Algebraic Operations-Based Secret-Key Design for Encryption Algorithm (ASKEA) for Energy Informatics and Smart Internet of Things (IoT) Applications

Abbas M. Al-Ghaili[1,2](✉), Hairoladenan Kasim[2], Ridha Omar[2],
Zainuddin Hassan[2], Naif M. Al-Hada[3](✉), and Jihua Wang[3]

[1] Institute of Informatics and Computing in Energy (IICE), Universiti Tenaga Nasional (UNITEN), 43000 Kajang, Selangor, Malaysia
[2] College of Computing and Informatics (CCI), UNITEN, 43000 Kajang, Selangor, Malaysia
abbas@uniten.edu.my
[3] Shandong Key Laboratory of Biophysics, Institute of Biophysics, Dezhou University, Dezhou 253023, China
alhada@dzu.edu.cn

Abstract. This paper designs a secret key that has the ability to behave in a different way based on plaintext input. This paper proposes two schemes of secret key length which are: 2^i and $2^i + 2^{i-1}$. The aim is to increase unpredictability and privacy of plaintext. A series of algebraic operations has been used to create a specific coefficient by which the relation between plaintext size and key length is adjustable. This design of secret key increases privacy of sensitive data used with energy informatics and smart Energy Internet-of-Things (EIoT) applications. The proposed design has been evaluated in terms of time complexity computation time to generate one secret key and encrypt plaintext. Additionally, results show that the proposed Algebraic Operations-based Secret-Key Design for Encryption Algorithm (ASKEA) has less computation time and complexity than other competitive research works. Privacy and unpredictability of plaintext and secret key could be preserved and achieved.

Keywords: Security scheme · Secret key · QR-code · Energy informatics · Energy Internet-of-Things

1 Introduction

Many Internet-of-Things (IoT) applications have lately been using the technique of Quick Response Code (QR-Code) to allow many smart services and tasks be done in a very short period of time [1–3]. The number of researches [4–8] focusing on the use of QR-Code is gradually increasing [9, 10]. Such researches produce smart services to the user whereas private data are used. Thus, there is a continuous need to protect data from being used by an unauthorized party.

Usually, Many IoT-based applications have become widely used to do many smart services and tasks which require a sufficient and precise process in a short period of time.

H. Badioze Zaman et al. (Eds.): IVIC 2021, LNCS 13051, pp. 587–599, 2021.
https://doi.org/10.1007/978-3-030-90235-3_51

Thus, a lot of information, caused by the huge portion of data being stored, requires fast processing.

The QR-Code is a very effective tool and has been used by many applications to with different purposes. Usually, QR tags store data in an encrypted form where scanners are used to extract data. Then, depending on the system or application the QR-Code is used for, a verification procedure follows the extraction step to make sure those extracted values are identical to original values before being encrypted [11].

Hence, various factors are taken into account when such a smart application is designed e.g., privacy, authentication, availability, data integrity, data recovery, responsiveness, reliability … etc. Depending on the kind of the smart application being used for, certain security factors rise up to make that application as perfect as possible, more than other factors do. Smart home applications, for example, require privacy, safety, and responsiveness …etc. Embedded systems such as health systems require more responsiveness and reliability. Sensor based collection systems reliability and maintainability. Hence, there is a need to consider all above mentioned security factors using a strong encryption scheme to include as many factors as it could.

Proposed works in [12, 13] have designed two different methods to generate a QR-Code. In [12], a graphical design has been used to make sure that documents being verified are secure whereas the document authentication factor has been considered. The design in [13] has considered both privacy and usability factors by using array of patterns that includes no much details. Similarly, another proposed QR-Code design [14] to protect private data, a print-and-scan (P&S) operation has been addressed whereas the data privacy factor has been investigated. Additionally, QR-Code designs have been used with root applications. In order to measure distances accurately and reliably, landmark images and image recognition processes have been used to head the robot to estimate the accurate location detection [15].

On the other hand, smart and IoT applications have numerous uses reviewed in literature. Some researches, for example, have proposed a security purposed method for smart home [16], monitoring purposed application [17], biometrics-based home access system [18], embedded physical evaluation method [19] … etc. These IoT applications require, for example, reliable responsiveness, security, and authentication. IoT technology relies on some other essential tools to successfully and accurately build such a system. One of these tools is the encryption scheme by which the smart system is robust. The verification procedure is however another important tool for a robust smart application. Usually, the encryption scheme needs to be followed by a strong verification procedure. Therefore, any the encryption scheme is so essential for smart and IoT applications because it affects the whole smart system including the verification procedure as well.

In general, proposed methods have utilized the security and privacy of contents however the computation time and code complexity are still of importance to be considered by many IoT applications.

This paper focuses on designing a different way of encryption for both plaintext and secret key. It proposes an Algebraic Operations-based Secret Key Design for Encryption Algorithm (ASKEA) focuses on the authentication and integrity. ASKEA is used to increase privacy of several information-centric applications such as energy informatics, IoT, and smart home applications. In this work, QR-Code has been acted as a landmark image to ease the recognition process considering both reliability and responsiveness factors to attain integrity and availability. To protect data stored in a QR-Code which is used with IoT based smart applications, this paper proposes an unpredictable secret key design by which the QR-Code tag is encrypted.

ASKEA has used a strong encryption procedure in order to increase the time needed to decrypt the QR-Code information in case the secret key has been deduced. ASKEA however has focused to use a 1-session secret key policy with the most of encryption steps so that encrypted data will not be decrypted; this is to increase the privacy and secrecy of the user's information. In order to attain the availability, the encryption policy generates a random unpredictable key. Furthermore, the proposed design has considered a very long encryption key to reduce the vulnerability of being deduced; it takes very long time to attack and crack the key. So that data integrity could be achieved.

This paper is organized as follows: Sect. 2 is dedicated to explain the schematic representation for ASKEA. Section 3 is dedicated for Results and Discussion. Conclusion has been drawn in Sect. 4.

2 The Proposed ASKEA Schematic Representation

The proposed ASKEA is relevant to user information encryption. Its encryption scheme deals with the way the collected information is being protected. ASKEA is proposed to protect information from being modified in an unauthorized manner. This paper aims to design the secret key based on the ASKEA plaintext. A generalized schematic representation is shown in Fig. 1 to show ASKEA main steps and its relationship with such a verification tool.

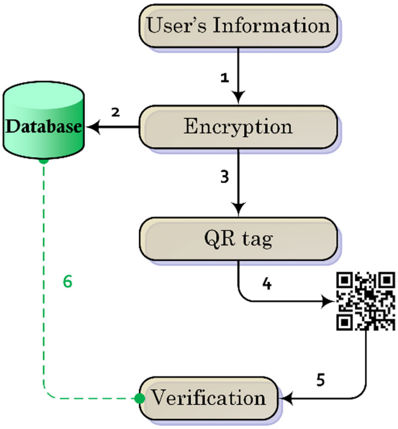

Fig. 1. A generalized ASKEA schematic representation

In Fig. 1, information will be encrypted and stored. Then, the QR-Code is generated based on the encrypted values. Then, the user interface starts. Once, an access is needed by the user, the ASKEA immediately is called to enable the verification. However, the encryption process is called two times; the first one, when it is used to encrypt original information collected from the user at the first time. Another call happens when there is a need to update the database with new values, such as SQ values, new hash values …etc. One of the main uses for ASKEA, as marked in Fig. 1 with numerical labels: 5 and 6, is to be used as a verification tool for a number of IoT applications using QR-Code.

2.1 The Proposed ASKEA Design

This section is dedicated to explain the ASKEA basic design and how this proposed design could be used with a verification tool for energy informatics related applications.

2.2 Proposed ASKEA Inputs' Design

Initially, ASKEA encrypts whole information collected from the user. Information includes certain values that consist of pure characters, numbers, and a mixture of characters and numbers as described in Fig. 2.

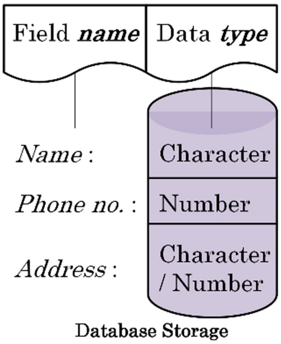

Fig. 2. User's information category

In Fig. 2, data types of 'character', 'number', and 'character/ number' are assigned to be stored in the 'name', 'phone no., fax …etc.', 'address, website, email …etc.' fields, respectively.

To increase the security of encryption scheme, further matter has been considered that is a variable with a property that is capable to frequently change its value and data type. Usually, the user has different activities. Thus, the user could be asked to answer a security question. This feature is so useful to create different answers with different values producing amounts by which the field can be increased or decreased in case the answer is a number. As expected, in case the SQ answer alters, for example, from text to number, its data type will be changing accordingly. This proposed scheme is illustrated in Fig. 3.

Unlike values of fields, shown in Fig. 2, are constants and related data types are fixed; the field in Fig. 3 is dynamically and periodically changing in value and data type. This has added a feature that is, the value is difficult to be predicted when a threat is trying to access or modify data contents.

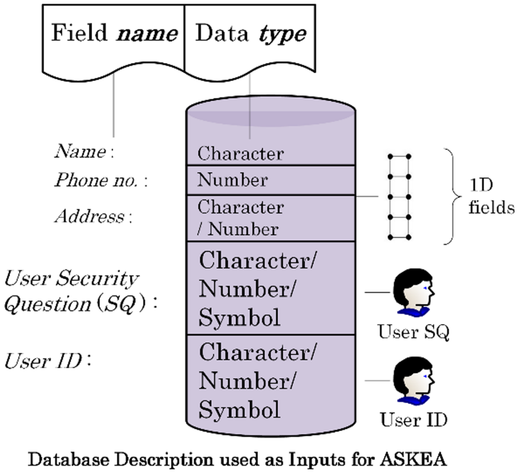
Field name
Data type
Name :
Phone no. :
Address :
User Security Question (SQ) :
User ID :
Character
Number
Character / Number
Character/ Number/ Symbol
Character/ Number/ Symbol
1D fields
User SQ
User ID
Database Description used as Inputs for ASKEA

Fig. 3. Proposed user's information category for ASKEA

As can be seen in Fig. 3, there will be two attributes considered as inputs for ASKEA which are: User SQ (USQ), and User ID (UID). Description of fields of which ASKEA inputs consist is provided in Table 1.

Table 1. Description of input items of ASKEA

		Data descriptor			
		Data type	Array size	Status	
				Data type	Field value
ASKEA Input Attribute/Field	User SQ (USQ)	num/char/ symbol	1	alt	var
	User ID (UID)	num/char	1	fix	const

As can be seen in this table, the field of '*User SQ (USQ)*' has the following data description: a *Data_Type* = 'number, character or symbol', *Array_Size* = '1D', *Status_of_Data_Type* can *alter* between 'numbers', 'characters', and 'symbols', and *Field_Value* = 'variable'.

2.3 Proposed ASKEA Secret-Key Length Design

In order to implement the ASKEA, a series of cascade encryption processes are applied on fields related values. The output of ASKEA is the QR-Code for User ID, User SQ. The technical procedures will be discussed in this section.

ASKEA has adopted to use two schemes for key length, which are 2^i and $2^i + 2^{i-1}$. However, the value of i is designed to be adjustable between key length and encrypted data size. The increment denoted by 2^{i-1} for the second key length is to take into account any increment caused by a potential change inside the ‘field value’ or ‘data type’. Referring to Table 1, for example, the SQ increment could probably happen when it is increased or altered between numbers and characters. However, this design includes a limitation relates to the maximum length of the secret key by the value $2^i + 2^{i-1}$. Thus, the key length must be greater than or equal to the data size.

Therefore, the following two formulas are considered for the secret key length, shown in Eq. (1) and Eq. (2):

$$2^i \le key_1 \ length \tag{1}$$

$$key_2 \ length \ge 2^i + 2^{i-1} \tag{2}$$

2.4 Mathematical Design of ASKEA Input (Plaintext)

The key length has a directly proportional relationship with user information being encrypted. That is, and from this information, the encryption scheme is going to extract a certain amount of data (i.e., plaintext) to be encrypted; and not whole user’s information. This amount of data (plaintext) being encrypted is, as mentioned before, mutable due to it depends on alteration of data type(s). As a result, this amount of data is considered as an ASKEA plaintext input whereas it has a variable length producing two different sizes in bits/ Bytes. The design of key length has considered both minimum and maximum sizes of the plaintext input. This criterion is a must and is expressed in inequalities (1) and (2):

$$u_{data_{min}} < 2^i \tag{3}$$

$$u_{data_{max}} < 2^i + 2^{i-1} \tag{4}$$

whereas $u_{data_{min}}$ and $u_{data_{max}}$ minimum and maximum size(s) of data entered by the user, respectively.

Thus, in order to derive the relationship between the key length and data size with consideration of the adjustable variable (i): firstly, inequalities (2) and (4) are used to produce inequality (6); secondly, inequalities (3) and (5) are used to produce inequality (7), as follows:

$$u_{data_{min}} < 2^i \le key_1 \ length \tag{5}$$

$$u_{data_{max}} < 2^i + 2^{i-1} \le key_2 \ length \tag{6}$$

Thus, inequalities (6) and (7) could produce the following expression:

$$size \ of \ \ data < adjustable \ variable \le key \ length \tag{7}$$

2.5 Proposed Mathematical Operations Based ASKEA Design

As discussed, there is a proportional relationship between the secret key size and data size with consideration of adjustable variable, therefore these two mathematical formulas are true:

$$f\,(key) \propto f\,(data) \tag{8}$$

$$f\,(key) = f\,(data) + c_{HEA} \tag{9}$$

where,

- $f\,(data)$ is an independent function to which user's information is assigned
- $f\,(key)$ is a dependent function and varies based on $f\,(data)$
- c_{HEA} is an ASKEA constant designed exclusively to normalize the size of $f\,(data)$ to a fixed size that is proportional to the length of $f\,(key)$

This equation relates mainly to the length of secret key and size of data being encrypted. Meaning, Eq. (9) is mathematically represented as in Eq. (10):

$$size_{f\,(key)} = size_{f\,(data)} + size_{cHEA} \tag{10}$$

where,

- $size_{f\,(key)}$ is the function by which the key size is calculated
- $size_{f\,(data)}$ is the function used to store a piece of data extracted from user's information; data to be sent to ASKEA for encryption purpose
- $size_{c_{HEA}}$ calculates the size of c_{HEA} by which the plaintext size is normalized to the key length

Equation (10) could be mathematically re-formulated as:

$$k_{size} = P_{text_{size}} + c_{HEA_{size}} \tag{11}$$

where,

- k_{size} has two secret key sizes which are represented in Eq. (12):

$$k_{size} = \left\{ \begin{array}{ll} 2^i & f\,(data) = minimum\ size\ of\ \ data \\ 2^i + 2^{i-1} & f\,(data) = maximum\ size\ of\ \ data \end{array} \right\} \tag{12}$$

- $P_{text_{size}}$ is an aggregation function relates to data entered by user, refer to Table 1, as mathematically represented in Eq. (13):

$$P_{text_{size}} = f\,(input_{user}) \tag{13}$$

- $c_{HEA_{size}}$ is an ASKEA constant and can be calculated using Eq. (14) producing two different values based on minimum and maximum size of data entered by the user:

$$c_{HEA_{size}} = \left\{ \begin{array}{ll} 2^i - P_{text_{size}} & 2^i \geq u\_data_{min} \\ 2^i + 2^{i-1} - P_{text_{size}} & 2^i < u\_data_{max} \end{array} \right\} \tag{14}$$

2.6 $P_{text_{size}}$ Design

In this subsection, ASKEA inputs are in detail discussed. That aims to explain how to obtain appropriate values for length and sizes of ASKEA inputs. Following considerations are in detail discussed:

As for data entered by user, it is a function in which whole entered information is aggregated, see Eq. (13). Once the user has entered and filled up required fields, a series of algebraic operations extracts a piece of data from user's information in accordance with 'ASKEA Input Field' mentioned in Table 1. It is an aggregation function of several user inputs whereas it is simply formulated in (15):

$$f(P_{text}) = f(USQ, UID) \tag{15}$$

where,

- $f(P_{text})$ represents assigned data to ASKEA as a plaintext
- $f(USQ, UID)$ represents data-required input-fields (DRIFs).

In order to determine the size specified for each function, a size-purposed algebraic function has been proposed to obtain the bits-length needed (BLN) for each DRIF. To calculate BLN for each DRIF, a certain piece-of-data (PoD) is extracted from the related DRIF based on algebraic operations and then PoD's size is passed to the BLN.

Thus, to ease the procedure mentioned in Eq. (15), each BLN could be calculated for every DRIF using the following mathematical formulas shown in Eq. (16) and Eq. (17):

$$f(USQ) = inp(u_{SQ}) \tag{16}$$

$$f(UID) = inp(u_{ID}) \tag{17}$$

where,

- $f(USQ)$, and $f(UID)$ are DRIFs for user SQ and user ID, respectively
- $inp(u_{SQ})$ and $inp(u_{ID})$ are alphanumeric values of PoD for USQ and UID RFIDs, respectively.

Once $inp(u_{SQ})$ and $inp(u_{ID})$ have been calculated, the P_{text} is ready to feed the ASKEA, using Eq. (18). In order to find the appropriate ASKEA secret key size, BLNs need to be firstly calculated for each DRIF by using Eqs. (19) - (20):

$$f(P_{text}) = inp(u_{SQ}) + inp(u_{ID}) \tag{18}$$

$$BLN_{USQ} = size_{bits}\{inp(u_{SQ})\} \tag{19}$$

$$BLN_{UID} = size_{bits}\{inp(u_{ID})\} \tag{20}$$

By obtaining values of BLNs in Eqs. (19) - (20), Eq. (13) is re-formulated as in Eq. (21):

$$P_{text\,size} = BLN_{USQ} + BLN_{UID} \tag{21}$$

Usually, a PoD for each DRIF consists of a number of alphanumeric/ numeric values. Thus, each PoD has a 'value' and 'data type'. The relation between PoD and DRIF items is provided in Table 2.

Table 2. DRIF and PoD

DRIF	PoD	
	Value	Data type
User SQ	Variable	alphanumeric
User ID	Constant	alphanumeric

As mentioned before in Table 1, fields requiring data entries whether by automatically or manually ways are supersets of ASKEA inputs. Thus, a series of mathematical subsets which can be derived from Table 2 are mathematically represented, as follows: $\{DRIFs\} \subset \{HEA\ Input\ Attributes\}$, the set DRIF is represented by Eq. (22):

$$\{DRIF\} = \{\{USQ\}, \{UID\}\} \tag{22}$$

Each subset is defined using the following algebraic description:

$$\{USQ\} = \{inp_{SQ} : \forall inp \in\ user\,inputs\,for\,SQ\}$$

$$\{UID\} = \{inp_{ID} : \forall inp \in\ user\,inputs\,for\,ID\}$$

Each PoD is a subset of its related DRIF's superset, which is mathematically represented as follows: $\{PoDs\} \subset \{DRIFs\}$.

As a result, further defined descriptions could be derived, as follows:

$$\left\{PoD_{USQ}\right\} \subset \{USQ\};\ \Leftrightarrow \left\{PoD_{USQ}\right\} \cap \{USQ\} = \left\{PoD_{USQ}\right\}$$

$$\{PoD_{UID}\} \subset \{UID\};\ \Leftrightarrow \{PoD_{UID}\} \cap \{UID\} = \{PoD_{UID}\}$$

Each of abovementioned algebraic definitions is true with every condition accompanied.

As per the design limitation, the user is allowed to enter an answer for the SQ with a length up to 8 Bytes.

In this design, the minimum size of an answer the user can type is '1' alphanumeric e.g., '1', '#', 'a', or 'z'. Thus, '1' alphanumeric can be written as: 2^b; whereas $b = 3$.

Therefore, 1 Byte is a minimum length of the SQ answer. As a result, the proposed algorithm has adopted that 1 character of 8-bits length is assigned to 'i'. That is represented in by Eq. (23):

$$i = 2^b \tag{23}$$

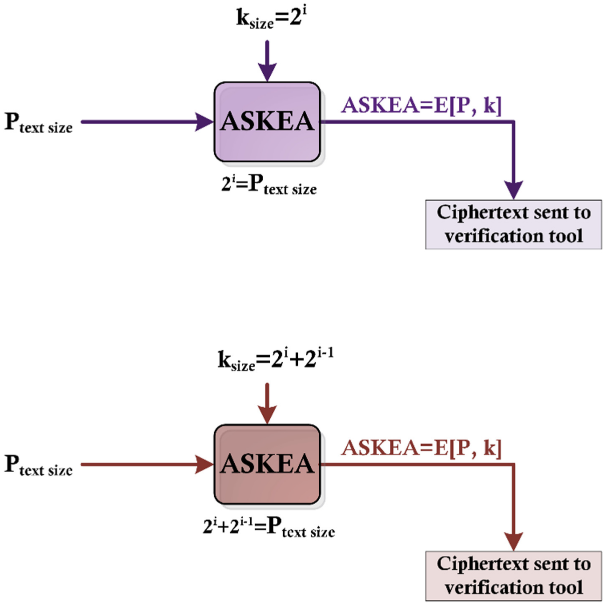
k_{size}=2ⁱ
P text size
ASKEA
ASKEA=E[P, k]
2ⁱ=P text size
Ciphertext sent to verification tool
k size=2i+2i-1
P text size
ASKEA
ASKEA=E[P, k]
2i+2i-1=P text size
Ciphertext sent to verification tool

Fig. 4. ASKEA block-diagram – security scheme design

2.7 Proposed ASKEA Block-Diagram Design

A simple model of ASKEA ingredients for two schemes of plaintext size(s), which are $P_{text_{size}} < 2^i$ and $P_{text_{size}} < 2^i + 2^{i-1}$, is illustrated in Fig. 4.

3 Results and Discussion

This part presents results in terms of plaintext time complexity, computation time, security factors. Then, it evaluates the proposed design performance based on abovementioned obtained results.

3.1 Time Complexity

Big-O-Notation will be used for a further evaluation and comparative analysis. The running time for the proposed secret key design will be described. Two main procedures will be evaluated which are plaintext entries values and secret key variables. Usually, O(n) with the proposed design depends on user entries and values being extracted.

As for USQ and UID, O(n) depends on values being generated from equations. An array $A_{USQ}[k1]$ and $A_{UID}[k2]$ is dedicated to store k elements. Therefore: O(n) for this case is defined as in Eqs. (24) and (25):

$$O(n)_{\mathrm{AUSQ}[k1]} = O(k1) \tag{24}$$

$$O(n)_{\mathrm{AUID}[k2]} = O(k2) \tag{25}$$

Totally, $O(n)$ for ASKEA is a summation formula of Eqs. (24) and (25). Since $k1$ and $k2$ are constants, therefore Eq. (26) is used to obtain value of $O(n)_{\text{ASKEA}}$:

$$O(n)_{\text{ASKEA}} = O(k) \tag{26}$$

A time complexity value equals to $O(k)$ looks reasonable when compared to other time complexity schemes.

3.2 Time Based Efficiency

The proposed design scheme of secret key design in term of computation time needed to generate one secret key. The proposed ASKEA in term of secret key and plaintext processing computation times has been compared to other methods which are Password and Certificate techniques. The obtained result is given in Fig. 5.

In this comparison, the procedure of how samples are taken is explained as follows: many tests using different samples of entries have been considered. There have been seven tests at which different number of samples is considered. In regard to the proposed design, the user entries of plaintext have been fed to the ASKEA inputs which are $P_{text_{size}}$ and k_{size}. For the first implemented test, 7 samples are considered to produce 7 secret keys accordingly. The computation time for each sample is recorded; whereas the computation time mentioned in Fig. 5 is the average computation time for whole samples of the related test. Similarly, this is repeated for the rest of tests with different number of samples. As for Password and Certificate techniques, both are used to create the same number of words.

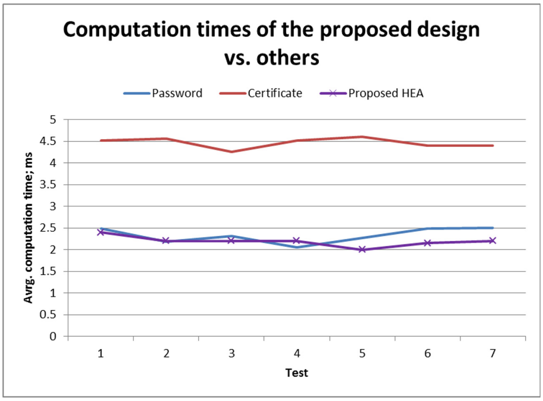

Fig. 5. Average computation times of the proposed secret key design vs. others

There are seven tests in which 10, 20, 50, 75, 100, 200, and 500 samples have been considered. Thus, the computation time for each sample is calculated and then the average computation time is obtained for whole samples for every test. The related result is provided as mentioned in Fig. 5.

As noticeable, the proposed design has less time than Certificate and Password techniques unless two tests in case of Password. The reason is that the length of secret key is the maximum length which is considered acceptable when comparing to the Certificate technique.

3.3 Security Factors Analysis - Confidentiality and Privacy vs. Unpredictability

The aim of choosing 2 schemes of key length is to increase the privacy of encryption algorithm by building an unpredictable behavior of key. Thus, the way the secret key changes help increase the confidentiality of plaintext being encrypted.

4 Conclusion

This paper has proposed a dynamical design for a secret key based on user plaintexts. Original inputs are encoded. To perform such a design, a number of algebraic operations have been derived and proposed. The essential part in designing such a secret key is to keep an unpredictable behavior of ASKEA in terms of key length and the way the values are generated. This design could be followed by a verification tool in which a QR-Code is used to achieve a high level of security and privacy to such energy informatics, smart home, and EIoT applications.

Acknowledgement. This research work is funded by Universiti Tenaga Nasional (UNITEN) with Grant Code: (J510050002/2021048).

References

1. Liu, L., Wang, Q., Wu, Y.: QR code positioning algorithm. In: The 2nd International Conference on Computing and Data Science, Stanford, CA, USA (2021)
2. Al-Ghaili, A.M., Kasim, H., Othman, M., Hassan, Z.: A New Encryption Scheme Method (ESM) using capsulated-layers conception for verified QR-Tag for IoT-based smart access systems. In: Balas, V.E., Solanki, V.K., Kumar, R., Khari, M. (Eds.) Internet of Things and Big Data Analytics for Smart Generation, Cham, Springer International Publishing, pp. 77–103 (2019)
3. Ramalho, J.F.C.B. et al.: Super modules-based active QR codes for smart trackability and IoT: a responsive-banknotes case study. npj Flexible Electron. **4**, (1), 11 (2020)
4. Liu, Z., Choo, K.K.R., Grossschadl, J.: Securing edge devices in the post-quantum internet of things using lattice-based cryptography. IEEE Commun. Mag. **56**(2), 158–162 (2018)
5. Neisse, R., Baldini, G., Steri, G., Ahmad, A., Fourneret, E., Legeard, B.: Improving Internet of Things device certification with policy-based management, In: 2017 Global Internet of Things Summit (GIoTS). Vol. 6–9, pp. 1–6 (2017)
6. Rane, S., Dubey, A., Parida, T.: Design of IoT based intelligent parking system using image processing algorithms. In: 2017 International Conference on Computing Methodologies and Communication (ICCMC), pp. 1049–1053 (2017)
7. Xiao-Long, W., Chun-Fu, W., Guo-Dong, L., Qing-Xie, C.: A robot navigation method based on RFID and QR code in the warehouse. In: 2017 Chinese Automation Congress (CAC), vol. 20–22, pp. 7837–7840 (2017)
8. Ghaffari, M., Ghadiri, N., Manshaei, M.H., Lahijani, M.S.: P4QS: a peer-to-peer privacy preserving query service for location-based mobile applications. IEEE Trans. Veh. Technol. **66**(10), 9458–9469 (2017)

9. Sha, K., Yang, T.A., Wei, W., Davari, S.: A survey of edge computing-based designs for IoT security. Digit. Commun. Netw. **6**(2), 195–202 (2020)
10. Parikh, S., Dave, D., Patel, R., Doshi, N.: Security and privacy issues in cloud, fog and edge computing. Procedia Comput. Sci. **160**, 734–739 (2019)
11. Al-Ghaili, A.M., Kasim, H., Othman, M., Hassan, Z.: Security factors based evaluation of verification algorithm for an IoT access system. In: Saeed, F., Gazem, N., Mohammed, F., Busalim, A. (eds.) Recent Trends in Data Science and Soft Computing, Cham, Springer International Publishing, pp. 384–395, (2019)
12. Tkachenko, I., Puech, W., Destruel, C., Strauss, O., Gaudin, J.M., Guichard, C.: Two-Level QR code for private message sharing and document authentication. IEEE Trans. Inf. Forensics Secur. **11**(3), 571–583 (2016)
13. Lin, S.S., Hu, M.C., Lee, C.H., Lee, T.Y.: Efficient QR code beautification with high quality visual content. IEEE Trans. Multimedia **17**(9), 1515–1524 (2015)
14. Lin, P.Y.: Distributed secret sharing approach with cheater prevention based on QR code. IEEE Trans. Industr. Inf. **12**(1), 384–392 (2016)
15. Nazemzadeh, P., Fontanelli, D., Macii, D., Palopoli, L.: Indoor localization of mobile robots through QR code detection and dead reckoning data fusion. IEEE/ASME Trans. Mechatron. **22**(6), 2588–2599 (2017)
16. Kirkham, T., Armstrong, D., Djemame, K., Jiang, M.: Risk driven Smart Home resource management using cloud services. Futur. Gener. Comput. Syst. **38**, 13–22 (2014)
17. Chen, Y.H., Tsai, M.J., Fu, L.C., Chen, C.H., Wu, C.L., Zeng, Y.C.: Monitoring elder's living activity using ambient and body sensor network in smart home. In: 2015 IEEE International Conference on Systems, Man, and Cybernetics, pp. 2962–2967 (2015)
18. Kanaris, L., Kokkinis, A., Fortino, G., Liotta, A., Stavrou, S.: Sample size determination algorithm for fingerprint-based indoor localization systems. Comput. Netw. **101**, 169–177 (2016)
19. Gentili, M., Sannino, R., Petracca, M.: BlueVoice: voice communications over Bluetooth Low Energy in the Internet of Things scenario. Comput. Commun. **89–90**, 51–59 (2016)