Intelligent Automation & Soft Computing
DOI:10.32604/iasc.2022.020422
Article

Tech Science Press

Efficient Key Management System Based Lightweight Devices in IoT

T. Chindrella Priyadharshini[1,*] and D. Mohana Geetha[2]

[1]Department of Information and Communication Engineering, Anna University, Chennai, 600025, India
[2]Department of Electronics and Communication Engineering, Sri Krishna College of Engineering and Technology, Coimbatore, 641008, India
*Corresponding Author: T. Chindrella Priyadharshini. Email: chindrellaphd@gmail.com
Received: 23 May 2021; Accepted: 21 July 2021

Abstract: The Internet of Things (IoT) has changed our lives significantly. Although IoT provides new opportunities, security remains a key concern while providing various services. Existing research methodologies try to solve the security and time-consuming problem also exists. To solve those problems, this paper proposed a Hashed Advanced Encryption Standard (HAES) algorithm based efficient key management system for internet-based lightweight devices in IoT networks. The proposed method is mainly divided into two phases namely Data Owner (DO) and Data User (DU) phase. The DO phase consists of two processes namely authentication and secure data uploading. In authentication, the registration process consists of three phases namely KGC, SCC and DVC. In Key Generation Center (KGC), the device ID is converted into a 128-bit hash key using the Point on Curve based Fowler–Noll–Vo (PoC-FNV) algorithm. In DVC, the Data Access Policy (DAP) is created by using the selected attributes which is in turn selected by using the Chen Chaotic Chimp Optimization Algorithm(CCCOA). Thus, the authentication information is stored in the blockchain. If the person is an authorized, then the data is securely stored and uploaded into the cloud server using HAES. In the DU phase, the DU sends the data access request to the blockchain then the blockchain forwards the request to the DO, if the DO accepts the request then the DU obtains the data by using the DAP. Finally, the performance of the proposed method is compared with the existing methods and attains better result than the existing research methodologies.

Keywords: Point on curve based Fowler–Noll–Vo; hashed advanced encryption standard; Chen chaotic chimp optimization algorithm; data access policy; blockchain

1 Introduction

The IoT is globally connected networks, which are associated with each other via the Internet [1]. IoT is rapidly growing, as the number of devices increases in the network [2]. IoT broadens the likelihood of the Internet and makes it unavoidable [3]. The IoT communication environment is used in various types of applications such as “smart health care”, “smart traffic monitors” and “smart homes”, etc. [4]. However,

the sensor nodes in an IoT environment have certain limitations, such as low storage capacity, limited battery and computing capability. Data involved in IoT ecosystem contains high sensitive information that requires high confidentiality. To address these issues, the Cloud computing paradigm offers unlimited technology to store, process, analyze, deliver, distribute and secure critical data [5,6]. Therefore, users can also quickly and easily access the cloud computing center and process data through the Internet [7]. The process of sending and storing sensitive data on the server over insecure cloud-IoT network becomes open to many attacks. The sensitive data must be protected from unauthorized users when it is transferred, collected, processed and stored [8]. Data security is the most important concern in the cloud and it can be ensured by setting up a secure channel between different devices. A number of security operations (authentication, authorization and data integrity) are needed. Algorithms are available for data security like symmetric and asymmetric algorithms in cryptography. A bunch of these algorithms contains AES, BLOWFISH, RSA, DES and triple DES, etc. [9,10]. But devices used in IoT are not strong enough to process these bunch of algorithms and not capable of storing more data on it as well. But they can process a lightweight complex algorithm supporting a smaller number of bits at a time. Hence, it becomes necessary to perform lightweight cryptography in an IoT-cloud environment [11–13]. Current models for data sharing and user privacy are commonly based on the trusted third party. The key management schemes based on the pre-shared key framework and key pool framework are operated at the Key Generation Center (KGC) and are not scalable for large number of entities. Management of keys is one of the most critical components of the cryptographic system. The various benefits afforded by blockchain technology makes it an attractive solution for addressing the aforementioned problems in IoT [14–16]. Blockchain and distributed ledger are pivotal part of Bitcoin that eliminates the necessity for a central authority as it works as a distributed database that is capable of storing every single operation performed by participating parties in a given system [17,18]. Blockchain integration in IoT will strengthen the overall IoT network and create a secured distributed network on combining general and blockchain-specific security features [19]. To improve the security level of the IoT devices, this paper proposed HAES-based key management for an efficient key management system for internet-based lightweight devices in IoT networks.

2 Literature Review

Reem et al. [20] presented a lightweight and secure multi-factor device authentication protocol for IoT devices. The scheme was based on two concepts, namely configurable Physical Unclonable Functions (PUF) within IoT devices and channel-based parameters. It uses simple cryptographic operations such as the bit-wise exclusive-OR operation and a one-way hash function. The unique PUF value serves as the mutual secret identifier between the pair of users, which frequently changes for every session. Moreover, the protocol had exploited the random channel characteristics to provide high robustness against different kinds of attacks, while maintaining low complexity. The framework combined physical layer security with PUFs to authenticate communicating devices, dynamically. However, as the number of bytes increase, more time is required to process (encrypt) them. Bander et al. [21] developed an Improved Lightweight Authentication Scheme for IoT deployments (ILAS-IoT). The approach had slightly modified the IoT device enrollment phase and some changes were made in the login and authentication phases. The security of ILAS-IoT was carried out by formal and informal methods. Although the ILAS-IoT increased some computation and communication overheads as compared the other schemes, ILAS-IoT provides resistance to all known attacks including stolen verifier attacks and completes the process correctly. Moreover, ILAS-IoT also provides an access control mechanism and more desirable in IoT-based access control scenarios. Deepsubhra et al. [22] developed blockchain technology for risk identification along with risk prevention using RSA encryption and decryption techniques in IoT devices. The Mosquitto (MQTT) broker acts as a negotiator to convey data between cloud consumers and cloud providers. By using MQTT protocol and a virtual private network, the man-in-the-middle attack was

prevented successfully. As per the experimental results, this method is able to recover 100% RAM, 97.64% CPU utilization and 78.7% storage using encryption and algorithm on blockchain infrastructure. This method has an impact of the high power consumption and processing power due to the usage of the MQTT protocol which needs more power and memory. Deebak et al. [23] presented a Lightweight-based Authentication and Key Management (L-AKM) scheme for Smart IoT-Assisted Systems. The performance analysis proved that the L-AKM scheme achieved better security to resist various attacks such as forgery, replay, password guessing, etc. However, the experimental analysis shows that the L-AKM consumes more transmission delay and throughput rate due to extra authentication phases involved in the working process of this scheme. Khalid et al. [24,25] introduced a Light-Weight Structure-based Data Aggregation Routing (LSDAR) protocol for IoT integrated Next-generation Sensor Networks for the improvement of energy routing performance with data protection against malicious threats. Firstly, the network nodes were decomposed into independent clusters based on varying radii and preventing energy holes around the locality of the Base Station (BS). Effective and loop-free routing paths are constructed based on the A-star heuristics algorithm. This method tolerates with complexity problems as A-star heuristics algorithm is prone to time complexity.

3 Key Management System for Internet-Based Lightweight Devices in IoT

The Internet-of-Things (IoT), which refers to the interconnection of heterogeneous devices, has gained a lot of interest and it has witnessed a growth in the number of IoT devices connected due to the importance of such system in today's communication networks. However, the IoT has an enormous threat to security and privacy due to its heterogeneous and dynamic nature. Authentication is one of the most challenging security requirements in the IoT environment, where a user (external party) can directly access information from the devices, provided the mutual authentication between the user and devices takes place. For authentication, this research paper proposes an efficient key management system for internet-based lightweight devices in IoT networks using Hashed AES algorithm. The proposed method is mainly divided into two phases namely, data owner and data user phase. The data owner phase has the authentication and secure data uploading phase. In the authentication phase, the data owner registers their details and their sensor device details. The registration consists of three steps such as, Key Generation Centre (KGC), Signature Creation Centre (SCC) and Data Verification Centre (DVC). In the KGC phase, the device ID is changed into a 128-bit hash key, which is generated by using the Point on Curve-based Fowler–Noll–Vo hash function (PoC-FNV). In the SCC phase, the signature is derived from the device ID and the hash value of the device ID. In the DVC phase, the attributes are extracted from the data owner and device details using Chen Chaotic Chimp Optimization Algorithm (CCCOA). By using the selected attributes, the data access policy is created using Attribute-Based Access Control (ABAC). The data of those three phases are stored in the blockchain. During login time, the data owner enters their Device ID, username, password and signature. In the verification process, the Cloud Service Provider (CSP) checks the previously stored information with login information and if it gets matched, then the data is uploaded into the cloud server. The next phase is the secure data uploading phase where the data owner securely uploads the data by using the Hashed Advanced Encryption Standard (HAES) algorithm. The next phase is the data user phase, that has authentication, data request and secure downloading steps. Data-user requests the data from the blockchain which in-turn forwards the request to the data owner. If the data owner accepts the request, the access policy and the decryption key are sent to the data user. Then, the data user downloads the data with help of obtained access policy and decrypts the data by using the decryption key. The block diagram for the proposed method is shown in Fig. 1.

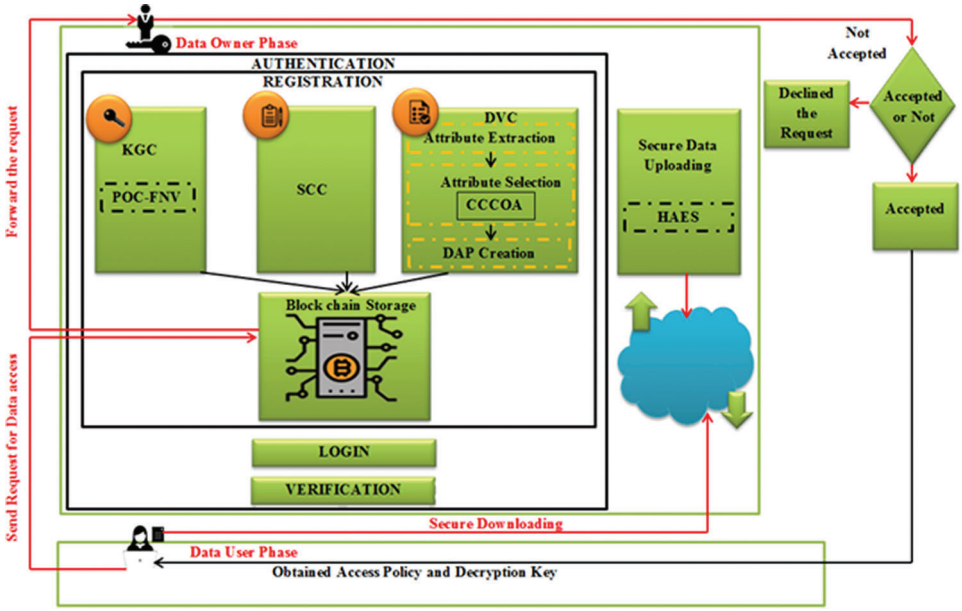

Figure 1: Block diagram of the proposed method

3.1 Data Owner

Data Owner (DO) is the act of having legal rights and complete control over a single piece or set of data elements. Here, the data owner has the sensor devices. This data owner has two processes for storing their data into the cloud namely authentication and secure data uploading. The data owner is expressed as follows,

$$DO_s = d_i, i = 1, 2, \cdots, n \tag{1}$$

where, DO_s denotes the data owner sets and d_i defines the n-individual data owners.

3.1.1 Authentication

Authentication is the process of verifying the identity of a person or device. The data owner authentication phase consists of four sub-phases namely, registration, blockchain storage, login and verification.

3.1.1.1 Registration

In this registration phase, the data owner enters their details along with the sensor device details in the Cloud Service Provider (CSP). At the registration time, the Data Owner(DO) must provide the data owner name, data owner password, device ID, device name, password, Device Type, Frequency, Data Rate, Range, Power, etc. The registered details are stored on the BlockChain, which is expressed in Eq. (2).

$$R_g = \left\{ \overrightarrow{t_1}, \ \overrightarrow{t_2}, \ \cdots, \ \overrightarrow{t_n} \right\} \tag{2}$$

where, R_g indicates the registered information and t_n defines the n-number of details. At this registration time, the CSP generates the key in KGC, generates the signature in SCC and verifies in DVC and these steps are explained as follows,

(a) Key Generation Centre

During this registration process, the key is generated from the device ID. Key generation is a promising technique to bootstrap secure communications for the Internet of Things (IoT) devices that have no prior knowledge of each other. Here, the 128-bit hash code key is generated by using PoC-FNV algorithm. The Fowler–Noll–Vo is a non-cryptographic hash function. It creates a non-zero FNV offset basis. One of FNV key's advantages is that it is very simple to implement. Start with an initial hash value of FNV offset basis. But if the input size is larger, then the original input is unpredictable. Hence, the Point on Curve value is used to solve the unpredictable problem for larger size of input . The steps of the PoC-FNV is described as follows,

Step 1: DO registration details, such as device ID t_1 is considered as the input value *(I_k)*.

$$I_k = \left\{ \overrightarrow{t_1} \right\} \tag{3}$$

Step 2: Start with an initial hash value *(H)* of the FNV offset basis. There are several different FNV offset basis for various input bit lengths. Eq. (4) describes the offset value of the input. Each byte in the input hash is XOR-ed with an octet of data, which is shown in Eq. (5). In the traditional FNV algorithm, the hash is multiplied with the FNV prime, which is simple hashing and can be easily hacked by others and the unpredictable original input may exist for the larger input size. Instead of only multiplying the FNV prime, POC is multiplied with the hash code as shown in Eq. (6). This method gives lengthy and strong hash codes. Thus, the complexity of the hash key is increased by this novel hash algorithm.

$$H1 \; = \; offset(Ik) \tag{4}$$

$$H_2 = H_1 \oplus D_0 \tag{5}$$

$$H = H_2 * P * P_\infty \tag{6}$$

where, H stands as the final hash value, H_1 is offset value of the input, H_2 indicates the XOR value of the initial offset value with D_0, D_0 represents octet_of_data (i.e., the input device ID is split into octet size), P stands for selected FNV prime, P∞ represents POC. The POC is generated based on the elliptic curve. Thus, the equation for the elliptic curve is derived as follows,

$$v^2 = u^2 + \text{au} + \text{b} \tag{7}$$

where, v and u are the standard variables that define the function while a and b are the constant coefficients that define the curve. Finally, the 128-bit hash code is generated by using these steps.

(b) Signature Creation Centre

After hash code key generation, the CSP creates the signature for DO. A cloud-based signature is considered as a paradigm for proper, reliable, secure infrastructure, with flexible access to the network. This signature is helpful for providing a high level security. Here, the signature is created by using the 128-bit hash code that is generated and the device ID, is expressed as follows,

$$\delta = H \oplus \overrightarrow{t_1} \tag{8}$$

where, δ defines the generated signature and t_1 defines the device ID.

(c) Data Verification Centre (DVC)

In this section, with the data owner and device details, the Data Access Policy (DAP) is created. An access control policy would be a policy that defines the kind of user has the permission to read the documents. This DVC has three processes namely, attribute extraction, attribute selection and attribute-based access policy control.

First, the sensor device and the data owner-related attributes are extracted as data owner name, data owner password, device ID, device name, password, Device Type, Frequency, Data Rate, Range, Power, etc. Thus, the attributes are expressed as follows in Eq. (9),

$$A = \left\{ \overrightarrow{t_1},\ \overrightarrow{t_2},\ \overrightarrow{t_3}\ \cdots,\ \overrightarrow{t_n} \right\} \tag{9}$$

where, A defines the set of attributes. The important attributes are only selected for reducing the verification process time using Chen Chaotic Chimp Optimization Algorithm (CCCOA). A novel hunting-based optimization algorithm called Chimp Optimization Algorithm (ChOA) is taken, which is inspired by their individual intelligence and sexual motivation of chimps in their group hunting, which is different from the other social predators. In a chimp colony, there are four types of chimps entitled for the hunting process known as a driver, barrier, chaser and attackers. They all have different abilities and this diversity is necessary for a successful hunt. This algorithm has a faster convergence speed, but the search capability is presented with in the particular boundary level to extend the searching capability. This proposed method uses the Chen Chaotic method in this ChOA. Here, the chimps are initialized as the attributes, as expressed in Eq. (9). After that, the fitness value is calculated for the attributes. The fitness value is calculated based on the access policy structure. The expression of the fitness calculation is given as follows,

$$Fit = \tau(APS) \tag{10}$$

where, Fit defines the fitness function and $\tau(APS)$ indicates the better structure of the access policy. The better fitness value is initially fixed as the threshold value. Then, the fitness is calculated for the given input attributes and if the calculated fitness value of the input is equal to the originally-fixed fitness value, then the attributes are taken, otherwise, the attributes are updated by using the further steps of the chimp optimization process.

In this Choa, prey is hunted during the exploration and exploitation phases. The mathematical model of driving and chasing the prey is expressed as follows,

$$Z = |C \cdot B_A(g) - l \cdot B_c(g)| \tag{11}$$

$$B_c(g+1) = B_A(g) - k \cdot Z \tag{12}$$

where, Z defines the driver's prey, A represents the DO and device registration details and B_c defines the access policy structure. Next, g stands for number of iteration, k, l and C are the coefficient vectors, B_A is the vector of attributes position and Y_i is the position vector of an input. Then K, l and C vectors are calculated by Eqs. (13)–(15), respectively.

$$k = 2 \cdot Q \cdot s_1 - Q \tag{13}$$

$$C = 2 \cdot s_2 \tag{14}$$

$$l = \psi \tag{15}$$

where, Q denotes the value which is reduced non-linearly from 2.5 to 0 through the iteration process (in both exploitation and exploration phase), S_1, S_2 are the random vectors in the range of [0,1], l represents Chen's chaotic vector calculated based on the various chaotic map and ψ represents the chaotic value. Thus, Chen's chaotic system is described as,

$$w_1 = -aw_1 + aw_2 \tag{16}$$

$$w_2 = -w_1 w_3 - (\gamma - \alpha) w_1 + \gamma w_2 \tag{17}$$

$$w_3 = w_1 w_2 - \beta w_3 \tag{18}$$

where, α, β, γ are positive real numbers, w_1, w_2 and w_3 are the point of the chaotic map. The mathematical model of the attacking behaviour of chimps is given by two approaches as follows: The chimps are capable of exploring the prey's location (by driving, blocking and chasing) and then encircling it. Hence, best of the four solutions is obtained and stored while the other chimps are forced to update their positions according to the best chimp's locations. This relationship is expressed as follows,

$$Z_{\text{att}} = | C_1.B_{att} - l_1.B| \tag{19}$$

$$Z_{\text{bar}} = | C_2.B_{bar} - l_2.B| \tag{20}$$

$$Z_{\text{cha}} = | C_3.B_{cha} - l_3.B| \tag{21}$$

$$Z_{\text{dri}} = | C_4.B_{dri} - l_4.B| \tag{22}$$

where Z_{att} , Z_{bar}, Z_{cha} and Z_{dri} represents the attacker, barrier, chasing and driver chimps respectively (i.e., access policy structure) and B_{att}, B_{bar}, B_{cha} and B_{dri} indicates the vector attacker, barrier, chasing and driver of attribute position. The position of the attributes are expressed as follows,

$$B_1 = B_{att} - l_1 \ (Z_{\text{att}}) \tag{23}$$

$$B_2 = B_{bar} - l_2 \ (Z_{\text{bar}}) \tag{24}$$

$$B_3 = B_{cha} - l_3 \ (Z_{\text{chat}}) \tag{25}$$

$$B4 = B_{dri} - l4 \ (Z_{\text{dri}}) \tag{26}$$

At last, the position of the attributes are updated by using Eq. (27),

$$B(g + 1) = B1 + B2 + B3 + B4 \ / \ 4 \tag{27}$$

The chaotic behaviour in the final stage helps chimps to further alleviate the two problems of entrapment in local optima and slow convergence rate in solving high dimensional problems. The position of the attribute policy structure is updated by using Eq. (28),

$$B_c(g+1) = \begin{cases} B_A(g) - k \cdot Z, & if \ \omega < 0.5 \\ l \quad , & if \ \omega \geq 0.5 \end{cases} \tag{28}$$

where, η is a random number in [0,1]. By using this optimization process, the important attributes are selected from the extracted attributes.

Fig. 2 shows the pseudo-code of the CCCOA. The pseudo-code explains the initialization, fitness calculation, driving, chasing prey and attacking prey functions. This process is related to the attributes selection. After that, the Data Access Policy (DAP) is generated by using the Attribute-Based Access Control (ABAC). The combination of the selected attributes is known as the access policy. This Access Policy will be created for the uploaded file and a copy will be given to the user uploaded for future authentication. Access policy creation is derived as follows,

$$APS = \sum_{i=1}^{n} \overrightarrow{t_i} \tag{29}$$

where, *APS* describes the created access policy.

Input: Extracted attributes, $A=\{\vec{t}_1,\vec{t}_2,\vec{t}_3,.....,\vec{t}_n\}$
Output: Select important attributes

Begin
Initialize population $\vec{t}_1,\vec{t}_2,\vec{t}_3,.....,\vec{t}_n$, and maximum iteration M_t.
Calculate fitness function *Fit*
Set iteration $I_t=1$
While $(I_t \le M_t)$ **do**
for each chimp
Use its group strategy to update k,l and C
End for
for each search chimp
if $(\omega<0.5)$ {
Update the position of the current search agent by,
$B_c(g+1)=B_A(g)-k.Z$
} **else** {
Update by using equation (28)
} **end if**
end for
Update k,l and C
Update B_{att}, B_{bar}, B_{cha}, and B_{dri}
$I_t=I_t+1$
End while
Return important attributes
End

Figure 2: Pseudo code for the CCCOA

3.1.1.2 Details Stored into Blockchain

In this section, the registered information is stored in blockchain technology for enhancing the security level of cloud computing. Blockchains are write-only data structures with no administrative permissions for editing or deleting the data. The data structures are known as blocks and are distributed in a P2P network. Each block contains the cryptographic hash function of the previous block and is used to develop a link between them. The linked blocks form a complete chain, hence the term blockchain. The hash function maintains security, integrity and immutability of the blockchain.

$$R_g\left(\vec{\zeta}\right) \xrightarrow{\text{moved}} (B_c) \tag{30}$$

where, $R_g\left(\vec{\zeta}\right)$ defines the registered details and B_c indicates the blockchain.

3.1.2 Login

After DO registration, when the owner wants to upload the resources on the cloud, the owner should log in to the cloud. At the time of login, the DO enter the sensor device ID, username, password and the obtained signatures during the registration process. Thus, the login details are represented as follows,

$$LO_d = \left\{\overrightarrow{t_1},\ \overrightarrow{t_2},\ \overrightarrow{t_3},\ \delta\right\} \tag{31}$$

where, LO_d describes the login details and $\left\{\overrightarrow{t_1},\overrightarrow{t_2},\overrightarrow{t_3},\delta\right\}$ are the sensor device ID, username, password and the signature.

3.1.2.1 Verification

At the time of the verification process, the saved details of the user are matched with the login details and then the user is allowed to access the cloud server. If the login details are not matched with the stored details, then the access gets denied.

3.1.3 Secure Data Uploading

If the person is authenticated, then the cloud server is allowed to upload their document. Here, the data is securely uploaded by using the Hashed Advanced Encryption Standard (HAES) algorithm. AES is a block cipher well-known round-based symmetric, where sizes of the input and output are equal to 128 bits and fixed. In this research method, the input key is considered as the 128-bit hashed key generated by the PoC-FNV algorithm. Primitive four functions are executed in a sequence $Nr - 1$ time compose a loop called a round, Sub-Byte, Shift Row, Mix-Column and AddRoundKey. These four steps are done in the encryption phase.

(a) Substitute Bytes (Sub-bytes Operation)

The Sub-Byte method is a non-linear byte substitution, using a replacement table (S-box), ordered on the basis of multiplication. The S-box converts the byte into another value using its hexadecimal code.

(b) Shift Rows transformation:

It is a simple byte transposition. The bytes in the last three rows of the state depending upon the row location are cyclically shifted. For the 2nd row, the 1-byte circular left shift is performed. For the 3rd and 4th row, 2-byte and 3-byte left circular left shifts are performed respectively.

(c) Mix-columns transformation:

This round is equivalent to a matrix multiplication of each Column of the states. Each column of four bytes is now transformed using a special mathematical function. This function takes four bytes of one column as input and then outputs four completely new bytes, which replace the original column. The result is another new matrix consisting of 16 new bytes. It should be noted that this step is not performed in the last round.

(d) Addroundkey transformation:

It is a bitwise XOR between 128 bits of the present state and 128 bits of the round key. This transformation is its own inverse.

3.2 Data User

The data user accesses the data from the cloud server. The data user phase consists of three steps, namely the authentication phase, data request phase and secure data downloading phase. Here, the authentication phase is different from the data owner phase, where the user enters only their details. If the user is an authorized user, then the user sends the request to the blockchain and the blockchain forwards the request to the DO for accessing the data. If DO accepts the request, then the data access policy and the decryption key are sent to the Data User (DU) else, the access policy is not forwarded to the DU. After that, the DU tries to download the data from the cloud server, if the access policy given by the DU is matched with the stored access policy of the data, then the data is downloaded. Next, the DU decrypts the data by using the decryption key of the HAES. The decryption process is the reverse of the encryption process namely, AddRoundKey, MixColumn, ShiftRow and SubByte.

4 Result and Discussion

The performance of the proposed key management system for internet-based lightweight devices in IoT networks using Hashed AES algorithm is analyzed. The proposed method is implemented in the working platform of JAVA.

4.1 Performance Analysis

In this sub-section, the performance analysis is done in three parts namely, (a) hash key generation, (b) secure data uploading and (c) attribute selection.

4.2 Performance Analysis for Hash Key Generation

Here, the performance of the proposed PoC-FNV is analyzed with existing RACE Integrity Primitives Evaluation Message Digest (RIPEMD), Message Digest Algorithm 5 (MD5), Spooky Hash and FNV algorithms with respect to the hash code generation time which is shown in Tab. 1.

Table 1: Performance analysis of proposed PoC-FNV with the existing algorithms based on hash code generation time

Algorithms	Hash code generation Time (ms)
Proposed PoC-FNV	1022
FNV	1895
Spooky Hash	2784
MD5	3115
RIPEMD	3892

Tab. 1 describes the performance of the proposed PoC-FNV algorithm with the existing FNV, spooky hash, MD5 and RIPEMD algorithms in terms of hash code generation time. The hash code generation time defines how much time is taken for converting the device ID into a 128-bit hash key. Here, the proposed PoV-FNV algorithm only takes 1022 ms time to convert the input device ID into hash key, but the existing algorithms such as FNV, spooky hash, MD5 and RIPEMD takes time of 1895, 2784, 3115 and 3892 ms respectively. Thus, it denotes that the existing RIPEMD takes more time when compared with other existing algorithms and proposed algorithms. From this analysis, the existing FNV and spooky hash algorithms are much better than the other existing algorithms but those algorithms also take more time than the proposed method. Finally, it is deduced that the proposed takes only less time than the existing methods. The pictorial representation of this Tab. 1 is shown in Fig. 3.

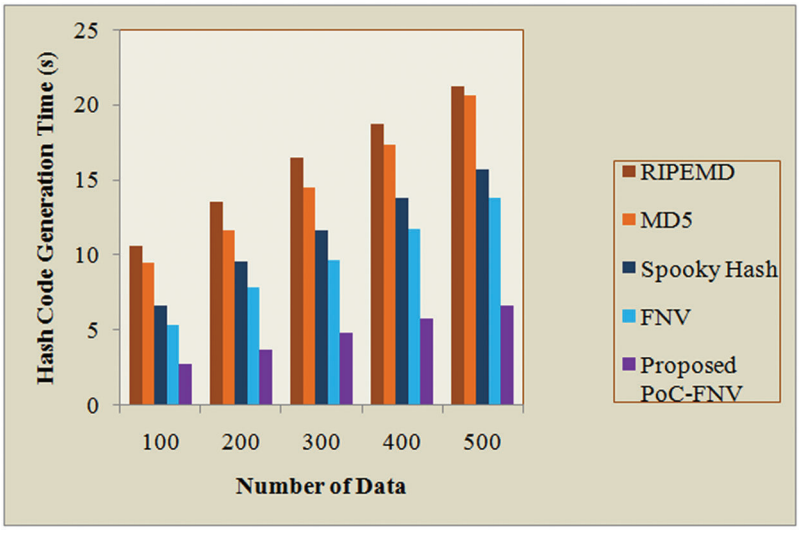

Figure 3: Hash code generation time analysis for the hash key generation algorithms

4.3 Performance Analysis for Secure Data Uploading

Here, the performance of the proposed HAES is analyzed with the existing Blowfish, Data Encryption Standard (DES), Rivest Cipher 4 (RC4) and AES based on encryption time, decryption time, security level, memory usage on encryption and memory usage on decryption.

Fig. 4 shows the comparative graph for the proposed HAES with the existing DES, RC4, Blowfish and the AES algorithm based on encryption time. The encryption time is considered as the time that an encryption algorithm takes to produces a ciphertext from plain text. Here, the encryption time varies based on data obtained from a number of sensor nodes. When the sensor node count is 100, the proposed takes 1.0178 s time to encrypt the data. But the existing Blowfish, DES, RC4 and AES take 7.278, 5.892, 4.324 and 2.897 s of time respectively. For the 500 nodes also, the proposed takes 3.023 s, the existing algorithms Blowfish, DES, RC4 and AES takes 14.456, 10.673, 9.874 and 8.541 s of time respectively. Here, the proposed takes less time, the existing blowfish and DES takes more time to encrypt the data. Similarly, for the remaining node counts also, the proposed takes less time to encrypt the data. Thus, it deduces that the HAES attain better performance than the existing research methods.

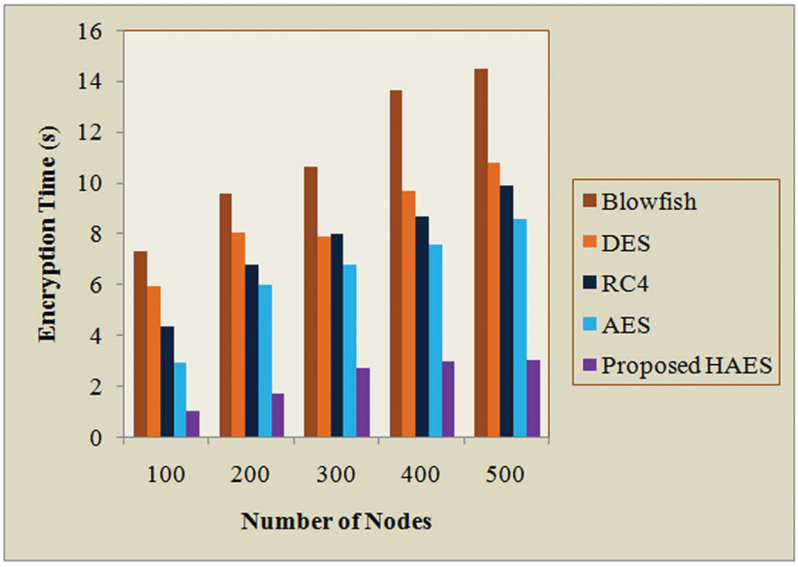

Figure 4: Compared the performance of the proposed with existing algorithms based on encryption time

Fig. 5 displays the decryption time analysis for the HAES algorithm with the existing cryptographic algorithms. Decryption time refers to the time taken for converting the ciphertext into plain text. Thus, the decryption time is calculated by taking subtraction between the decryption ending time and decryption starting time. The less time of system ensures the better performance of the system. In this way, the HAES algorithm takes less time to decrypt the data. Here the performance is analyzed for the particular set of data from the number of sensor nodes. In this, the performance analysis is done for the range of 100 sensor nodes to 500 sensor nodes. For 500 sensor node count the HAES takes 4.004 s to decrypt the data, whereas the existing Blowfish, DES, RC4 and AES have 14.756, 10.733, 8.374 and 8.841 s of time respectively. Here, the existing algorithms take more time to decrypt the data than the proposed algorithm. For the remaining sensor nodes count also the proposed takes less time to decrypt the input data. Thus, the description proves the improved performance of the proposed HAES algorithm.

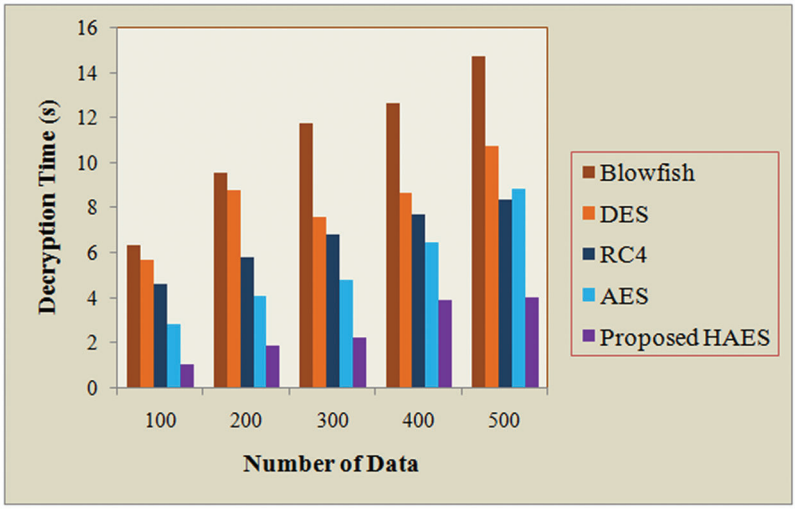

Figure 5: Decryption time analysis for the HAES and the existing algorithms

Tab. 2 analyzes the security level of the proposed HAES with the AES, RC4, DES and Blowfish algorithms. Security is very important for cloud storage. The security level is defined as the hacked data divided by the number of the original text. The security level of the HAES is 97.89% and the security level of the AES is 96.56%, RC4 is 93.75%, DES is 90.76% and Blowfish is 88.66%. Here, the security level of the HAES is higher than the other algorithms. The existing blowfish provides very low-level of security than the other existing algorithms and the proposed algorithm. The existing AES algorithm is much better than the other existing algorithms and it also has the lowest security than the HAES. Hence, the security level analysis proves that the HAES performance is better than the other algorithms. The graphical representation of the Tab. 2 is shown in Fig. 6.

Table 2: Security analysis

Algorithms	Security level (%)
Proposed HAES	97.89
AES	96.56
RC4	93.75
DES	90.76
Blowfish	88.66

Fig. 7 displays the CPU memory usage on encryption time for the HAES with the AES, RC4, DES and Blowfish algorithms. Different algorithms occupy various size of memory to proceed with the process. Thus, the memory requirements are based on the number of operations performed by the specific algorithms. If the algorithm takes less amount of space then that algorithm is considered to be the best algorithm for encrypting the data. Here, the performance is also analyzed based on the number of sensor nodes. For all the node count such as 100, 200, 300, 400 and 500 the proposed algorithm occupies 5577455, 5848796, 6125897, 6529784 and 6735587 KB of memory respectively. If the node count is increased then the algorithm occupies more space. But the existing algorithms occupy more space for all the sensor node counts than the proposed algorithm. Thus, the proposed algorithm occupies less memory space than the existing algorithms.

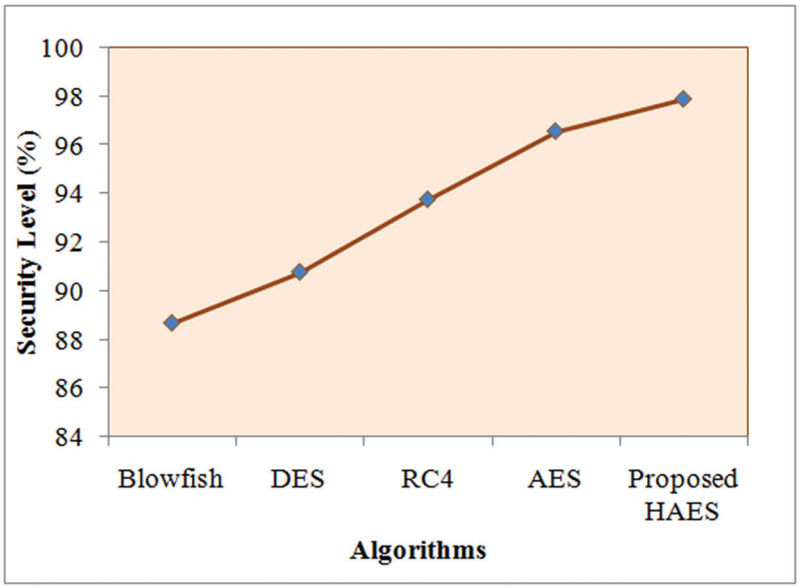

Figure 6: Demonstrates the security level of the proposed and existing algorithms

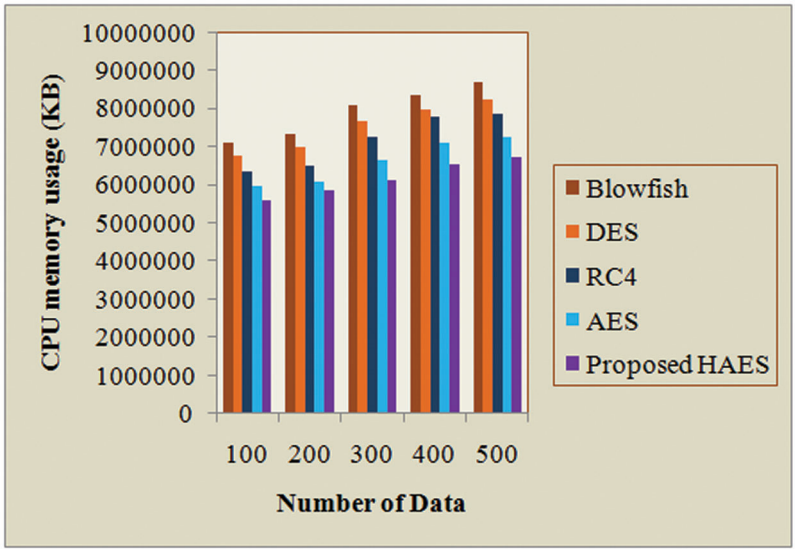

Figure 7: Illustrate the CPU memory usage on encryption time for the HAES with existing algorithms

Fig. 8 denotes the graphical representation analysis of CPU memory usage on decryption for the HAES and the existing algorithms. When the node count is 400, the existing algorithms AES, RC4, DES and Blowfish algorithms occupy 7366257, 7687854, 8124577 and 8578455 KB of memory respectively during the decryption time. But the proposed AHES occupies 6657845 KB memory at the decryption process. For the remaining node counts also, the proposed algorithm occupies less memory space than the other existing algorithms. Thus, the proposed provides better performance than the existing research methodologies.

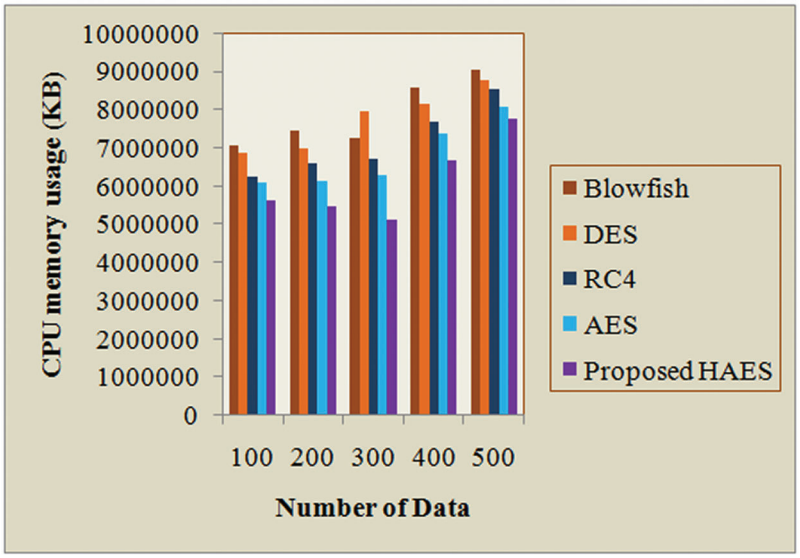

Figure 8: CPU memory usage on decryption time analysis for the proposed and existing algorithms

4.4 Performance Analysis for Attribute Selection

Here, the performance of the proposed CCCOA algorithm is analyzed with the existing Mayfly Optimization Algorithm (MOA), Rat Swarm Optimization Algorithm (RSOA), Deer Hunting Optimization Algorithm (DHO) and Chimp Optimization Algorithm (COA).

Fig. 9 displays the fitness *versus* iteration analysis for the CCCOA with the other existing algorithms. Here, 5, 10, 15, 20 and 25 iteration levels are considered. At the 25th iteration level, the fitness level of the proposed CCCOA is 8240 and the fitness level of the existing MOA, RSOA, DHO and COA is 6247, 6874, 7214 and 7807 respectively. Here, the fitness level is higher for the CCCOA when compared with the other algorithms. Similarly, for the remaining iteration level, the fitness level is higher than the existing algorithms. It is concluded that the CCCOA based attribute selection attains the better performance than the existing research algorithms.

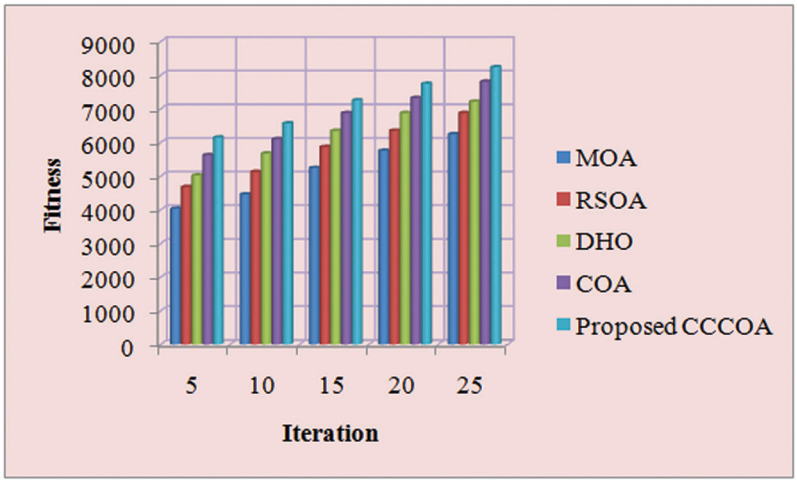

Figure 9: Fitness *vs*. Iteration

5 Conclusion

The IoT is the most recent Internet evolution that integrates many smart devices. However, the IoT has an enormous threat to security and privacy. Hence, the authentication process is most important for ensuring the security level. The proposed method is divided into two phases namely, the data owner phase and the data user phase. The data owner phase consists of two processes such as authentication and secure data uploading. In that authentication, the registration process has the KGC, SCC and DVC phases. In KGC, the 128-bit hash key is generated by using PoC-FNV in SCC the signature is created by the generated hash value in DVC, the DAP is created by the selected attributes thus the attributes are selected by using CCCOA. In performance analysis, the performance of the proposed algorithms is analyzed with the existing algorithm. The performance analysis is done by three steps namely (a) hash key generation, (b) secure data uploading and (c) attribute selection. In the hash key generation, the performance of the PoC-FNV is compared with the FNV, spooky hash, MD5 and RIPEMD algorithms in terms of hash code generation time. The proposed PoC-FNV takes 1022 ms time to generate the hash key, which is lesser than the other algorithms. In the secure data uploading analysis, the performance of the HAES is compared with the existing DES, RC4, Blowfish and the AES algorithm based on encryption time, decryption time, memory usage on encryption time, memory usage on decryption time and security level. The security level of the HAES is 97.89%, which is higher among the other existing algorithms. In attribute selection, the performance of the CCCOA is compared with the existing MOA, RSOA, DHO and COA, in which the proposed CCCOA attains better fitness. Hence, the suggested work provides the better result in the key management of internet-based lightweight devices in IoT network. In future, the proposed method can be extended by using distributed key management process and advanced algorithms to improve the security level.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] P. M. Chanal and M. S. Kakkasageri, "Security and privacy in IoT: A survey," *Wireless Personal Communications*, vol. 115, no. 2, pp. 1667–1693, 2020.

[2] J. Tournier, F. Lesueur, F. L. Mouël, L. Guyon and H. Ben-Hassine, "A survey of IoT protocols and their security issues through the lens of a generic IoT stack," *Internet of Things*, vol. 26, no. 2, pp. 100264, 2020.

[3] R. Palanivelu and P. S. S. Srinivasan, "Safety and security measurement in industrial environment based on smart IOT technology based augmented data recognizing scheme," *Computer Communications*, vol. 150, no. 2, pp. 777–787, 2020.

[4] M. Wazid, A. K. Das, S. Shetty, J. J. P. C. Rodrigues and Y. Park, "LDAKM-EIoT: Lightweight device authentication and key management mechanism for edge-based IoT deployment," *Sensors*, vol. 19, no. 24, pp. 5539, 2019.

[5] B. Karthikeyan, T. Sasikala and S. B. Priya, "Key exchange techniques based on secured energy efficiency in mobile cloud computing," *Applied Mathematics & Information Sciences*, vol. 13, no. 6, pp. 1039–1045, 2019.

[6] V. D. A. Kumar, K. Abhishek and C. Veluvolu, "Active volume control in smart phones based on user activity and ambient noise," *Sensors*, vol. 20, no. 15, pp. 4117, 2020.

[7] L. Ding, Z. Wang, X. Wang and D. Wu, "Security information transmission algorithms for IoT based on cloud computing," *Computer Communications*, vol. 155, no. 1, pp. 32–39, 2020.

[8] D. Paulraj, "A gradient boosted decision tree-based sentiment classification of twitter data," *International Journal of Wavelets, Multiresolution and Information Processing, World Scientific*, vol. 18, no. 4, pp. 1–21, 2020.

[9] C. Li and C. Yang, "(WIP) authenticated key management protocols for internet of things," in *IEEE Int. Congress on Internet of Things (ICIOT)*. IEEE, pp. 126–129, 2018

[10] J. Gokul Anand, "Trust based optimal routing in MANET's," in *Int. Conf. on Emerging Trends in Electrical and Computer Technology*, pp. 1150–1156, 2011.

[11] P. Subbulakshmi and M. Prakash, "Mitigating eavesdropping by using fuzzy based MDPOP-q learning approach and multilevel Stackelberg game theoretic approach in wireless CRN," *Cognitive Systems Research*, vol. 52, no. 4, pp. 853–861, 2018.

[12] P. Chintan and D. Nishant, "Cryptanalysis of ECC-based key agreement scheme for generic IoT network model," in *Int. Conf. on Computing, Communication and Networking Technologies (ICCCNT)*, IEEE, pp. 1–7, 2019.

[13] P. Mohan and S. Chelliah, "An authentication technique for accessing de-duplicated data from private cloud using one time password," *International Journal of Information Security and Privacy (IJISP)*, vol. 11, no. 2, pp. 1–10, 2017.

[14] M. Mingxin, S. Guozhen and L. Fenghua, "Privacy-oriented blockchain-based distributed key management architecture for hierarchical access control in the IoT scenario," *IEEE Access*, vol. 7, pp. 34045–34059, 2019.

[15] O. Pal, B. Alam, V. Thakur and S. Singh, "Key management for blockchain technology," *ICT Express*, vol. 7, no. 1, pp. 76–80, 2019.

[16] S. Neelakandan, "Large scale optimization to minimize network traffic using map reduce in big data applications," in *Int. Conf. on Computation of Power, Energy Information and Communication (ICCPEIC)*, pp. 193–199, 2016.

[17] H. Seyoung, C. Sangrae and K. Soohyung, "Managing IoT devices using blockchain platform," in *Int. Conf. on Advanced Communication Technology (ICACT)*, IEEE, pp. 464–467, 2017.

[18] R. Victor, H. Raimir, R. Alex and J. P. C. Rodrigues, "Enhancing key management in LoRaWAN with permissioned blockchain," *Sensors*, vol. 20, no. 11, pp. 3068, 2020.

[19] M. Sana, K. Ahmad, S. Zanab, S. Kalsoom, A. Ejaz *et al.,* "Securing IoTs in distributed blockchain: Analysis, requirements and open issues," *Future Generation Computer Systems*, vol. 100, no. 15, pp. 325–343, 2019.

[20] M. Reem, N. N. Hassan and C. Ali, "Lightweight multi-factor mutual authentication protocol for IoT devices," *International Journal of Information Security*, pp. 1–16, 2019.

[21] A. Bander, A. C. Shehzad, B. Ahmed, X. Wenjing, C. Min *et al.,* "ILAS-IoT: An improved and lightweight authentication scheme for IoT deployment," *Journal of Ambient Intelligence and Humanized Computing*, vol. 14, no. 4, pp. 1–13, 2020.

[22] G. P. Deepsubhra, D. Puja, D. Debashis and B. Rajkumar, "QoS-aware secure transaction framework for internet of things using blockchain mechanism," *Journal of Network and Computer Applications*, vol. 144, no. 2, pp. 59–78, 2019.

[23] B. D. Deebak, "Lightweight authentication and key management in mobile-sink for smart IoT-assisted systems," *Sustainable Cities and Society*, vol. 63, no. 1, pp. 102416, 2020.

[24] H. Khalid, I. Naveed, S. Tanzila, R. Amjad and M. Zahid, "LSDAR a light-weight structure based data aggregation routing protocol with secure internet of things integrated next-generation sensor networks," *Sustainable Cities and Society*, vol. 54, no. 2, pp. 101995, 2020.

[25] S. Neelakandan and D. Paulraj, "An automated exploring and learning model for data prediction using balanced CA-SVM," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 5, pp. 4979–4990, 2021.