International Journal of Innovative
Computing, Information and Control
Volume **13**, Number **5**, October **2017**

 ISSN 1349-4198
pp. **1623–1635**

AN ONLINE MOBILE SIGNATURE VERIFICATION SYSTEM BASED ON HOMOMORPHIC ENCRYPTION

Huixiang Zhang[1], Xinxin Liu[1] and Chunlei Chen[2]

[1]School of Automation
Northwest Polytechnical University
No. 127, West Youyi Road, Xi'an 710072, P. R. China
zhanghuixiang@nwpu.edu.cn

[2]School of Computer Engineering
Weifang University
No. 5147, Dongfeng Street, Weifang 261061, P. R. China

Received January 2017; revised May 2017

Abstract. *An online mobile signature verification system was proposed based on homomorphic encryption in this paper. A secure iterative protocol was designed to perform dynamic time warping calculation in homomorphic encryption domain between the mobile device and verification server. A restriction window-based dynamic time warping strategy was leveraged to improve the computational efficiency. A three-party storage mechanism was designed to remove the risk of data leakage in a single-point failure situation. The experimental results show that the verification performance was not degraded by the use of homomorphic encryption. In addition, the response time of our system for single verification significantly decreases. Our system is secure and reliable, and can be deployed in a general mobile device.*

Keywords: Signature verification, Biometric template protection, Homomorphic encryption, Dynamic time warping, Mobile device, Three-party storage

1. **Introduction.** Online handwritten signature is a general biometric trait for identity authentication. During the process of online handwritten signature, a sequence of x-y coordinates is captured by a dedicated sampling device, such as a pen tablet [1]. In recent years, smart mobile devices equipped with touch-based interface become popular. Since the touch-based interface is suitable for handwritten signature, smart mobile devices are turned into an attractive target for the deployment of a signature verification system [2,3].

The signature verification process is performed in a client/server manner [4]. The client captures users' handwritten signature sample, and then sends the sample to the server. The server verifies a signature sample by computing the similarity between the sample and the stored signature template. Any information leakage resulting from an insecure transportation of signature sample or an inappropriate storage of the templates can lead to severe privacy leakage issues. Thus, a biometric template protection mechanism must be provided to prevent the template leakage in a signature verification system [5].

Biometric template protection has been a research hotsopt. [6] proposed a cryptosystem for signature templates protection using fuzzy commitment. They adopted error correcting codes to generate a protected representation of the biometrics template. However, the scheme of fuzzy commitment is vulnerable to several attacks [7]. [8] applied multiple linear convolution to transforming the original signature templates. It is computationally hard to invert the transformation even if its defining parameters are known. [9] demonstrated an implementation of the fuzzy vault scheme for online signatures. As the length of template grows, the computational complexity increases significantly. As a result, their

method only applied to protecting template consisting of small number of signature characters. These template protection schemes rely on irreversible transformations to obscure the extracted features, thus resulting in some performance degradation [10].

An alternative approach is to use cryptographic techniques. The traditional encryption can be employed to secure the transmission and storage of biometric data. Nevertheless, the verification process has to be performed after decryption, and therefore no protection is provided during user authentication. Fortunately, homomorphic encryption (HE) can be used to overcome this weakness. HE is a form of encryption that satisfies following characteristics [11]. First, ciphertext can be generated by executing encryption operation on plaintext. Second, a set of calculation is executed on the ciphertext and the generated result is denoted as $R1$. Third, a result is obtained by decrypting $R1$, denoted as $R1'$. Fourth, $R1'$ matches the result that is generated by executing the same calculation on the plaintext. This is a desirable feature in a biometric template protection mechanism. On one hand, the template does not need to be restored in the verification process. As a result, the security of the signature template is ensured. On the other hand, the consistency of the computation performed on ciphertexts and plaintexts guarantees that the authentication performance will not degrade. Moreover, the homomorphic nature also ensures that a portion of the computation can be delegated to the other computers without causing data leakage. This property is beneficial to smart mobile devices, and the battery consumption can be reduced by transferring some calculation to the server side.

[12] proposed an efficient method to compute the Hamming distance on encrypted data using the homomorphic encryption, and designed a privacy-preserving biometric authentication protocol for online signature verification system. [10] presented an online signature verification system based on homomorphic encryption. In their implementation, most of the calculations are performed on the client. The time consumption of one-time verification is more than one minute. That is not appropriate for mobile devices which have limited computation resources and energy. Moreover, these works did not consider how to secure the biometric template in a single-point situation. The encrypted template and encryption keys are stored in the same server. If the server is compromised, there is a risk of the template leakage.

In this paper, we proposed an online signature verification system for mobile device users. Our goal is to achieve fast verification process and secure storage in a single-point situation. The main contributions of this paper are summarized as follows.

- We designed a secure iteration protocol to perform cooperative dynamic time warping based verification in homomorphic encryption domain between the mobile device and the sever.
- We accelerated the verification process by applying a restriction window based dynamic time warping matching strategy.
- We designed a three-party storage mechanism to exclude the risk of data leakage in a single-point failure situation.
- We implemented the proposed online mobile signature verification system, and the client system runs on Android mobile device.

The rest of the paper is organized as follows. Section 2 designs the signature verification process, including the iteration protocol and window-based acceleration strategy. Section 3 demonstrates the proposed three-party secure storage mechanism. Verification performance is evaluated in Section 4 and final conclusions are drawn in Section 5.

2. The Mobile Signature Verification Process with HE.

2.1. **Paillier homomorphic encryption scheme.** The proposed template protection mechanism is built on the Paillier cryptosystem [13] which is a probabilistic asymmetric algorithm for public key cryptography. Here, (pk, sk) is a pair of public key and private key for unsymmetrical encryption. A plain message m is encrypted into its ciphertext m^* by an encryption function Enc, that is $m^* = Enc_{pk}(m)$. And Dcp denotes as its corresponding decryption function, that is $m = Dcp_{sk}(m^*)$. Two properties of Paillier cryptosystem will be used in our system.

First, the product of two ciphertexts, m_1^* and m_2^*, will be decrypted to the sum of their corresponding plaintexts.

$$Dcp_{sk}\left(m_1^* \cdot m_2^* \bmod n^2\right) = m_1 + m_2 \bmod n \tag{1}$$

Second, an encrypted plaintext, m_1^*, raised to a constant l, will be decrypted to the product of the plaintext and the constant.

$$Dcp_{sk}\left((m_1^*)^l \bmod n^2\right) = m_1 \cdot l \bmod n \tag{2}$$

Here, n is a number used to generate the public key and it is the product of two large prime numbers. For more detail on Paillier homomorphic probabilistic encryption scheme, we refer readers to [10,13].

2.2. **The DTW-based verification.** Dynamic time warping (DTW) is an algorithm for measuring similarity between two temporal sequences which may vary in speed. The temporal sequences of signature will be denoted as matrices like $\mathbf{CX}_{P\times Z}$, where P is the number of points in the signature sequence and Z the number of features extracted from each point. Therefore, the i-th point in the sequence is a Z-dimensional vector, that is $\mathbf{CX}[i] = x^i = \{x_1^i, x_2^i, \ldots, x_z^i\}$. For mobile devices, the feature vector mainly consists of the horizontal and vertical coordinates. The Euclidean distance between two points x and y is denoted as $dist_{euc}(x, y)$. Thus, $dist_{euc}^2(x, y)$ is the square of Euclidean distance between x and y.

In order to verify whether a sample ($\mathbf{CX}_{P\times Z}$) matches its corresponding template ($\mathbf{SY}_{Q\times Z}$), a dissimilarity score is computed between them based on the DTW algorithm. Here, a cost matrix ($\mathbf{D}_{P\times Q}$) is used to minimize the distance between signature points in terms of their Euclidean distance. The standard DTW algorithm is shown in Algorithm 1. First, DTW initializes the first row, first column, and first cell from line 1 to 5. Second, DTW calculates the remaining cells iteratively from line 6 to 10. The dissimilarity score between $\mathbf{CX}_{P\times Z}$ and $\mathbf{SY}_{Q\times Z}$ is the last cell of the cost matrix, namely $dis = \mathbf{D}[P, Q]$. As can be seen in Algorithm 1, *dis* is a cumulative distance of the two signature sequences, and we have to traverse the whole cost matrix to calculate the value of *dis*. If *dis* is smaller than a threshold value denoted as *score*, we believe that the sample matches the template.

In our proposed system, the Paliwal's window-based DTW matching strategy is used to speed up the DTW process [14]. As illustrated by Figure 1, an adjustment window is used for restricting the warping function. In order to calculate the dissimilarity score, we only traverse the matrix space restricted by the adjustment window. As a result, the iteration times of computing the dissimilarity score are reduced significantly. Meanwhile, the dissimilarity score calculated in window-based DTW strategy is probably greater than the score calculated in standard algorithm. From the perspective of window-based DTW algorithm, it should have a smaller dissimilarity score if a sample is considered to match the template. Therefore, the window-based DTW algorithm not only consumes less computation time but also improves the recognition accuracy. This algorithm is shown in

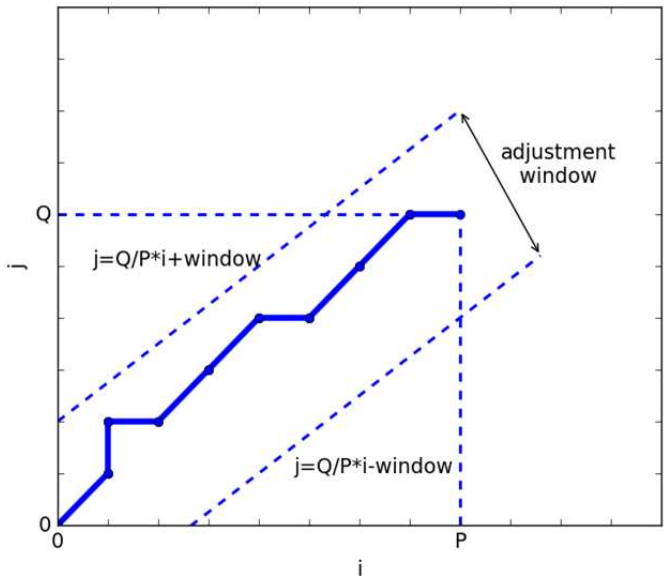

FIGURE 1. The effect of adjustment window

Algorithm 1 *DTW: The standard DTW algorithm.*

Input: $\mathbf{CX}_{P\times Z}$, $\mathbf{SY}_{Q\times Z}$
Output: *The dissimilarity score.*

1: *for* $0 \le i \le P$:
2: $\quad \mathbf{D}[i,0] = \text{infinity}$
3: *for* $0 \le j \le Q$:
4: $\quad \mathbf{D}[0,j] = \text{infinity}$
5: $\mathbf{D}[1,1] = dist^2_{euc}(\mathbf{CX}[1], \mathbf{SY}[1])$
6: *for* $1 \le i \le P$:
7: $\quad$ *for* $1 \le j \le Q$:
8: $\quad\quad cost = dist^2_{euc}(\mathbf{CX}[i], \mathbf{SY}[j])$
9: $\quad\quad minimum = \min(\mathbf{D}[i-1,j], \mathbf{D}[i,j-1], \mathbf{D}[i-1,j-1])$
10: $\quad\quad \mathbf{D}[i,j] = cost + minimum$
11: return $\mathbf{D}[P,Q]$

Algorithm 2 *Paliwal's DTW: The window-based DTW algorithm.*

Input: $\mathbf{CX}_{P\times Z}$, $\mathbf{SY}_{Q\times Z}$, ws
Output: *The dissimilarity score.*

1: *for* $0 \le i \le P$:
2: $\quad \mathbf{D}[i,0] = \text{infinity}$
3: *for* $0 \le j \le Q$:
4: $\quad \mathbf{D}[0,j] = \text{infinity}$
5: $\mathbf{D}[1,1] = dist^2_{euc}(\mathbf{CX}[1], \mathbf{SY}[1])$
6: *for* $1 \le i \le P$:
7: $\quad$ *for* $1 \le j \le Q$:
8: $\quad\quad$ *if* $\text{abs}(i \times Q/P - j) < ws$:
9: $\quad\quad\quad cost = dist^2_{euc}(\mathbf{CX}[i], \mathbf{SY}[j])$
10: $\quad\quad\quad minimum = \min(\mathbf{D}[i-1,j], \mathbf{D}[i,j-1], \mathbf{D}[i-1,j-1])$
11: $\quad\quad\quad \mathbf{D}[i,j] = cost + minimum$
12: return $\mathbf{D}[P,Q]$

Algorithm 2. In line 8, a window condition is added to restrict the traversal space. Here, ws is a predefined empirical value of the restriction window.

2.3. **The encrypted DTW-based verification process.** In our system, the DTW algorithm is used to identify the sample signature, and HE is used to protect the signature template. The sample signature is captured in a mobile device with touch-based interface. The signature template is encrypted and stored in a server. The mobile device and the sever cooperate to complete the encrypted DTW-based verification process. We proposed an interaction protocol for the cooperative verification process. This protocol aims to

achieve two objectives: 1) most of the computing is performed in the server to reduce the load of the mobile device; 2) the signature sample and template are shielded from being compromised in the verification process.

The interaction protocol is depicted in Figure 2. In the first step, a random list $R_N = \{t_0, t_2, \ldots, t_{N-1}\}$ is introduced to hide the true value of sample signature sequence $\mathbf{CX}_{P\times Z}$. When the mobile device extracts the user signature sequence, a random offset $t_i = R[i\%N]$ is added to all Z feature values of the i-th point in $\mathbf{CX}_{P\times Z}$ as Equation (3).

$$\mathbf{CT}[i] = \mathbf{CX}[i] + T_i = \left\{x_1^i + t_i, x_2^i + t_i, \ldots, x_z^i + t_i\right\}, \quad i \in [0, P-1] \tag{3}$$

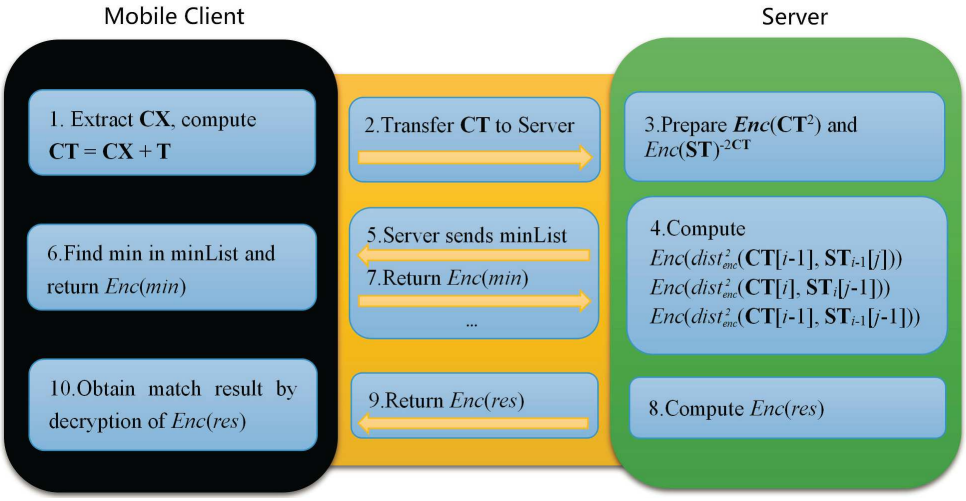

FIGURE 2. The interaction protocol

Then the blurred sequence $\mathbf{CT}_{P\times Z}$ is sent to the server in the second step. The original sequence $\mathbf{CX}_{P\times Z}$ is hidden using the random offsets. Even if the attacker intercepted the sequence $\mathbf{CT}_{P\times Z}$, the original signature $\mathbf{CX}_{P\times Z}$ cannot be restored. Thus, the signature sample is protected from being compromised in the transmission process.

Regarding two points $\mathbf{CX}[i] = x$ and $\mathbf{SY}[j] = y$, the square of Euclidean distance between them can be computed as Equation (4).

$$dist_{euc}^2(\mathbf{CX}[i], \mathbf{SY}[j]) = \sum_{z=1}^{Z}(x_z - y_z)^2 = \sum_{z=1}^{Z} x_z^2 + \sum_{z=1}^{Z} y_z^2 - 2\sum_{z=1}^{Z} x_z y_z \tag{4}$$

The signature template sequence denoted as $Enc(\mathbf{SY}_{Q\times Z})$ is encrypted in the server. Thus, an encrypted square of Euclidean distance between x and y can be calculated as Equation (5) by using the two properties of HE in Equations (1) and (2).

$$\begin{aligned} Enc\left(dist_{euc}^2(\mathbf{CX}[i], \mathbf{SY}[j])\right) &= Enc\left(\sum_{z=1}^{Z}(x_z - y_z)^2\right) \\ = \prod_{z=1}^{Z} Enc\left(x_z^2\right) \cdot \prod_{z=1}^{Z} Enc\left(y_z^2\right) &\cdot \prod_{z=1}^{Z} Enc(y_z)^{-2x_z} \end{aligned} \tag{5}$$

For the point of $x = \mathbf{CT}[i]$ in the blurred sample sequence, the same offset can be added to the point $y = \mathbf{SY}[j]$ in signature template sequence. Thus, a new vector $\mathbf{ST}_i[j]$ is obtained:

$$\mathbf{ST}_i[j] = \mathbf{SY}[j] + T_i = \left\{y_1^j + t_i, y_2^j + t_i, \ldots, y_z^j + t_i\right\}, \quad i \in [0, P-1],\ j \in [0, Q-1] \tag{6}$$

The encrypted square of Euclidean distance between x and y can be transformed to the distance of $\mathbf{CT}[i]$ and $\mathbf{ST}_i[j]$ as shown in Equation (7).

$$\begin{aligned} &Enc\left(dist^2_{euc}(\mathbf{CX}[i], \mathbf{SY}[j])\right) = Enc\left(\sum_{z=1}^{Z}(x_z - y_z)^2\right) \\ &= Enc\left(\sum_{z=1}^{Z}[(x_z + T_i) - (y_z + T_i)]^2\right) = Enc\left(dist^2_{euc}(\mathbf{CX}[i] + T_i, \mathbf{SY}[j] + T_i)\right) \\ &= Enc\left(dist^2_{euc}(\mathbf{CT}[i], \mathbf{ST}_i[j])\right) \end{aligned} \tag{7}$$

Consequently, the encrypted square of Euclidean distance between x and y can be computed pursuant to Equation (5).

$$\begin{aligned} &Enc\left(dist^2_{euc}(\mathbf{CX}[i], \mathbf{SY}[j])\right) = Enc\left(dist^2_{euc}(\mathbf{CT}[i], \mathbf{ST}_i[j])\right) \\ &= \prod_{z=1}^{Z} Enc\left[(x_z + T_i)^2\right] \cdot \prod_{z=1}^{Z} Enc\left(y_z^2\right) \cdot Enc\left(T_i^2\right) \cdot Enc(y_z)^{2T_i} \\ &\cdot \prod_{z=1}^{Z} Enc(y_z + T_i)^{-2(x_z + T_i)} \end{aligned} \tag{8}$$

In step 3, the first and the last item in Equation (8) are calculated in the sever. The first item can be calculated using $\mathbf{CT}_{P\times Z}$ after a square and encrypted operation. $Enc(y_z)$, $Enc(T_i)$, $Enc\left(y_z^2\right)$, $Enc\left(T_i^2\right)$ and $Enc(y_z)^{2T_i}$ are generated in the template enrollment and stored in the server for accelerating the computation. $Enc(y_z + T_i)$ can be calculated by the product of $Enc(y_z)$ and $Enc(T_i)$ pursuant to Equation (1). Thus, the last item can be calculated by a normal exponential operation in the server. So far, we can securely compute the encrypted square of Euclidean distance between x and y in the server.

As described in Algorithm 2, the minimum cumulative distance, i.e., the variable *minimum*, needs to be determined in each iteration computation. The variable *minimum* is the smallest value among $\mathbf{D}[i-1, j]$, $\mathbf{D}[i-1, j-1]$ and $\mathbf{D}[i, j-1]$. These three distances are computed in encrypted domain on the server during the fourth step. As the computing results are encrypted, a list called minList consisting of $Enc(\mathbf{D}[i-1, j])$, $Enc(\mathbf{D}[i-1, j-1])$ and $Enc(\mathbf{D}[i, j-1])$ is sent back to the mobile client in step 5. The minList can be decrypted in the client, and the minimum is determined by comparing these values in step 6. Afterwards, the minimum is encrypted to $Enc(minValue)$ and return to the server in step 7. The process from step 4 to step 7 operates iteratively, and finally the encrypted dissimilarity score between $\mathbf{CX}_{P\times Z}$ and $\mathbf{SY}_{Q\times Z}$ is obtained, namely $Enc(dis) = Enc(\mathbf{D}[P, Q])$. In step 8, dis is compared to a threshold *score* to determine whether the sample sequence matches the template sequence. Again, the property of HE is used to calculate the encrypted match result $Enc(res)$ as Equation (9).

$$Enc(res) = Enc(dis - score) = Enc(dis) * Enc(score)^{-1} \tag{9}$$

Then, the result of $Enc(res)$ is sent back to the mobile client in step 9. The client obtains the match result by decryption of $Enc(res)$. The verification process is finished.

As can be seen from the entire iteration protocol, the computation on the server is all executed in the encryption domain. The template and decision threshold score stored on the server are always encrypted, and the user samples are passed to the server after being blurred. Only the client device held by the user can decrypt the user data. This eliminates the risk of data leakage in cooperated verification process. In addition, the main DTW calculation process is performed by the server, and the client only decrypts

the results for comparison. Thus, the computation load in mobile client is relatively light, which is conducive to deploying the protocol in mobile devices.

3. **The Secure Three-party Storage Mechanism.** A three-party storage mechanism is designed to remove the risk of data leakage in a single-point failure situation. The user templates would not leak in the case that any single party was compromised. The storage mechanism is illustrated in Figure 3.

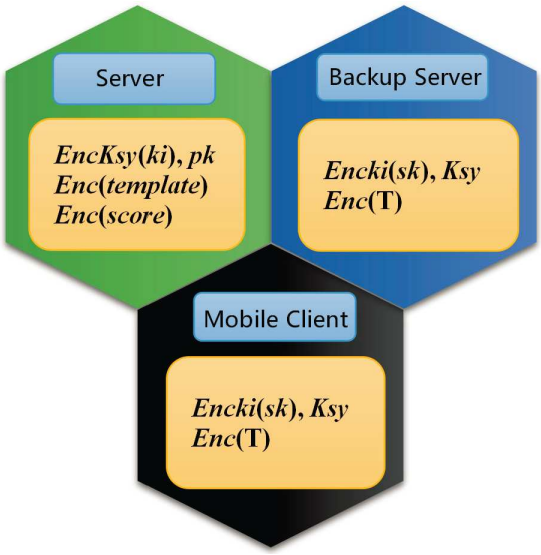

FIGURE 3. A three-party storage mechanism

As depicted in Subsection 2.3, a private key of HE, denoted as sk, is used in client device to perform decryption. However, we did not keep sk in the client to prevent leakage of the key in case the client is compromised. The private key of sk is encrypted with a key of ki; the encrypted sk, namely $Encki(sk)$, is stored in the client. Furthermore, ki is encrypted by a key of Ksy; the encrypted ki, namely $EncKsy(ki)$, is stored in the server, and the key of Ksy is stored in the client. Moreover, the encrypted offset list $Enc(T)$ is also stored in the client. When a mobile client starts a process of signature verification, the verification server will return the data of $EncKsy(ki)$ to the client. Subsequently, the mobile client can use the local storage of Ksy information to obtain ki. By using the key of ki, the private key sk is obtained by deciphering and used for subsequent verification calculation. If the client was compromised, the attacker only gets the key of Ksy and $Encki(sk)$. It would not lead to leakage of the private key sk.

Besides the encrypted key $Enc_{Ksy}(ki)$, $Enc(y_z^2)$, $Enc(T_i^2)$ and $Enc(y_z)^{2T_i}$ are stored in the server for the purpose of computing acceleration. The $ST_i[j]$ and $Enc(T_i)$ are stored in the server for calculation of the encrypted square of Euclidean distance. These five values are denoted as $Enc(template)$ in Figure 3. The encrypted matching threshold $Enc(score)$ and the public key of HE pk are also stored in the server. All the data except pk are stored in encrypted domain. The compromise of the server would not lead to leakage of the user templates.

In addition to the mobile client and server, a backup server is involved in the mechanism. The back server is used for user data recovery in case that the mobile client is lost. As shown in Figure 3, the back server stores the same data as the mobile client. The compromise of the back server would not result in leakage of the private key sk.

With such a three-party storage mechanism, the pk and sk are separately stored in the server and client. Once the server is compromised, the signature template is protected

by HE and would not cause leakage of template. Meanwhile, *sk* and the offset list are encrypted in the client, and they cannot be leaked if the mobile client is lost. This storage mechanism can effectively ensure the security and reliability of the verification system.

4. **Performance Evaluation.** The signature subset used in our work was captured in two sessions over a half-year period. In the first session, 200 genuine signatures from 20 persons (10 signatures per person) are acquired. These signatures are enrolled as the signature template. In the second session, 200 genuine signatures are acquired as the first session. These signatures were used to evaluate the false rejection rate (FRR), and all these samples from all other individuals were used to evaluate the false acceptance rate, namely FAR-rf. Moreover, 200 skilled forgery samples were captured by randomly choosing 10 people to imitate the signature of the others. These forgery samples were used to evaluate the false acceptance rate, namely FAR-sf. The equal error rate (EER), the rate at which FAR and FRR are equal, was also used to compare the verification performances in different scenarios. All the signatures are captured in a touch-based mobile phone. The feature vector consists of the horizontal and vertical coordinates in a handwritten process.

4.1. **The effect of restriction window.** In this subsection, we evaluate the effect of restriction window in the window-based DTW algorithm (i.e., Algorithm 2). The window size is denoted as *ws* in Algorithm 2 in Subsection 2.2, and it varied from 10 to 40. The result is shown in Figure 4.

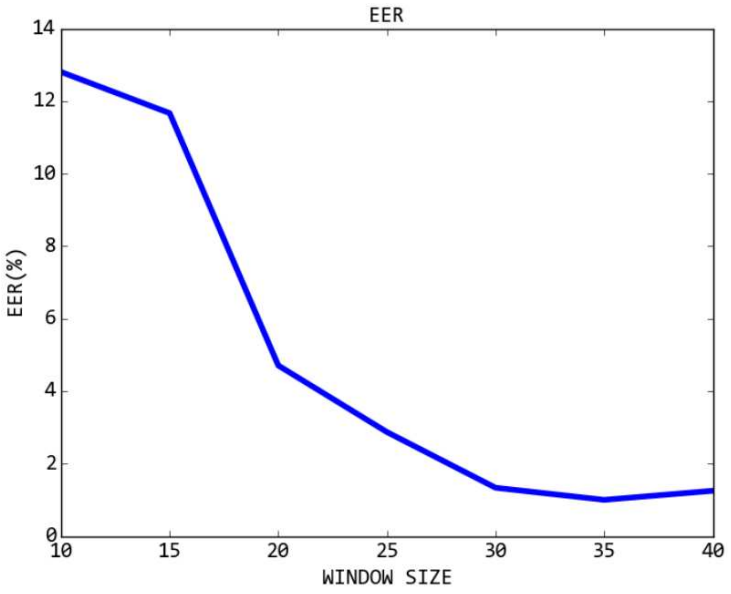

FIGURE 4. The effect of restriction window

In Figure 4, the EER value of the algorithm keeps dropping as the restriction window grows. When the restriction window is relatively small, the DTW distance between the forgery samples and signature templates increases. This increase in distance helps to distinguish forgery signatures, but also lead to false rejection to genuine signatures. Thus, the EER value is relatively higher when the window is small. With the increment of window, the genuine and the forgery samples can be distinguished gradually, and the EER value also reduced and tend to be consistent with standard DTW algorithm. The larger the restriction window is, the closer the effect is to the standard DTW algorithm. However, larger restriction window means much more computation cost. For our system,

it is necessary to choose a relatively small window to reduce time overhead and suppress verification error.

4.2. **Performance comparison.** In this subsection, we compared the performance of DTW and encrypted DTW algorithms in online signature verification. The Detection Error Trade-off (DET) curves are depicted in Figure 5 and Figure 6, respectively for the random forgeries and the skilled forgeries scenarios. The restriction window varied from 10 to 40 with a step of 5 in these experiments.

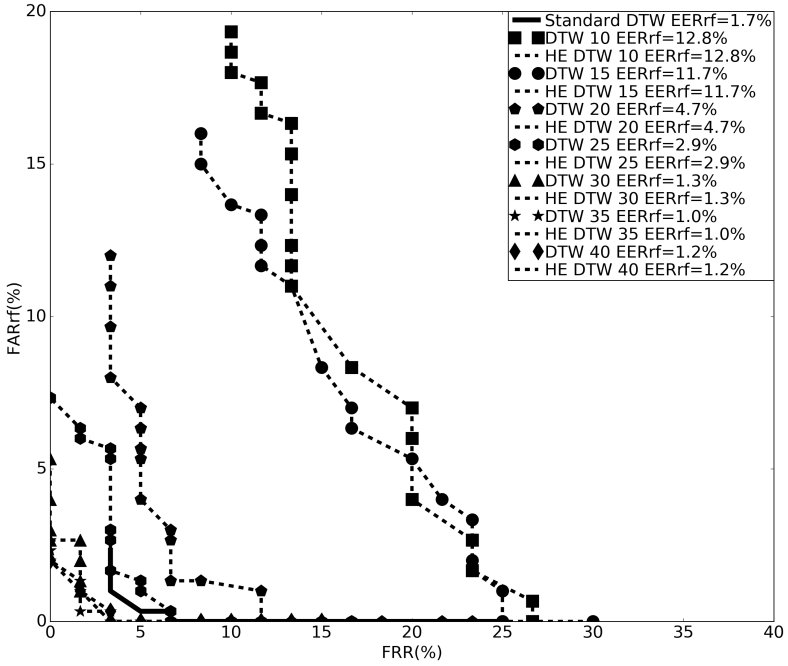

FIGURE 5. DET curves for random forgeries

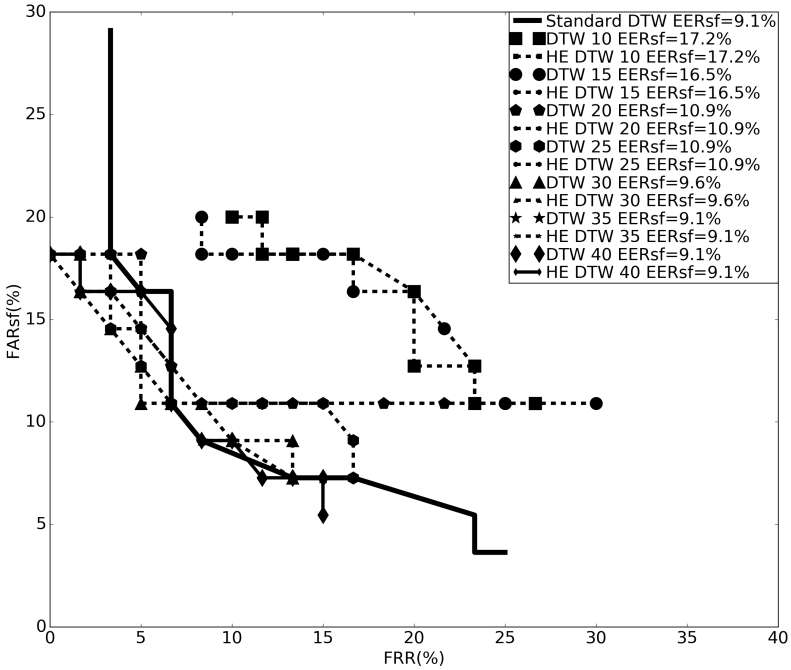

FIGURE 6. DET curves for skilled forgeries

As illustrated by Figure 5 and Figure 6, the curves of encrypted DTW and the curves of DTW with the same restriction window completely overlap. That means the performance of DTW in HE domain is not degraded at any operating point. In general, the performance is getting better as the restriction window grows. The EER value tends to be consistent with standard DTW both for random forgeries and skilled forgeries scenarios. When the size of restriction window is set to 35, the metric of ERR is only 1.0% for random forgeries experiment, and 9.0% for skilled forgeries experiment. By introducing a certain size of the restriction window, the performance of window-based DTW is slightly less efficient than the standard DTW, but the efficiency of the algorithm exhibits a substantial increase. From the curves in Figure 5 and Figure 6, the choice of 30 for the restriction window can basically achieve a balance between authentication performance and computational efficiency.

4.3. **Verification time.** In this subsection, the response time for one time verification is evaluated. The Gomez-Barrero's system [10] is used for comparison purpose. The restriction window is set to 30. A user signature sequence is captured by handling the touch event triggered in Android device. The size of the signature sequence obtained is determined by the user's handwriting behavior. Signature sequence sizes of two inputs can be different even if the inputs are generated by the same user. According to the statistics of the signature subset used in our work, the largest signature sequence size is around 800, and the smallest is around 100. To investigate the relationship between performance and sequence size, we resampled the signature sequence with different lengths that range from 100 to 800.

The total response time for one time verification consists of the computation time in server and the computation time in client. As can be seen in Figure 8, the time complexity of Gomez-Barrero's system is $O(n^2)$, the time complexities of computation time on client and server are both quadratic with regards to the sequence size. By using the window based DTW matching strategy, the time complexity of our proposed system is reduced

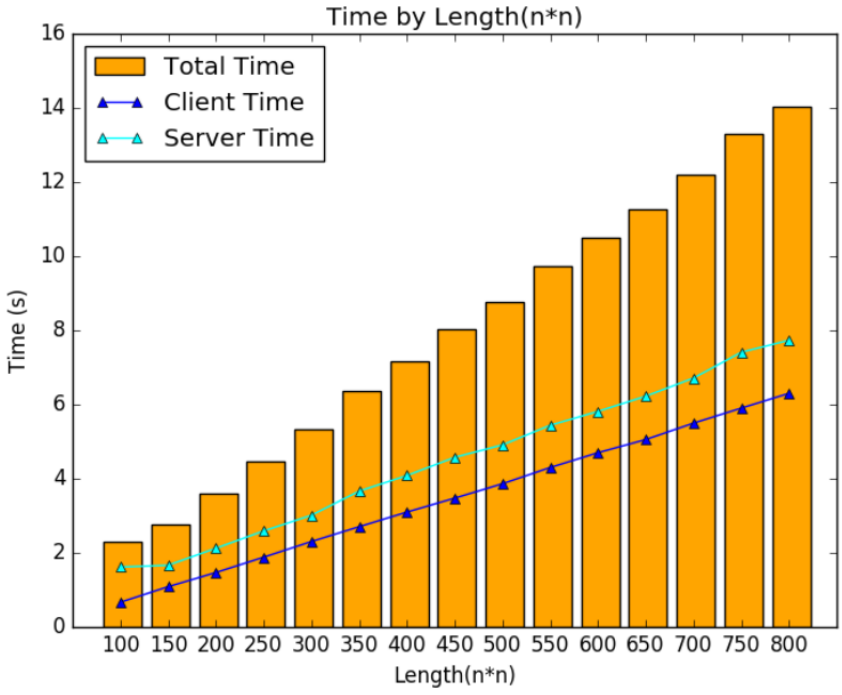

FIGURE 7. Response time of proposed system

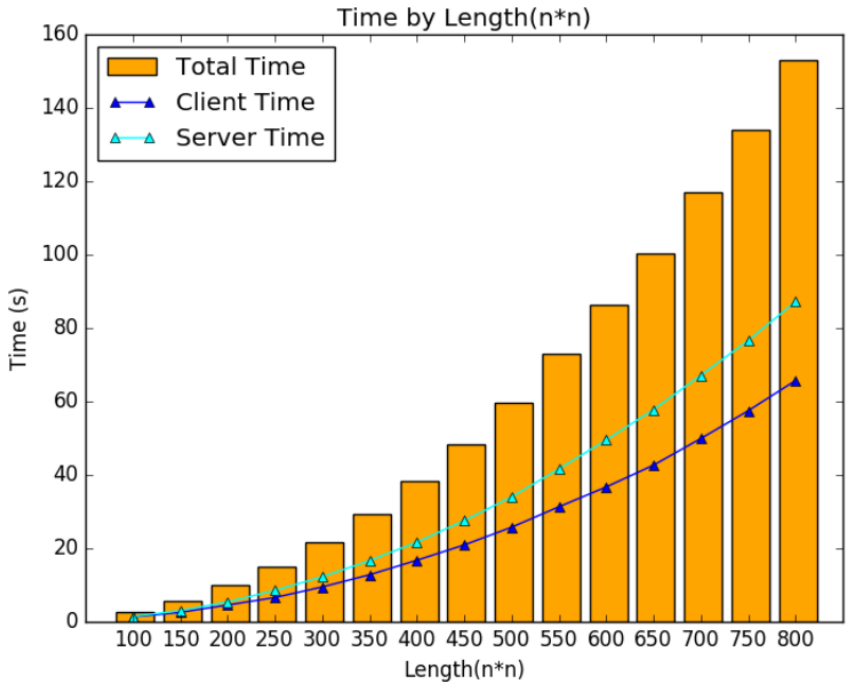

FIGURE 8. Response time of Gomez-Barrero's system

to $O(n)$. As shown in Figure 7, the total time and computation time in client and server are both linear to the sequence sizes.

The response time for one time verification is reduced from minutes to seconds. As can be seen from Figure 7 and Figure 8, the response time of our proposed system is only less than one tenth of the Gomez-Barrero's system when the signature sequence length is set to 800. Thus, our system is deployable in a mobile device which has less computation resource and energy.

We implemented our system in Android OS, and deployed the system on two models of mobile phones, which are Samsung GALAXY Note2 and Meizu MX2. These two mobile phones are both of 1.6GHz CPU and 2GBytes RAM. The signature sequence length varied in $[100, 800]$. The verification time performance is illustrated in Figure 9. Due to the limited computing capability, the total response time increases in Android mobile phones comparing to that time in PCs. However, the total response time for one time verification is less than 15 seconds with the sequence size of 100. We think that is tolerable for an online mobile signature verification system.

5. **Conclusions.** In this paper, we investigated on how to implement a secure and reliable online mobile signature verification system. The homomorphic encryption is used to protect signature template. A secure iterative model is designed to perform encrypted DTW calculation between the mobile client and verification server. A restriction window-based strategy is leveraged to achieve a balance between authentication performance and computational efficiency. Moreover, we design a three-party storage mechanism to remove the risk of data leakage in a single-point failure situation. Through experimental analysis, the performance of DTW in HE domain is not degraded. Comparing to Gomez-Barrero's system, the response time of our system for single verification decreased significantly. Our proposed system is usable and deployable in a general mobile device.

As can been seen in Figures 5 and 6, the performance gap between the random forgeries experiment and the skilled forgeries experiment is still large. That is because the current feature vector only consists of the horizontal and vertical coordinates. Our future work

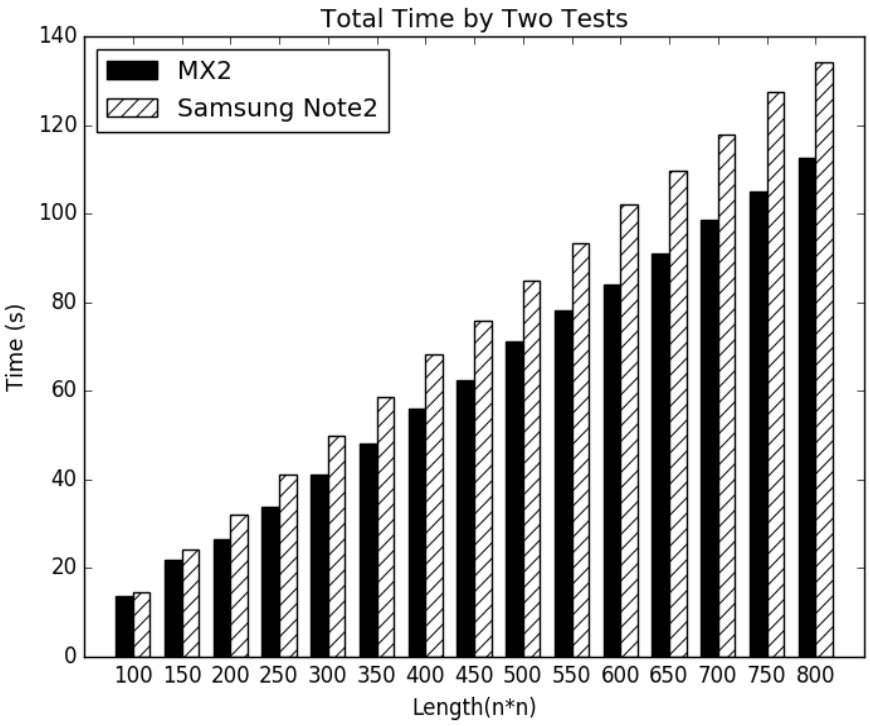

FIGURE 9. Response time in Android mobile phones

will try to introduce some statistical features in our HE scheme to reduce the effect of skilled forgeries.

Acknowledgment. This work is partially supported by the National Natural Science Foundation of China (No. 61672433), and the Fundamental Research Funds for the Central Universities (Project No. 3102016JKBJJGZ07) and Science and Technology Development Program of Weifang (2015GX008).

REFERENCES

[1] M. Lech and A. Czyzewski, A handwritten signature verification method employing a tablet, *Signal Processing: Algorithms, Architectures, Arrangements, and Applications (SPA)*, Poznan, Poland, pp.45-50, 2016.

[2] F. J. Zareen and S. Jabin, Authentic mobile-biometric signature verification system, *IET Biometrics*, vol.5, no.1, pp.13-19, 2016.

[3] N. Sae-Bae and N. Memon, Online signature verification on mobile devices, *IEEE Trans. Information Forensics and Security*, vol.9, no.6, pp.933-947, 2014.

[4] H. Zhu, X. Meng and G. Kollios, Privacy preserving similarity evaluation of time series data, *Proc. of EDBT*, pp.499-510, 2014.

[5] K. Nandakumar and A. K. Jain, Biometric template protection: Bridging the performance gap between theory and practice, *IEEE Signal Processing Magazine*, vol.32, no.5, pp.88-100, 2015.

[6] E. Maiorana, Biometric cryptosystem using function based on-line signature recognition, *Expert Systems with Applications*, vol.37, no.4, pp.3454-3461, 2010.

[7] C. Rathgeb and A. Uhl, Statistical attack against fuzzy commitment scheme, *IET Biometrics*, vol.1, no.2, pp.94-104, 2012.

[8] E. Maiorana, P. Campisi and A. Neri, Template protection for dynamic time warping based biometric signature authentication, *IEEE the 16th International Conference on Digital Signal Processing*, Santorini-Hellas, pp.1-6, 2009.

[9] A. Kholmatov and B. Yanikoglu, Biometric cryptosystem using online signatures, *International Symposium on Computer and Information Sciences*, pp.981-990, 2006.

[10] M. Gomez-Barrero, J. Fierrez and J. Galbally, Variable-length template protection based on homomorphic encryption with application to signature biometrics, *The 4th International Conference on Biometrics and Forensics (IWBF)*, pp.1-6, 2016.

[11] R. L. Lagendijk, Z. Erkin and M. Barni, Encrypted signal processing for privacy protection: Conveying the utility of homomorphic encryption and multi-party computation, *IEEE Signal Processing Magazine*, vol.30, no.1, pp.82-105, 2013.

[12] M. Yasuda, T. Shimoyama, J. Kogure, K. Yokoyama and T. Koshiba, Packed homomorphic encryption based on ideal lattices and its application to biometrics, *Lecture Notes in Computer Science*, vol.8128, 2013.

[13] P. Paillier, Public-key cryptosystems based on composite residuosity classes, *Proc. of Eurocrypt*, pp.223-238, 1999.

[14] K. K. Paliwal, A. Agarwal and S. S. Sinha, A modification over Sakoe and Chiba's dynamic time warping algorithm for isolated word recognition, *Signal Processing*, vol.4, no.4, pp.329-333, 1982.

[15] X. Yao and H. L. Wei, Off-line signature verification based on a new symbolic representation and dynamic time warping, *Proc. of the 22nd International Conference on Automation and Computing*, Colchester, pp.108-113, 2016.