke dynamics profiles. *Heliyon*, **7,** 11 (2021) e08413. ⇒12

[10] D. Gunetti, C. Picardi. Keystroke analysis of free text. *ACM Trans. Inf. Syst. Secur.*, **8,** 3 (2005) 312–347. ⇒11

[11] R. Joyce, G. Gupta. Identity authentication based on keystroke latencies. *Commun. ACM*, **33,** 2 (1990) 168–176. ⇒11

[12] K. S. Killourhy, R. A. Maxion. Comparing anomaly-detection algorithms for keystroke dynamics. In *2009 IEEE/IFIP Int. Conf. on Dependable Systems Networks*, pages 125–134, 2009. ⇒11, 20

[13] R. A. Maxion, K. S. Killourhy. Keystroke biometrics with number-pad input. In *2010 IEEE/IFIP Int. Conf. on Dependable Systems & Networks (DSN)*, pages 201–210, 2010. ⇒11, 12

[14] D. Neubrandt, K. Buza. Pro,jection-based person identification. In *Proceedings of the 10th Int. Conf. on Computer Recognition Systems CORES 2017*, pages 221–228. Springer, 2018. ⇒10

[15] L. Xu, M. Skoularidou, A. Cuesta-Infante, K. Veeramachaneni. Modeling tabular data using conditional gan. In *Proceedings of the 33rd Int. Conf. on Neural Information Processing Systems*, pages 7335–7345, 2019. ⇒19

*Received: March 13, 2023 • Revised: April 11, 2023*