Type^2: A Secure and Seamless Biometric Two-Factor Authentication Protocol Using Keystroke Dynamics

Pia Bauspieß[1,2][0000−0003−0225−1674], Patrick Bours[2][0000−0001−5562−6957], Christian Rathgeb[1][0000−0003−1901−9468], and Christoph Busch[1,2][0000−0002−9159−2923]

[1] Hochschule Darmstadt, da/sec biometrics and security research group, Department of Computer Science, 64295 Darmstadt, Germany
[2] NTNU – Norwegian University of Science and Technology, Department of Information Security and Communication Technology, 7034 Trondheim, Norway

Abstract. Password-based user authentication comes with impersonation risks due to poor quality passwords or security breaches of service providers. An additional layer of security can be provided to the authentication through keystroke dynamics, i.e., measuring and comparing users' typing rhythm for their password. While this two-factor authentication is efficient and unobtrusive, the privacy of the biometric characteristics must be ensured. Therefore, we present the Type^2 protocol for secure two-factor authentication based on keystroke dynamics, where the anomaly detection of the latter is executed in the encrypted domain. In an experimental evaluation, we show that our proposed protocol achieves real-time efficiency with an overhead of less than 130 milliseconds compared to password-only authentication.

Keywords: Keystroke dynamics · fully homomorphic encryption · two-factor authentication

1 Introduction

Reliable user authentication is an important building block in an increasingly digital world [12]. In many authentication scenarios, it is important to ensure that data is disclosed only to the intended receiver, and not to a third party using the receiver's device with their stolen authentication credentials. This applies, e.g., to the disclosure of medical data, but also the agreement of legal contracts or financial transactions.

One of the most common digital authentication methods, passwords, do not inherently provide this security. Trust in password-authenticated communication can be impaired by the fact that many users choose simple passwords that are easy to brute-force [31], or their password may have been compromised by a large-scale attack on a service provider [25].

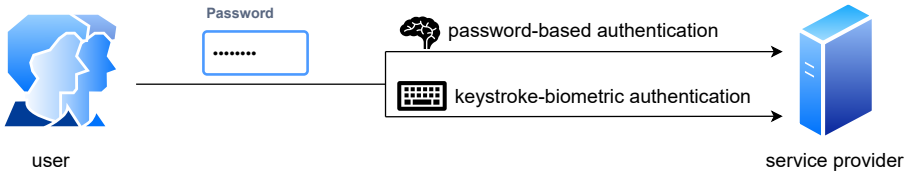

Fig. 1. Seamless integration of biometric authentication using keystroke dynamics.

Biometrics can make such impersonation attacks harder and provide additional confidence in the authentication. In particular, one efficient and unobtrusive way of adding a second trust factor to password-based authentication are keystroke dynamics, i.e., measuring and comparing the users' typing rhythm for their password [24]. Thereby, a second authentication factor can be derived from the already provided password through extracting the timing information from the user's typing rhythm. This motivation is visualized in Figure 1.

However, such biometric characteristics are classified as sensitive by the European Union's General Data Protection Regulation (GDPR) [13] and must be protected according to the ISO/IEC 24745 international standard on biometric information protection [20]. The latter defines the following three requirements for secure biometric authentication: *i) unlinkability*, two protected biometric templates stored in different applications cannot be linked to the same subject, *ii) renewability*, new templates can be created from the same biometric instance without the need to re-enrol, and *iii) irreversibility*, it is impossible to retrieve original templates given only protected templates. In addition, the biometric performance (i.e., accuracy) as well as the computational performance of the unprotected system should be preserved.

In this work, we present the Type^2 protocol for secure two-factor authentication based on keystroke dynamics, where the biometric comparisons are executed in the encrypted domain. To this end, *Fully Homomorphic Encryption* (FHE) [15] is applied to the biometric features both during enrolment and verification. More concretely, we investigate the compatibility of established anomaly detectors for keystroke dynamics [24], and present an analysis of the applicability and feasibility of FHE to these detectors. Further, we give a comprehensive security analysis of Type^2 with regard to adaptations that have to be made in order to apply FHE to different detectors. We evaluated our Type^2 protocol experimentally on publicly available data [24] and libraries [29]. Our proposed protocol can be instantiated with detectors that achieve real-time user authentication at an overhead of less than 130 milliseconds per authentication attempt.

The rest of this article is structured as follows: Section 2 discusses works that are closely related to ours, before Section 3 gives more technical background information. Our protocol and main contribution is presented in Section 4, together with its experimental evaluation given in Section 5. Finally, we draw our conclusions in Section 6.

2 Related Work

One of the first to discuss the application of homomorphic encryption to keystroke dynamics were Šeděnka et al. [30]. In their work, the authors indicate that their key generation protocol could also be instantiated with FHE, but that they refrained from this choice due to the significant computational overhead of FHE, in particular with respect to the schemes and implementations that were available in 2014. Therefore, they use additively homomorphic encryption only [10], which only allows for additions of ciphertexts, and therefore limits the complexity of detectors. In their evaluation, they use an in-house data set that does not allow for reproducibility of their research. Nevertheless, we can estimate a comparison of the efficiency, as the authors of [30] achieve execution times in the magnitude of minutes, whereas our Type^2 protocol can be executed in the order of milliseconds.

More recently, Acar et al. [1] presented a privacy-preserving multi-factor authentication system named *PINTA*, where they consider keystroke dynamics as one potential authentication factor. The authors evaluate their protocol on the established and publicly available keystroke dynamics dataset provided by [24], in addition to other modalities such as mouse movements. Their multi-factor authentication protocol uses fuzzy hashing in combination with FHE, which impairs the accuracy of the system. Furthermore, the FHE scheme used by [1] is the BFV [8, 14] encryption scheme, which operates on integers and therefore requires a quantisation of keystroke dynamic features. The computational cost of their authentication decision was evaluated at around 370 milliseconds.

The most closely related work to ours was presented by Loya et al. [26] in 2021. In their work, the authors evaluate a neural network with differential privacy during the training process, while the keystroke dynamic features are protected using the CKKS [9] encryption scheme. This is the same FHE scheme we will use for our experimental evaluation. In addition, the work by [26] utilizes the same established data set for keystroke dynamic evaluation provided by [24]. However, the execution times of [26] are not applicable for real-time applications, as they are no lower than 14 seconds.

Table 1. Qualitative comparison of related works on keystroke dynamic authentication with (fully) homomorphic encryption.

	Encryption Scheme	Accuracy Preservation	Performance Preservation	Post-quantum Security
Šeděnka et al. 2014 [30]	DGK	✓	✗	✗
Acar et al. 2019 [1]	BFV	✗	✓	✓
Loya et al. 2021 [26]	CKKS	✓	✗	✓
Ours	CKKS	✓	✓	✓

3 Background

3.1 Password-Authenticated Key Exchange

For the first component of our Type^2 protocol, *Password-Authenticated Key Exchange* (PAKE) [21] is used. Compared to traditional hashing and salting of passwords, PAKE provides additional security against offline attacks and can be considered the state-of-the-art in password authentication. Popular approaches include the SRP protocol [32] used among others in the Apple iCloud, or the more recent the OPAQUE [21] protocol. Similar to biometric authentication, a PAKE protocol is defined through a registration phase, where the user's password information is enrolled into the system in a protected manner, and an authentication phase, where a cryptographic key is exchanged successfully if and only if the correct password is provided again. The PAKE component in our protocol can be easily exchanged and we therefore do not focus on it further for the scope of this work, but refer the reader to the works of [32] and [21] directly.

3.2 Fully Homomorphic Encryption

FHE allows for the evaluation of arithmetic circuits on encrypted data [15] and has been determined to fulfil the ISO/IEC 24745 [20] requirements for biometric information protection [33, 16, 7, 5]. For the scope of our work, we define an FHE scheme through the following algorithms:

- $(sk, pk) \leftarrow \mathit{KeyGen}(1^\lambda)$: on input of the security parameter λ, generates a secret key sk and public key pk, where pk includes the homomorphic evaluation keys.
- $c_m \leftarrow \mathit{HomEnc}(pk, m)$: on input of pk and a message m, outputs a ciphertext c_m.
- $c_{f(m_1,m_2)} \leftarrow \mathit{HomEval}(pk, f, c_{m_1}, c_{m_2})$: on input of pk, a public function f, and two ciphertexts c_{m_1} and c_{m_2}, outputs an encryption $c_{f(m_1,m_2)}$ of the evaluation of f on the underlying plaintext messages m_1 and m_2.
- $m' \leftarrow \mathit{HomDec}(sk, c_m)$: on input of sk and ciphertext c_m, outputs a message m'.

These operations can be applied to vectorized data, where all evaluations will be performed element-wise, yielding an improvement in terms of computational overhead [7]. It holds that $\mathit{Dec}(sk, \mathit{HomEval}(pk, f, c_{m_1}, c_{m_1})) = f(m_1, m_2)$ [9].

3.3 Keystroke Dynamics

In this work, we focus on keystroke dynamic features that can be extracted from password timings measured using the same keyboard for each authentication attempt. For a given password, this feature set will always be of fixed length n, and the order of typed letters will be the same, easing the task of anomaly detection. Different features that can be measured from password typings are [24]:

(i) keydown-keydown time: time interval between a key is pressed and the consecutive key is pressed, *(ii) keyup-keydown time:* time interval between a key is released and the consecutive key is pressed, and *(iii) hold time:* time interval between a key is pressed and the same key is released.

Using these timings, the typical typing pattern of a user is established during the enrolment or training phase. In this step, the mean vector over a set of timing vectors is stored, with additional information such as the covariance of the features. For neural network-based approaches, this step corresponds to the training of the weights. For a verification transaction, a fresh probe timing vector is captured from the data subject. The probe features are compared against the stored reference template and a distance score or *anomaly score* [24] is computed. Using a pre-defined threshold, the anomaly score can be used to grant or deny the subject access to the system. The combined algorithms of enrolment and verification are referred to as an *anomaly detector* in the following.

4 Proposed System

In this Section, we describe the Type^2 protocol with FHE protection and necessary modifications and limitations for all of the anomaly detectors described in [24]. An overview of our proposed system is given in Figure 2.

In the enrolment phase, both the password and biometric reference of a subject are enrolled into the system. For the password w, the PAKE registration is performed according to the chosen approach [32, 21]. Additionally, an FHE key pair (sk, pk) is generated by the key server, and the public key pk is shared with the other parties. We assume that an attacker has access to the public key. The client uses pk to encrypt the keystroke timing features after the reference vector r has been established in the training process. The *Computation Server* (CS) stores $c_r \leftarrow \mathit{HomEnc}(pk, r)$.

In the first step of the verification protocol, the subject provides a password w', which is input to the PAKE protocol. If the PAKE authentication phase is successful, the system proceeds to the keystroke anomaly detection. For an optimized user experience, both processes can also be run in parallel. Using the timing features t' extracted from w', the client computes the probe ciphertext $c_p \leftarrow \mathit{HomEnc}(pk, t')$ and sends it to the CS. Using the encrypted reference template c_r corresponding to the biometric claim, the CS computes the detector $d(c_r, c_p)$, and sends d to the key server. Here, d can be decrypted and the threshold comparison of the decrypted anomaly score against threshold τ is computed. The system outputs a bit $b = 1$ if the anomaly score is smaller than τ, and $b = 0$ otherwise.

4.1 Adversary Model

In our work, we consider all parties to operate in the *semi-honest adversary model*. In this model, the participating parties do not deviate from the given protocol, but may aim to collect information that is available to them. It can

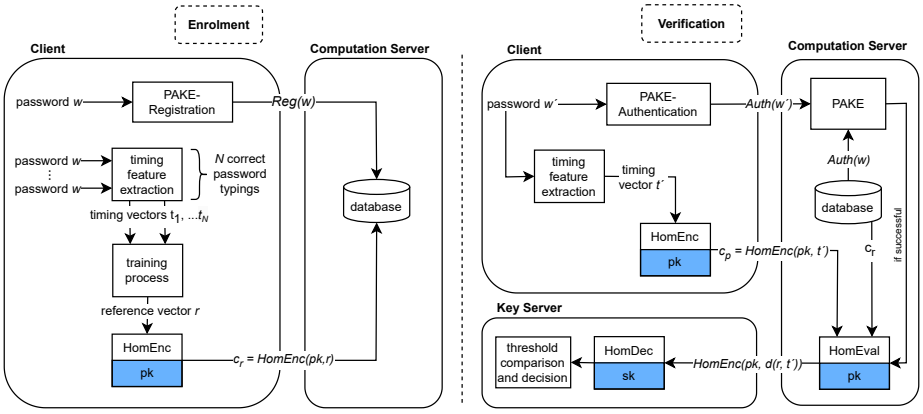

Fig. 2. Enrolment (left) and verification (right) transactions in the Type^2 protocol.

be argued that a more realistic model is given through the *malicious adversary model*, where parties are allowed to deviate from the given protocol to gain further information. This model has been discussed in the context of biometric template protection [4], where zero-knowledge proofs are applied for the protection against malicious adversaries. Our proposed Type^2 protocol is compatible with such proofs, however, we do not focus further on malicious adversaries in our work. We assume that the capture process of the timing features takes place in a controlled environment during enrolment, resulting in trusted reference vectors. During verification, the system may be confronted with presentation attacks. However, this work focuses on the application of FHE to keystroke dynamic features in a manner that does not alter the unencrypted accuracy of the system.

4.2 Euclidean Detector

The squared Euclidean distance used in the Euclidean detector [11] has been studied for FHE-based template protection for other biometric modalities, mostly for face [5, 7]. As the square-root operation is not supported by FHE, the squared Euclidean distance is preferred to the original Euclidean distance. During the enrolment phase, [24] describe that the mean vector over the set of training vectors is computed and stored as reference vector. As the enrolment is considered an offline process, the mean vector is computed on the unencrypted training vectors. Then, the client enrolls a subject by encrypting the mean reference vector r as $c_r \leftarrow \mathit{HomEnc}(pk, r)$, and sends c_R to CS.

For a verification transaction, the client encrypts the probe feature vector p as $c_p \leftarrow \mathit{HomEnc}(pk, p)$, which is sent to CS. Here, CS computes the squared Euclidean distance

$$d_{Euclid}(r, p) = \sum_{i=0}^{n-1} (r_i - p_i)^2 \tag{1}$$

as established in the recent literature [7]: the two ciphertexts c_r and c_p are subtracted, yielding an element-wise subtraction of their elements. The resulting

vector is multiplied with itself, corresponding the square of elements in the vector. To facilitate the computation of the sum over the packed vector, the established rotate-and-add technique is applied [7]. The total cost of FHE operations required for the Euclidean detector is summarized in Table 2.

4.3 Normed Euclidean Detector

The normed Euclidean detector expands upon the Euclidean detector through normalizing the final anomaly score, i.e., dividing it by the multiplied norm of the probe and reference feature vectors [6]. During the enrolment phase, the norm of the reference template is computed and encrypted to an additional ciphertext $c_r^{||\cdot||} \leftarrow \mathit{HomEnc}(pk, ||r||)$, which is sent to CS together with the encrypted reference template c_r, and both ciphertexts are stored at CS.

During verification, the client computes $c_p^{||\cdot||} \leftarrow \mathit{HomEnc}(pk, ||p||)$ in addition to c_p. The computation of the Euclidean distance follows the description in Section 4.2. In addition to the Euclidean distance, one homomorphic multiplication $c_p^{||\cdot||} \cdot c_r^{||\cdot||}$ is performed. Both the encrypted squared Euclidean distance and the encrypted multiplied norms are sent to the key server for decryption, which calculates the final anomaly score. Ideally, the division would also be computed in the encrypted domain. However, division is not directly supported by FHE operations. In the following, we note the described approach as *approach A* and describe a second option yielding a more private computation (*approach B*).

For a fully private computation of the normed Euclidean distance *(approach B)*, the client can compute the inverted Euclidean norms $1/||r||$ and $1/||p||$ from the reference and probe feature vectors during enrolment and verification, respectively. Then, it can produce ciphertexts $c_r^{1/||\cdot||} \leftarrow \mathit{HomEnc}(pk, 1/||r||)$ and $c_p^{1/||\cdot||} \leftarrow \mathit{HomEnc}(pk, 1/||p||)$. The computation on the FHE ciphertexts described above then corresponds to

$$\frac{d(c_p, c_r)}{c_p^{||\cdot||} \cdot c_r^{||\cdot||}} = d(c_p, c_r) \cdot c_p^{1/||\cdot||} \cdot c_r^{1/||\cdot||}, \tag{2}$$

where d denotes the squared Euclidean distance described in Section 4.2. This more private computation comes at a cost of one additional homomorphic multiplication and thereby an increased multiplicative depth of the circuit. The cost for all homomorphic operations is higher for parameter sets that allow this additional circuit depth. Therefore, approach B must be expected to have higher computational workload than approach A.

4.4 Manhattan Detector

The Manhattan detector utilizes the Manhattan distance, which is another established distance metric in pattern recognition [11]. It is defined as

$$d_{Manhattan}(r, p) = \sum_{i=0}^{n-1} |r_i - p_i|. \tag{3}$$

On unencrypted data, only a bit shift is required for the computation of the absolute difference between the reference and probe feature vector elements. However, a bit shift is not an available computation in FHE. When computing on integer or float values, the computation of the absolute value corresponds to a conditional statement. The evaluation of conditional statements is by design infeasible on encrypted data, as the result of the computation needs to be known in order to evaluate the statement. Recent approaches have explored conditional statements in FHE, however, they cannot be considered feasible for real-world applications [19].

Therefore, the only encrypted computation that can be performed during verification for the Manhattan detector is the difference between the reference and probe feature vectors, i.e., $c_r - c_p$, and the absolute values and the sum are computed on the plaintext data at the key server. It can be argued that the protected computation of the difference yields an additional protection of the features, in particular during storage, but also during the comparison, as it can be challenging for an attacker to determine the original features based on the difference alone. However, the aforementioned privacy limitations apply. Further privacy protection could be given through a random negation of both probe and reference feature vectors. However, this approach would correspond to an additional multiplication of the encrypted probe feature vector during verification, thereby increasing the authentication workload.

For the filtered Manhattan distance, outliers are excluded during the training phase [22]. As the enrolment phase is computed on plaintext data however, this does not impact the application of FHE to the detector in question.

4.5 Scaled Manhattan Detector

The scaled Manhattan distance utilizes mean absolute deviation a_i of the i-th feature of the training vector as a scale factor for the final anomaly score [3]. Similarly to the normed Euclidean distance, this additional vector a can be computed on the plaintext reference vectors during enrolment. Due to the lack of a division operation in FHE, we apply the same transform as in Section 4.3 and let the client encrypt the inverse $1/a$ into a ciphertext $c_r^{1/a} \leftarrow \mathit{HomEnc}(pk, 1/a)$, which is stored at CS alongside the encrypted reference vector c_r. Then, we can express the anomaly score as

$$d^{scaled}_{Manhattan}(r,p) = \sum_{i=0}^{n-1} \frac{|r_i - p_i|}{a_i} = \sum_{i=0}^{n-1} \left| \frac{r_i - p_i}{a_i} \right| = \sum_{i=0}^{n-1} \left| (r_i - p_i) \cdot \frac{1}{a_i} \right| \quad (4)$$

and calculate the values $\frac{r_i - p_i}{a_i}$ in the encrypted domain at the following cost for a verification transaction (see Table 2): first, one encryption of c_p is computed, then one subtraction of $c_r - c_p$. Subsequently, the inverted mean absolute deviation vector $c_r^{1/a}$ is multiplied to the difference, and the result is decrypted. As in Section 4.4, the absolute values and computation of the sum must be conducted on plaintext data, as the evaluation of conditional statements such as the absolute value are not feasible on FHE-encrypted data.

For the computation of the scaled Manhattan distance, the mean absolute deviation vector a should be stored in encrypted form at CS. It can be assumed that a encodes sensitive information about the biometric reference stored at CS, and can therefore be considered to be of similar sensitivity as the feature vectors themselves. Scaling on the decryption comparison score in plaintext can therefore not be considered a secure approach.

4.6 Mahalanobis Detector

The Mahalanobis detector [11] is based on the Mahalanobis distance:

$$d_{Mahalanobis}(r,p) = (r-p)^\top S^{-1}(r-p), \tag{5}$$

where S denotes the covariance matrix over the training vectors. Both S and the mean reference vector r are computed in plaintext. Then, the following ciphertexts are computed by the client: an encryption of the mean reference feature vector c_r, and each column of the inverted covariance matrix S^{-1} into a ciphertext c_i^S, where $\{c_i^S \leftarrow \mathit{HomEnc}(pk, S_i^{-1})\}_{i=0}^{n-1}$. During verification, the client obtains and encrypts a probe feature vector and sends the resulting ciphertext c_p to CS. In the first step of the distance computation, CS computes $(r-p)^\top S^{-1}$ on the corresponding ciphertexts through one subtraction of $c_r - c_p$, and n multiplications of the resulting vector with each of the ciphertexts c_i^S. The vector-matrix multiplication is completed by computing the sum over each $(c_r - c_p) \cdot c_i^S$, which is computed as described in Section 4.2. The total cost for the Mahalanobis detector is given in Table 2. The approach to the normed Mahalanobis detector [6] follows the same procedure as the normed Euclidean detector described in Section 4.2 as approach B. In addition to the computations for the Mahalanobis distance score, the inverted probe and reference feature vector norms $c_r^{1/||\cdot||} \leftarrow \mathit{HomEnc}(pk, 1/||r||)$ and $c_p^{1/||\cdot||} \leftarrow \mathit{HomEnc}(pk, 1/||p||)$ are encrypted. Then, the final comparison score is obtained after a multiplication by both ciphertexts to the original score, i.e., $d(c_p, c_r) \cdot c_p^{1/||\cdot||} \cdot c_r^{1/||\cdot||}$. The additional encryption (of $c_p^{1/||\cdot||}$) and two multiplications can be observed in Table 2.

4.7 Nearest-neighbor Detector

The nearest-neighbor approach [18] expands the Mahalanobis detector described in Section 4.6 by computing the Mahalanobis distance to every training vector (instead of the mean reference vector), and choosing the lowest out of these comparison scores as the final outcome. Its cost with regard to FHE operations can therefore be determined as the N-fold effort of the Mahalanobis detector, where N is the number of training vectors. As discussed above, conditional statements cannot be evaluated efficiently in FHE. Therefore, all N distance scores need to be decrypted, and the lowest score is determined in the plaintext domain. The nearest-neighbour approach can therefore not be fully realized in FHE, and furthermore has an infeasible overhead in terms of the number of required FHE operations.

4.8 Neural-Network Detector

The neural network detector utilizes a simple fully connected neural network with one hidden layer. The enrolment phase corresponds to the training phase of the network, while the comparison score is achieved through inference over one probe sample [11]. This inference can be expressed as two matrix multiplications with the encrypted probe feature vector, and can therefore be computed similarly to the Mahalanobis distance. As the network only has one output node, the second multiplication corresponds to a similar vector multiplication as in Section 4.6. The total cost with regard to the originally proposed parameter choices [24] can be viewed in Table 2. The FHE protection for the auto-associative neural-network detector introduced by [18] is similar to the previously described approach with the difference of n output nodes and an additional distance computation. These additional costs can be viewed in Table 2.

4.9 Fuzzy Logic Detector

The fuzzy logic detector [17] applies a succession of logical statements, i.e., conditional statements, to classify the probe feature set instead of classic distance metric. While the reference and probe features can still be sent and stored encrypted, all computations can only be computed in plaintext due to the challenge of evaluating conditional statements on encrypted data. FHE protection can therefore not be meaningfully applied to this detector.

4.10 Outlier-Counting Detector

The outlier-counting detector presented by [17] is derived from the scaled Manhattan distance. However, the final score is a count of element-wise scores above a predefined threshold, rather than the distance scores itself. For every feature in the feature vector, a so-called z-score defined as

$$z_i = \frac{|r_i - p_i|}{\sigma_i}, \tag{6}$$

where σ_i is the standard deviation of the i-th feature calculated during the training phase. We therefore apply the same transformation as in Sections 4.3 and 4.5, and store a ciphertext $c_{1/\sigma} \leftarrow \mathit{HomEnc}(pk, 1/\sigma)$ at CS during enrolment. Here, the vector $1/\sigma$ contains all inverse standard deviations $1/\sigma_i$ for every feature i. During verification, client and CS proceed as in Section 4.5 and obtain the encrypted result of the computation $c'_z = (c_r - c_p) \cdot c_{1/\sigma}$. As argued above, neither the absolute value nor the threshold comparisons can be computed in the encrypted domain. Therefore, c'_z is decrypted and the remaining computations are executed over the plaintext vector.

4.11 One-Class Support Vector Machine Detector

For the one-class *Support Vector Machine* (SVM) detector [34], the training phase is again conducted on the unencrypted training vectors. After training is completed, the determined hyperplane h used as the separator is encrypted into a ciphertext $c_h \leftarrow \mathit{HomEnc}(pk, h)$ and stored at CS. A verification transaction then corresponds to a projection of the encrypted probe feature set p into the higher-dimensional separator space of the SVM, i.e., a matrix multiplication, the cost of which is presented in Table 2.

4.12 k-Means Detector

The application of the established k-means clustering algorithm [28] has been proposed for keystroke dynamics by [23]. In terms of the application of FHE to this detector, the approach corresponds to the Euclidean detector described in Section 4.2. For each of the k centroids, the Euclidean distance between the centroid and the probe feature vector is computed, and the closest distance is determined to be the final comparison score. However, as the evaluation of this last conditional statement is not feasible within FHE, all three distances are decrypted, and the minimal distance is determined over the plaintext data. This means that final comparison score was fully computed in the encrypted domain, however, the algorithm reveals additional information in plaintext that may impact the privacy of the enrolled subjects, i.e., the discarded distances to the remaining $k - 1$ centroids. This limitation is also indicated in Table 2 for better transparency with regard to the different approaches.

4.13 Workload and Feasibility Discussion

We have now described all keystroke anomaly detectors from the seminal study by [24] and their challenges and adaptions under FHE encryption. Due to the limitations of FHE computations discussed so far, we can classify these detectors into three categories: *(1)* vector-based distance metrics such as the Euclidean and Manhattan distance, *(2)* detectors requiring matrix-vector or matrix multiplications, which introduce a significantly higher workload in FHE operations than the detectors discussed above. These include the (normed) Mahalanobis detector [11] as well as neural network-based approaches, including SVMs, as evaluated in [26]. And finally, *(3)*, detectors require the evaluation of conditional statements, which cannot be realized efficiently in FHE [19]. These include the nearest-neighbour [18], fuzzy logic and outlier counting [17], and k-means [23] detectors. We give the computational workload of all detectors in Table 2. Further context to Table 2 is provided through the relative cost of FHE operations given in Table 3. With regard to their feasibility however, detectors from categories *(2)* and *(3)* are not evaluated them experimentally. The experimental workload for some detectors of category *(3)* however can be estimated based on the Euclidean and Manhattan distance. E.g, the workload of outlier counting can be estimated as the workload of the scaled Manhattan distance, while the workload of the k-means detector corresponds to the k-fold workload of the Euclidean detector.

Table 2. FHE operations during verification for keystroke anomaly detectors [24], where n is the feature dimension, N is the number of training vectors, k is the number of centroids in the k-means clustering, and m is the dimension of the SVM projection space. Detectors marked with ** can only be partly computed on encrypted data, while detectors marked with * reveal more information than the final comparison score.

Detector	Enc	EvalAdd	EvalSub	EvalMult	EvalAtIndex	Dec
Euclidean	1	$n-1$	1	1	$n-1$	1
Euclidean (normed) (appr. A)*	2	$n-1$	1	2	$n-1$	2
Euclidean (normed) (appr. B)	2	$n-1$	1	3	$n-1$	1
Manhattan**	1	—	1	—	—	1
Manhattan (filtered)**	1	—	1	—	—	1
Manhattan (scaled)**	1	—	1	1	—	1
Mahalanobis	1	$2n(n-1)$	—	n^2	$2n(n-1)$	1
Mahalanobis (normed)	2	$2n(n-1)$	—	n^2+2	$2n(n-1)$	1
Nearest-neighbour*	N	$2Nn(n-1)$	—	N^2n	$Nn(n-1)$	N
Neural-network (standard)	1	$\lceil \frac{2n}{3} \rceil n-1$	—	$\lceil \frac{2n}{3} \rceil^2$	$2n(n-1)$	1
Neural-network (auto-assoc)	1	$2(n^2-n)$	1	n^2+n+1	$n-1(2n+1)$	1
Outlier-counting**	1	—	1	1	—	1
SVM (one-class)	1	$n+m-2$	m	$m \cdot m$	$(n-1)$	1
k-means*	1	$k(n-1)$	k	k	$k(n-1)$	k

Table 3. Relative cost of CKKS [9] operations implemented in PALISADE [29, 5].

Operation on encrypted data	Add	Subtract	Rotate	Decrypt	Multiply	Encrypt
Relative cost	1	5	24	33	46	52

5 Experimental Evaluation

We implemented our Type^2 protocol using the CKKS [9] scheme implemented in the PALISADE [29] C++ FHE library at a security level of 128bits for all variants of the Euclidean and Manhattan detectors. All execution times were measured on an Intel i7 CPU @ 2.60GHz with 32GB RAM and an Ubuntu 20.04 operating system. As a data set, we used the established CMU keystroke dynamics data set provided by [24] and maintain all features and the split into training and testing data. For the 400 timing vectors captured from each of the 51 subjects in the data set, the first 200 password timings were used for the training of each detector, and samples from the remaining timings for verification.

The execution times for enrolment and verification for the five discussed detectors are given in Table 4, where N is the number of subjects to be enrolled

Table 4. Experimentally determined execution times in milliseconds for the evaluated detectors. Detectors marked with ** can only be partly computed on encrypted data, while detectors marked with * are computed on encrypted data, but reveal more information than the final comparison score.

Detector	**Enrolment (ms)**	**Verification (ms)**
Euclidean	$4N$	117
Euclidean (normed) (appr. A)*	$8N$	125
Euclidean (normed) (appr. B)	$21N$	338
Manhattan**	$4N$	4
Manhattan (filtered)**	$4N$	4
Manhattan (scaled)**	$8N$	8

Table 5. Biometric performance for the evaluated detectors taken from [24]. Detectors marked with ** can only be partly computed on encrypted data, while detectors marked with * are computed on encrypted data, but reveal more information than the final comparison score.

Detector	**Equal-Error Rate (EER)**	**Standard Deviation**
Euclidean	0.171	0.095
Euclidean (normed) (appr. A)* Euclidean (normed) (appr. B)	0.215	0.119
Manhattan**	0.153	0.092
Manhattan (filtered)**	0.136	0.083
Manhattan (scaled)**	0.096	0.069

in the system. As discussed in Section 4, the Manhattan detectors have the fastest execution times as they use the lowest number of homomorphic operations. However, they cannot be considered fully secure, as the pre-computation step is decrypted before anomaly score can be calculated. The Euclidean detectors grant more privacy, with the plain Euclidean and the normed Euclidean (approach B) being the only fully private detectors with regard to evaluation under FHE. For the latter, the impact of the increased multiplicative depth of 2 instead of 1 can be observed. The encryption of reference or probe data, which consists of two encryption operations for the feature vector and its norm (or inverted norm) for both approach A and B to the normed Euclidean detector, therefore increases to 21 milliseconds instead of 8 milliseconds due the parameter set required to accommodate the increased circuit depth.

In terms of the biometric performance, we refer the reader to the original evaluation conducted in [24], which we give in Table 5. Through the application of the CKKS [9] with correct parameter choices, the biometric performance if not altered in the encrypted domain. In particular, we chose a scaling factor of 50 bits for the CKKS scheme, such the accuracy of the detectors is not affected by the application of the encryption scheme. Therefore, the accuracy evaluations given by [24] are maintained.

5.1 Security Analysis

Our proposed Type^2 protocol fulfils the ISO/IEC 24745 [20] requirements unlinkability, renewability, and irreversibility. Firstly, irreversibility is given through the hardness of the Ring-Learning with Errors (R-LWE) problem [27], which the CKKS [9] FHE scheme builds upon. As R-LWE is believed to be secure against attacks implemented on a quantum computer [2], our Type^2 protocol inherits this post-quantum security. Secondly, unlinkabilty and renewability are provided through the IND-CPA security of the CKKS scheme, i.e., its indistinguishability under chosen-plaintext attacks. Thereby, an attacker cannot distinguish between two encryptions of the same featrue vector and two encryptions of different feature vectors. Finally, our protocol preserves both the biometric and computational performance of the unprotected authentication as shown in Section 5. The choice of the PAKE, which is an independent component of the protocol next to the FHE protection, determines the security of the authentication as a second factor. However, post-quantum protection may not be necessary for the PAKE component, as the user password does not require long-term protection as sensitive biometric features do. This yields more flexibility with regard to the chosen PAKE approach, where computational efficiency lower than the workload for the biometric authentication should be considered [21, 32].

6 Conclusion

In this work, we have presented the Type^2 protocol for secure two-factor authentication based on keystroke dynamics as second trust factor, where the protection of sensitive biometric data is ensured through fully homomorphic encryption. For five established keystroke anomaly detectors, we showed the potential and limitations of their evaluation under fully homomorphic encryption. In an experimental evaluation, we show that our protocol outperforms the state-of-the-art with execution times of under 130 millisecond per authentication attempt. While the assumption of the semi-honest adversary model remains a limitation, the cryptographic principles applied in this work can be used to extend the Type^2 protocol in more realistic adversary models. With advances of the cryptographic components, more complex detectors, e.g., neural networks, could be investigated in future research. Furthermore, it would be interesting to extend the Type^2 protocol to other behavioral features using mobile phones as the capture device.

References

1. Acar, A., Liu, W., Beyah, R., Akkaya, K., Uluagac, A.S.: A privacy-preserving multifactor authentication system. Security and Privacy **2**(5), e88 (2019)
2. Alagic, G., Apon, D., Cooper, D., Dang, Q., Dang, T., Kelsey, J., Lichtinger, J., Miller, C., Moody, D., Peralta, R., et al.: Status report on the third round of the nist post-quantum cryptography standardization process. National Institute of Standards and Technology (NIST) (2022)
3. Araújo, L.C.F., Sucupira, L.H.R., Lizarraga, M.G., Ling, L.L., Yabu-Uti, J.B.T.: User authentication through typing biometrics features. IEEE Trans. on Signal Processing **53**(2), 851–855 (2005)
4. Bassit, A., Hahn, F., Peeters, J., Kevenaar, T., Veldhuis, R., Peter, A.: Fast and accurate likelihood ratio-based biometric verification secure against malicious adversaries. IEEE Trans. on Information Forensics and Security (TIFS) **16**, 5045–5060 (2021)
5. Bauspieß, P., Olafsson, J., Kolberg, J., Drozdowski, P., Rathgeb, C., Busch, C.: Improved homomorphically encrypted biometric identification using coefficient packing. In: Proc. Intl. Workshop on Biometrics and Forensics (IWBF) (2022)
6. Bleha, S., Slivinsky, C., Hussien, B.: Computer-access security systems using keystroke dynamics. IEEE Trans. on Pattern Analysis and Machine Intelligence **12**(12), 1217–1222 (1990)
7. Boddeti, V.N.: Secure face matching using fully homomorphic encryption. In: Intl. Conf. on Biometrics Theory, Applications and Systems (BTAS). pp. 1–10. IEEE (2018)
8. Brakerski, Z.: Fully homomorphic encryption without modulus switching from classical gapsvp. In: Advances in Cryptology–CRYPTO 2012: 32nd Annual Cryptology Conference, Santa Barbara, CA, USA, August 19-23, 2012. Proceedings. pp. 868–886. Springer (2012)
9. Cheon, J.H., Kim, A., Kim, M., Song, Y.: Homomorphic encryption for arithmetic of approximate numbers. In: 23rd International Conference on the Theory and Applications of Cryptology and Information Security. pp. 409–437. Springer (2017)
10. Damgard, I., Geisler, M., Kroigard, M.: Homomorphic encryption and secure comparison. International Journal of Applied Cryptography **1**(1), 22–31 (2008)
11. Duda, R.O., Hart, P.E., Stork, D.: Pattern classification. John Wiely and Sons (2001)
12. European Council: Regulation of the european parliament and of the council on electronic identification and trust services for electronic transactions in the internal market (eidas regulation) (July 2014)
13. European Council: Regulation of the european parliament and of the council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (general data protection regulation) (April 2016)
14. Fan, J., Vercauteren, F.: Somewhat practical fully homomorphic encryption. IACR Cryptol. ePrint Arch. **2012**, 144 (2012)
15. Gentry, C.: Fully homomorphic encryption using ideal lattices. In: Proceedings of the 41st annual ACM symposium on Theory of Computing. pp. 169–178 (2009)
16. Gomez-Barrero, M., Maiorana, E., Galbally, J., Campisi, P., Fierrez, J.: Multi-biometric template protection based on Homomorphic Encryption. Pattern Recognition **67**, 149–163 (July 2017)
17. Haider, S., Abbas, A., Zaidi, A.K.: A multi-technique approach for user identification through keystroke dynamics. In: Proceedings of the IEEE Intl. Conference on Systems, Man and Cybernetics (SMC). vol. 2, pp. 1336–1341. IEEE (2000)

18. Han, S.C.C., Han, D.H., Kim, H.: Web-based keystroke dynamics identity verification using neural network. Journal of Organizational Computing and Electronic Commerce (JOCEC) **10**(4), 295–307 (2000)
19. Iliashenko, I., Zucca, V.: Faster homomorphic comparison operations for bgv and bfv. Proceedings on Privacy Enhancing Technologies **2021**(3), 246–264 (2021)
20. ISO/IEC JTC1 SC27 Security Techniques: ISO/IEC 24745:2022. Information Technology - Security Techniques - Biometric Information Protection. International Organization for Standardization (2022)
21. Jarecki, S., Krawczyk, H., Xu, J.: OPAQUE: an asymmetric pake protocol secure against pre-computation attacks. In: 37th Ann. Intl. Conf. on the Theory and Applications of Cryptographic Techniques. pp. 456–486. Springer (2018)
22. Joyce, R., Gupta, G.: Identity authentication based on keystroke latencies. Communications of the ACM **33**(2), 168–176 (1990)
23. Kang, P., Hwang, S.s., Cho, S.: Continual retraining of keystroke dynamics based authenticator. In: Proceedings of the 2nd International Conference on Biometrics (ICB). pp. 1203–1211. Springer (2007)
24. Killourhy, K., Maxion, R.A.: Comparing anomaly-detection algorithms for keystroke dynamics. In: IEEE/IFIP International Conference on Dependable Systems & Networks (2009)
25. Krebs, B.: Facebook Stored Hundreds of Millions of User Passwords in Plain Text for Years (2019), https://krebsonsecurity.com/2019/03/facebook-stored-hundreds-of-millions-of-user-passwords-in-plain-text-for-years/
26. Loya, J., Bana, T.: Privacy-preserving keystroke analysis using fully homomorphic encryption & differential privacy. In: International Conference on Cyberworlds (CW). pp. 291–294. IEEE (2021)
27. Lyubashevsky, V., Peikert, C., Regev, O.: On ideal lattices and learning with errors over rings. In: Annual Intl. Conf. on the Theory and Appl. of Cryptographic Techniques. pp. 1–23. Springer (2010)
28. MacQueen, J.: Some methods for classification and analysis of multivariate observations. Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability **1**, 281–297 (1967)
29. Ruhloff, K., Cousins, D., Polyakov, Y.: The PALISADE Lattice Cryptography Library (2017), https://git.njit.edu/palisade/PALISADE
30. Šeděnka, J., Balagani, K.S., Phoha, V., Gasti, P.: Privacy-preserving population-enhanced biometric key generation from free-text keystroke dynamics. In: IEEE International Joint Conference on Biometrics. pp. 1–8. IEEE (2014)
31. Shen, C., Yu, T., Xu, H., Yang, G., Guan, X.: User practice in password security: An empirical study of real-life passwords in the wild. Computers & Security **61**, 130–141 (2016)
32. Wu, T.: The secure remote password protocol. In: Proc. 1998 Internet Society Symposium on Network and Distributed Systems Security. pp. 97–111. Citeseer (1998)
33. Yasuda, M., Shimoyama, T., Kogure, J., Yokoyama, K., Koshiba, T.: Packed homomorphic encryption based on ideal lattices and its application to biometrics. In: Intl. Conf. on Availability, Reliability, and Security. pp. 55–74. Springer (2013)
34. Yu, E., Cho, S.: Ga-svm wrapper approach for feature subset selection in keystroke dynamics identity verification. In: Proceedings of the International Joint Conference on Neural Networks (IJCNN). vol. 3, pp. 2253–2257. IEEE (2003)