Secure, Usable and Privacy-Friendly User Authentication from Keystroke Dynamics

Kimmo Halunen(✉) and Visa Vallivaara

VTT Technical Research Centre of Finland Ltd., Oulu, Finland
{kimmo.halunen,visa.vallivaara}@vtt.fi

Abstract. User authentication is a key technology in human machine interaction. The need to establish the legitimacy of transactions and possibly the actors behind them is crucial for trustworthy operation of services over the internet. A good authentication method offers security, usability and privacy protections for the users and the service providers. However, achieving all three properties with a single method is a difficult task and such methods are not in wide use today. We combine methods from biometrics, secure key exchange algorithms and privacy-protecting authentication to build an authentication system that achieves these three properties. Our system uses keystroke dynamics to authenticate the user and cryptographic methods to protect the privacy of the templates and samples and to extend the authentication to key exchange. The results show that the system can be used for user authentication, but more work is needed to protect against impersonation in some cases. Our work is extensible to many other biometrics that can be measured and compared in a similar manner as keystroke dynamics and with further research to larger classes of authentication methods.

1 Introduction

User authentication is one of the key technologies in human machine interaction. The services provided in many contexts both locally and over the internet require the user to provide assurance that she is authorised to access the service. To this effect, a good authentication method provides security, usability and privacy protection both for users and the service providers alike.

Generally, user authentication is done via three different types of factors. The most common in web authentication is *something you know*, which is manifested in the ubiquitous passwords that users need to type in order to gain access. The second category of factors is *something you have* such as a key to a lock, a list of one time passwords, a USB token or a mobile phone. These can be used to authenticate the user towards services, usually through a challenge-response system. The third factor is *something you are* such as a biometric, e.g., fingerprint or a facial image. These factors are more common also in the identification of individuals and in authentication between humans.

Different factors and methods of authentication offer different levels of security, usability and privacy protection. A great study of various methods used

B.B. Brumley and J. Röning (Eds.): NordSec 2016, LNCS 10014, pp. 256–268, 2016.
DOI: 10.1007/978-3-319-47560-8_16

in web authentication can be found in [6]. However, no single method can offer the best from all of these categories. Thus, new methods and combinations are needed.

To provide a good level of security, privacy and usability there need to be systems that offer protection in all of these categories. For example, a randomly chosen 16 character password is quite secure and offers good privacy, but it is very hard to use. Thus the usability of such a scheme is rather low, although some progress has been made to help people remember random secrets [7]. On the other hand, many biometrics such as fingerprints and facial recognition systems offer good usability and even security. The privacy protection of such systems is often very poor and the templates can be easily used for surveillance and identification in addition to the original authentication use case.

In this work we present a novel combination of known methods that can achieve good performance in all three categories. By measuring the keystroke dynamics of the user (i.e. timings related to keystrokes), when typing her username, we can use this biometric to authenticate the user. We will then combine this with privacy protection mechanisms from [27] to protect the biometric templates and samples and the protocol from [13] to combine all these into a secure key exchange protocol.

The paper is organised in the following way. In the next section, we present previous work on keystroke dynamics and privacy protecting authentication. In Sect. 3 we describe the methods that we have used to build our authentication system in more detail. In the fourth section we present the results of a user study that we conducted with our system and we end the paper with discussion and conclusions on our work.

2 Previous Work

Keystroke dynamics as a biometric have been researched extensively for many years with earliest results already from the 1980s, e.g., [14]. They can be captured both from regular keyboards, e.g., [19–21] or from mobile devices, e.g., [10,26]. The measurements are usually related to timings between keypresses and these can be used for fairly accurate authentication results. On the other hand, these can also be used for profiling and identifying users, e.g., [9], which causes privacy considerations. For a more thorough survey on different methods of keystroke dynamics see for example [3].

Privacy-friendly authentication, especially with biometrics, has been a topic of research for some time. Some biometric features can be protected either by specific schemes (many of which have been evaluated in [25]) or more generic constructions such as fuzzy vaults [18] or fuzzy extractors [12]. In these ways, the biometric information is either protected at the template level or only used to generate cryptographic keys, which are then used for authentication. However, the generic methods need to be tailored to suit each specific biometric and measurement type and also contain some limitations of their own. This means that applying these methods is not necessarily straightforward to any given biometric and type of measurement.

Another way to protect the privacy of biometrics is to build a privacy protection system specifically for some biometric, e.g., symmetric hash functions for fingerprints [28]. Keystroke dynamics have not been extensively studied from the privacy preserving authentication point of view. In [22], the authors present a method for using keystroke dynamics on mobile phones and trusted computing technologies to provide some guarantees against privacy invasive attacks. Their attack considers the situation, where the user is profiled by her keystroke patterns by some (web) applications, without the user's consent. Our method protects against this type of attack, because the samples are encrypted before sending them to the application or service provider. Furthermore, our methods also protect the templates that are stored at the server both against breaches to the server by malicious parties and from insider in the server end. This is not covered in the threat model of [22].

From the usability perspective, authentication has also received a lot of scrutiny. Security and usability have been seen as contradictory goals and in many cases this can be validated, although it is not an absolute truth [6,8]. In general, the pinnacle of usability would be that the authentication would not impose any extra interaction between the user and the system. This is captured fairly well in the concept of *implicit authentication* in [17]. This type of authentication is possible with many biometrics such as face, speaker or gait recognition. There are also systems that provide privacy-friendly implicit authentication, such as [27], which we will utilise in our constructions. The work in [27] is concentrated on profiling mobile phone users and protecting templates gathered from these and our work adapts their system to the case of keystroke dynamics.

3 Methods

This section describes our methods for measuring the keystrokes, protecting the templates and securing the authentication. We begin by briefly stating our threat models.

3.1 Threat Model

Because our system is designed for both privacy and security, we have two different goals and threat models. For privacy, we consider an *honest-but-curious* adversary at the server end. This is similar to the adversary of [27]. The adversary is bound to respect the protocol, but may try to learn additional information from the content of the messages and the internal computations that it carries out during authentication.

For security, we have a more powerful adversary as described in [13]. To break the security of the authenticated key exchange, the adversary can read, send and modify messages in transit and, if multiple authentication methods and/or factors are used, learn the secrets from these for any given client. The adversary can also learn the secret key of any given server, i.e., corrupt a server, if mutual authentication between the server and the client is provided. The results

of [13] show, that the authenticated key exchange protocol is still secure, if at least one factor or method remains unbroken or not corrupted.

3.2 Measuring Keystrokes

As can be seen from Sect. 2, there are many ways to measure the keystroke dynamics of the user. Our approach started with the work from [1], where four different features were measured from the keystrokes: the entered string of characters and down-down, up-down and down-up times of each keystroke. To simplify our approach, we decided to test whether we could achieve good performance with only some of the timing measurements.

We chose to use the down-up times, up-down times and the entered string of characters in our solution. The entered string of characters is an obvious choice as it prevents the adversary from typing in just any combination of characters with correct timings. However, we do not assume that this string of characters is kept secret and thus it is not just another password. The down-up time measures how long each individual character was held down and up-down time measures how long was the difference from letting go of previous key to pressing down the next one. We did not use down-down times since they aren't independent from the down-up and up-down times.

Furthermore, because the methods from [27] measure the distance of the sampled vector to each of the template vectors, we could not use the averaging of the times in our templates, which was done in [1]. In our templates, each measurement was retained, in order to measure the sample against all these values. This then resulted in slightly larger templates than in the original paper, where averages could be used.

3.3 Protecting Privacy

The privacy protection of the templates was done according to the implicit authentication scheme of [27]. Each of the templates was protected with two methods, each component separately with partially homomorphic encryption and with order-preserving symmetric encryption (OPSE). The partially homomorphic part of the template enabled computing the distance from the average absolute deviation (AAD) of the sample from the template values and the OPSE encrypted part enabled comparisons between the sample and some threshold values, which resulted in the final score that was used for deciding the successfulness of the authentication attempt.

Like in [27], we used Paillier encryption [23] for the homomorphic encryption and the results of Boldyreva *et al.* from [4,5] for the order-preserving encryption. These are needed to protect the privacy of the templates and samples both in transit between the client and server and also at the server end from the honest-but-curious server itself. The details of the application of these are postponed to Sect. 3.5, where our system is described in more detail.

3.4 Providing Security

The security of our authentication is based on the multi-factor authenticated key exchange (MFAKE) protocol from [13]. The protocol specifies three subprotocols, one for each type of factor (passwords, biometrics and tokens). Each of these can be run in parallel as many times as there are different authentication methods. In our work, we used the keystroke dynamic as the first line of authentication. If the biometric measurement was successful, the authentication proceeded with the key exchange. If it was unsuccessful, the user could enter a password and if it was correct the key exchange proceeded with all the information linked to the key. This is the approach that is suggested also in [27] to be utilised with the implicit authentication scheme. In a real world implementation the system could enhance its performance by learning also from the false negatives (i.e. cases where the implicit authentication fails, but password authentication succeeds). If also the password authentication failed, the authentication was considered completely failed.

The security of the MFAKE protocol is based on tag-based authentication from [16] and this required some changes to the privacy-preserving authentication of [27]. These changes were minor and are discussed in more detail later in this paper. The changes did not affect the security of the MFAKE or the level of privacy protection.

3.5 Our Implementation

We implemented the above system using Python 2.7 with the help of some open source libraries for cryptography: `pycrypto`[1], `paillier`[2] and `pyope`[3]. We used the `getch`[4] and `clock`[5] Python functions for the keystroke timing measurements and those gave us 0.44 µs precision on the timing.

The first part of the program prompted for the user to provide a username. If this was a new username, the system asked the user to type the username in again for 9 times. This resulted in a total of ten vectors of timings that were combined into a template of the user and the username.

All the measured times were then encrypted with both the homomorphic and order-preserving encryptions and then sent for the server to store as the template for this given username just as in [27]. Naturally, only the measurements from the successful replication of the same username were stored.

The testing part was implemented in a way that the user was requested to type their chosen username and the typing pattern was matched against the template. Two thresholds were chosen from the empirical data to provide approximately 90 % acceptance rate. This test was repeated 10 times for each participant. The scores were computed following the methods presented in [27] as follows.

[1] https://www.dlitz.net/software/pycrypto/.
[2] https://github.com/mikeivanov/paillier.
[3] https://github.com/psviderski/pyope.
[4] https://pypi.python.org/pypi/getch.
[5] https://docs.python.org/2/library/time.html.

Let $HE = (KeyGen^{HE}, E^{HE}, D^{HE})$ be a homomorphic encryption scheme, such as Paillier, and $OPSE = (KeyGen^{OPSE}, E^{OPSE}, D^{OPSE})$ be an order preserving symmetric encryption scheme. During system setup $KeyGen^{HE}$ and $KeyGen^{OPSE}$ are used to generate the HE key pair (pk, sk) and the $OPSE$ key k.

The user profile is a pair $U = (DU, UD)$, where $DU = (V_1, \ldots, V_n)$ and $V_i = (v_i(1), \ldots, v_i(10))$, where n is the length of the username and $v_i(j)$ is the time for down-up keystroke for the j^{th} measurement of the i^{th} character in the template. Similarly, $UD = (W_1, \ldots, W_n)$ and $W_i = (w_i(1), \ldots, w_i(10))$ is defined as above, but for up-down timings. The accumulated user profiles contain tuples

$$\left(E_{pk}^{HE}(v_i(j)), E_k^{OPSE}(v_i(j)), E_{pk}^{HE}(w_i(j)), E_k^{OPSE}(w_i(j)\right)$$

for $j = 1, \ldots, 10$ and $i = 1, \ldots, n$.

The server can precompute the AAD, which will be used in the comparisons between the template and the sample. The AAD from the encrypted values can be computed as follows:

$$E_{pk}^{HE}(AAD(V_i) \times n) = \sum_{i=1}^{n} \left| E_{pk}^{HE}(v_i(j)) - E_{pk}^{HE}(\text{Med}(V_i)) \right|$$

From the above it is straightforward to use the homomorphic properties of HE and remove the scalar factor n by multiplying with n^{-1}. The $\text{Med}(V_i)$ denotes the median of the values in V_i and this can be found by comparing the OPSE encrypted values in the templates. This is done similarly to the values in W_i.

The actual scoring algorithm for authentication proceeds in two stages. First, the samples taken by the client are encrypted into tuples

$$e_i(t) = \left(E_{pk}^{HE}(c_{\text{tag}} \times v_i(t)), E_k^{OPSE}(v_i(t)), E_{pk}^{HE}(c_{\text{tag}} \times w_i(t)), E_k^{OPSE}(w_i(t))\right)$$

for each variable v_i. We extended the system from [27] by multiplying each of the vector components of the measurement with the tag c_{tag} computed by the client. This was included in the computation of the template to provide tag-based authentication, which is the basis for MFAKE and all its subprotocols. The tags are required in order to guarantee the security of the system in MFAKE by binding all the utilised authentication factors and methods to a single session of the MFAKE protocol. To initialise the MFAKE protocol the client and server perform an unauthenticated Diffie-Hellman key exchange [11]. The tag is the hash value of the original Diffie-Hellman key, the identities of the two communicating parties and all the messages exchanged by that time. We computed the tag using the SHA-256 hash function.

In the server end, the server computes its own tag s_{tag} from its view on the key and the transcript of messages. Then it can compute the inverse of s_{tag} and multiply the vector components with that value to get to the "untagged" values that are used for comparison. This could be done in a similar fashion as with the AAD computation described earlier due to homomorphism ($E_{pk}^{HE}(s_{\text{tag}}^{-1} \times$

$c_{\text{tag}} \times v_i(t))$, which equals $E_{pk}^{HE}(v_i(t))$ if the tags agree). After that, the server computes the end points of the allowable interval for up-down times by

$$E_{pk}^{HE}(b_l^i(t)) = E_{pk}^{HE}(v_i(t)) - E_{pk}^{HE}(AAD(V_i))$$
$$E_{pk}^{HE}(b_h^i(t)) = E_{pk}^{HE}(v_i(t)) + E_{pk}^{HE}(AAD(V_i))$$

and similarly $E_{pk}^{HE}(c_l^i(t))$ and $E_{pk}^{HE}(c_h^i(t))$ for down-up times from the values in W_i.

However, the server does not know how these can be compared with the template values it holds. Thus in the second phase the server delivers the values $E_{pk}^{HE}(b_l^i(t)), E_{pk}^{HE}(b_h^i(t)), E_{pk}^{HE}(c_l^i(t))$ and $E_{pk}^{HE}(c_h^i(t))$ to the client. The client decrypts these getting $b_l^i(t), b_h^i(t), c_l^i(t)$ and $c_h^i(t)$. These are encrypted with E_k^{OPSE} and returned to the server. Now the server can count the number of occurrences in the OPSE encrypted template that fall within the interval defined by these two values. This number is then compared to the given threshold values.

If the matching was within the thresholds, the system would immediately compute the authenticated key with the MFAKE [13] protocol. To this end the client and server combined the transcripts of the message contents that had been sent between them and hashed them with SHA-256 to generate the key. If the username typing pattern was not recognised, the system would prompt for a password. This password was tested through the MFAKE protocol and again the authenticated key exchange would be completed, if a correct password was entered. Otherwise, the authentication failed and a common key was not established between the client and the server.

4 Empirical Results

In the user tests, we tested only the keystroke biometric and left the password authentication part out. Thus, the user taught the system for keystroke dynamics without using the secondary password authentication. This was reasonable, because there is a lot of research on the benefits and weaknesses of password authentication and how users perform with them. Our work would not bring new insights on that front and we decided to focus on the effect of privacy protection on the accuracy of the keystroke dynamics.

We tested our implementation with two groups of volunteers in two different locations. Users were volunteers from academic institutions. The first group had 11 participants and the second 9 participants, making the total number of test subjects 20. We ran the tests on both Windows and Linux operating systems on two different laptops (one for each location).

Each of the subjects chose a username of 8–12 characters. Although in [1] it is stated that less than ten characters is not sufficient for good accuracy, we decided to let the subjects use also shorter usernames that might be more familiar to them. The choice was not restricted in any way and the subjects were free to choose completely new usernames or ones that they used in other systems.

The tests were performed on hardware that was not the same in both locations, but that was available for the authors at the time. Some participants asked to use their own keyboards to better reflect their real typing patterns. We allowed this as our method could be used with any keyboard and a real implementation would be used with wide variety of keyboards. Most of the participants used the keyboard on the laptop.

The testing was divided into two parts with first the learning part and then testing users after a short period of time. This interval varied from 2–7 days due to the schedules of some participants not permitting them to participate at a specific time. The results are further analysed and discussed in Sect. 5.

One third of the subjects chose an 8 character username, probably due to legacy reasons from Unix machines. Others chose longer usernames with 10 and 12 characters being in the second place for popularity. The length of the username did not have any significant impact on the accuracy of the measurements, although the sample sizes are very small for reliable statistical analysis.

4.1 Authentication

In Fig. 1 we can see the scores from the first phase, the learning phase. These scores were used to decide what would be feasible thresholds for authentication. We set our target to 90 % accuracy for the authentication. From the data we computed for down-up time 3.5 as the threshold. For up-down time the threshold

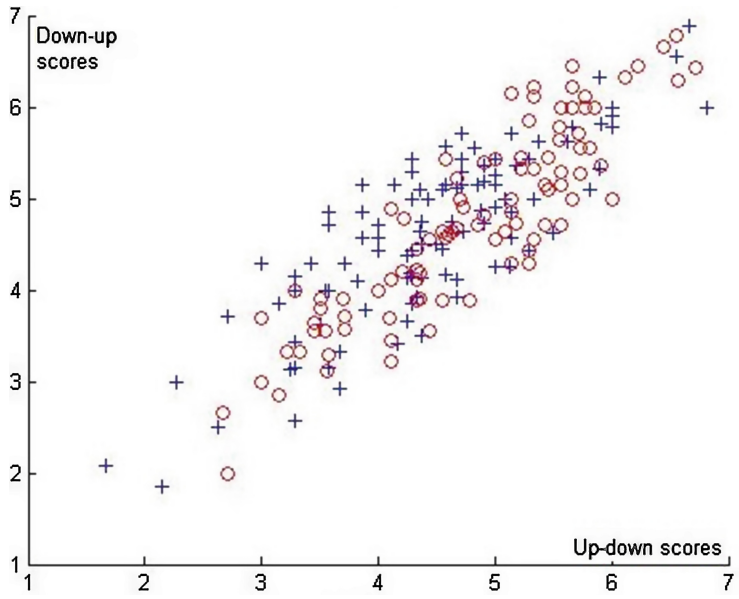

Fig. 1. Scores from the learning phase. Scores from the two different locations are differentiated with crosses and circles.

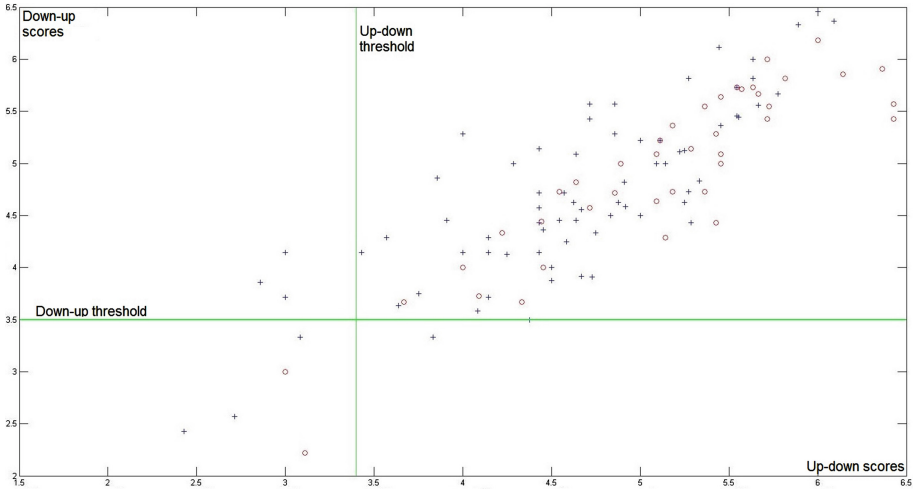

Fig. 2. Scores from the testing phase. Scores from the two different locations are differentiated with crosses and circles. The lines denote the values of the thresholds.

was 3.4 and in order to successfully authenticate the user had to pass both thresholds while typing the correct username.

In Fig. 2 we can see the results from the second phase of tests. The horizontal and vertical lines denote the thresholds chosen from the data of the first phase for the up-down and down-up times respectively. During the second test phase the users tried to authenticate themselves against the chosen thresholds. The participants in the first location managed to authenticate themselves correctly with 91 % success rate with a standard deviation $\sigma = 7\,\%$. In the second location success rate was over 92 % and the a standard deviation was $\sigma = 10\,\%$. In total the authentication success rate was slightly over 91.5 % with a standard deviation of $\sigma = 9\,\%$.

4.2 Impersonation

We also carried out a small impersonation test together with the actual authentication test. The test was conducted on three different usernames from the usernames of the other test group. This simulated the situation of an attacker trying to impersonate someone that she does not know. This is arguably the hardest case of impersonation and attackers that are more familiar with the victim or that can try to learn the typing pattern by some form of observation could have better chances of success. In any case, we saw this as a good indicator of the security of our system despite the small sample sizes.

After the participant had completed the test with her own username, we displayed a username from the other group of participants and asked the current participant to type that username 5 times. After each attempt the scores from that attempt were displayed to the participant attempting the impersonation. We did this for 3 different randomly selected usernames for each participant.

The impersonation challenge succeeded 42 times out of 195 tries so the success rate was little below 22 %. The success probability between different kind of usernames varied greatly. The variance between the results of impersonation attempts usernames is $\sigma^2 = 8.1\,\%$ and thus the standard deviation is $\sigma \approx 28.5\,\%$. In the best case the impersonation succeeded 9/10 of the time and in the worst case it succeeded with 0/15 attempts.

Because the challenges were selected at random without any effort to balance the occurrences of different usernames, the results are only indicative of the security and not easily comparable. With seven out of 18 usernames there were no successful impersonation attempts (two usernames were not assigned for impersonation at all). On the other hand, none of the three usernames, that were impersonated by four different participants (total of 20 attempts) withstood all attempts. The length of the usernames did not have a significant impact on the security against impersonation.

5 Discussion

Our results show that this type of authentication can be used and it provides usability, security and privacy protection to the user. However, more care needs to be taken to make the impersonation harder for all usernames as the results of Sect. 4.2 clearly show. The weakest usernames were too prone to impersonation to provide any meaningful protection against adversaries. On the other hand, more than half of the usernames were impersonated less than 10 % of the time, when the acceptance threshold was tuned for 90 %.

In any case, our sample of some 20 people is too small to draw definitive conclusions. Especially, if the length of the username would be considered as a factor in both accuracy and security, then a much larger pool of users would be needed. If the increased length would improve security and accuracy, it could act as a positive sign for continuous authentication, where the typing would be measured continuously and authentication would be based on the totality of all things typed. Of course in this type of authentication the content would no longer matter for the authentication unlike in our case with usernames.

One improvement could be to use some more recent methods of measuring keystroke dynamics. This would most likely require readjusting the thresholds to get the same level of accuracy. It might also generate larger templates and decrease the overall performance of the system if more different values need to be stored in the template. Thus, there is a trade-off that needs to be considered, if future work is built upon this system.

Our inclusion of the tag in the keystroke dynamics vectors was necessary to fit the methods of [27] to those of [13]. The tags provide security in the original MFAKE protocol and are based on results of [16]. However, in our case, due to homomorphism, these tags do not protect against a man-in-the-middle tag change. That is, an active attacker could change the original tag t to another tag t' for the keystroke dynamics. The full MFAKE protocol mitigates against this by including in the final computation not only the tag, but also all communications

between the server and the client and thus, this change of tag can only lead into a denial of service (i.e. failing authentication), which an active attacker can always do against MFAKE.

Furthermore, the method of [27] also only provides security against honest-but-curious adversaries and as such may constitute even more serious threat. There is a version of the protection method of [27] that is targeted against malicious attackers, but it does not provide much more security for the overhead that it generates and thus we decided not to use it for our work. In a more optimised implementation, the overhead might not become an issue.

The system could be improved also in several other ways. First of all, one could apply more recent partially homomorphic or even fully homomorphic encryption (FHE) schemes, e.g., [15] to provide more versatility to the comparison methods. This would mean that also other biometrics than keystroke dynamics based on simple timing would probably be applicable. Also the order-preserving encryption could be generalised with other property preserving encryption schemes such as [24]. Such a system could offer better privacy protection. Especially with FHE the adversary could not learn even the relative order of the different values in the tempalates and samples. This would make the OPSE encrypted parts of the vector unnecessary and make the comparison much simpler. In this way it would increase the efficiency of the system provided that the FHE is efficient.

Keystroke dynamics provide also an interesting opportunity for continuous authentication while typing. This type of authentication has been discussed for example in [2]. The methods described in this paper are not yet efficient enough for continuous use. This provides an excellent venue for further research, because the system described in [2] does not offer privacy protection at the same level as our work. Also protecting other types of continuous authentication systems such as facial images with privacy safeguards is an important topic. Generalising the results of [27] to this direction would be an interesting topic of research.

Our solution to this privacy-friendly, usable and secure authentication is also generalisable to many other biometrics. Our specific implementation only assumes that the measurements are given in numeric form as a vector and an identification threshold that can be adjusted through experiments or taken from the literature, if such can be found for the biometric method at hand. This makes it suitable to many other biometric authentication applications. The main constraint is that it can only measure the distance of the sample measurement from the template with the AAD metric. This provides a good venue for further research on the topic, e.g., finding ways to use other metrics in a similar manner.

6 Conclusion

In this paper we have shown that it is possible to realise a secure, usable and privacy-friendly authentication from keystroke dynamics. Our method results in fairly good performance even with a simple biometric measurement. The security and privacy features are also provided, although for some usernames the impersonation was very easy.

Further development could make this type of authentication even applicable in a continuous manner, measuring the user constantly while she is typing and still assuring privacy and security. It is fairly straightforward to generalise the system to work with other biometrics that use simple distance metrics for comparison and even more complicated systems can be realised with further research. On the other hand, the overall efficiency of the system should be improved through optimised implementation.

Acknowledgements. We would like to thank Tekes – the Finnish Funding Agency for Innovation, DIMECC Oy, and the Cyber Trust research program for their support of this research. Furthermore, we thank all the volunteers that participated in the experimental study for their time and also the anonymous reviewers for their valuable comments and suggestions that helped in improving this paper.

References

1. Araújo, L.C., Sucupira, L.H., Lizarraga, M.G., Ling, L.L., Yabu-Uti, J.B.T.: User authentication through typing biometrics features. IEEE Trans. Sig. Process. **53**(2), 851–855 (2005)
2. Arias-Cabarcos, P., Almenarez, F., Trapero, R., Diaz-Sanchez, D., Marin, A.: Blended identity: pervasive IdM for continuous authentication. IEEE Secur. Priv. **13**(3), 32–39 (2015)
3. Banerjee, S.P., Woodard, D.L.: Biometric authentication and identification using keystroke dynamics: a survey. J. Pattern Recognit. Res. **7**(1), 116–139 (2012)
4. Boldyreva, A., Chenette, N., Lee, Y., O'Neill, A.: Order-preserving symmetric encryption. In: Joux, A. (ed.) EUROCRYPT 2009. LNCS, vol. 5479, pp. 224–241. Springer, Heidelberg (2009). doi:10.1007/978-3-642-01001-9_13
5. Boldyreva, A., Chenette, N., O'Neill, A.: Order-preserving encryption revisited: improved security analysis and alternative solutions. In: Rogaway, P. (ed.) CRYPTO 2011. LNCS, vol. 6841, pp. 578–595. Springer, Heidelberg (2011). doi:10.1007/978-3-642-22792-9_33
6. Bonneau, J., Herley, C., van Oorschot, P., Stajano, F.: The quest to replace passwords: a framework for comparative evaluation of web authentication schemes. In: 2012 IEEE Symposium on Security and Privacy (SP), pp. 553–567, May 2012
7. Bonneau, J., Schechter, S.: Towards reliable storage of 56-bit secrets in human memory. In: 23rd USENIX Security Symposium (USENIX Security 2014), pp. 607–623 (2014)
8. Braz, C., Robert, J.M.: Security and usability: the case of the user authentication methods. In: Proceedings of the 18th International Conferenceof the Association Francophone d'Interaction Homme-Machine, pp. 199–203. ACM (2006)
9. Brown, M., Rogers, S.J.: User identification via keystroke characteristics of typed names using neural networks. Int. J. Man Mach. Stud. **39**(6), 999–1014 (1993)
10. Clarke, N.L., Furnell, S.: Authenticating mobile phone users using keystroke analysis. Int. J. Inf. Secur. **6**(1), 1–14 (2007)
11. Diffie, W., Hellman, M.E.: New directions in cryptography. IEEE Trans. Inf. Theory **22**(6), 644–654 (1976)
12. Dodis, Y., Ostrovsky, R., Reyzin, L., Smith, A.: Fuzzy extractors: how to generate strong keys from biometrics and other noisy data. SIAM J. Comput. **38**(1), 97–139 (2008)

13. Fleischhacker, N., Manulis, M., Sadr-Azodi, A.: Modular design and analysis framework for multi-factor authentication and key exchange. In: Cryptology ePrint Archive, Report 2012/181 (2012). http://eprint.iacr.org/
14. Gaines, R.S., Lisowski, W., Press, S.J., Shapiro, N.: Authentication by keystroke timing: some preliminary results. Technical report, DTIC Document (1980)
15. Gentry, C.: A fully homomorphic encryption scheme. Ph.D. thesis, Stanford University (2009)
16. Jager, T., Kohlar, F., Schäge, S., Schwenk, J.: Generic compilers for authenticated key exchange. In: Abe, M. (ed.) ASIACRYPT 2010. LNCS, vol. 6477, pp. 232–249. Springer, Heidelberg (2010). doi:10.1007/978-3-642-17373-8_14
17. Jakobsson, M., Shi, E., Golle, P., Chow, R.: Implicit authentication for mobile devices. In: Proceedings of the 4th USENIX Conference on Hot Topics in Security, p. 9. USENIX Association (2009)
18. Juels, A., Sudan, M.: A fuzzy vault scheme. Des. Codes Crypt. **38**(2), 237–257 (2006)
19. Mäntyjärvi, J., Lindholm, M., Vildjiounaite, E., Mäkelä, S.M., Ailisto, H.: Identifying users of portable devices from gait pattern with accelerometers. In: IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005, Proceedings (ICASSP 2005), vol. 2, pp. ii/973–ii/976. IEEE (2005)
20. Monrose, F., Rubin, A.: Authentication via keystroke dynamics. In: Proceedings of the 4th ACM Conference on Computer and Communications Security, pp. 48–56. ACM (1997)
21. Monrose, F., Rubin, A.D.: Keystroke dynamics as a biometric for authentication. Future Gener. Comput. Syst. **16**(4), 351–359 (2000)
22. Nauman, M., Ali, T., Rauf, A.: Using trusted computing for privacy preserving keystroke-based authentication in smartphones. Telecommun. Syst. **52**(4), 2149–2161 (2013)
23. Paillier, P.: Public-key cryptosystems based on composite degree residuosity classes. In: Stern, J. (ed.) EUROCRYPT 1999. LNCS, vol. 1592, pp. 223–238. Springer, Heidelberg (1999). doi:10.1007/3-540-48910-X_16
24. Pandey, O., Rouselakis, Y.: Property preserving symmetric encryption. In: Pointcheval, D., Johansson, T. (eds.) EUROCRYPT 2012. LNCS, vol. 7237, pp. 375–391. Springer, Heidelberg (2012). doi:10.1007/978-3-642-29011-4_23
25. Rathgeb, C., Uhl, A.: A survey on biometric cryptosystems and cancelable biometrics. EURASIP J. Inf. Secur. **2011**(1), 1–25 (2011)
26. Saevanee, H., Bhattarakosol, P.: Authenticating user using keystroke dynamics and finger pressure. In: 6th IEEE Consumer Communications and Networking Conference, CCNC 2009, pp. 1–2. IEEE (2009)
27. Safa, N.A., Safavi-Naini, R., Shahandashti, S.F.: Privacy-preserving implicit authentication. In: Cuppens-Boulahia, N., Cuppens, F., Jajodia, S., Abou El Kalam, A., Sans, T. (eds.) SEC 2014. IFIP AICT, vol. 428, pp. 471–484. Springer, Heidelberg (2014). doi:10.1007/978-3-642-55415-5_40
28. Tulyakov, S., Farooq, F., Mansukhani, P., Govindaraju, V.: Symmetric hash functions for secure fingerprint biometric systems. Pattern Recogn. Lett. **28**(16), 2427–2436 (2007)