Privacy Preserved Authentication: A Neural Network Approach

Ray R. Hashemi
Amar Rasheed
Jeffrey Young
Department of Computer Science
Georgia Southern University,
Savannah, GA, USA
e-mails: {rayhashemi, amarrasheed, alanyoung7}@gmail.com

Azita A. Bahrami
IT Consultation
Savannah, GA, USA
e-mail: Azita.G.Bahrami@gmail.com

***Abstract*—The anonymity of users during the authentication process for accessing computer-based Safety-Critical Systems (SCSs) is crucial for two reasons: (i) ever growing dependency of users on SCSs and (ii) Internet of Things (IoT), social media, and marketers put the privacy of users of SCS in jeopardy more than ever. The goal of this research effort is to introduce and develop a novel neural network-based system that is able to (a) employ Extracted Eelectro-Cardiogram (ECG) feature vectors of the user as biometric credentials for authentication, (b) preserve the privacy of users during the authentication process and (c) attest the authenticity of clients on a continuous basis during the time that the SCS serves the client. Such attestation is necessary to make sure the user, after initial successful authentication, has not been replaced by an entity with malicious intent. Ten datasets with the total of 246,690 synthesized ECG feature vectors were created to test the system. These vectors were generated out of borrowed real ECG feature vectors for 90 users. Each dataset had 2,169 legitimate users' credentials and 22,500 illegitimate ones. Our neural network-based system revealed the accuracy of (99.98%), precision of (100%), and sensitivity of (99.82%).**

Keywords-Anonymous Authentication; Encryption; Neural Network; Dynamic Authentication; Neural Network-based Authentication; Continuous Attesting Authenticity.

I. INTRODUCTION

The failure of Safety-Critical computer-based Systems (SCSs) may cause economic loss, loss of life, or both. These systems are at work in every segment of society including banking, state and federal elections, business, travel, service, military, manufacturing, insurance, hospitals, medicine, etc. There is a large class of SCSs with the following desired properties:

(a) Being accessed by a set of legitimate clients frequently,
(b) Preserving the clients' privacy during the authentication process,
(c) Attesting the authenticity of clients on a continuous basis during the time that the SCS serves the client.

The first property is innate in all SCSs because, in general, all SCSs are exposed to some degree of controlled access. The second property is crucial for two reasons: (i) ever growing dependency of users on SCSs and (ii) Internet of Things (IoT), social media, and marketers put the privacy of users of SCSs in jeopardy more than ever.

The second property also inherently proposes a major challenge. The challenge stems from the fact that the preservation of privacy and enforcement of security are at odds with each other. To support such oddity, the credentials by which a client seeks access to a SCS is transformed before being presented to the SCS. The intention is to make sure that the actual credentials can neither be seen by the SCS nor can the SCS get the actual credentials through the process of reverse engineering. Therefore, the transformation process totally takes place on the client side. In addition, every time that client wishes to access the SCS, the transformed version of the credentials must be different, although the client credentials remains the same. This is necessary to discourage any discovery attempt of the client credentials. At the SCS side, there is a depository of client credentials that are used for confirming the authenticity of the legitimate clients. Since the client credentials are transformed, each credential in the SCS depository also needs to be transformed. (Obviously, the transformation function used on the client side cannot be used on the SCS side.) The similarities between the transformed client's credentials and each one of the transformed credentials in the depository of the SCS are measured. The authentication is confirmed if the similarities are above a predefined threshold.

The third property is an antidote to laxation of the SCS after initial authentication. In other words, SCS requires a test of assurance that, during the period of service, the client has not been replaced by an entity with malicious intent. Although the credentials of the user are not changing during the service period, the transformation of the credentials has to change.

The goal of this research effort is to introduce and develop a novel neural network-based system that is able to (a) employ Extracted Electro-Cardiogram (ECG) feature vectors of a user as credentials for authentication, (b) preserve privacy and enforce the authentication process and (c) attest continuously the authenticity of clients who are using the SCS service. Since the transformation functions

 ISBN: 978-1-61208-746-7

for transforming credentials on the client side and SCS side ought to be different, two new neural networks (one for the client side and one for the SCS side) are introduced.

One may ask why the ECG feature vectors are chosen for authentication. The answer is that ECG vectors are much less susceptible to compromise in comparison to the other biometric measures such as fingerprint, iris etc. [1].

The rest of the paper is organized as follows. The Previous Works are the subject of Section 2. The Methodology is presented in Section 3. The Empirical Results are discussed in Section 4. The Complexity Analysis of the system is the subject of Section 5. The Conclusions and Future Research are covered in Section 6.

II. PREVIOUS WORKS

Due to explosion of IoT and social media, privacy-preserved authentication has received tremendous attention over the last two decades. In general, four different paradigms are used: *hamming distance* paradigm, *oblivious* paradigm, *zero-knowledge proof* paradigm, *verifiable common secret encoding* paradigm, and *hybrid* paradigm. Secured weighted hamming distance and its modified versions are the nucleus of the hamming distance paradigm [2][3].

According to the oblivious paradigm, SCS has several strings of information and transfers one of the strings to the receiver and after that remains inattentive and or unconcerned (oblivious) about the transferred string of information [4][5].

According to the zero-knowledge proof paradigm, the client is able to prove his/her credentials to the SCS many times using polynomial authentication [6]-[8].

According to the verifiable common secret encoding paradigm, clients are arranged in groups and groups are dynamically formed by using a set of public keys ids. The privacy of the client is preserved through proving that it is an active member of a certain group [9][10].

According to the hybrid paradigm, a combination of more than one of the above mentioned paradigms are used [1] [11]-[13]. For example, a combination of hamming distance paradigm and oblivious paradigm are used frequently. We introduce and evaluate a novel privacy-preserved authentication paradigm—Neural Network-based paradigm.

III. METHODOLOGY

The methodology for meeting the three-prong goal of this research is explained in detail in this section. Let X be the binary vector of length N representing features for one biometric measurement of a user and also let Γ be a set of binary vectors on the SCS side that if the similarity of X with one of the elements in Γ is within an acceptable range then, X is valid. We separately explain steps taken in both client side and SCS side to provide access to SCS while preserving the privacy of users in the following two subsections. The reader needs to be reminded that providing access to SCS and preserving the privacy of users are the first two prongs of the goal for this study. The details of the last prong of the goal are the subject of the third sub-section.

A. Actions on the Client Side

We take X and divide it into equal size sections of $x_1, \ldots, x_n$. Let the number of bits in x_i be m. We build a *semi-feed forward neural network* with two layers of input and output. A feed forward neural net is unidirectional. That is, the difference between the output vector and target vector is not fed backward. Therefore, the weight matrix for the connections between the nodes of input layer and the nodes of output layer do not change. However, we call our neural network a semi-feed forward net because the weight matrix is updated after each input vector completes its journey through the net. We shortly introduce the updating process for the weight matrix.

The input layer has (m+1) nodes (the extra node is a bias node) and output layer has only m nodes. Sections x_i (for i = 1 to n) are used as input vectors to the net. The input for the bias node is always one, as shown in Figure 1.

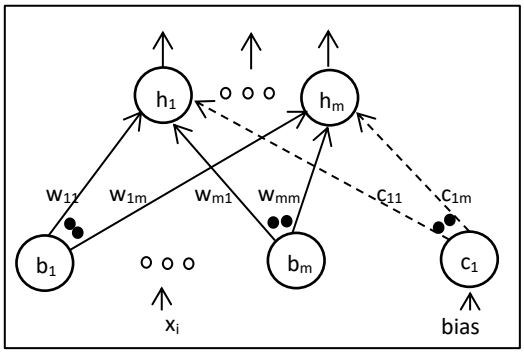

Figure 1. The client side feed forward neural network architecture

The initial weight matrix, W is:

$$W=\begin{bmatrix} a_{1,1} & e_{1,2} & \ldots\ldots & & e_{1,m} \\ e_{2,1} & a_{2,2} & e_{2,3} & \ldots\ldots & e_{2,m} \\ e_{3,1} & e_{3,2} & a_{3,3} & e_{3,4}\ \ldots\ldots & e_{3,m} \\ & \ldots\ldots & & & \\ & \ldots\ldots & & & \\ e_{m,1} & e_{m,2} & e_{m,3} & \ldots\ldots & a_{m,m} \\ e_{m+1,1} & e_{m+1,2} & e_{m+1,3} & \ldots\ldots & e_{m+1,m} \end{bmatrix}$$

Elements $a_{i,i}$ of W are calculated using (1), where d is a random integer value >2.

$$a_{i,i} = 2_{k-i+1},\ k \geq d*m+1 \qquad (1)$$

Elements $e_{i,j}$ of W are calculated using (2), where $c_{i,j}$ is a non-negative integer random number less than 2_m. (c_{ij} is randomly generated for each $e_{i,j}$.)

$$e_{i,j} = c_{i,j} * 2_m \quad (2)$$

The output of j-th node of the output layer for the input vector of I is calculated using (3), where $W_{*,j}$ means the j-th column of weight matrix W.

$$o_j = \sum_{j=1}^{m} IW_{*,j} \quad (3)$$

After an input vector completes the feed forward step, before the next input vector be fed to the net, all the e_{ij} elements of W are replaced by a new e_{ij} that is randomly generated using (2). This is an extra effort in preserving the privacy of the client.

The input vector of x_i generates an output vector that serves as a column of a new matrix. That is, using the neural net for input vectors of x_i (for i = 1 to n) generates a new matrix G such that the output for input vector x_i is the i-th column of matrix G. Therefore, G is a matrix of m rows and n columns. Matrix G is the one that leaves the client side as the transformation of X. SCS has neither the knowledge of input vectors nor weight matrix W and its updates.

B. Actions on the SCS side

The SCS receives only matrix G from the client side and the process of authentication is completed in two phases. Details of each phase are the subject of the following two subsections.

1) Phase I: Let Y be one of the several existing binary vectors in the depository of the SCS side representing a user. We shortly introduce another *semi-feed forward neural network* (different from the one used on the client side) that transforms Y. The differences between the transformed Y and transformed X are measured and if the difference is higher than an acceptable threshold then, X matches Y. The same process is repeated for every users' binary vectors until the authenticity of X is either validated or denied.

The vector Y is divided into equal size sections of $y_1, \ldots, y_n$ such that the number of bits in y_i is m. We convert y_i to a matrix of m by 1 and as a result Y is converted to a matrix, Z, of m rows and n columns.

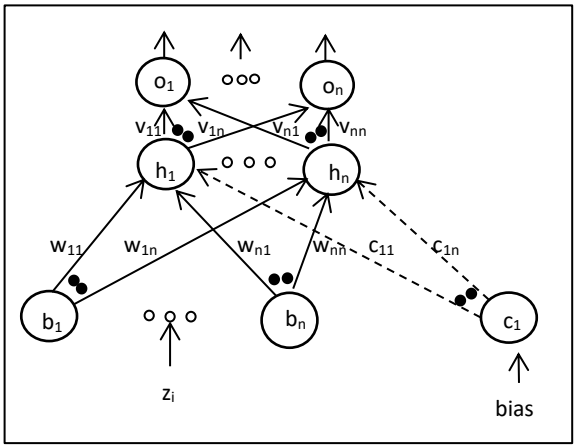

Figure 2. The SCS side feed forward neural network architecture

We introduce the *semi-feed forward neural network* for use in the SCS side that is different from the one introduced for the client side. The new neural net has three layers of input, hidden and output, as shown in Figure 2.

Two weight matrices are needed (one for connections between the nodes of input –hidden layers, W, and one for the connections between the nodes of hidden-output layers, V.) Creating and updating processes of the weight matrices are also different from the one for the client side. The input layer has (n+1) nodes (the extra node is a bias node), both the hidden layer and the output layer have n nodes. Each row of matrix Z along with an input of 1(for the bias node) serves as input to the feed forward neural network.

The initial weight matrix, W, has n+1 rows and n columns and it is built in two steps. During the first step, $W = (2_{m-1}) * W'$, where W' is an identity matrix of n X n. During the second step, the k-th row of G (k =1 for the initial weight matrix) is added to W to serve as the $(n+1)_{th}$ row of W. The value of k is increased by one for each incoming input vector.

$$W = \begin{bmatrix} a_1 & 0 & \ldots\ldots & & 0 \\ 0 & a_2 & 0 & \ldots\ldots & 0 \\ 0 & 0 & a_3 & 0 \ldots\ldots & 0 \\ & \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot & & & \\ & \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot\ \cdot & & & \\ 0 & 0 & 0 & \ldots\ldots & a_n \\ g_{k,1} & g_{k,2} & g_{k,3} & \ldots\ldots & g_{k,n} \end{bmatrix}$$

For the first input vector, the initial W is used. For the next input vector, W changes using (4) and (5).

$$new(w_{ij}) = old(w_{ij})/2 \quad (4)$$
(for i = 1 to m and j =1 to n)

$$new(w_{ij}) = g_{k,j} \quad (5)$$
(for i = m+1, j = 1 to n, and k = k+1)

The output of the j-th node of the hidden layer for the input vector of I is calculated using (1).

The weight matrix, V, for connections between nodes of the hidden layer and output layer has n rows and n columns. V is a binary matrix and it is randomly created such that every row and every column of V contains only one 1. Therefore, the total number of ones in V is equal to n. For each input vector, a new V is randomly generated. Let vector H be the output of the hidden layer nodes, the output of the j-th node in the output layer is calculated by (1). Using this neural net for all input vectors delivers a matrix of m rows and n columns, P.

2) Phase II: Let the largest element in matrix P be L bits long when it is converted into binary. We take the binary equivalence of p_{11}, first element in P, and padded with zeros (if needed) to make its length equal to L. We rotate the binary number to the left i=1 places and take the m+1 least

significant bits as a value (q). The counter h is set using (6), where $r = q \bmod (2^m + 1)$.

$$h = (m_{r+1}) * \lfloor Cos(r) \rfloor \quad (6)$$

The above process is repeated for each remaining element, p_{ij}, in P using (7).

$$h = h + (m_{r+1}) * \lfloor Cos(r) \rfloor \quad (7)$$

The ultimate outcome of the second phase is h. Let N be the length of X and, thus, the length of Y. The threshold value $T = m(N - \beta)$. Let us explain what β is. In reality, X and Y are extracted feature vectors of a biometric of interest for a user. The extracted feature vectors are not always exactly the same and they may differ by negligible number of bits—β. The value for β is selected in such a way that the ratio of β/N is extremely small. If h > T then, X and Y are matched.

C. Attestation

Our neural network-based system employs Extracted Electro-Cardiogram (ECG) feature vectors of users as credentials for authentication. An ECG shows the electrical signals of a human heart as a waveform and it is unique for each individual. The components of the ECG waveform are named P, Q, R, S, T, U, and V as illustrated in Figure 3.

The uniqueness of this waveform for each person makes it a vital biometric candidate for authentication. The physical characteristics of the ECG such as relationships among R, Q, and S peaks individually or collectively, duration and shape of P, T, and U waves and their relationships (that represent depolarization and repolarization phases of human heart) may be used as features of an ECG. To obtain such features: (i) several ECG waveforms of a person are recorded for a short period of time and (ii) recorded waveforms are analyzed using either fiducial points based approaches [14] [15] or pattern recognition based approaches [16][17], to conclude the features of ECG for the person. Such features are claimed to be independent of the heartrate [16].

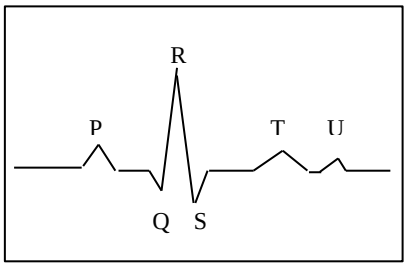

Figure 3. Components of ECG waveform

To complete attestation, a portable health device with sensors that are able to read constantly the ECG waveforms of a person may be used. Let us assume that the ECG of a person is read for t_1 units of time. These waveforms go to the process of ECG analysis for extraction of ECG features and the analysis process takes t_2 units of time. The authentication process using the neural network-based paradigm takes t_3 units of time. Thus, for the very first reading of ECG, it takes $t = t_1 + t_2 + t_3$ units of time that authentication be completed. However, ECG reading is done by a different device than the one in charge of all computations; therefore, the attestation is repeated every $t' = t_2 + t_3$ units of time, after the authentication is completed for the first reading of ECG.

IV. EMPIRICAL RESULTS

For a given authentication system, let us assume that out of U_1 number of users who have valid credentials only T_P of them were positively authenticated by the system and, therefore, F_N of them were rejected ($U_1 = T_P + F_N$.) Let us also assume that out of U_2 number of users who have invalid credentials only T_N of them were rejected by the system and, F_P of them were not ($U_2 = T_N + F_P$). The accuracy, precision, and sensitivity of the authentication for the system are calculated using (8), (9), and (10), respectively.

$$\text{Accuracy} = (T_P + T_N)/(U_1 + U_2) \quad (8)$$
$$\text{Precision} = T_P/(T_P + F_P) \quad (9)$$
$$\text{Sensitivity} = T_P/U_1 \quad (10)$$

Bhutra et al. [1] reported extraction of a 240-bit long feature vector from a person's ECG that is recorded for 20 seconds and digitized at 500 Hz with 12-bit resolution over a nominal ±10 mV range. We assumed that the negligible number of bits is one (β = 1). To examine the behavior of our neural network-based authentication, the test dataset was borrowed from [1]. This dataset included feature vectors for 90 different eligible users and it was used as the depository of credentials on the SCS side.

For the client side, we repeated the following process for every one of the 90 feature vectors, FV_i. Out of the 240 bits in FV_i, randomly k ($2 \le k \le 10$) bits of the FV_i were chosen and corrupted (i.e., flipped) to generate a new FV_i. We generated 2500 new corrupted feature vectors out of each FV_i and thus, the total of 225,000 corrupted ones out of the original 90 feature vectors. (The reader needs to be reminded that the total possible corrupted vectors that can be created out of one FV_i using $2 \le k \le 10$, is more than 10^{17}. We just randomly generate 2500 of them.) Ten datasets of equal size were created randomly out of the corrupted feature vectors and named $D_1 \ldots D_{10}$ such that D_i contained 22,500 corrupted vectors (invalid credentials).

Following the same procedure for k =1, the total of 21,600 corrupted feature vectors generated using the original 90 vectors. The new corrupted vectors were treated as the original vectors with a negligible number of corrupted bits (β = 1). These vectors were randomly and equally added to the $D_1 \ldots D_{10}$ such that every vector appeared in only one of the datasets. (Each D_i was expanded by 2160 new vectors. In addition the original 90 vectors were randomly and

equally divided among the ten datasets such that each vector appeared in only one dataset. As a result, each dataset D_i had the total of 24,669 vectors of which 22500 of them were invalid and 2169 vectors were valid. We measured the accuracy, precision, and sensitivity of our neural network-based authentication approach using the ten datasets and results are shown in Table I.

TABLE I: THE AVERAGES OF ACCURACY, PRECISION, AND SENSITIVITY MEASURES FOR OUR NEURAL NETWORK-BASED AUTHENTICATION APPROACH USING THE TEN DATASETS OF $D_1 \ldots D_{10}$.

Dataset	Accuracy	Precision	Sensitivity
D1	100	100	99.95
D2	99.99	100	99.91
D3	99.96	100	99.54
D4	100	100	99.95
D5	99.97	100	99.63
D6	100	100	99.95
D7	99.97	100	99.68
D8	99.98	100	99.72
D9	99.99	100	99.91
D10	99.99	100	99.91
Average	99.98%	100%	99.82%

We also examined whether the rejection or acceptance of a feature vector was dependent on the locations of those bits that are different between the two transformed vectors of X and Y. To explain further, let us assume that X and Y are both divided into 6 sections of (x1, x2, x3, x4, x5, and x6) and (y1, y2, y3, y4, y5, and y6) and each section is 3 bits long (thus, n = 6 and m = 3). Let us also assume that the number of bits that are different between X and Y is 6. These six bits may have several different distributions within X. For example, in one distribution, one bit is different in each corresponding section of x_i and y_i (for i =1 to 6). In another distribution, all the bits in x1and x2 (total of 6 bits) are different from the bits in y1 and y2, respectively. In a third distribution, two bits in each section of x1, x2, and x3, (total of 6 bits) are different from two bits in each section of y4, y5, and y6. Is the authentication influenced by distribution of the different bits between the vectors of X and Y?

To find the answer to the proposed question we selected randomly one of the 90 original feature vectors, Vector Y. We used m = 4 and n =60 to create 4-bit long 60 sections for Y. A new vector, X, was created. We assumed that the number of bits that are different between X and Y is B_i (for i = 3 to 30). The vector Y was considered as the true credentials and vector X was the one in which distributions of B_i flipped bits took place. For each B_i 200 different distributions of B_i flipped bits in Y were generated that collectively made a group of distributions for B_i—G_{Bi}. Each vector in the group is a new X vector and all X vectors in the group have the same number of B_i bits that are different from Y. The total number of distributions' groups was 27. We used every new X against the true credentials vector Y and findings showed that the authentication process is not influenced by the locations of B_i flipped bits.

V. COMPLEXITY ANALYSIS

On both the client side and the SCS side, the credentials are divided into n sections of m-bit long. The weight matrix W, on the client side has m +1 rows and m columns. Therefore, smaller m means smaller W. However, smaller m means larger n, which makes the weight matrices of W and V on the SCS side larger because they have n+1 rows and n columns. As a result, in choosing n and m one may pay attention to the size of the weight matrices. The Reader needs to be reminded that the size of m and the number of input records are moving in two different directions.

Let us assume that N = n and n >> m. The worst case is when the size of m becomes equal to the size of n and the size of the weight matrix is, therefore, N_2. The time complexity for the neural network on the client side, the neural network of phase I on the SCS side, and computation of phase II of SCS are $O(N_2)$, $O(2(N)_2)$, and $O(N_2)$, respectively. The total time complexity is $O(4N_2)$. Since N is a very small number so is the time complexity.

VI. CONCLUSIONS AND FUTURE RESEARCH

Two well-known paradigms of privacy based authentications are Zero Knowledge Proof (ZKP) and Verifiable Common Secrete Encoding (VCSE). In general, the former one uses a set of hardcoded parameters that are essential to its performance. The device that runs ZKP could be profiled by a *Side-Channel Attack* [18], which in turn could be used to disclose the set of parameters. The consequences of the parameters' disclosure are compromising the device and subsequently back engineering the authentication secrets. In contrast, our methodology (use of a neural network) creates a different set of weights every time it is used. This means that no matter how much profiling is done, no intrinsic data can be compromised.

VCSE uses public/private key encryption. It maintains anonymity by the use of a dummy list of keys, which is sent to the server. The server encrypts every key in the list using the same session key, concatenates each encrypted key with the same random number, r, and returns the list to the client. The client, in turn, decrypts and sends back r to be validated by the server. This makes the anonymity provided by VCSE considerable. The problem is that due to the overhead, the VCSE consumes a large amount of system resource, which results in a slow execution especially on devices with a small amount of resources. This is not the case with our neural network approach, which uses very little system resource to achieve anonymity.

In addition, our privacy preserving authentication approach also shows almost a perfect accuracy, precision, and sensitivity.

As future research, development of a neural network-based hybrid authentication system is in progress that will be

 ISBN: 978-1-61208-746-7

tailored toward authentication at Boundaries of Cyber-Physical Systems.

REFERENCES

[1] G. Bhutra, A. Rasheed, and R. Mahapatra, "Privacy-Preserving ECG based Active Authentication (PPEA2) for IoT Devices", IEEE International Performance Computing and Communication Conferences (IPCCC 2018), Special Session on Networking in Cyber Physical Systems, pp. 1-8, 2018.

[2] R Kulkarni and A. Namboodiri, "Secure hamming distance based biometric authentication", Proceedings of 2013 International conference on Biometrics, pp. 1-6, 2013.

[3] M. Yasuda, "Secure Hamming distance computation for biometrics using ideal-lattice and ring-LWE homomorphic encryption", Online Information Security Journal: A Global Perspective, 26(2): 85-103, 2017.

[4] M. S. Kiraz, B. Schoenmakers, and J. Villegas, "Efficient committed oblivious transfer of bit strings", in Information Security, Lecture Notes in Computer Science, Springer, 4779:130–144, 2007.

[5] T. Tassa, "Generalized oblivious transfer by secret sharing", International Journal of Designs, Codes and Cryptography, Springer, 58(1):11–21, 2011.

[6] O. Goldreich and Y. Oren, "Definitions and Properties of Zero-Knowledge Proof Systems", Journal of Cryptology, Springer Verlag, 7:1-32, 1994.

[7] H. Wu, F. Bao, D. Ye, and R. Deng, "Cryptanalysis of Polynomial Authentication and Signature Scheme", Proceedings of the 5th Australasian Conference on Information Security and Privac0, Brisbane, Australia, 1841: 278-288, 2000.

[8] C. Rackoff and D. R. Simon, "Non-Interactive Zero-Knowledge Proof of Knowledge and Chosen Ciphertext Attack", Proceedings of the International Cryptology Conference, Springer Lecture Notes in Computer Science, 576: 433-444, 1991.

[9] T. Rabin and M. Ben-Or, "Verifiable secret sharing and multiparty protocols with honest majority", Proceedings of the twenty-first annual ACM symposium on Theory of computing, Seattle, Washington, pp. 73-85, 1989.

[10] C. Cachin, K. Kursawe, A. Lysyanskaya, and R. Strobl, "Asynchronous verifiable secret sharing and proactive cryptosystems", Proceedings of the 9th ACM conference on Computer and communications security, Washington Dc, pp. 88-97, 2002.

[11] M. S. Kiraz, Z. A. Genc, and S. Kardas, "Security and Efficiency Analysis of the Hamming Distance Computation Protocol Based On Oblivious Transfer", Security and Communication Networks, Wiley Online Library, 8(18):4123-4135, 2015.

[12] J. Bringer, H. Chabanne, and A. Patey, "Shade: Secure hamming Distance Computation from Oblivious Transfer", in Financial Cryptography and Data Security, Lecture Notes in Computer Science, Springer, 7862: 164–176, 2013.

[13] Y. Xi, K. Sha, W. Shi, L. Schwiebert, and T. Zhang, "Probabilistic Adaptive Anonymous Authentication in Vehicular Networks", Journal of Computer Science and Technology, 23(6):916-928, 2008.

[14] L. Biel, O. Pettersson, L. Philipson, and P. Wide "ECG analysis: a new approach in human identification", IEEE Transactions on Instrumentation and Measurement, 50(3): 808-812, 2001.

[15] S. Yazdani and J. Vesin, "Extraction of QRS Fiducial points from the ECG using adaptive mathematical morphology", Journal of Digital Signal Processing, Academic Press, 56:100-109, 2016.

[16] S. Israel, J. Irvine, A. Cheng, M. Wiederhold, and B. Wiederhold, "ECG to identify individuals." Pattern recognition, 38(1):133-142, 2005.

[17] K. N. Plataniotis, D. Hatzinakos, and J. K. M. Lee, "ECG biometric recognition without fiducial detection", Biometrics Symposium: Special Session on Research at the Biometric Consortium Conference, Baltimore, pp. 1-6, 2006.

[18] C. Carlet and E. Prouff, "Polynomial Evaluation and Side Channel Analysis", in the New Codebreakers, Lecture Notes in Computer Science, Springer, 9100: 315-341, 2016.

 ISBN: 978-1-61208-746-7