International Journal on Advanced Science Engineering Information Technology

Vol.9 (2019) No. 3
ISSN: 2088-5334

Generating and Validating DSA Private Keys from Online Face Images for Digital Signatures

Asraa Safaa Ahmed[#1], Firas A. Abdullatif [*], Taha Mohammad Hasan[#2]

[#] *Department of Computer Sciences, Diyala University, Baghdad, 10062, Iraq*
E-mail: [1]altamimiasra@gmail.com; [2]dr.tahamh@sciences.uodiyala.edu.iq

[*] *Department of Computer Sciences, Baghdad University, Baghdad , 10053, Iraq*
E-mail: Firas.alobaedy@gmail.com

***Abstract*— Signing digital documents is attracting more attention in recent years, according to the rapidly growing number of digital documents being exchanged online. The digital signature proves the authenticity of the document and the sender's approval on the contents of the document. However, storing the private keys of users for digital signing imposes threats toward gaining unauthorized access, which can result in producing false signatures. Thus, in this paper, a novel approach is proposed to extract the private component of the key used to produce the digital signature from online face image. Hence, this private component is never stored in any database, so that, false signatures cannot be produced and the sender's approval cannot be denied. The proposed method uses a convolutional neural network that is trained using a semi-supervised approach, so that, the values used for the training are extracted based on the predictions of the neural network. To avoid the need for training a complex neural network, the proposed neural network makes use of existing pretrained neural networks, that already have the knowledge about the distinctive features in the faces. The use of the MTCNN for face detection and Facenet for face recognition, in addition to the proposed neural network, to achieved the best performance. The performance of the proposed method is evaluated using the Colored FERET Faces Database Version 2 and has achieved robustness rate of 13.48% and uniqueness of 100%.**

***Keywords*— digital signature algorithm; face biometrics; artificial neural networks; semi-supervised learning.**

I. Introduction

The high availability and ease of access to digital devices have encouraged the use of digital documents as a substitute for the use of papers. Digital documents are easier to store, organize and exchange, especially with the digital revolution that has enabled exchanging digital information easily [1, 2]. However, storing or exchanging these documents imposes the risk of tampering with them, by changing the contents of the original document [3]. Hence, the digital signature is proposed to certify the authenticity of such documents and provide a proof of the sender's approval to the contents of the document. Thus, a digital signature contains unique information about the document and the sender [4].

A unique string for the digital document can be generated using a unidirectional hash function, so that, the hash of the document is different when its contents are tampered with [5]. In a digital signature, this hash is encrypted using public-key cryptography, which requires private and public keys, one is used for the encryption and the other for decryption. The hash of the document is encrypted using the sender's private key to generate the digital signature. As the private key is only known by the sender, using it to sign the document proves the sender's approval on the contents of the document [6]. Thus, even when a new hash is calculated for the tampered document, a signature cannot be produced as the sender's private key is unknown.

According to this importance, different techniques have been proposed to deny any unauthorized access to the private keys. Among several types of authentication techniques, biometric authentication systems have shown significantly better security, according to the high uniqueness and robustness of biometric features. Therefore, these systems are being widely used to control the users' access to the digital signature systems, so that, users can only access their private keys [7, 8]. However, such an approach still requires storing the private key of the user in the database to identify and retrieve the private key to sign the documents when the user is authenticated. Such storage imposes a security threat, as intruders that gain access to the database can produce false signatures as soon as the private keys are retrieved [9, 10].

In this paper, a novel method is proposed to extract the private key of the users directly from their online face images. Hence, these keys are never stored in the database, which significantly improves the security of the digital signing system. Moreover, according to the variation in the images collected for the same user, imposed by different illumination and posing conditions, the extracted keys must be validated before being used to sign a document. The validation process relies on the public key of the user, which is stored in the database and known to all the receivers of the documents signed by the user to verify the signature.

II. Material and Method

A. Digital Signature Generation

As mentioned earlier, generating a digital signature for a certain document by a user requires two main components, the hash of the document and the private key of the user. According to the Digital Signature Standard DSS [11], the US secure hash algorithm (SHA-2) can be used to calculate the hash of the document, which is one of the widely used in digital signature generation, according to its non-colliding nature, i.e. it produces different hashes for different documents. Hence, tampering can be easily detected as this hashing algorithm ensures producing a different hash when contents are changed [12]. Several versions of the SHA have been proposed, which differ from each other by the computations executed to produce the hash and the size of the produced has [13-15]. However, as this study emphasizes on the private key extraction for the encryption phase of the digital signature generation and according to the popularity of the SHA-256 in digital signing [16, 17], this version of the SHA is used for signatures generation throughout the paper.

Normally, a ciphered message is encrypted using the public key, so that, acquiring the encrypted message does not result in revealing its original contents. However, in digital signature, the original content of the message is known, which is the hash of the digital document. Moreover, as the aim of the signature is to prove the identity of the sender, the hash is encrypted using the senders private key. Hence, the public key, which can be revealed to the users, can be used to decrypt the contents of the message [6]. Then, the received encrypted hash can be compared to the hash calculated for the received document to verify its authenticity. The Digital Signature Algorithm (DSA), which is a public-key cryptography method, is widely used to cipher the hash of the document and generate the digital signature, which is also recommended by the DSS.

To sign a document using the DSS, a private key is calculated, as shown in Algorithm 1, and used to generate a signature that can be verified using the sender's public key. To generate the private key, three numbers with specific characteristics are required, which as the *p*, *q* and *x*. According to the recommendations of the DSS [18], the length of *p*, i.e. *L* in algorithm 1, is required to be between 512 and 3072 bits and the selected value in that range is dividable by 64. This standard also recommends the use of *q* value with number of bits not greater than the number of bits in the produced hash.

Algorithm 1: Generating DSA's private and public keys.

Input: **Output:**	Length of *p* (L), length of *q* (N). User's private and public keys.
Step1:	$q \leftarrow$ N-bit random prime number. $p \leftarrow$ L-bit random number that satisfies the rule ((p-1) is multiple of q).
Step2:	$g \leftarrow$ A random integer that satisfies the rule ($g^q=1 \bmod p$).
Step3:	$x \leftarrow$ A random integer number as the private component $0<x<q$.
Step4:	$y \leftarrow g^x \bmod p$. The private component of the key.
Step5:	Return (p, q, g, x, y)

The parameters *p*, *q* and *g* are shared among the users of a certain digital signing system, i.e. the same parameters are used to calculate the signatures of documents for any users. Hence, the only private component that defines the signing user is *x*, which is required to be protected and never revealed to any person other than the owner of the key. Thus, this paper emphasizes on extracting the value of *x* from the online face image, so that, it is never stored in a database to improve the security of the system and the liability of the produced signature.

To produce a digital signature for a certain document by a specific user, the private key of the user is used to compute the digital signature using the hash of the document being signed as shown in Algorithm 2.

Algorithm 2: Generating the digital signature using DSA.

Input: **Output:**	Digital document (D), user U signing parameters (p, q, g, x) Digital signature for the document D by the user U.
Step1:	$H \leftarrow$ calculate the hash of document (D).
Step2:	$k \leftarrow$ Generate any random per-document value $0<k<q$. $r \leftarrow (g^k \bmod p) \bmod q$.
Step3:	If r=0 go to Step 2.
Step4:	$s \leftarrow k^{-1}(H+x\times r) \bmod q$
Step5:	If s=0 go to Step 2.
Step6:	Return digital signature (r, s).

To verify the signature generated by the user U for the document D, the steps shown in Algorithm 3 are executed, which does not require revealing the private component *x* of the parameters to the receiving user.

Algorithm 3: Signature verification procedure.

Input: **Output:**	Digital document (D), the digital signature of the document (r, s), user U public parameters (p, q, g, y). Validity of the received signature.
Step1:	If r<0 OR s<0 OR r>q OR s>q: Return **False.** Invalid signature.
Step2:	$H \leftarrow$ calculate the hash of document (D). $w \leftarrow s^{-1} \bmod q$
Step3:	$u_1 \leftarrow H\times w \bmod q$ $u_2 \leftarrow r\times w \bmod q$
Step4:	$v \leftarrow (g^{u_1} \times y^{u_2} \bmod p) \bmod q$
Step5:	If v=r: Return **True**. Valid signature exit.
Step6:	Return False.

B. Convolutional Neural Networks

Recently, the good performance of artificial neural networks in different applications has brought significant attention to these networks. Several types of neural networks have been proposed for different applications, where the distribution of the neurons in the network and the connections among them defines the type of the neural network. In a Convolutional Neural Network (CNN), the weights of a neuron are distributed in a two-dimensional array, known as a filter. By convoluting the filters of the neurons over the two- or three-dimensional input, local two-dimensional features can be detected, regardless of their positioning with respect to the border of the input. Hence, the performance of these networks when used to process images have shown significantly better results than any other type of neural networks.

Different convolutional neural networks have been proposed to detect and recognize faces in images. A face detection technique extracts the region of the image that contains the face while a recognition technique produces a descriptor that defines the face and can be used to match the face image with model images in the dataset. The use of Multi-Task Convolutional Neural Network (MTCNN), which detects certain face-features, such as eyes and mouth and define the region that contains it, is one of the widely used techniques in this field [19]. Another face detection method based on convolutional neural networks is the faster-RCNN [20]. Two of the popular CNN-based descriptor generation methods, which generate a fixed size vector per each input face, as the Facenet [21] and Arcface [22]. According to the intensive training required to allow CNNs to recognize the distinctive features of faces, the proposed method is implemented on top of the existing face detection and recognition techniques.

C. The Proposed Neural Network

The proposed neural network uses the 128-dimesional descriptors outputted from the pretrained face recognition neural network, i.e. Facenet or Arcface networks. As the values of these descriptors are expected to be very similar for face images from the same user and different for face images from different users, a simple deep neural network can process these inputs and produce similar numbers for the same user, which are different from those produced for other users. This number is used as a seed for a random generation algorithm that is used to generate the *x* value, which is the private component of the user in the DSA. The use of the generated value as a seed for the random generator, instead of using it as the value of *x* directly, is to allow the generation of the value that satisfies the requirement of *x*, as shown in Algorithm 1. As the random seed is required to be an integer positive number, the output layer is set to have a single neuron with the Rectified Linear Unit (ReLU) activation function. The topology of the proposed neural network is described in Table 1.

TABLE I
TOPOLOGY OF THE IMPLEMENTED NEURAL NETWORK.

Layer	Neurons	Activation
Input	128	-
Hidden1	512	ReLU
Hidden2	128	ReLU
Hidden3	64	ReLU
Output	1	ReLU

D. Training the Neural Network

The proposed neural network is required to learn to neglect the inter-class variations and consider the intra-class variations, so that, the outputted values are similar per each user and different for different users. However, the value that is suitable for each individual are unknown, so that, supervised training approach cannot be used for the implemented neural network. Assigning numbers to the users randomly can impose challenges toward training the neural network, as no rigid relations between the assigned number and the face images exist. Hence, a semi-supervised training approach is used to train the proposed neural network. This approach relies of the values produced by the neural network itself to recognize the value assigned per each user.

First, the neural network is initialized with random weights, which produces random outputs. These outputs are collected and distributed in a one-dimensional space with a predefined limit. The centroid of the predictions collected from the neural network per each user is calculated using the median function, which has the ability to neglected extremely different values, i.e. noise. After recognizing these centroids, their positions are normalized to the dimension of the space, i.e. the farthest prediction is set to the farthest edge of the space while the least value is set to position zero.

Next, the optimal distribution of the centroids is calculated by dividing the length of the space by the number of individuals in the dataset, so that, the maximum possible value between any two consequent centroids is maximized. The value assigned for each individual in the training dataset is equal to the value of the optimal centroid nearest to that point. However, to ensure linear expansion of the values, the assignment procedure is initialized from the farthest edge, so that, the largest value in the predictions is assigned with the largest optimal value. Such linear expansion allows the neural network to adjust the weights and biases to produce the required value without the need to change the features detected in the inputs. Algorithm 4 shows the procedure required to calculate the appropriate value per each individual in the training dataset.

Algorithm 4: Generating the optimal value per each user in the training dataset.

Input:	Predictions (P), Individuals (I), Space Length (S).
Output:	Labels for individuals (L)
Step1:	P ← Collect Predictions from NN. I ← List of the owner of each prediction. Optimal Values O ← List of values that split the space into the number of unique individuals.
Step2:	L ← Empty list with identical dimensions of P.
Step3:	For i in unique(I): C_i ← [i, median(P[I==i])] //calculate the median of the predictions per each user.

```
Step4:  While len(C)>0:
            Mi ← index of maximum value in C.
            Mo ← Max(O)
            L[I==C[Mi, 0]] ← Mo
            C ← C[C[:,0]≠C[Mi,0]]
            O = O[O≠Mo]
Step5:  Return L
```

Using the calculated optimal value per each user, the neural network is trained for a set of epochs. Then, new predictions are collected from the neural network and remapped on the space, so that, the value assigned for each user is updated. This update ensures the accommodation of any changes in the features detected by the neural network, while maintaining linear expansion. This procedure is repeated for a set of iterations to ensure that the neural network learns the intra- and inter-class variations.

E. Key Generation and Validation

Several face images are collected for the same user and used to generate several values of the random seed using the neural network. The most frequent value in the predictions is selected as the random seed to produce the private *x* value. However, as the key values are being extracted based on the face image, it is important to validate the generated private key per each user before generating the public key, and storing it in the database, and signing any digital document. Before generating the public key and storing it in the database, it is important to ensure that the generated private key does not conflict with any other keys stored in the database for other users. This validation can be achieved by generating a random hash value, i.e. a random value with the same size of the hash produces by the hash function. This random hash is used to generate a signature using the private key produces based on the random seed extracted from the face. Then, the public key of each user is used to. If any of the public keys in the database validates the generated signature, the new private key is considered invalid and the next most frequent value is selected to generate a new key. This process is repeated until a valid private key is generated. Thus, the values generated by the neural network are required to be unique, i.e. different users have different value, to reduce the time required to validate the private key in this stage.

Per each signing operation, the key generated for the user signing the document is validated using the user's public key. A random hash value is generated and signed using the extracted private key. If the user's public key validates the signed hash, the extracted key is considered valid and is used to sign the actual document. Otherwise, the key is considered invalid and the next most frequent value is used to generate a new key, until a valid key is created. Thus, the values produced by the neural network are required to be robust, i.e. identical for each user, to reduce the time required to calculate the private key per each signing procedure.

Moreover, as the random seed guarantees producing unique sequences of numbers, instead of unique numbers, it is important to use multiple numbers to calculate the value of *x* in the private key. In the proposed method, a window is defined to pass over a certain number of values in a sequence generated by the random generator. The random generator is set to output a value in the interval [0, 255], i.e. 8-bit integer. These bits are merged to produce a c×8-bit value, where c in the number of bytes the *x* value is required to have.

F. Performance Evaluation

According to the novelty of the proposed method, a benchmark is defined in this paper to evaluate the proposed method and to be used by any future researchers to evaluate their methods and compare their performances to the method proposed in this paper. The benchmark is defined according to the requirement of the random seed described in the previous section. The first performance measure is the robustness of the values produced by the neural network. Per each user u, the robustness of the predictions is equal to the frequency of the most frequent value f in the predictions divided by the number of predictions provided by the neural network p, which is equal to the number of face images collected from that user, as:

$$Robustness_u = \frac{|f_u|}{|p_u|} \quad (1)$$

The overall robustness of the system is calculated as the average of the robustness for all the users in the evaluation dataset. The other important performance measure is the uniqueness of the values produced for the user, where the value produced for a certain user is required to be unique in the predictions. Hence, the uniqueness of the values produced for the user *u* is calculated as:

$$Uniqueness_u = \frac{1}{\sum_{i=1}^{U} \begin{cases} 1 & f_i = f_u \\ 0 & f_i \neq f_u \end{cases}} \quad (2)$$

As the formula shows, the robustness for a unique value is equal to 100%, while if this value is found as the most frequent value for another individual the robustness becomes 50%. The overall robustness of the system is also calculated as the average of robustness values of all users.

Moreover, for accurate evaluation, the images used for the evaluation phase must be for individuals that are not included in the training phase. Instead of splitting the face images into training and testing sets, the individuals are split, then the images of the users in the training set are used for training while those for individuals in the training sets are used for the evaluation. This split ensures that the face images included in the evaluation have never been recognized in the training of the neural network. Hence, the produced evaluation measures are accurate and unbiased.

III. Results and Discussion

To evaluate the performance of the proposed method, the Color FERET face database version 2 [23] is used for training and evaluation. This dataset contains face images of 269 different individuals, where face images of the last 54 individuals are excluded from the training and used for the evaluation. In total, this database contains 3528 face images, where 585 images are for the 54 individuals included in the testing dataset.

Both face detection techniques, the MTCNN- and Faster RCNN-based, each with both the Facenet and Arcface face recognition methods. The descriptors outputted from these neural networks are used to train and evaluate the proposed neural network, to avoid the need for training a convolutional neural network to recognize facial features. The length of the axis that the produced values are distributed over is set to 10^9, as the seed of numpy's random generator, which is used to generate the *x* values, has a limited size of 32 bits, i.e. a maximum value of 4,294,967,295. The proposed neural network is trained for 1000 iterations, per each iteration the value required per each user is updated and the neural network is trained for 50 epochs. The performance measures of each combination are summarized in Table 2.

TABLE II
PERFORMANCE MEASURES OF THE PROPOSED METHOD USING DIFFERENT FACE DETECTION AND RECOGNITION TECHNIQUES.

Face Detection	Face Recognition	Detected Faces	Overall Robustness (%)	Overall Uniqueness (%)
Faster-RCNN	Facenet	2519	12.48	100
Faster-RCNN	Arcface	2519	11.07	98.15
MTCNN	Facenet	2641	13.58	100
MTCNN	Arcface	2641	12.96	98.15

The results show that the use of the descriptors generated by the Facenet neural network for the face images detected by the MTCNN has the highest performance, with 13.58% robustness and 100% uniqueness. Moreover, the 98.15% uniqueness in the results indicates that there are two individuals that have shared the same value of the private key.

To illustrate the training process, the centroids of the predictions provided by the neural network, the normalized centroids over the length of the space and the optimal value assigned for each user are shown in Figure 1 for the first, 500th and the last iterations. The values of each group are distributed in a different vertical level for better illustration, where these values actually have only one dimension. As Figure 1 shows, the predictions produced by the neural network before being trained are very close to zero. Then, be normalizing these values, individuals that have higher predictions' centroid are assigned with the higher values. During the training, the predictions start to expand linearly on the defined space, while maintaining the same relativity to ensure that the assigned values are suitable for the features detected by the neural network. By the end of the training, most of the centroids are positioned according to the optimal distribution, with the existence of some noisy values, which is expected behavior with biometric-based features. However, the 100% uniqueness show that to intersections have happened among the most frequent values of the users.

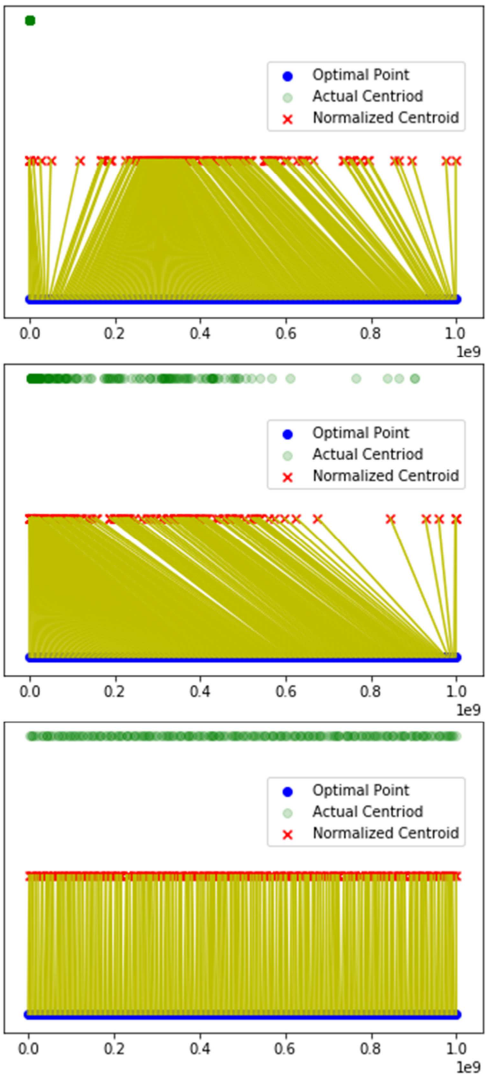

Fig. 1 Distribution of the centroids for the actual and normalized predictions and the optimal values; Top: First iteration; Middle: 500th iteration; Bottom: 1000th iteration.

The MTCNN has been able to detect 438 face images, out of the total 585 images. The histogram shown in Figure 2 shows the distribution of the number of individuals per each number of detected faces.

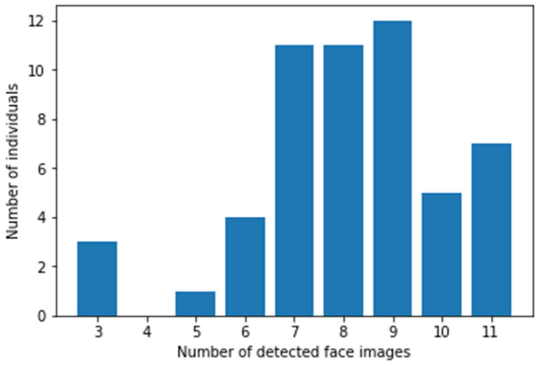

Fig. 2 Histogram of detected face images in the testing dataset.

As the extraction of the private component *x* in the key requires the extraction of the same random seed, such extraction from the face images of the same individual indicates a successful key generation and validation. Thus, per each individual in the testing dataset, the private key extraction is considered successful if the same random seed value is extracted from more than one face image. The average success rate per each number of face images per individual is shown in Figure 3. These results show that the existence of more face images increases the success rate, i.e. collecting more face images increases the probability of producing a valid private key component. However, the extraction of the private component for the single user with five face images has not been successful.

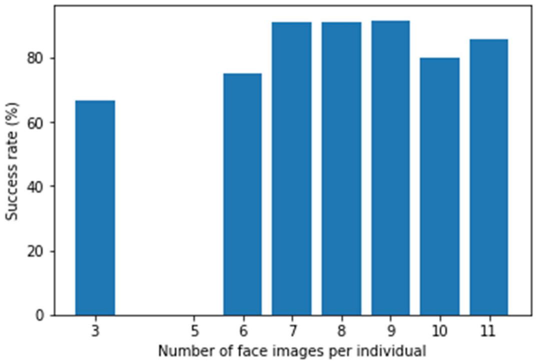

Fig. 3 Success rate versus the number of face images per individual.

IV. Conclusion

Digital signatures are being widely used to sign digital documents, according to the exponentially increasing number of documents being communicated over the internet. A digital signature can be used to prove the authenticity of the document and the approval of the sender upon the contents of that document. Storing the private key of the signer imposes security threats, as gaining access to the database by an attacker enables producing false signatures. Thus, a novel method is proposed in this paper to extract the private component of the DSA from the user’s face. Using such approach, the private component of the key is never stored in the system’s database, so that, even if access to the database is acquired, a false signature can never be produced and approval to the contents of the document cannot be denied. The proposed method uses a neural network to produce a random seed that is used to generate the private key of the user. This neural network is trained using a semi-supervised approach, so that, the predictions of the neural network are used to produce the labels used to train the network. This approach ensures that the values used for training are based on the features extracted by the neural network itself. Different face detection and recognition techniques are evaluated for the proposed method, where the use of MTCNN for face detection and Facenet for face recognition has achieved the highest performance with 13.48% robustness and 100% uniqueness.

In future work, the use of fingerprints is going to be evaluated, as the features in such biometric templates are more robust than those from a face image. Hence, the use of fingerprint images is expected to improve the robustness of the system.

References

[1] J. Rothenberg, "Ensuring the longevity of digital documents," Scientific American, vol. 272, pp. 42-47, 1995.

[2] A. M. French and J. P. Shim, "The Digital Revolution: Internet of Things, 5G, and Beyond," CAIS, vol. 38, p. 40, 2016.

[3] P. Singh, B. Raman, and P. P. Roy, "Detection of seal and signature entities with hierarchical recovery based on watermark self embedding in tampered digital documents," Displays, vol. 54, pp. 47-59, 2018.

[4] S. Saxena and D. Anand, "A Novel Digital Signature Algorithm based on Biometric Hash," International Journal of Computer Network and Information Security, vol. 9, p. 12, 2017.

[5] H. E. Michail, G. S. Athanasiou, G. Theodoridis, A. Gregoriades, and C. E. Goutis, "Design and implementation of totally-self checking SHA-1 and SHA-256 hash functions’ architectures," Microprocessors and Microsystems, vol. 45, pp. 227-240, 2016.

[6] D. E. Denning, "Digital signatures with RSA and other public-key cryptosystems," Communications of the ACM, vol. 27, pp. 388-392, 1984.

[7] Y. Isobe, Y. Seto, and M. Kataoka, "Development of personal authentication system using fingerprint with digital signature technologies," in Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001, p. 9 pp.

[8] E. Rahmawati, M. Listyasari, A. S. Aziz, S. Sukaridhoto, F. A. Damastuti, M. M. Bachtiar, et al., "Digital signature on file using biometric fingerprint with fingerprint sensor on smartphone," in 2017 International Electronics Symposium on Engineering Technology and Applications (IES-ETA), 2017, pp. 234-238.

[9] M. Malik and T. Patel, "Database securityattacks and control methods," International Journal of Information, vol. 6, pp. 175-183, 2016.

[10] R. M. Thiyab, M. A. Ali, and F. Basil, "The impact of SQL injection attacks on the security of databases," in Proceedings of the 6th International Conference of Computing & Informatics, 2017, pp. 323-331.

[11] C. Kerry and P. Gallagher, "FIPS PUB 186-4: Digital Signature Standard (DSS)," Federal Information Processing Standards Publication. National Institute of Standards und Technology, 2013.

[12] K. Kumar Raghuvanshi, P. Khurana, and P. Bindal, "Study and comparative analysis of different hash algorithm," Journal of Engineering Computers & Applied Sciences, vol. 3, pp. 1-3, 2014.

[13] D. Eastlake 3rd and P. Jones, "US secure hash algorithm 1 (SHA1)," 2070-1721, 2001.

[14] G. Gupta and S. Sharma, "Enhanced SHA-192 Algorithm with Larger Bit Difference," in 2013 International Conference on Communication Systems and Network Technologies, 2013, pp. 152-156.

[15] T. Lakshmanan and M. Madheswaran, "Security and robustness enhancement of existing Hash algorithm," in 2009 International Conference on Signal Processing Systems, 2009, pp. 253-257.

[16] R. Kaur and A. Kaur, "Digital signature," in 2012 International Conference on Computing Sciences, 2012, pp. 295-301.

[17] T. Pornin, "Deterministic usage of the digital signature algorithm (DSA) and elliptic curve digital signature algorithm (ECDSA)," 2070-1721, 2013.

[18] G. Locke and P. Gallagher, "Fips pub 186-3: Digital signature standard (dss)," Federal Information Processing Standards Publication, vol. 3, pp. 186-3, 2009.

[19] K. Zhang, Z. Zhang, Z. Li, and Y. Qiao, "Joint face detection and alignment using multitask cascaded convolutional networks," IEEE Signal Processing Letters, vol. 23, pp. 1499-1503, 2016.

[20] H. Jiang and E. Learned-Miller, "Face detection with the faster R-CNN," in 2017 12th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2017), 2017, pp. 650-657.

[21] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in Proceedings of the IEEE conference on computer vision and pattern recognition, 2015, pp. 815-823.

[22] J. Deng, J. Guo, N. Xue, and S. Zafeiriou, "Arcface: Additive angular margin loss for deep face recognition," arXiv preprint arXiv:1801.07698, 2018.

[23] NIST. (2016, 28/04/2019). The Color FERET Database Version 2. Available: http://www.nist.gov/itl/iad/ig/colorferet.cfm