OPEN ACCESS
Indo-JC
Indonesian Journal on Computing
ISSN 2460-9056
socj.telkomuniversity.ac.id/indojc

Ind. Journal on Computing
Vol. 9, Issue. 1, April 2024. pp. 30-45
doi:10.34818/indojc.2024.9.1.909

Securing KTP Data Using QR Code Modification and Elliptic Curve Cryptography

Rakha Aditya Nugraha [#1], Ari Moesriami Barmawi [#2]

 # School of Computing, Telkom University
Jl. Telekomunikasi, Terusan Buahbatu, Bandung, Indonesia

[1] rakhaadityanugraha@student.telkomuniversity.ac.id
[2] mbarmawi@melsa.net.id

Abstract

Identity Cards or Kartu Tanda Penduduk (KTP) are essential for Indonesian people. KTP contains personal information, such as National Identity Number (NIK), Name, Address, Gender, etc. Since KTP has essential data and is still printed conventionally, there is a vulnerability if the KTP is lost, and the owner's data is disclosed so that if an irresponsible person finds it, the data can be used for impersonating the owner. In the previous method proposed by Haque et al., [1], the data was stored in a QR Code. However, there was no verification method to legitimize the original owner, and the system did not have a login feature. To overcome the weakness of Haque et al., method [1], the owner's NIK is encrypted using the Elliptic Curve El-Gamal (ECEG) and further signed using ECDSA by the owners before storing it in the QR Code. For obtaining the owner's data in the database, the verification process should be done after the QR Code is scanned. Using the proposed method, the probability of success for a guessing attack is 1 / (n - 1). Meanwhile, the probability of success for an impersonation attack is 1 / (q1 * q2 * l).

Keywords: QR Code, Identity Card, KTP, ECDSA, Elliptic Curve El-Gamal

Abstrak

Kartu Tanda Penduduk (KTP) merupakan hal yang sangat penting bagi masyarakat Indonesia. KTP memuat informasi pribadi, seperti Nomor Induk Kependudukan (NIK), Nama, Alamat, Jenis Kelamin, dll. Karena KTP memiliki data penting dan masih dicetak secara konvensional, maka rawan terhadap pencurian data jika KTP hilang. Jika KTP ditemukan oleh orang yang tidak bertanggung jawab, maka data dari pemilik KTP dapat digunakan orang tersebut untuk menyamar sebagai pemiliknya. Dalam metode sebelumnya yang dikemukakan oleh Haque et al., [1], data disimpan dalam Kode QR. Namun, tidak ada metode verifikasi untuk melegitimasi pemilik aslinya, dan sistem tidak memiliki fitur login. Untuk mengatasi kelemahan Haque et al., metode[1], NIK pemilik dienkripsi menggunakan Elliptic Curve El-Gamal (ECEG) dan selanjutnya ditandatangani menggunakan ECDSA oleh pemilik sebelum disimpan dalam QR Code. Untuk mendapatkan data pemilik di database, proses verifikasi harus dilakukan setelah QR Code dipindai. Dengan menggunakan metode yang diusulkan, probabilitas keberhasilan serangan tebakan adalah 1 / (n - 1). Sedangkan probabilitas keberhasilan serangan peniruan identitas adalah 1 / (*q1* * *q2* * l).

Kata Kunci: QR Code, Kartu Identitas, KTP, ECDSA, Elliptic Curve El-Gamal

I. Introduction

Kartu Tanda Penduduk (KTP) as an identity card for Indonesians is very important because it can be used for health, education, economic needs, and activities that need government involvement. Currently, the KTP is represented in the form of a conventional card containing the Identification Number (in Indonesia NIK),

Received on January, 2024. Accepted on April, 2024

Name, Place, Date of Birth, Gender, Address, Religion, Marital Status, Occupation, and Citizenship. Since the information stored on the KTP is essential, it is necessary to secure this data. Currently, KTP is still in conventional form, and this information is easy to use by someone who finds the KTP. One way to secure data on identity cards (in Indonesia is KTP) has been proposed by Haque et al. [1], Ayeleso et al. [2], and Saheed et al. [3].

In the previous research proposed by Haque et al. [1] and Saheed et al. method [3], the QR Code is used for saving the details of the owner (including the identification number) directly, such that everybody can obtain the owner's data by scanning the QR Code. However, there is no method to verify the owner's legitimacy, and the system does not have a feature to log in for the owner, such that the security of personal data is weak.

Ayeleso et al.'s method [2] used a QR Code authentication. For authentication, Ayeleso compares the encrypted identification number stored in the QR Code and the result of the encrypted identification number, where the owner enters the identification number manually. Since the identification number is also printed on the identity card, an attacker can directly succeed in the authentication process.

To overcome this problem, this study proposes to utilize a QR Code whose data has been secured using encryption and signature. Not all KTP data will be printed using the secured QR Code, and some data will be encrypted using the Elliptic Curve El- Gamal scheme [11] and stored in the QR Code[12]. The encrypted data is signed using the Elliptic Curve Digital Signature Algorithm (ECDSA) [7]. This data will be used to verify the validity of the KTP owner [8]. Using the proposed method, the system's security level against guessing and impersonation attacks increases because the probability of those attacks decreases.

II. Literature Review

This chapter discusses the previous method proposed by Haque et al. entitled the security analysis of this method, namely the Advanced QR Code Based Identity Card: A New Era for Generating Student ID Cards in Developing Countries proposed by Haque et al. [1].

A. Overview of Previous Methods

To overcome the problem, namely the disclosure of essential data on identity cards, Haque et al. proposed an Advanced QR Code Based Identity Card: A New Era for Generating Student ID Cards in Developing Countries [1]. The Haque et al. method[1] generates a QR Code which contains a link indicating the location of the data collected in a card.

With this Haque et al. method [1], identity data can be secured by applying access via the link contained in the QR Code, and this access can be obtained through permission from the admin. Identity information is stored in a database available in the administration section. Thus, the time needed to verify is less, and the data will be more difficult to manipulate. The application built by Haque et al. [1] can scan identity cards directly; so it no longer requires photos and other information stored on the identity card.

The Haque et al. method [1] has several processes to identify owners using their QR Codes. The first stage is to do the installation, then create a owner interface application and enter data related to the owner of the identity, including the photo, into the database. After that, the application can be used by opening the application's home screen, then scanning the QR Code through the application screen and accessing all data of the identity owner, including the owner's photo.

The identity owner information that needs to be provided is available in the form of students' names, class roll number, registration, number, session, name of the department, and hall name. The available data will be grouped based on basic student information and the feature, then modify the department name and hall name if necessary. Next, take a student snap from a real-time video streaming, check the printable form, edit the total amount of information, and go to the print process. Finally, search for the student's unique ID number.

The complete use case diagram of the system proposed by Haque et al. [1] is shown in Figure 1.

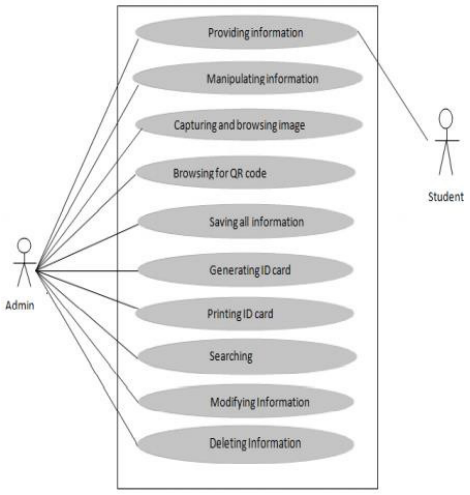

Fig. 1. Usecase Diagram of Haque et al. method

Similar to the Haque et al. method, the Saheed et al. method [3] also uses the QR Code to save the owner's details (including the identification number) directly, such that everybody can obtain the owner's data by scanning the QR Code.

Ayeleso et al.'s method [2] used a QR Code authentication. The authentication process begins by encrypting the identification number and saving it into the QR Code. The identification number is also printed on the identity card. Finally, the identification number is conducted by comparing the encrypted identification number stored in the QR Code and the result of the encrypted identification number, where the owner enters the identification number manually.

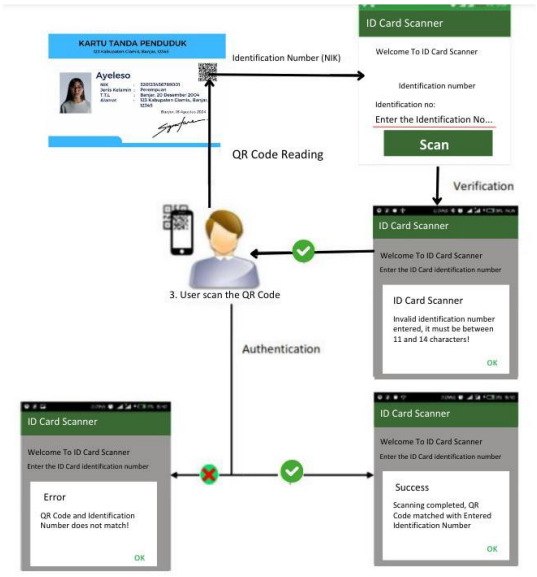

Fig. 2. Framework of Identity Card Authentication System of Ayeleso et al [2].

B. Security Analysis of Previous Methods

In the previous methods [1][2][3], the system reads (scans) the QR Code, which displays general data from the owner's identity. For Haque et al.'s method [1] and Saheed et al.'s method [3], the server is used to obtain data from the owner's identity. In this case, only the admin can access the owner's identity data through the login process.

For Ayeleso et al. method [2], authentication is conducted by comparing the encrypted identification number store into QR Code and the result of the encrypted identification number, where the owner enters the identification number manually. The weakness of Ayeleso et al. method is that the authentication can be succeeded by an attacker who has stole the identity card. This condition occurs because it has the identification number and the QR Code containing the identification number.

The weakness of Haque et al.'s [1] and Saheed et al.'s [3] method is that there is no verification of the legitimacy of the owner of the identity card by the admin, so the data representing the identity can be seen directly by the party after scanning the QR Code, and the admin permits access. Also, the system does not have a feature for the owner to log in. Thus, the security of personal data is weak.

C. Elliptic Curve Cryptography

This section discuss about the elliptic curve cryptography and its variants.

1) Elliptic Curve Cryptography: Elliptic Curve Cryptography (ECC) [4][9] has a shorter key than RSA but with the same security [5][10], so it is more efficient [10] to use because it saves storage and bandwidth. ECC uses Finite Field (Fp). A finite field is a set of numbers that contains a finite number of elements. Elliptic curve cryptography has security parameters: a, b, and p where a and b are the coefficients of the elliptic curve while p is the modulo number that limits the finite field (Fp). Also, in the ECC there are orders, which in the implementation represented by len_orde. Order is the maximum number of multipliers of points on a curve, which still lie on that curve.

In elliptic curve cryptography, two keys are generated: the public and private keys. The identity owner will choose the private key whose magnitude is less than p. For example, if the number the identity owner selects is dA, then the public key will be generated according to the key generation process discussed in section 2.C.2.

2) Key Generation: The key generation process aims to generate the public key from the private key. The inputs in this process are the private key (dA), p, and base point. The output generated is the public key (pub_key). Furthermore, the public key is calculated by multiplying dA by the base point, as shown in Equation 1.

$$pub\_key = dA * base\ point \tag{1}$$

where dA = private key

3) Encryption Using Elliptic Curve El-Gamal: The process of encrypting message using Elliptic Curve El-Gamal aims to encrypt the message. The message is represented in digit. The input for this process is message, pub_key, and base point data. Meanwhile, the output is each encrypted message character. Thus, the overall output is a collection of points (x_i, y_i), where i = 1 – the number of digits of the message. The details of the stages carried out are as follows:

i. Changing *message* into points based on the digit-point mapping. The result of converting all message is represented as a collection of $mPoint_i$.
ii. Choose k values randomly in the range 1 – (n – 1) where n is the order of the *base point*.
iii. Calculate $x_i = k * base\ point$.

iv. Calculating $y_I = mPoint_I + (k * pub\_key)$, if the result is infinity, then return to step ii.

4) Signing Encrypted Message using ECDSA: The signing encrypted message process using ECDSA aims to sign encrypted message. Input in this process is encrypted message, *p*, base point, *dA*, and len_orde. Meanwhile, the output is an encrypted message signed (*r*, *s*). The stages of the process are as follows:

i. Choose *k* values randomly in the range 1 – (*n* – 1) by private key owner.
ii. Calculating *rPoint* = *k* * *base point*, if the result is infinity then return to step i.
iii. Calculating the value of *r* using Equation 2.

$$r = (ordinate\ x \text{ from } rPoint) \bmod len\_orde \tag{2}$$

where *ordinate x* is the *x* value of *rPoint*.

iv. Calculating the value of *t* using Equation 3.

$$t = k^{-1} \bmod len\_orde \tag{3}$$

v. Calculating the value of *s* using Equation 4. If the value of *s* = 0 then it will return to step i.

$$s = t\ (message + r * dA) \bmod len\_orde \tag{4}$$

vi. The signature for encrypted message is (*r*, *s*).

5) Decrypt Encrypted Message by Private Key Owner: The decrypt encrypted message by Private Key Owner process aims to decrypt the encrypted message sent by Disdukcapil to the identification number. The inputs to this process are encrypted message (set points (w_i, z_i)), owner's private key, and *p*. While the output is the message which is the result of converting the points from the $mPoint_i$ decryption results into digits. Suppose *dA* is the owner's private key, *dA* is modulated with n. Then each pair of set points (w_i, z_i) will be calculated by Equation 5. Then $mPoint_i$ will be converted into digits (message) using the method discussed in section 3.A.3.

$$mPointi = zi - (dA * wi) \tag{5}$$

where i = index decryption result of encrypted message.

6) Verify the Owner's Signature: The Verify Owner Signature process aims to verify this is the original owner. The input entered in this verification is the signature (*r*, *s*) obtained from the QR Code, message *mPoint*, base point, *p*, *dA*, and signature (*r'*, *s'*) from the identity owner. Meanwhile, the output is True / False. The stages of the process are carried out as follows:

i. Calculating the value of *u1* using Equation 6.

$$u1 = (mPoint * s^{-1}) \bmod p \tag{6}$$

ii. Calculating the value of *u2* using Equation 7.

$$u2 = (r * s^{-1}) \bmod p \tag{7}$$

iii. Calculating the value of *O* using Equation 8.

$$O = u1 * base\ point + u2 * pub\_key \tag{8}$$

iv. Calculating the value of *V* using Equation 9.

$$V = O.x \bmod p \tag{9}$$

v. Comparing the value of *V* with the value of *r'*. If they are equal, then the verification is successful, and the data appears to the identity owner. Furthermore, the owner can use the features in this system. If they are not similar, then the verification fails and is suspended.

III. Research Method

To overcome the weaknesses of the method proposed by Haque et al. [1], Ayeleso et al. [2], and Saheed et al. [3], a system was created to verify the identity owner's validity before the person concerned was given access rights to the identity owner's data.

In the proposed method, there are four stages of the process, namely, the preprocessing, the registration, the login, and the verification stages. The overview of the proposed method is shown in Figure 3. The details of the preprocessing stage are discussed in Section A, the registration stage is discussed in Section B, and login and verification are discussed in Section C. The preprocessing stage is conducted to generate the elliptic curve, including its parameters. The registration stage generates the QR Code, and the login stage is conducted to generate the signed encrypted NIK. Finally, the verification stage is conducted to verify the data owner by validating the signed encrypted NIK.

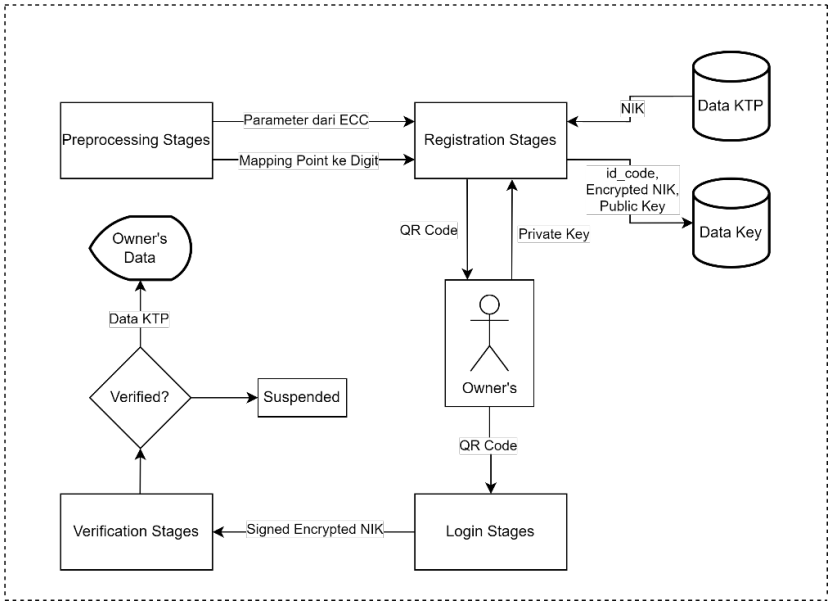

Fig. 3. Overview of the Proposed Method.

In this method, data from the identity owner in the form of a NIK (Identification Number) will be encrypted using Elliptic Curve El-Gamal [11] by the identity owner at the registration stage. The encrypted data will be digitally signed using the Elliptic Curve Digital Signature Algorithm (ECDSA) [7] by the identity owner at the registration stage and the login & verification stage. The data that has been signed is stored in the QR Code [12], and data from the identity owner will be stored in the database owned by Disdukcapil. Disdukcapil is Direktorat Jenderal Kependudukan dan Pencatatan Sipil, which carries out the policy's formulation and implementation, especially for civil registration, by the provisions of statutory regulations. Meanwhile, the public key of the owner is stored in the key database owned by Disdukcapil.

A. Preprocessing Stage

The preprocessing stage discusses the setup parameter system, creating the mapping, and converting from point to digit.

1) Setup Parameter System: The system parameter setup process aims to find and select a base point value. Enter security parameters (a, b, and p for elliptic curve $y^2 = x^3 + ax + b \bmod p$) in this process. Meanwhile, the output that will be generated is in the form of a base point. The stages of the process consist of the following steps:

a. The system checks the security parameters of this system by evaluating whether the parameters form singular values or not. The parameter is a singular value if $4a^3 + 27b^2 \neq 0$ [4][9].
b. If the security parameter does not form a singular value, a set of points that can be used as a base point should be observed.
c. Base point selection is carried out from a set of points by doing step a.
d. Steps b and c are carried out until singular values are obtained from the security parameters.

2) Create Mapping: The process of creating mapping aims to map between the digits of the NIK (Nomor Induk Kependudukan), which have a digit range of 0 – 9 to be the points inside the elliptic curve. The points that have been generated are entered into the *pointHashing* list. A *pointHashing* list is a table that maps digits into points on an elliptic curve. The point in the *pointHashing* list results from multiplying the digits and the base point. At the beginning of the process, enter the input as an empty base point and *pointHashing* list. While the output generated from this process is a filled *pointHashing* list. The stages of the process are carried out in the following steps.

a. The first stage is mapping the digits to the points listed in the *pointHashing* list. Each point is calculated by multiplying the digit by the base point. The formulation used is shown in Equation 10.

$$newPoint_i = (i + 1) * basepoint \tag{10}$$

where i = digit 0 - 9

b. Insert $newPoint_i$ into the *pointHashing* list with indexes representing digits.
c. Processes a and b are carried out for the digit range 0 – 9.

3) Converting from Point to Digit: Converting from point to digit aims to transform from point to digit. In this process, the input entered is the point that will be converted to digits and the *pointHashing* list. The output is in the form of digits. In this process, point tracing is carried out in the *pointHashing* list based on the point to be changed to obtain results in the form of digits.

B. Registration Stage

This system has two databases, namely a database for identity owners and a database for keys, both owned by Disdukcapil. The identity owner database contains NIK, Name, etc. Meanwhile, the critical database contains id_code, encrypted NIK, and public key.

The registration stage is to register identity owner data into the Disdukcapil database. In this process, key generation is carried out by the identity owner, then the identity owner's public key is sent to Disdukcapil. Disdukcapil will store the public key in the key database. Then, disdukcapil will encrypt the identification number (NIK) using the identity owner's public key based on Elliptic Curve El-Gamal. Furthermore, the identity owner's data is stored in the identity owner's database owned by Disdukcapil. At the same time, the encrypted identity owner data is sent to the identity owner. Then the identity owner signs the NIK, which has been encrypted using ECDSA. Signed data is stored in a QR Code and concatenated with id_code. id_code consists of six A-Z characters, representing the identity owner code. An overview of this process is shown in Figure 4.

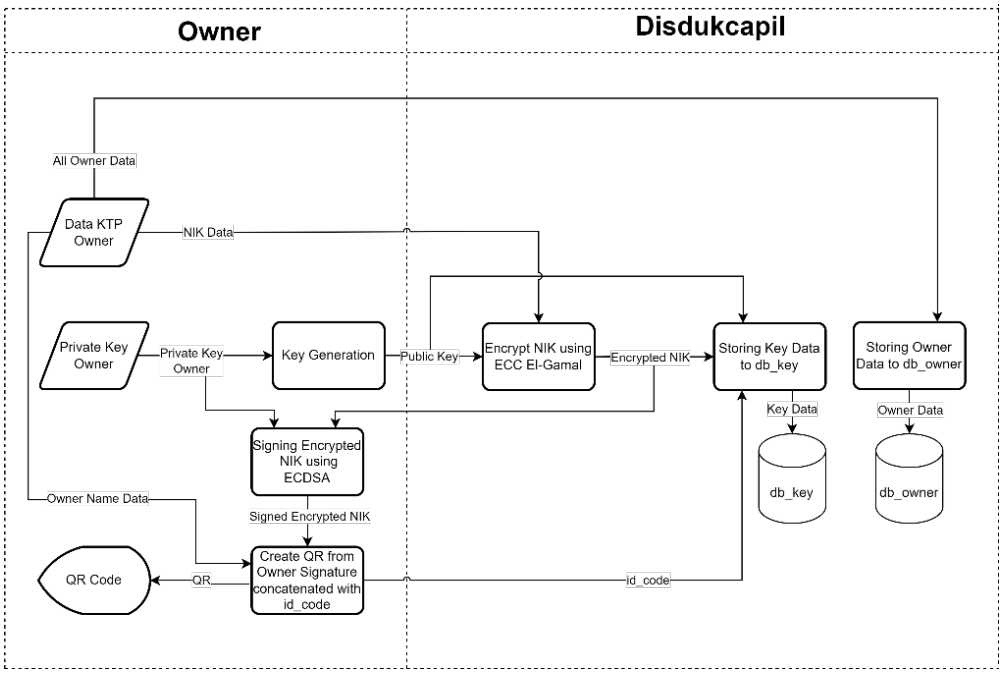

Fig. 4. Registration Stage

1) Key Generation: The key generation process aims to generate the public key from the owner's private key. The inputs in this process are the owner's private key (*dA*), *p*, and *base point*. The output generated is the owner's public key (*pub_key*). Furthermore, the owner's public key is calculated by multiplying *dA* by the base point, as shown in Equation 2.

2) Encrypting NIK using ECC El-Gamal: The process of encrypting NIK using ECC El-Gamal aims to encrypt NIK. The input for this process is NIK, *pub_key*, and *base point* data. Meanwhile, the output is each encrypted NIK character. Thus the overall output is a collection of points (x_i, y_i), where i = 1 – the number of digits of the NIK. The details of the stages carried out are as follows:

i. Changing NIK digits into points in the manner discussed in section 3.A.2. The result of converting all digits is represented as a collection of $mPoint_i$.
ii. Choose k values randomly in the range 1 – (n – 1) where n is the order of the base point.
iii. Calculate $x_i = k * base\ point$.
iv. Calculating $y_i = mPoint_i + (k * pub\_key)$, if the result is infinity, then return to step ii. The encrypted NIK is (x_i, y_i).

3) Storing Owner Data to Database db_owner: Storing Owner Data in the Database db_owner aims to store identity owner data in the db_owner database owned by Disdukcapil. The input in this process is the identity owner's data. Meanwhile, the output is data verified and stored in the db_owner database. In this process, the first thing to do is check the similarity between the NIK entered by the identity owner and the NIK stored in the db_owner database so that it can be guaranteed that there are no duplicate NIK data. If the NIK data is not duplicated, then the data will be stored in the db_owner database.

4) Signing Encrypted NIK using ECDSA: The Encrypted NIK Signing process using ECDSA aims to sign encrypted NIK. Input in this process is encrypted NIK, *p*, *base point*, and *len_orde*. Meanwhile, the output is an encrypted NIK signed (*r*, *s*). The stages of the process are as follows:

i. Calculating the hash value of encrypted NIK (x_i, y_i) using SHA-3 [6], and if the result of encrypted NIK hashed is 0, then the last of encrypted NIK is added by "1".
ii. Calculating the hash value generated from the *i*-th stage is modulo *p*, where *p* is a large prime number. The calculation can be represented by Equation 11.

$$h = Hash(xi, yi) \bmod p \tag{11}$$

iii. Choose *k* values randomly in the range 1 – (*n* – 1) by private key owner.
iv. Calculating *rPoint* = *k* * *base point*, if the result is infinity then return to step iii.
v. Calculating the value of *r* using Equation 2.
vi. Calculating the value of *t* using Equation 3.
vii. Calculating the value of *s* using Equation 4. If the value of *s* = 0 then it will return to step iii.
viii. The signature of encrypted NIK is (*r*, *s*).

5) Create QR Code from the Signed Encrypted NIK Concatenated with id_code: The Create QR Code from the Signed Encrypted NIK concatenated with *id_code* process aims to generate a QR Code which contains an id_code concatenate with the NIK, which is encrypted with the public key of the identity owner and signed by the identity owner. The input in this process is a signed encrypted NIK (*r*, *s*) and the identity owner's name. The output of this process is a QR Code. The process steps are to create *id_code*. Inside the *id_code* are six random characters (A – Z) obtained from the name of the identity owner. Next, build the mQR shown in Equation 12. Then, generate the QR Code filled with mQR.

$$mQR = id\_code \,||\, \# \,||\, r \,||\, \# \,||\, s \tag{12}$$

where *id_code* is six random characters (A-Z) from the owner's identity and (*r*,*s*) is an encrypted NIK that has been signed. *r* is obtained from Equation 2 and *s* from Equation 4.

6) Storing Key Data to Database db_key: Storing Key Data in Database db_key aims to store key data in Disdukcapil's db_key database. Inputs in this process are id_code, pub_key, and encrypted NIK. Meanwhile, the output is data verified and stored in the db_key database. In this process, a comparison is made between the id_code, which was constructed in the manner discussed in section 3.B.5, with the data stored in the db_key database. If id_code is not duplicated, the data will be stored in the db_key database such that it can be guaranteed that there are no duplicate id_code.

C. Login and Verification Stage

A QR Code scanning process will be carried out at the login and verification stage, and an *encrypted NIK* and *public key* from *db_key* will be retrieved. Furthermore, the encrypted NIK is then sent to the identity owner and decrypted using the identity owner's private key. After that, a comparison was made between the decryption results from the identity owner and the NIK data search in the identity owner's database. If the NIK data is available, the encrypted NIK obtained from *db_key* is signed using the identity owner's private key. This signing method is done using ECDSA. The resulting signature is represented by (*r'*, *s'*). After that, a comparison was made between the signature (*r'*, *s'*) and the encrypted NIK that had been signed and obtained from the QR Code scan, namely (*r*, *s*). If the results are the same, verification will be carried out on the signature of the identity owner. An overview of the process is shown in Figure 5.

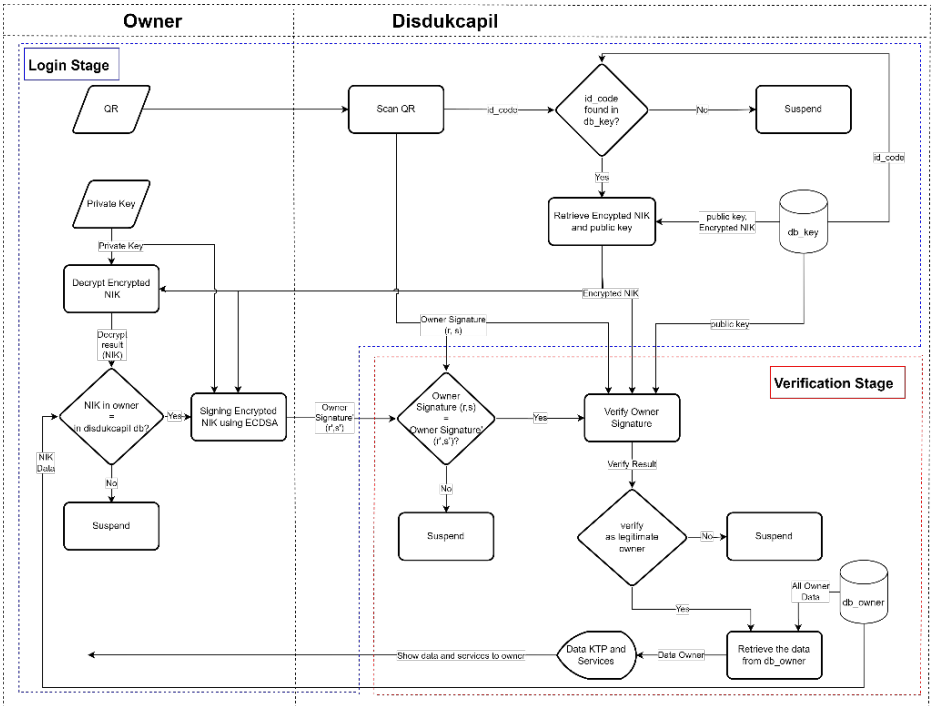

Fig. 5. Login and Verification Stage

1) Scan QR: The QR Scan process aims to read the QR Code so that the contents of the QR Code are read. The input in this process is a QR Code. Meanwhile, the output is id_code and signed encrypted NIK (r, s). The steps in this process are the first to extract the mQR, which contains the id_code concatenated with an encrypted NIK signed by the owner of the identity.

2) Retrieve Encrypted NIK and Public Key from db_key: Retrieving encrypted NIK and Public Key Owner from db_key aims to retrieve encrypted NIK data and public keys from db_key. Input in this process is id_code. Meanwhile, this process outputs an encrypted NIK and a public key. The stage of the process is to compare the id_code obtained in the manner discussed in section 3.C.1 with the id_code in the db_key. If the data is available, then the encrypted NIK and public key are taken from db_key. However, if the data is not available, then it is suspended.

3) Decrypt Encrypted NIK by Private Key Owner: The decrypt encrypted NIK by Private Key Owner process aims to decrypt the encrypted NIK sent by Disdukcapil to the identity owner. The inputs to this process are encrypted NIK (set points (w_i, z_i)), owner's private key, and p. While the output is the NIK which is the result of converting the points from the $mPoint_i$ decryption results into digits. Suppose dA is the owner's private key, dA is modulated with n. Then each pair of set points (w_i, z_i) will be calculated by Equation 5. Then $mPoint_i$ will be converted into digits (NIK) using the method discussed in section 3.A.3.

4) Signing ECDSA Encrypted NIK by Owner: The ECDSA Encrypted NIK by Owner Signing process aims to sign the encrypted NIK sent by Disdukcapil. The input in this process is an encrypted NIK obtained from Disdukcapil. While the output of this process is a signed encrypted NIK (r', s'). The stages of the process are carried out to sign the signature in the manner discussed in section 3.B.4.

5) Verify the Owner's Signature: The Verify Owner Signature process aims to verify this is the original owner. The input entered in this verification is the signature (r, s) obtained from the QR Code, encrypted NIK, base

point, p, private key, and signature (r', s') from the identity owner. Meanwhile, the output is True / False. The stages of the process are carried out as follows:

i. Verify the signature (r, s) in the following way:
   a. Calculating the hash value of encrypted NIK with SHA-3 [6], and if the result of encrypted NIK hashed is 0, then the last of encrypted NIK is added by "1".
   b. Calculating the hash value generated from stage a modulo p, where p is a large prime number. The calculation can be represented by Equation 11.
   c. Calculating the value of $u1$ using Equation 6.
   d. Calculating the value of $u2$ using Equation 7.
   e. Calculating the value of O using Equation 8.
   f. Calculating the value of V using Equation 9.
   g. Comparing the value of V with the value of r. If they are the same, then the verification is successful, and the data appears to the identity owner. Furthermore, the owner can use the features in this system. If they are not similar, then the verification fails and is suspended.
ii. Comparing (r, s) and (r', s'). If they are equal then do ii-stage. Otherwise, the process is suspended.

IV. Results And Discussion

This chapter discusses the performance of the proposed method. The proposed method is evaluated based on two aspects. The first is calculating the total operation the registration, login, and verification stages. The second is security analysis. These aspects will be discussed in the following subchapters

A. Total Operation Analysis

This section analyzes the total operation (doubling and adding points) used in the registration, login, and verification stages. There are three subprocesses in the registration process: key generation, encryption, and signing process. The login and verification process has three subprocesses: decryption, signing, and verification.

The experiment is conducted using 12 NIK data taken with details, namely 4 NIK data encrypted using a public key that was constructed using a 4-digit private key, 4 NIK data encrypted using a public key that was constructed using a 5-digit private key, and 4 NIK data encrypted using a public key that was constructed using a 6-digit private key. From the 12 NIK data, an analysis was carried out for the total operation for each process.

The total operation chart for the key generation process is shown in Figure 6. As shown in Figure 6, we can see that the higher the private key length, the higher the total operation is needed. However, there is one outlier of the total operation where the length of the private key is 157218. This condition occurred because the number of point doubling is increased such that the number of point additions decreases. Thus, the total operation is decreased.

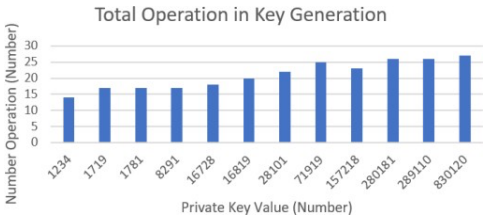

Fig. 6. Total Operation in Key Generation

The total operation for the encryption process is shown in Figure 7. As shown in Figure 7, we can see that the number of operations used to encrypt the message is not directly related to the private key length because for

the encryption, the random number k is used, and this number will influence the number of operations used in the encryption process.

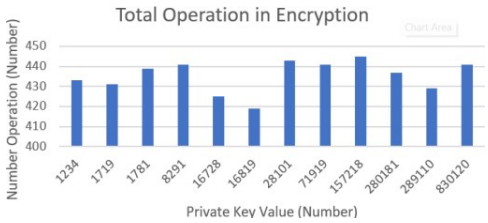

Fig. 7. Total Operation in Encryption

The total operation for the signing process is shown in Figure 8. As shown in Figure 8, we can see that the number of operations used for signing the message is not directly related to the private key length because for the signing process, the random number k is used, and this number will influence the number of operations used in the signing process.

Fig. 8. Total Operation in Signing

The total operation for the decryption process is shown in Figure 9. As shown in Figure 9, we can see that the higher the private key length, the higher the total operation is needed. However, there is one outlier of the total operation where the length of the private key is 157218. This condition occurred because the number of point doubling is increased such that the number of point additions decreases. Thus, the total operation is decreased.

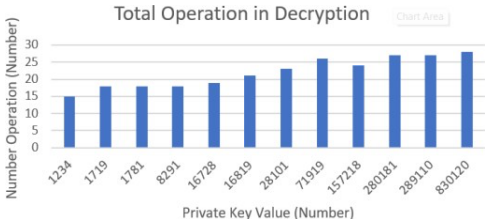

Fig. 9. Total Operation in Decryption

The total operation for the signature verification process is shown in Figure 10. As shown in Figure 10, the number of operations used for the verification process is random and does not depend on k or private key length.

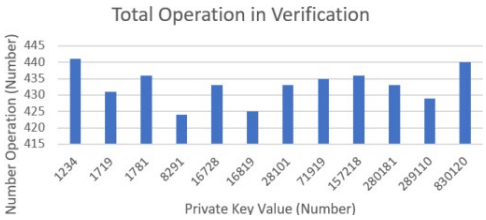

Fig. 10. Total Operation in Verification

Table 1 is used for analyzing the range of the total operation in each process. Table 2 is used for analyzing the average time in each digit in each process.

TABLE I
Process And Total Operation Required For Each Process

Process	Total Operation Required
Key Generation	Shown in Figure 6 in the range 14 - 27
Encryption	Shown in Figure 7 in the range 419 - 445
Signing	Shown in Figure 8 in the range 208 - 221
Decryption	Shown in Figure 9 in the range 15 - 28
Verification	Shown in Figure 10 in the range 424 - 441

TABLE II
Process And Time Required For Each Process

Process	Time Required
Key Generation	The average Time required for 4 Digits is 7106172,8 nanoseconds, for 5 Digits is 7338196 nanoseconds, and for 6 Digits is 13776218,25 nanoseconds.
Encryption	The average Time required for 4 Digits is 13594955771,3 nanoseconds, for 5 Digits is 14281221176,3 nanoseconds, and for 6 Digits is 14441609336,5 nanoseconds
Signing	The average Time required for 4 Digits is 8115066,75 nanoseconds, for 5 Digits is 8334607,5 nanoseconds, and for 6 Digits is 8734152,25 nanoseconds.
Decryption	The average Time required for 4 Digits is 201977450,5 nanoseconds, for 5 Digits is 346820654,8 nanoseconds, and for 6 Digits is 551619612,3 nanoseconds.
Verification	The average Time required for 4 Digits is 456618532 nanoseconds, for 5 Digits is 574533387,8 nanoseconds, and for 6 Digits is 865026599,3 nanoseconds.

For calculating the time average, addition and doubling time are necessary. In this case, the addition time ranges between 3679949 and 6545587 nanoseconds, and the doubling time ranges between 288821 and 330396 nanoseconds.

Table 1 and Figure 6 show that the key generation process takes between 14 and 27 total operations. From Table 2, the key generation process for 4-digits private key takes time about 7106172,8 nanoseconds, in 5-digits is 7338196 nanoseconds, and in 6-digits is 13776218,25 nanoseconds. Meanwhile, the highest time duration is in the key generation process for a 6-digit private key, while the lowest time is in the key generation process for a 4-digit private key.

Table 1 and Figure 7 show that the encryption process takes between 413 and 449 total operation. From Table 2, the encryption process for a 4-digit private key takes time about 13594955771,3 nanoseconds, in 5-digits is

14281221176,3 nanoseconds, and in 6-digits is 14441609336,5 nanoseconds. Meanwhile, the highest time duration is in the encryption process for a 6-digit private key, while the lowest time is in the encryption process for a 4-digit private key.

Table 1 and Figure 8 show that the signing process takes between 208 and 221 total operation. From Table 2, the signing process for a 4-digit private key takes time about 8115066,8 nanoseconds, in 5-digits is 8334607,5 nanoseconds, and in 6-digits is 8734152,25 nanoseconds. Meanwhile, the highest time duration is in the signing process for a 6-digit private key, while the lowest time is in the signing process for a 4-digit private key.

Table 1 and Figure 9 show that the decryption process takes between 15 and 28 total operation. From Table 2, the decryption process for a 4-digit private key takes time about 201977450,5 nanoseconds, in 5-digits is 346820654,8 nanoseconds, and in 6- digits is 551619612,3 nanoseconds. Meanwhile, the highest time duration is in the decryption process for a 6-digit private key, while the lowest time is in the decryption process for a 4-digit private key.

Table 1 and Figure 10 show that the signature verification process takes between 424 and 441 total operation. From Table 2, the signature verification process for a 4- digit private key takes time about 456618532 nanoseconds, in 5-digits is 574533387,8 nanoseconds, and in 6-digits is 865026599,3 nanoseconds. Meanwhile, the highest time duration is in the signature verification process for a 6-digit private key, while the lowest time is in the signature verification process for a 4-digit private key.

B. Security Analysis

For analyzing the security of the proposed method, two attacks are applied: guessing and impersonation attacks.

1) Guessing Attack

The guessing attack scenario is applied in the proposed method to guess the identity owner's private key after scanning the QR Code and verifying its validity. Thus, the security depends on the attacker's ability to find the correct private key from all possible private keys. The probability of success in guessing the private key is shown in Equation 14. The probability of guessing from the previous methods [1][2][3] is 1. This condition occurred because the previous method did not use a key such that the desired data could be seen immediately.

In this case, the possibility of guessing the private key is shown in Equation 12.

$$P(guessing\ private\ key) = \frac{1}{n-1} \tag{12}$$

where n is the order of the base point.

Thus, our proposed method introduces a significant improvement in security over the previous methods [1][2][3], making it a safer alternative.

2) Impersonation Attack

The impersonation attack scenario is applied in the proposed method by imitating the owner's identity in the QR Code. The prevention of this attack is carried out by verifying the validity of the signature contained in the QR Code which is represented by (r, s). Thus, the security depends on the signature contained in the QR Code. It means the security strength depends on the value of k, the private key, and the number of base points. The probability of success in this impersonation is shown in Equation13. The probability of impersonation from the previous methods [1][2][3] is 1. This occurs because the previous method did not verify the validity of identifying data from the owner of the identity.

$$P(impersonation) = \frac{1}{q1 * q2 * l} \quad (13)$$

where *q1* is the value of *k*, *q2* is the value of the private key, and *l* is the number of base points.

Thus, the security level of the proposed method against impersonation attacks is higher than previous methods [1][2][3].

V. Conclusion

This research aims to verify the legitimacy of the identity owner. Haque et al. [1] and Saheed et al. [3] propose methods to achieve this objective. In the method proposed by Haque et al. [1] and Saheed et al.[3], the owner's data cannot be validated because there is no authentication after scanning the QR Code. So, it is still vulnerable for cheaters to attack the scheme. In the Ayeleso et al. method [2], authenticity exists. However, the data used for the authentication is written on the card such that the attacker who stole the Identification Number can succeed in the authentication process.

To overcome the problem of the previous methods [1][2][3], the identity data has to be stored in a QR Code. However, before storing the data in a QR Code, it has to be encrypted and signed by the owner.

The number of operations needed for the key generation and decryption process depends on the private key length. Meanwhile, the number of operations used for encryptions and signing processes is not directly dependent on the private key length but on the k value. Based on the results of the research that has been done, it is found that the verification process for KTP owners requires time for the key generation in the range of 7106172,8 nanoseconds – 13776218,25 nanoseconds, for the encryption in the range of 13594955771,3 nanoseconds - 14441609336,5, for the decryption in the range of 201977450,5 nanoseconds - 551619612,3 nanoseconds, for the signingin the range of 8115066,75 nanoseconds - 8734152,25 nanoseconds, and the verification in the range of 456618532 nanoseconds - 865026599,3 nanoseconds. Meanwhile, the probability of success in the guessing attack is 1 / (*n* - 1), and the probability of success in carrying out an impersonation attack is 1 / (*q1* * *q2* * *l*).

This research proposes an identity data encryption method (NIK) using the elliptic curve El-Gamal encryption method, which is then signed using the elliptic curve digital signature algorithm (ECDSA) method. The results from ECDSA will be included in the QR Code. When the owner has scanned the QR Code and if it is proven that the authenticity of the data can be authenticated, the owner data will be shown. Otherwise, it will be rejected.

This method is proven more secure than previous methods because the probability of private key guessing and impersonation attacks using the proposed method is less than the probability of private key guessing and impersonation attacks using Haque et al. method.

The proposed method requires an integrated database so that all Indonesian people, wherever they are, can access their data using a QR code.

This research can be improved by adding data to be encrypted, not limited to NIK, and maximizing the use of the codewords in the QR Code. In addition, the message used as the private key can be letters or symbols.

Acknowledgment

I would like to express my deepest gratitude to my thesis supervisor, Prof. Ari Moesriami Barmawi, Ir., M.Sc., Ph.D., for their unwavering support, guidance, and invaluable feedback throughout the entire process of completing this thesis. Their expertise and encouragement played a pivotal role in shaping the direction of my research.

My sincere appreciation extends to my family and friends for their patience, understanding, and continuous encouragement during the challenges of thesis writing.

References

[1] S. Haque and R. Dybowski, “Advanced QR Code Based Identity Card: A New Era for Generating Student ID Card in Developing Countries,” in *IEEE SIMS2014*, Sheffield, UK, 2014, pp.76–82.

[2] E. C. Ayeleso, A. Adekiigbe, N. C. Onyeka, and M. O. Oladele. (2017). Identity Card Authentication System Using QR Code and Smartphone. *International Journal of Science, Engineering & Environmental Technology*.2(9). 61-68.Available:https://www.repcomseet.org/journal/AYELESO.pdf

[3] Y. K. Saheed, T. T Salau-Ibrahim, and A. F. Kadri. (2016, December). Student Identity Card Based On Advanced Quick Response Code Technology. *Computing, Information Systems, Development Informatics & Allied Research Journal*. 7(4). 149-158. Available: https://www.researchgate.net/publication/328430187_Student_Identity_Card_Based_On_Advanced_Quick_Response_Code_Technology

[4] N. Koblitz. (1987, January). Elliptic Curve Cryptosystems. *Mathematics of Computation*. *Volume 48*, 203-209. Available: https://www.ams.org/mcom/1987-48-177/S0025-5718-1987-0866109-5/S0025- 5718-1987-0866109-5.pdf

[5] D. Mahto and D. K. Yadav. (2017). RSA and ECC: A Comparative Analysis. *International Journal of Applied Engineering Research*. *Volume 12*, 9053-9061. Available: https://www.ripublication.com/ijaer17/ijaerv12n19_140.pdf

[6] S. Ghoshal, P. Bandyopadhyay, S. Roy, and M. Baneree,"A Journey from MD5 to SHA-3," in *Trends in Communication, Cloud, and Big Data*, 2020, pp. 107–112.

[7] N. J. G. Saho and E. C. Ezin, "Securing Document by Digital Signature through RSA and Elliptic Curve Cryptosystems," *2019 International Conference on Smart Applications, Communications and Networking (SmartNets)*, Sharm El Sheikh, Egypt, 2019, pp. 1-6.

[8] T. Wellem, Y. Nataliani, and A. Iriani. (2022, September). Academic Document Authentication using Elliptic Curve Digital Signature Algorithm and QR Code. *JOIV : International Journal on Informatics Visualization*, *6(3)*, 667-675. Available: https://www.joiv.org/index.php/joiv/article/view/872

[9] J. Holden, "Elliptic Curve Cryptography,” in *The Mathematics of Secrets: Cryptography from Caesar Ciphers to Digital Encryption*, 1st ed. New Jersey , US: Princeton University Press, 2017.

[10] F. Mallouli, A. Hellal, N. S. Saeed, and F. A. Alzahrani, "A Survey on Cryptography: Comparative Study between RSA vs ECC Algorithms, and RSA vs El-Gamal Algorithms," *2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/ 2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom)*, Paris, France, 2019, pp. 173-176.

[11] L. C. Washington, "Elliptic Curve Cryptography," in *Elliptic curves: Number Theory and Cryptography*, 2nd ed. Florida, US: Chapman and Hall/CRC, 2008.

[12] International Organization for Standardization & International Electrotechnical Commission, *ISO/IEC 18004:2015 : Information technology - Automatic identification and data capture methods - QR Code bar code symbology specification*, 3rd ed. Geneva, Switzerland: International Organization for Standardization & International Electrotechnical Commission, 2015.