Wireless Personal Communications (2024) 136:1675–1703
https://doi.org/10.1007/s11277-024-11353-3

End-to-End Self-organizing Intelligent Security Model for Wireless Sensor Network based on a Hybrid (AES–RSA) Cryptography

V. Anusuya Devi[1] · T. Sampradeepraj[1]

Accepted: 13 June 2024 / Published online: 25 June 2024

Abstract

In wireless sensor networks, ensuring data confidentiality and integrity is paramount, particularly in sensitive environments. However, existing encryption methods encounter challenges, including key maintenance in symmetric encryption and lower security levels in asymmetric encryption. To address these issues, this paper proposes a novel security protocol employing a hybrid encryption approach, combining both symmetric (AES) and asymmetric (RSA) cryptographic techniques. The protocol introduces a two-phase cryptographic technique utilizing AES for robust encryption in Phase 1 and RSA for efficient key management in Phase 2. Experimental results demonstrate enhanced efficiency in encryption, decryption, and total execution time compared to existing algorithms, marking a significant advancement in cryptographic protocols for improved security in data transmission and management. Key contributions include the division of plaintext for simplified cryptographic processing, AES encryption for Phase 1, RSA encryption for Phase 2, and a seamless decryption process ensuring data integrity and security against attacks.

Keywords Cryptography · Hybrid cryptographic techniques (HET) · Symmetric key cryptography · Asymmetric key cryptography · Advanced encryption standards (AES) · Rivest–Shamir–Adleman (RSA) · Wireless sensor network (WSN)

1 Introduction

1.1 Wireless Sensor Network

A wireless network comprises numerous sensor nodes responsible for performing computations, sensing, and communication between each sensor and the base station[1]. WSNs find application across diverse fields including environment monitoring, traffic surveillance, building structure monitoring, military sensing and information gathering,

✉ V. Anusuya Devi
samanusuya23508@gmail.com

[1] Department of Computer Science and Engineering, School of Computing, Kalasalingam Academy of Research and Education, Krishnankoil, Tamilnadu 626126, India

habitat monitoring, wildfire detection, pollution monitoring, and more. Figure 1 illustrates the structure of a WSN.

The sensing unit comprises the physical sensors, serving as the sole interface to external environments. Within the processor unit reside microprocessors responsible for processing data received from the sensing unit. Due to limited storage capacity, memory is categorized into two types: user memory, utilized for storing personal data-related applications, and program memory, used for device programming and data identification purposes. Power for each sensor node is typically sourced from either the electric utility or a battery.

The communication unit incorporates a short-range transceiver responsible for transmitting and receiving data within the resource-constrained environment. Due to limited battery energy, the transceiver employs a hop-by-hop mechanism rather than the end-to-end mechanism used in legacy IP networks.

1.2 Symmetric Key Cryptography

Symmetric key cryptographic techniques rely on a single secret key for both encryption and decryption processes [2, 3]. However, if this key is lost, attackers can compromise the entire security of the system. Figure 2 illustrates the symmetric key cryptographic process.

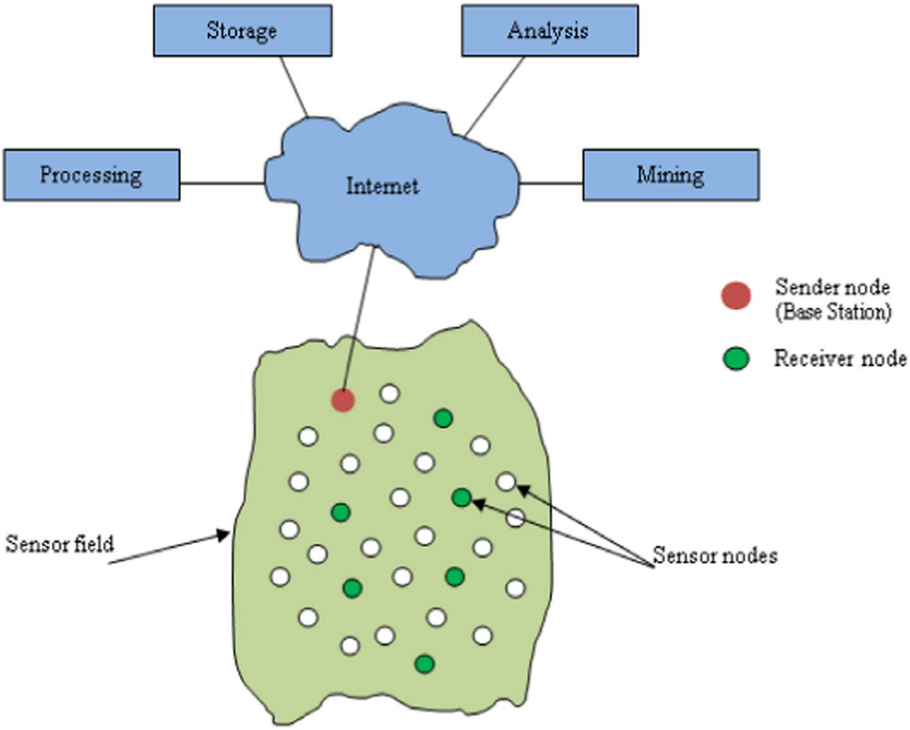


Fig. 1 Basic structure of wireless sensor network

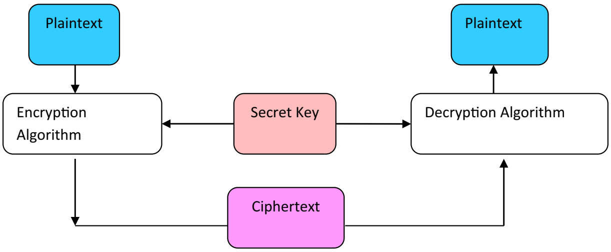


Fig. 2 Symmetric key cryptography

1.3 Advanced Encryption Standard (AES)

AES, a symmetric key cryptography technique, operates by dividing plaintext into blocks and subjecting these blocks to multiple transformations to generate ciphertext [13–15]. This process involves ten rounds applied to the plaintext, each comprising operations such as add round key, sub bytes, shift rows, and mix columns. However, the tenth round comprises only three operations: add round key, sub bytes, and shift rows. Each data block undergoes encryption through ten rounds of operations.

The encryption process involves the following steps:

1. Derive round keys from the key expansion steps.
2. Represent the plaintext as a state array.
3. Add the initial round key to the state array.
4. Manipulate the state array using sub bytes, shift rows, and mix columns in nine rounds.
5. Perform the tenth-round manipulation to encrypt the plaintext into ciphertext.

Figure 3 illustrates the AES encryption process, where the original message undergoes conversion into ciphertext through the execution of ten round functions.

1.4 Asymmetric Key Cryptography

Asymmetric key cryptographic techniques utilize two keys for encryption and decryption processes [4, 8, 10]. The key distribution among communicating parties poses a significant challenge. However, compared to symmetric key cryptographic processes, asymmetric key cryptography offers enhanced security across diverse applications. Figure 4 depicts the asymmetric key cryptographic process.

1.5 RSA

RSA is a form of asymmetric key cryptography, utilizing two keys for encryption and decryption [4, 8, 10–12]. The cryptographic process unfolds as follows:

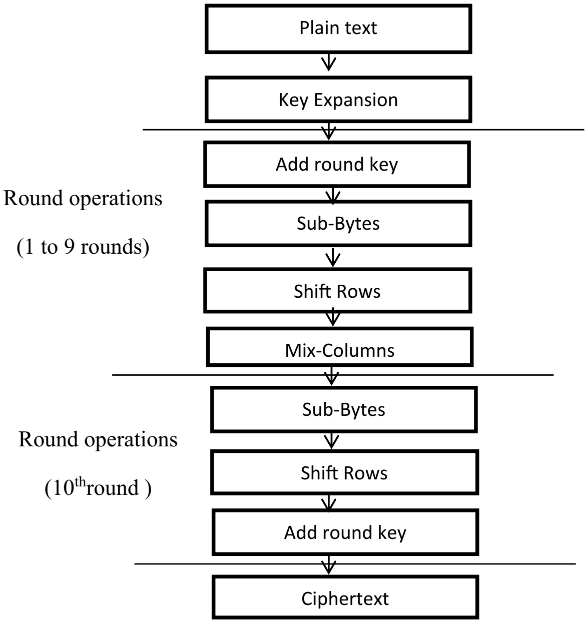


Fig. 3 AES—operations

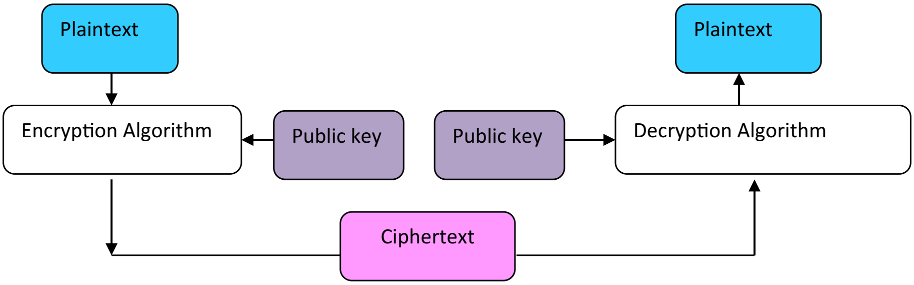


Fig. 4 Asymmetric key cryptography

Step 1: Choose two distinct prime numbers p and q.

Step 2: Compute their product $N = p * q$.

Step 3: Calculate Euler's totient function $\varphi(N) = (p - 1) * (q - 1)$.

Step 4: Choose an integer e such that $1 < e < \varphi(N)$ and $\gcd(e, \varphi(N)) = 1$.

Step 5: Compute the modular multiplicative inverse d of e modulo $\varphi(N)$, i.e., $d \equiv e^{\wedge}(-1) \bmod \varphi(N)$.

Step 6: The public key is (N, e) and the private key is (N, d).

Step 7: Encryption: To encrypt a plaintext message m, compute ciphertext c as $c \equiv m^{\wedge}e \bmod N$.

Step 8: Decryption: To decrypt ciphertext c, compute plaintext m as $m \equiv c^{\wedge}d \bmod N$.

1.6 Motivation and Contribution of this Work

The motivation for this work stems from the critical need to enhance security in wireless sensor networks, especially in sensitive environments where data confidentiality and integrity are paramount. Existing encryption methods face limitations in key maintenance and security levels, necessitating the development of a novel hybrid encryption approach to address these challenges effectively.

The primary contributions of this paper are as follows: (i) In Phase 1, the plaintext is divided into plaintext1 and plaintext2 to simplify the cryptographic process. (ii) AES encryption process is applied to plaintext1. (iii) RSA encryption process is applied to the remaining part of plaintext2. (iii) The resulting ciphertexts, ciphertext1 and ciphertext2, are transmitted to the receiver end. (iv) In Phase 2, ciphertext1 is decrypted into plaintext1 using the AES decryption process. (v) Ciphertext2 is decrypted into plaintext2 using the RSA decryption process. (vi) The plaintext1 and plaintext2 are combined to produce the original plaintext at the receiver end, ensuring security against attacks.

The remainder of this paper is structured as follows: Sect. 2 delves into a discussion of various hybrid security algorithms. Section 3 elucidates the operational workflow of the proposed model. The performance of the proposed system is scrutinized in Sect. 4. Finally, Sect. 5 presents concluding remarks on the proposed method and explores potential avenues for future research.

2 Related Work

Madhumita Panda [2] analyzed the suitable public key cryptography for wireless sensor networks to ensure the security in the wireless network concerning limitation of energy, computation capability and storage resources of the sensor nodes. The author compared two public key cryptography techniques RSA and Elliptic Curve Cryptography (ECC). Finally, the author concluded that ECC is suitable for WSN because it can reduce the computation time for data transmission.

Anusuya Devi et al. [3] proposed a new hybrid cryptography technique message authentication code and modified and enhanced lattice-based cryptography (MAC-MELBC) to ensure the security of the data transmission in the wireless body sensor networks. The merit of the proposed scheme is partitioning the plaintext into two parts, and two cryptographic algorithms MAC and MELBC, is used. Demerit of the system may include more complexity for making the plaintext into ciphertext.

Mallikarjunaswamy et al. [4] introduced an m-RSA for the encryption/decryption of data in wireless sensor networks. The advantages of this proposed scheme are execution time and security achieved by the wireless sensor network is more effective than other existing systems. Confidentiality about the data transmission is increased. The demerit of this method is that it can include more complexity in the encryption and decryption process.

T. Sampradeepraj [5] et al. proposed an efficient algorithm Minimum Connected Dominating Set for group communication in the wireless sensor network. The authors are used Random Linear Network Coding over Minimum Connected Dominating Set in On-Demand Multicast Routing Protocol to improve the reliability and throughput. The merit of the proposed scheme is to reduce the multiple numbers of transmissions in the wireless network. The authors may consider security parameters for data transmission in the wireless link.

BalasubramanianPrabhuKavin et al. [6] introduced a new hybrid cryptographic algorithm called Elliptic-Curve Cryptography and Diffie-Hellman based data security algorithm. The advantage of this hybrid algorithm is to secure the M-commerce data in fewer periods. It can establish the session key to produce more security in the insecure channel. The disadvantage of the author can design a lightweight cryptographic algorithm to secure the data transmission in the M-commerce field.

BalasubramanianPrabhuKavin et al. [7] proposed a new cloud-security framework. The authors used new elliptic curve cryptography based key generation algorithm to restrict the user's access to the cloud. The proposed scheme time consumed for the cryptographic function is less than the RSA. The authors may reduce the complexity of the cryptographic algorithm.

B. Sreevidya et al. [8] proposed a new security scheme to achieve minimal energy consumption for wireless sensor network-based applications. The proposed method ensures the original data in the WSN. It prevents incorrect data injection in the WSN. Advantages of the scheme are efficient reactive routing method, the authenticity of the nodes and the integrity of the data. The disadvantage of this method is routing do not depend on any updates in the activities and time.

Majid R et al. [9] used the 128-bit advanced encryption standard (AES-128) for encrypting and validating the transmitted data in the wireless sensor networks. The authors implemented the resource-constrained security algorithm. The merit of the methods is to consider the various parameters like energy consumption, storage overhead, key connectivity, replay attack, man-in-the-middle attack, and resiliency to node capture attack. The running time complexity of the proposed scheme is high compared to other existing systems.

SangeetaPatil et al. [10] proposed a clustering-based technique for routing in the wireless sensor network from the base station to sensor nodes. The advantages of this method are using the RSA algorithm to main the key and time schedule of the WSN. Route updating, packet generation, packet encryption, decryption methods are used to ensure the security of the networks. Using RSA for encryption and decryption will reduce the energy requirements of the WSN. The disadvantage of the system is delay parameter may be reduced.

Maha Salah Asaadet al. [11 proposed an adaptive cryptographic scheme for wireless sensor networks. This scheme operates on routing ad hoc on-demand vector routing (AODV) protocol. This scheme's merit is reduced complexity of the RSA algorithm. This light asymmetric cryptography technique of RSA operates security operations on sensor nodes with a low power ratio.

UtkuGulen et al. [12] implemented an RSA algorithm for a wireless sensor network. Here authors considered two key distribution problems in cryptography (i) for securing communication, distribution of shared private key among WSN nodes as a problem. (ii) Key distribution problem in the public key cryptography. The authors are used optimized arithmetic, low-level coding, and acceleration algorithms to reduce the RSA key generation, encryption, decryption time. The authors may consider larger key sizes for the cryptographic process in the WSN.

KavanA N et al. [13] introduced a simplified AES algorithm for wireless sensor networks. This proposed scheme is considered the sensor nodes' memory and processing speed to secure the communication in the wireless sensor network. The authors designed an S-AES ideal Cryptographic algorithm for a resource-constrained WSN environment. Cryptanalysis can operate on the S-AES.

SreenathThangarajan et al. [14] proposed a new power reduced scheme for AES operations. The authors used minimized energy by the parallel processing and reduced redundant hardware to secure the communication with high speed and minimum power. The advantage of this method is that it improves the speed as 2.04 times more than existing systems.

Dan Dragomir et al. [15] designed an AES algorithm with improved execution speed and with higher energy consumption. The authors used five modes of optimized operation for encryption and two modes for authentication to secure communication in the WSN. This scheme's disadvantage is the software implementation of the scheme is not done.

Meeta Gupta et al. [16] considered the substitution box lookup table in AES to reduce a time-consuming process. The authors improve the running time by splitting AES S-box. The proposed scheme uses parallel substituting in a hierarchical multicore sensor network. The advantages are the lifetimes of the sensors are improved, and data freshness is maintained.

Sangwon Shin et al. [17] proposed using pre-processed symmetric RSA for message authentication in WBAN (PSRSA). PSRSA is a variant of RSA that uses a minor key and numerous levels of pre-processing to expand the number of possible choices. PSRSA is being used to see if it's feasible and test a few situations. PSRSA with MAC allows RSA authentication with less data transmission than symmetric RSA. Because of the security frequency issue, the hash function can be used for authentication to improve PSRSA's security level.

Ramadan, M et al. [18] suggested an efficient and safe identity-based encryption system with an equality test based on the RSA assumption. As a consequence, our system outperforms the previously presented strategies in terms of computing performance and, more crucially, compatibility with WBAN applications. The suggested method is efficient and safe against one-way secure, chosen-identity, and chosen-ciphertext attacks, according to the performance assessment. However, when it comes to obstacles, one of the most significant is the lack of security in all WBAN systems.

AartiSangwan et al. [19] suggested a security model for the clustered WBAN network deployed in an integrated manner to increase communication dependability. RSA and hashing algorithms are coupled with specific roles to accomplish the security levels. The simulation is run on WBAN clusters with varying numbers of WBANs. The findings demonstrate that the suggested paradigm significantly increased packet communication and network life. The network's performance has increased thanks to the suggested secure communication paradigm. A comparison study is performed in terms of dead node analysis, packet communication, and network energy observation to assess network reliability and performance.

Singla R et al. [20] proposed an energy-efficient routing protocol (CSEER) has been designed for highly secure data transport in WBAN. The recommended technique employs two approaches. An arithmetic data compression technique was applied to compress patient data and add an encryption layer. The RSA technique was then utilized to encrypt the sensitive patient data securely. The CSEER regimen saves 10–11 percent more energy than Rel-AODV, according to the data, and is safe to use in medical settings. Furthermore, the CSEER protocol has a 3% greater throughput and a 10% lower packet loss rate than Rel-AODV in terms of transmission power. However, there is a disadvantage in security, which must be improved.

Sangwon Shin et al. [21] suggested a space-capable WBAN system that uses preprocessed symmetric RSA for message authentication (PSRSA). More potential variants of RSA encoded authentication messages will be created if the user can control the preprocessing. Compared to 1024 bit RSA, PSRSA is projected to result in a key size reduction of up to 256 bits while producing more real possibilities with no performance loss. PSRSA enables nodes to have reliable RSA message authentication while reducing processing time to a minimum. However, the tremendous computational power and complexity are a challenge to overcome.

UtkuGulen et al. [22] suggested that when efficient arithmetic, low-level coding, and various acceleration methods are implemented, RSA can be used in wireless sensor networks. On the limited MSP430 microcontroller, 1024-bit RSA is developed to solve the key distribution problem in wireless sensor networks. On the MSP430, the implementation completes 1024-bit RSA encryption and decryption operations in under 0.047 s and 1.14 s, respectively. The quickest in the literature are the timings for 1024-bit RSA encryption and decryption procedures for the MSP430 microcontroller. However, the suggested framework is designed to deal with high levels of complexity.

PriyabrataSatapathy et al. [23] proposed that NFC stands for Near Field Communication and is based on the idea of Radio Frequency Identification (RFID). The key splitting into two shares using the mediated RSA method must be determined. Identity-based encryption (IBE) is a type of open-key encryption that may be customized. The novel combination of NFC and SIM can enable the fulfillment of a slew of new use cases, and UICC has unlocked a world of possibilities. It offers the recipient additional convenience and security.

ArunaDeepthi et.al. [24] suggested a MATLAB implementation of the Low Energy Adaptive Clustering Hierarchy (LEACH). RSA encryption is used to provide secure data transit between nodes in a wireless sensor network. The paper's primary flaw is its lengthy processing time.

G. Leelavathi et al. [25] proposed a key size of 128 bits, the design and modeling of two distinct RSA encryption and decryption public-key systems are shown. The findings show that the updated MMM42 multiplier implementation is not suitable for WSN nodes since it requires 50% more hardware in the FPGA.

Surekha et al. [26] proposed that ELGAMAL and RSA implemented for wireless sensor for providing secure communication. In cluster-based wireless network topology environment, the performance is evaluated using ELGAMAL and RSA.The advantage of this work is less computation and good storage capacity.Moustafa M. Nasrall [26]introduceda novel methodology to identify the malfunctioning IoT devices to develop rehabilitation systems with virtual smart cities using time series analysis. The proposed method used self-management for IoT devices. The system uses an alternative IoT sensor to capture the missing data if any IoT device fails and also uses a novelty detector programmed with Python using the Scikitlearn framework to analyse the current data to determine whether it is probably malfunctioning. The proposed method's advantage is that time series automatically identify malfunctioning IoT devices.

Vankamamidi S. Naresh[27] proposed a three-tier architecture for a smart healthcare city. A multi-agent system is included in the proposed architecture. Communicating parties in this system can share their secret key using a quantum group key agreement based on quantum physics. Quantum Diffiehellmen key agreement is established between the communicating groups. This scheme's advantage is protecting healthcare data against various quantum attacks. Multi-agent systems in e-healthcare can increase the efficiency of patient health and quality of care, and it improves the education of physicians' online.

DrishtySobnath, Ikram Ur Rehman et al. [28] have provided a review and recommendations on how a smart city can offer better Quality of Life for visually impaired people (VIP).The authors reviewed VIP's ability to recognize people and how they can lead and interact with society. They reviewed VIP's routine activities using IoT and smartphones. With the development of ICT infrastructure, smartphones, wearable devices, Artificial

Intelligence (AI), Internet of Things (IoT), and Virtual and Augmented Reality (VR)/(AR), the VIP's life quality is improved.

SwarnAvinash Kumar et al.[29] proposed self-care programs with patients' artificial intelligence ecosystems. It can help doctors and patients to interact in an agile way. The authors included the machine-learning module for each mobile agent to know each patient's health. The authors used two agent-based simulators. This can control the heart rate and reduce negative emotions and stress. The disadvantage of this method maybe includes the doctor's feedback, which can help in selecting and guiding the self-care programs.

Muhammad AltafKha[30] et al. introduced an efficient multilevel probabilistic model for detecting abnormal traffic in wireless sensor networks. The authors used a Bayesian model to detect abnormal data traffic and discriminate distributed denial of service (DDoS) attacks from the flash crowd (FC). This method achieved more than 93% accuracy in detecting the DDoS attack. The advantage of this proposed system is using payload patterns and hop count information to build a strong detection system. This system may include prevention module c to the detection module to prevent all the attacker nodes in the network.

Mohit Kumar [31] et al. have also contributed to the discourse on optimizing cloud-fog-based environments for IIoT applications. They have proposed an AI-enabled intelligent and sustainable framework to address the challenges associated with latency-sensitive tasks in industries such as automotive, robotics, oil and gas, smart communications, and Industry 5.0. Their framework integrates fog computing to bring computational resources closer to the edge of the network, facilitating real-time offloading decisions tailored to the demands of IIoT applications.

In their approach, a fuzzy-based offloading controller is utilized to analyze offloading decisions, considering factors such as latency sensitivity and Quality-of-Service (QoS) parameters. Additionally, they incorporate an AI-based Whale Optimization Algorithm (WOA) to search for optimal resources and make accurate offloading decisions, thereby improving various performance metrics.Experimental results presented by authors, demonstrate significant enhancements in makespan time, energy consumption, and execution cost compared to benchmark offloading and allocation schemes. This underscores the importance of leveraging AI techniques to optimize cloud-fog-based environments for IIoT applications, addressing latency concerns and improving resource management efficiency.

Ananya Chakraborty [32] et al. have contributed to the advancement of security in cloud-fog environments for IoT applications. They recognize the critical importance of efficient offloading strategies to address latency-sensitive tasks while acknowledging the susceptibility of fog computing environments to security threats. Despite numerous security-based frameworks proposed in the literature, many fail to comprehensively address these challenges.

To fill this gap, authors propose a novel lightweight secure framework leveraging Deep Learning (DL) techniques for security attack detection and network traffic monitoring in IoT applications. Their approach involves deploying an Artificial Neural Network (ANN)-based model on a cloud platform and implementing the detection mechanism on fog nodes to minimize vulnerabilities and ensure timely response. Validation of the proposed framework is conducted using NSL-KDD datasets, with comparative evaluations against baseline techniques such as Support Vector Machine (SVM), Logistic Regression, and Decision Trees. Experimental results demonstrate the superior performance of the proposed framework, achieving a remarkable accuracy of 99.43%, precision of 99.26%, and a minimal False Alarm rate of 0.7396%.These findings highlight the effectiveness of Ananya Chakraborty [32] et al.'s approach in enhancing security in cloud-fog environments for

IoT applications. By leveraging Deep Learning techniques, their framework demonstrates robust defense against potential attacks while surpassing baseline algorithms in terms of Quality-of-Service (QoS) parameters, thus offering promising solutions to address security vulnerabilities in integrated cloud-fog environments.

Device-to-device communication has garnered significant attention from the research community due to its diverse range of applications and its potential to provide connectivity in low connectivity areas without relying on base stations. In this context, Prashant Kumar et al. [33] have proposed a new technique called Clustering Based Opportunistic Traffic Offloading (CBOT) for facilitating Device-to-device communication.

The CBOT technique introduced by Prashant Kumar et al. divides the network into small clusters and employs a hybrid scheme for data transmission. To enhance energy consumption and throughput, the authors have proposed cluster formation, cluster head selection, and rotation techniques. Simulation results presented in their study indicate that CBOT significantly improves network lifetime by achieving high energy efficiency and enhances system performance by achieving higher throughput.Furthermore, the performance of CBOT is evaluated under opportunistic network scenarios, and the simulation results demonstrate its superiority over existing approaches in terms of energy consumption and throughput. Prashant Kumar et al.'s [33] work contributes to advancing the field of Device-to-device communication by introducing an efficient and effective technique that improves energy efficiency and throughput in network environments, particularly in low connectivity areas.

Cloud computing has become a preferred choice for researchers owing to its versatile services, applications, and advantages. However, this accessibility of resources over the internet also introduces risks, as data transmission occurs over unsecured networks. This vulnerability is exacerbated by various cyber-attacks that threaten the confidentiality, integrity, and availability (CIA) of systems connected via the internet. Among these threats, Distributed Denial of Service (DDoS) attacks are particularly concerning due to their rapidly increasing frequency and sophistication.DDoS attacks pose a significant challenge because they can mimic legitimate traffic, making them difficult to detect. To mitigate these challenges, researchers have turned to implementing Intrusion Detection Systems (IDS). These systems play a crucial role in identifying and responding to suspicious activities within a network, including DDoS attacks.In this context, the authors Om Prakash Suman et al. [34] focus specifically on DDoS attacks and propose the implementation of state-of-the-art Machine Learning (ML) approaches to analyze performance. By leveraging ML techniques, such as anomaly detection or pattern recognition, researchers aim to enhance the capability of IDS to detect and mitigate DDoS attacks effectively.This article serves as a valuable resource for new researchers venturing into the field of cloud security, providing insights into the current challenges posed by DDoS attacks and the potential of ML-based approaches to address them. By exploring the intersection of cloud computing, cybersecurity, and machine learning, this work contributes to advancing our understanding and capabilities in safeguarding cloud-based systems against evolving threats.With the rapid development of mobile internet services, driven by applications such as augmented/virtual reality and vehicular networks, the demand for computational resources on mobile terminals is escalating. To address this demand, mobile edge computing (MEC) has emerged as a solution for offloading tasks to the edge of cellular networks. However, offloading remains a challenging issue due to the dynamism and uncertainty of upcoming Internet of Things (IoT) requests and the state of the wireless channel. Additionally, ensuring the security of offloaded data adds computational complexities to the process, necessitating a secure and efficient offloading technique.

In response to these challenges, researchers, including Jitendra Kumar Samriya et al. [35], have proposed a reinforcement learning-based Markov decision process offloading model. This model aims to optimize energy efficiency and mobile users' time while considering the constrained computation of IoT devices and ensuring efficient resource sharing among multiple users. Furthermore, the proposed approach employs the Advanced Encryption Standard (AES) to meet the security requirements of offloaded data.Simulation results presented in the study demonstrate that the proposed approach outperforms existing baseline models in terms of offloading overhead and service cost, as well as ensuring secure data offloading. By integrating reinforcement learning and Markov decision processes, the proposed model offers a promising solution for optimizing offloading decisions in dynamic and uncertain environments while addressing security concerns. This research, led by Jitendra Kumar Samriya et al., contributes to advancing the field of mobile edge computing and secure offloading techniques, offering valuable insights for future developments in this area.

3 Proposed System

The proposed work consists of 2 phases for encryption and decryption process. The security parameters are strongly framed in this work. The proposed system is implemented using AES and RSA algorithms. Here plaintext is divided into two blocks. The first part of the block is encrypted using AES, and the second is encrypted using RSA. Phase 1 producesciphertext1 and ciphertext2 at the sender side; then, these texts are transmitted through a wireless link to the receiver. Phase 2 performs the decryption operation and obtains the original text.

The proposed method is implemented in two phases: The first phase contains two processes: plaintext partition and encryption. The second phase also includes two decryption processes and combines the plaintext1,2 into plaintext. Figure 5 shows the general layout of the proposed method.

3.1 Phase 1: Encryption Process

Here the plain text is divided into two parts. The whole plaintext is divided into plaintext 1, plaintext 2.

Figure 6 shows the encryption process. The plain text one is converted into ciphertext 1 with the help of AES.AES contains a strong secret key for both encryption and decryption processes. Plaintext 2 is encrypted using RSA and produce ciphertext 2.

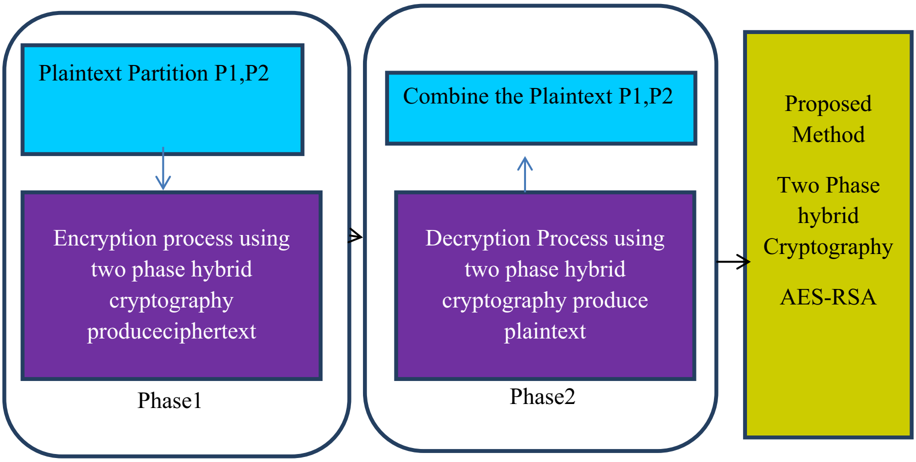
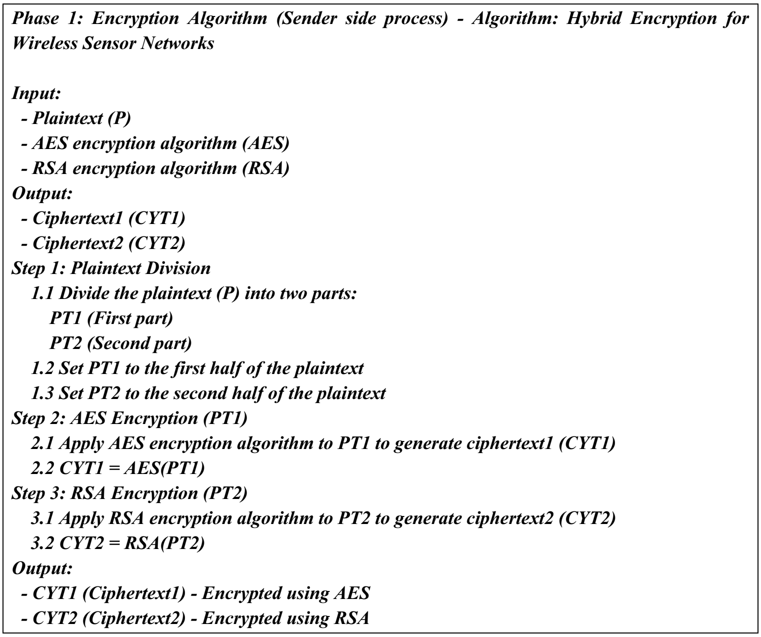

Plaintext Partition P1,P2
Encryption process using two phase hybrid cryptography produceciphertext
Phase1
Combine the Plaintext P1,P2
Decryption Process using two phase hybrid cryptography produce plaintext
Phase2
Proposed Method
Two Phase hybrid Cryptography
AES-RSA

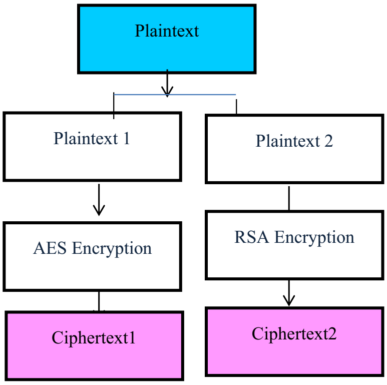
Fig. 5 Proposed method-general layout


```
Phase 1: Encryption Algorithm (Sender side process) - Algorithm: Hybrid Encryption for Wireless Sensor Networks

Input:
 - Plaintext (P)
 - AES encryption algorithm (AES)
 - RSA encryption algorithm (RSA)
Output:
 - Ciphertext1 (CYT1)
 - Ciphertext2 (CYT2)
Step 1: Plaintext Division
   1.1 Divide the plaintext (P) into two parts:
      PT1 (First part)
      PT2 (Second part)
   1.2 Set PT1 to the first half of the plaintext
   1.3 Set PT2 to the second half of the plaintext
Step 2: AES Encryption (PT1)
   2.1 Apply AES encryption algorithm to PT1 to generate ciphertext1 (CYT1)
   2.2 CYT1 = AES(PT1)
Step 3: RSA Encryption (PT2)
   3.1 Apply RSA encryption algorithm to PT2 to generate ciphertext2 (CYT2)
   3.2 CYT2 = RSA(PT2)
Output:
 - CYT1 (Ciphertext1) - Encrypted using AES
 - CYT2 (Ciphertext2) - Encrypted using RSA
```

Phase 1: Encryption Algorithm (Sender side process) - Algorithm: Hybrid Encryption forWireless Sensor Networks

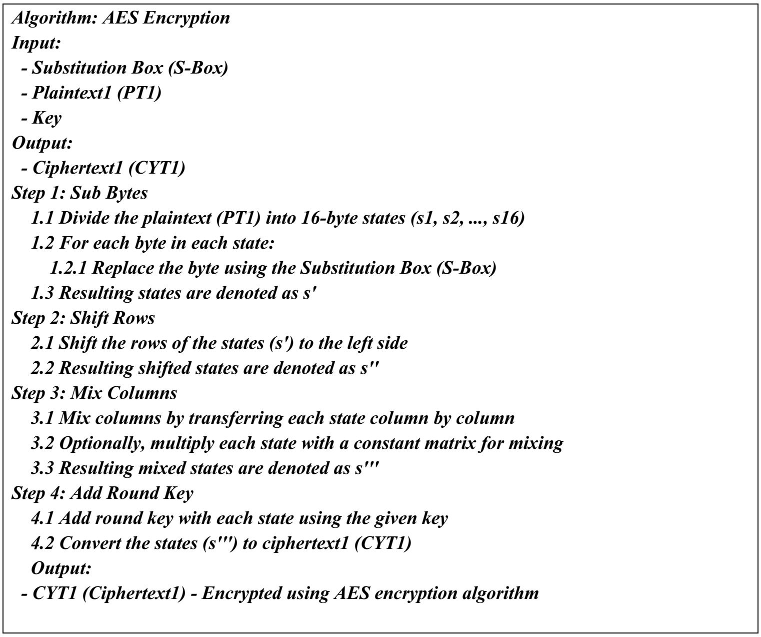
Fig. 6 Phase 1 encryption phase



3.1.1 AES Encryption Process

AES algorithm contains three parts encryption, decryption and key generation. Four different forms of transformations are applied to a plaintext. These are sub bytes, shift rows, mix columns and add round key.

Algorithm: AES Encryption

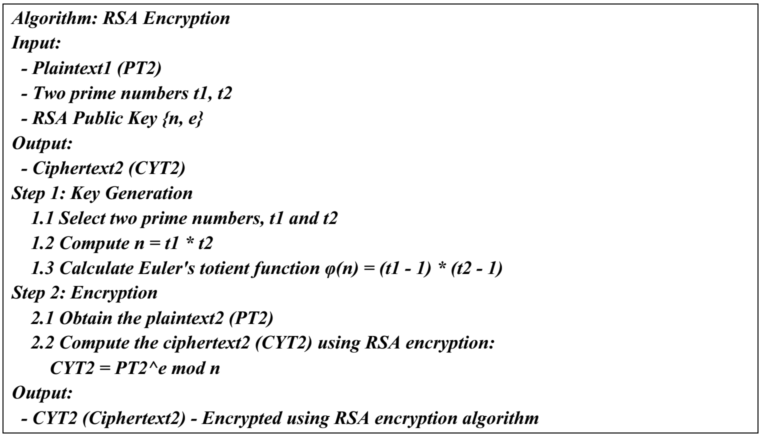


```
Algorithm: AES Encryption
Input:
 - Substitution Box (S-Box)
 - Plaintext1 (PT1)
 - Key
Output:
 - Ciphertext1 (CYT1)
Step 1: Sub Bytes
   1.1 Divide the plaintext (PT1) into 16-byte states (s1, s2, ..., s16)
   1.2 For each byte in each state:
      1.2.1 Replace the byte using the Substitution Box (S-Box)
   1.3 Resulting states are denoted as s'
Step 2: Shift Rows
   2.1 Shift the rows of the states (s') to the left side
   2.2 Resulting shifted states are denoted as s''
Step 3: Mix Columns
   3.1 Mix columns by transferring each state column by column
   3.2 Optionally, multiply each state with a constant matrix for mixing
   3.3 Resulting mixed states are denoted as s'''
Step 4: Add Round Key
   4.1 Add round key with each state using the given key
   4.2 Convert the states (s''') to ciphertext1 (CYT1)
   Output:
 - CYT1 (Ciphertext1) - Encrypted using AES encryption algorithm
```

The AES Encryption algorithm is a widely-used symmetric encryption technique for ensuring secure communication by transforming plaintext into ciphertext. The algorithm begins with the Sub Bytes step, where the plaintext PT1 is divided into 16-byte states, and each byte within these states undergoes a substitution process using the Substitution Box (S-Box). This process replaces each byte with a corresponding byte from the S-Box, resulting in transformed states denoted as s’. Subsequently, in the Shift Rows step, the rows of the states s’ are shifted to the left side, creating shifted states denoted as s″. Following this, the Mix Columns step involves mixing the columns of the states s″ by transferring each column in a specific manner. Optionally, each state may be multiplied with a constant matrix to further enhance mixing. The resulting mixed states are denoted as s‴. Finally, in the Add Round Key step, the round key is added to each state, utilizing the given key for encryption. This process results in the conversion of the states s‴ into ciphertext1 (CYT1), representing the encrypted form of the original plaintext PT1. Overall, the AES Encryption algorithm provides a robust method for securing data transmission, offering confidentiality and integrity protection.

3.1.2 RSA Encryption Process

Algorithm: RSA Encryption



```
Algorithm: RSA Encryption
Input:
  - Plaintext1 (PT2)
  - Two prime numbers t1, t2
  - RSA Public Key {n, e}
Output:
  - Ciphertext2 (CYT2)
Step 1: Key Generation
    1.1 Select two prime numbers, t1 and t2
    1.2 Compute $n = t1 * t2$
    1.3 Calculate Euler's totient function $φ(n) = (t1 - 1) * (t2 - 1)$
Step 2: Encryption
    2.1 Obtain the plaintext2 (PT2)
    2.2 Compute the ciphertext2 (CYT2) using RSA encryption:
        $CYT2 = PT2^e mod n$
Output:
  - CYT2 (Ciphertext2) - Encrypted using RSA encryption algorithm
```

The RSA Encryption algorithm is a widely used asymmetric cryptographic technique for ensuring secure communication. The algorithm begins with key generation, where two prime numbers, t1 and t2, are selected. The product of these primes, denoted as n, serves as the modulus for encryption. Additionally, the Euler’s totient function φ(n) is computed to determine the number of integers coprime to n, crucial for encryption. In the encryption phase, plaintext2 (PT2) is obtained, and then encrypted using the RSA public key {n, e}, where e is the public exponent. The ciphertext2 (CYT2) is computed as CYT2 = PT2^e mod n. This process ensures that the plaintext2 is transformed into an encrypted form, CYT2, which can only be decrypted using the corresponding RSA private key. Overall, the RSA Encryption algorithm provides a robust method for securing data transmission in various applications, offering confidentiality and integrity protection.

3.2 Phase 2: Decryption Process

Finally, this process produces plaintext1 and plain text2. Combine the two plaintexts on the receiver side. Figure 7 shows the decryption process.

The Phase 2 Decryption algorithm is a crucial process carried out at the receiver side to obtain the original plaintext from the received ciphertexts. The algorithm takes as input the ciphertexts CYT1 and CYT2, along with the AES and RSA decryption algorithms.

Algorithm: Phase 2 Decryption (Receiver side process)

```
Algorithm: Phase 2 Decryption (Receiver side process)
Input:
  - Ciphertext1 (CYT1)
  - Ciphertext2 (CYT2)
  - AES decryption algorithm (AES)
  - RSA decryption algorithm (RSA)
Output:
  - Plaintext (P)
  - Plaintext1 (PT1)
  - Plaintext2 (PT2)

Step 1: AES Decryption
    1.1 Decrypt Ciphertext1 (CYT1) using AES decryption algorithm to obtain Plaintext1 (PT1)
        PT1 = AES(CYT1)
Step 2: RSA Decryption
    2.1 Decrypt Ciphertext2 (CYT2) using RSA decryption algorithm to obtain Plaintext2 (PT2)
        PT2 = RSA(CYT2)
Step 3: Combine Plaintext1 and Plaintext2
    3.1 Combine the decrypted plaintexts PT1 and PT2 to produce the original plaintext (P)
        P = Concatenate(PT1, PT2)
Output:
  - P (Plaintext)
  - PT1 (Plaintext1)
  - PT2 (Plaintext2)
```

In the first step, AES Decryption is performed to decrypt CYT1, yielding Plaintext1 (PT1) using the AES decryption algorithm. Subsequently, RSA Decryption is applied to decrypt CYT2, resulting in Plaintext2 (PT2) using the RSA decryption algorithm. Finally, the decrypted plaintexts PT1 and PT2 are combined to produce the original plaintext (P) by concatenating them together. This process ensures the successful decryption of the received ciphertexts, thereby reconstructing the original plaintext, and enables the receiver to access the information securely transmitted over the network.

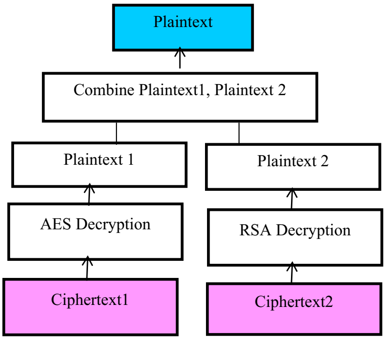


Fig. 7 Phase 2 decryption phase

3.2.1 AES decryption Process

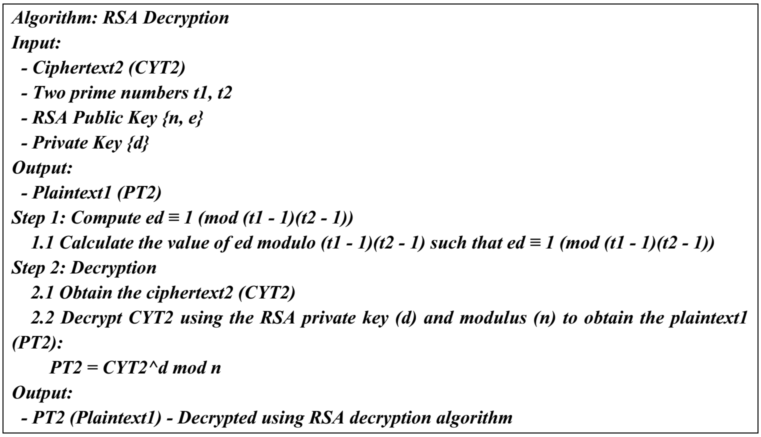
Algorithm: Reverse AES Decryption

```
Algorithm: Reverse AES Decryption
Input:
  - Ciphertext1 (CYT1)
  - Substitution Box (S-Box)
Output:
  - Plaintext1 (PT1)
Step 1: Reverse Sub Bytes
    1.1 Divide the ciphertext (CYT1) into states (s1, s2, ..., s16)
    1.2 For each byte in each state:
        1.2.1 Replace the byte using the reverse Substitution Box (S-Box)
    1.3 Resulting states are denoted as s'
Step 2: Reverse Shift Rows
    2.1 Shift the rows of the states (s') to the right side
    2.2 Resulting shifted states are denoted as s''
Step 3: Reverse Mix Columns
    3.1 Mix columns by transferring each state column by column in reverse order
    3.2 Optionally, multiply each state with the inverse of the constant matrix used in encryption
    3.3 Resulting mixed states are denoted as s'''
Step 4: Reverse Add Round Key
    4.1 Reverse the addition of round key with each state
    4.2 Convert the states (s''') to plaintext1 (PT1)
    Output:
  - PT1 (Plaintext1) - Decrypted using reverse AES decryption algorithm
```

The Reverse AES Decryption algorithm serves as a crucial process for decrypting ciphertexts encrypted using the Advanced Encryption Standard (AES). The algorithm takes as input the ciphertext CYT1 and the Substitution Box (S-Box) used in the AES encryption process. Initially, the ciphertext is divided into states, and then each byte within these states undergoes a reverse Sub Bytes operation, where the bytes are replaced using the reverse S-Box. Subsequently, the rows of the resulting states are shifted to the right in a reverse Shift Rows operation, followed by a reverse Mix Columns operation, where the columns of the states are mixed in reverse order. Additionally, each state may be optionally multiplied by the inverse of the constant matrix used in encryption. Finally, the reverse Add Round Key operation is performed to reverse the addition of the round key with each state, resulting in the decryption of the ciphertext into plaintext1 (PT1). This algorithm provides a mechanism for decrypting AES-encrypted data, thereby enabling the retrieval of the original plaintext from ciphertexts encrypted using the AES algorithm.

3.2.2 RSA decryption Process

Algorithm: RSA Decryption



```
Algorithm: RSA Decryption
Input:
  - Ciphertext2 (CYT2)
  - Two prime numbers t1, t2
  - RSA Public Key {n, e}
  - Private Key {d}
Output:
  - Plaintext1 (PT2)
Step 1: Compute $ed \equiv 1 \pmod{(t1 - 1)(t2 - 1)}$
    1.1 Calculate the value of ed modulo (t1 - 1)(t2 - 1) such that $ed \equiv 1 \pmod{(t1 - 1)(t2 - 1)}$
Step 2: Decryption
    2.1 Obtain the ciphertext2 (CYT2)
    2.2 Decrypt CYT2 using the RSA private key (d) and modulus (n) to obtain the plaintext1 (PT2):
        $PT2 = CYT2^d \bmod n$
Output:
  - PT2 (Plaintext1) - Decrypted using RSA decryption algorithm
```

The RSA Decryption algorithm is a fundamental process for decrypting ciphertexts encrypted using the RSA encryption scheme. The algorithm takes as input the ciphertext CYT2, along with the RSA public key {n, e} and the corresponding private key d. In the first step, the value of ed modulo (t1−1)(t2−1) is computed, where t1 and t2 are the prime numbers used in key generation. This computation ensures that the product of the public exponent e and the private exponent d is congruent to 1 modulo φ(n), where φ(n) is Euler's totient function of n. Subsequently, in the decryption phase, the ciphertext CYT2 is decrypted using the RSA private key d and modulus n to obtain the plaintext1 (PT2). This decryption process is achieved by raising CYT2 to the power of d modulo n. The resulting plaintext1 (PT2) represents the original plaintext that was encrypted using the RSAencryption algorithm, thereby enabling secure communication between parties.

4 Result Analysis

The proposed work has been developed and implemented using JAVA programming.

4.1 Key Length Versus Encryption Time

The table illustrates the relationship between key length in bits and encryption time in milliseconds for three cryptographic schemes: RSA–ECC, AES-MD5, and the proposed AES–RSA system. Across all key lengths tested, the proposed AES–RSA scheme consistently demonstrates shorter encryption times compared to both RSA–ECC and AES-MD5. Specifically, as the key length increases, the encryption time also increases for all three schemes, which is expected due to the increased computational complexity associated with longer keys. However, the proposed AES–RSA system consistently outperforms the other two schemes, showcasing its efficiency in cryptographic processing.

The results depicted in Fig. 8 and Table 1 indicate a correlation between encryption time and key size, with an increase in key size resulting in longer encryption times.

Notably, the proposed AES–RSA system exhibits a comparatively shorter encryption time compared to RSA–-ECC and AES-MD5. This efficiency can be attributed to the partitioning of the plaintext employed by AES–RSA, which allows for more streamlined processing. This division of the plaintext facilitates quicker encryption, as it enables parallelization andoptimization of the encryption process.

4.2 Key Length Versus Decryption Time

The provided table depicts the correlation between key length in bits and decryption time in milliseconds for three cryptographic approaches: RSA–ECC, AES-MD5, and the proposed AES–RSA system. Across varying key lengths, it's evident that the proposed AES–RSA scheme consistently exhibits notably shorter decryption times compared to both RSA–ECC and AES-MD5. As expected, decryption time tends to increase with longer key lengths due to heightened computational complexity. However, the proposed AES–RSA system consistently outshines the other methods, demonstrating its efficiency in cryptographic decryption tasks. This efficiency becomes particularly pronounced at larger key lengths, where the proposed scheme showcases a considerable reduction in decryption time compared to RSA–ECC and AES-MD5. These findings underscore the effectiveness of the AES–RSA hybrid approach, presenting it as a promising solution for applications requiring rapid and secure decryption processes.

The findings illustrated in Fig. 9 and Table 2 demonstrate a positive correlation between decryption time and key size, indicating that larger key sizes result in longer decryption times.

Notably, the AES–RSA system proposed in this study exhibits a reduced decryption time compared to RSA–ECC and AES-MD5. This efficiency can be attributed to the approach employed by AES–RSA, which involves partitioning the ciphertext. By dividing the ciphertext into partitions, AES–RSA enables more efficient decryption processes. This partitioning strategy facilitates parallelization and optimization of decryption operations, leading to quicker decryption times.

4.3 Total Execution Time

Table 3 and Fig. 10 provide a comprehensive comparative analysis of the total execution times for both the existing system and the proposed AES–RSA method across various key

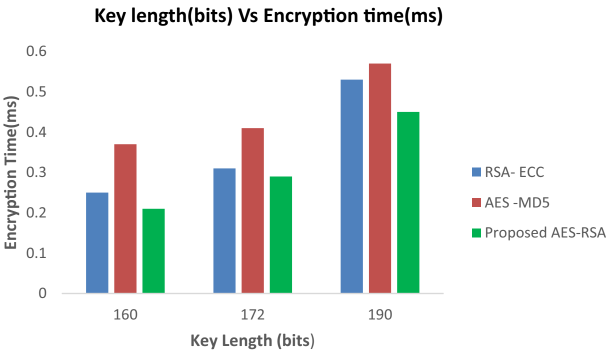


Fig. 8 Key length(bits) versus Encryption time (ms)

Table 1 Key length (bits) versus Encryption time(ms)

Key length bits	RSA- ECC	AES -MD5	RSA-OAEP (RSA with optimal asymmetric encryption padding)	Proposed AES–RSA
160	0.25	0.37	5.2	0.21
172	0.31	0.41	6.8	0.29
190	0.53	0.57	8.5	0.45

lengths. It's evident that the proposed AES–RSA approach consistently achieves shorter total execution times compared to both RSA–ECC and AES-MD5. This improvement in efficiency is particularly notable at larger key lengths, where the proposed method demonstrates a significant reduction in total execution time.

Such results underscore the effectiveness of the AES–RSA hybrid technique in enhancing overall system performance, offering a compelling solution for cryptographic applications where minimizing execution time is critical. These findings highlight the potential of the proposed AES–RSA approach to contribute to the advancement of cryptographic systems by providing a balance between security and efficiency.

The total execution time is contingent upon various factors, including key and text size. Remarkably, the proposed system showcases improved efficiency over RSA–ECC and AES-MD5, resulting in reduced execution times. This enhancement can be attributed to the following specific reasons: (i) In Phase 1, the plaintext is divided into plaintext1 and plaintext2 to simplify the cryptographic process, reducing computational overhead. (ii) AES encryption process is applied to plaintext1, leveraging its fast and efficient encryption algorithm. (iii) RSA encryption process is applied to the remaining part of plaintext2, utilizing its strength in handling larger data sizes. (iii) The resulting ciphertexts, ciphertext1 and ciphertext2, are transmitted to the receiver end, optimizing data transmission efficiency. (iv) In Phase 2, ciphertext1 is decrypted into plaintext1 using the AES decryption process, maintaining speed and accuracy. (v) Ciphertext2 is decrypted into plaintext2 using the RSA decryption process, ensuring robustness against cryptographic attacks. (vi) The plaintext1 and plaintext2 are combined to produce the original plaintext at the receiver end, ensuring security against attacks while maintaining efficiency throughout the cryptographic process.

Table 4 presents a comparative analysis of RSA-OAEP, McEliece-RSA, and the proposed AES–RSA hybrid cryptographic algorithms, highlighting differences in key length, encryption time, decryption time, and total execution time. While RSA-OAEP

Fig. 9 Key length (bits) versus Decryption time (ms)

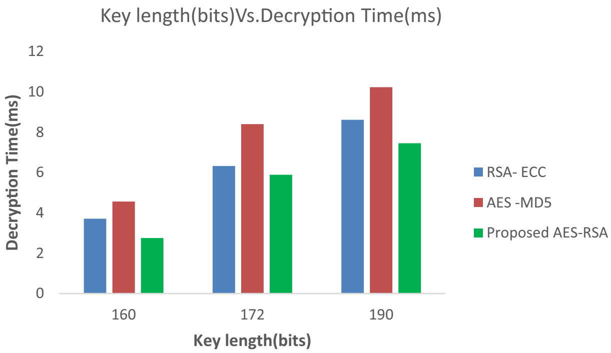


Table 2 Key length (bits) versus Decryption time (ms)

Key length Bits	RSA–ECC	AES -MD5	Proposed AES–RSA
160	3.71	4.56	2.75
172	6.32	8.4	5.89
190	8.61	10.23	7.45

Table 3 Key length (bits) versus Total execution time (ms)

Key length Bits	RSA–ECC	AES -MD5	Proposed AES–RSA
160	3.96	4.93	2.96
172	6.63	8.81	6.18
190	9.14	10.8	7.9

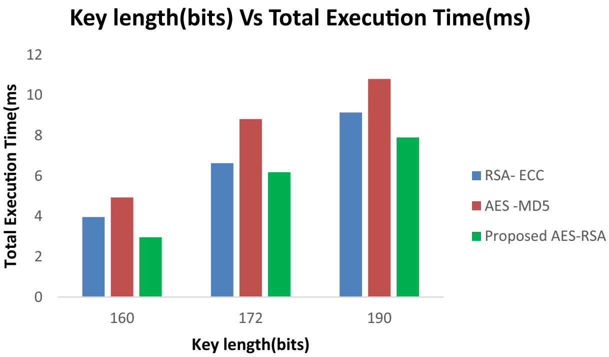
and McEliece-RSA show increased encryption and decryption times with larger key lengths, the proposed AES–RSA algorithm demonstrates superior efficiency with significantly lower times across all key lengths. This efficiency arises from combining AES for encryption and RSA for key management, resulting in faster overall execution. Thus, the AES–RSA hybrid emerges as a promising solution, offering expedited cryptographic operations compared to traditional RSA-based methods, addressing performance challenges in secure data transmission and storage in modern computing environments.

4.4 Security Requirements of the Proposed System

Every cryptographic algorithm must satisfy the security requirements of the applications. WSN is applicable in very sensitive applications like healthcare, defense, etc., The security parameter of the proposed system is mentioned in Table 5. Security parameters analysis is decided on which algorithm has strong security properties.

The proposed algorithm ensures data integrity, confidentiality, Authentication and freshness through several mechanisms embedded within its design, leveraging the contributions outlined in the paper.

Fig. 10 Key lengths (bits) versus Total execution time (ms)



- **Plaintext Division**

  The division of plaintext into plaintext1 and plaintext2 in Phase 1 simplifies the cryptographic process and allows for separate encryption using AES and RSA. This division ensures that each portion of the plaintext undergoes encryption independently, enhancing the integrity of the data.
- **AES Encryption**

  Plaintext1 undergoes AES encryption, a symmetric encryption process known for its strong security properties. AES ensures that plaintext1 is securely transformed into ciphertext1, making it resistant to tampering or unauthorized modification during transmission.
- **RSA Encryption**

  Plaintext2, the remaining part of the plaintext, is subjected to RSA encryption, a public-key encryption process. RSA encryption provides additional security by encrypting plaintext2 with the recipient's public key, ensuring that only the intended recipient can decrypt and access the original data.
- **Transmission and Decryption**

  The resulting ciphertexts, ciphertext1 and ciphertext2, are transmitted to the receiver end. During Phase 2, ciphertext1 is decrypted into plaintext1 using the AES decryption process, while ciphertext2 is decrypted into plaintext2 using the RSA decryption process.

  The combined plaintext1 and plaintext2 produce the original plaintext at the receiver end. This process ensures that the decrypted data matches the original plaintext, thus preserving data integrity.
- **Contributions to Data Integrity**

  The division of plaintext, coupled with separate encryption using strong cryptographic algorithms (AES and RSA), ensures that even if one portion of the ciphertext is compromised, the other remains secure.

  The use of AES and RSA encryption, along with the decryption processes in Phase 2, provides cryptographic integrity checks that detect any unauthorized modifications or tampering attempts during data transmission.

Table 4 Comparative analysis between recent hybrid cryptographic algorithm and proposed AES–RSA

Key length bits	RSA-OAEP (RSA with optimal asymmetric encryption padding)	McEliece-RSA	Proposed AES-RSA
Encryption time (ms)			
160	5.2	1	0.21
172	6.8	2	0.29
190	8.5	3	0.45
Decryption time (ms)			
160	6.8	8	2.75
172	8.5	12	5.89
190	10.2	16	7.45
Total Execution Time(ms)			
160	12	9	2.96
172	15.3	14	6.18
190	10.2	19	7.9

Table 5 Security requirement analysis between existing and proposed system

Security requirements	RSA–ECC	AES -MD5	Proposed AES–RSA
Data integrity	No	No	Yes
Data confidentiality	Yes	Yes	Yes
Data availability	No	Yes	Yes
Data authentication	Yes	No	Yes
Data freshness	Yes	No	Yes

4.5 Security Attacks Analysis Between Existing and Proposed AES–RSA

The proposed algorithm, as outlined in the contributions, incorporates several security measures to protect against various attacks. Here's how the contributions address the security attacks mentioned in Table 6

- **DoS Attack**
  The proposed algorithm mitigates DoS attacks by not being susceptible to denial-of-service attacks. This is achieved by ensuring efficient encryption and decryption processes, as well as strong key generation mechanisms. Additionally, the division of plaintext and separate encryption using AES and RSA contribute to the resilience against DoS attacks by distributing computational load and preventing resource exhaustion.
- **Masquerade Attack**
  The proposed algorithm does not explicitly address masquerade attacks in the contributions outlined. However, the use of AES and RSA encryption, along with proper key management practices, can help mitigate the risk of masquerade attacks by ensuring that only authenticated entities can decrypt and access the data.
- **Replay Attack**

The proposed algorithm prevents replay attacks by incorporating freshness mechanisms, as indicated in the contributions. By dividing the plaintext and incorporating data freshness checks during encryption and decryption processes, the algorithm ensures that replayed ciphertexts are detected and rejected, thereby preventing unauthorized access to outdated data.

- **Modification Attack**

  The proposed algorithm defends against modification attacks by ensuring data integrity through separate encryption of plaintext1 and plaintext2 using AES and RSA encryption, respectively. The decryption processes in Phase 2 include cryptographic integrity checks to detect any unauthorized modifications or tampering attempts during data transmission.

4.6 The Complexity Analysis of the Proposed Algorithm

The complexity analysis of the proposed algorithm is structured around the key contributions outlined:

- **Plaintext Division (Contribution i)**

  This step involves dividing the plaintext into two segments, plaintext1 and plaintext2. The computational complexity of this operation is linear and proportional to the size of the plaintext, denoted as O(n), where n represents the size of the plaintext.
- **AES Encryption (Contribution ii)**

  Applying AES encryption to plaintext1 introduces computational overhead primarily determined by the block size of AES. The time complexity is typically constant per block and can be represented as O(b), where b denotes the block size.
- **RSA Encryption (Contribution iii)**

  RSA encryption applied to plaintext2 involves modular exponentiation operations, resulting in a computational complexity of O(k^3), where k represents the size of the RSA modulus (key size).
- **Transmission (Contribution iii)**

  Transmitting ciphertext1 and ciphertext2 to the receiver end incurs minimal computational overhead, approximated as constant time, denoted as O(1).
- **Decryption Phase (Contributions iv, v)**

  Decrypting ciphertext1 using AES decryption and ciphertext2 using RSA decryption entails similar computational complexities to their encryption counterparts. AES decryption has a time complexity of O(b), while RSA decryption complexity is O(k^3).

Table 6 Security attacks analysis between existing and proposed system

Security attacks	RSA–ECC	AES -MD5	Proposed AES–RSA
DoS attack	Yes	Yes	No
Masquerade attack	No	No	No
Replay attack	Yes	Yes	No
Modification	Yes	Yes	No

- **Combination of Plaintexts (Contribution vi)**

  Merging plaintext1 and plaintext2 to reconstruct the original plaintext involves linear computational effort, proportional to the size of the plaintext, denoted as O(n).

4.7 State of Art Comparison Table

The comparison table outlines the state-of-the-art encryption techniques including the proposed hybrid encryption, AES, and RSA. The proposed hybrid encryption offers enhanced security by combining the strengths of symmetric and asymmetric encryption, providing robust protection against various attacks. AES, known for its fast encryption and decryption, is widely adopted and standardized, but requires secure key management to prevent vulnerabilities (Table 7).

On the other hand, RSA encryption, with its asymmetric nature, ensures strong security and is suitable for key management, albeit at a slower pace compared to symmetric encryption. However, RSA faces challenges in key distribution and is vulnerable to chosen plaintext attacks. This comparison highlights the trade-offs between speed, security, and complexity, helping stakeholders make informed decisions based on their specific security requirements and computational resources.

4.7.1 Security Mechanism Versus Processing Speed

In addressing the potential decrease in processing speed resulting from enhanced security mechanisms in the proposed hybrid encryption approach, several strategies can be employed. Firstly, algorithmic optimization can fine-tune encryption and decryption algorithms to balance security and processing speed, implementing efficient data structures and exploring algorithmic optimizations specific to the hybrid encryption technique. Secondly, parallelization techniques can distribute computational tasks across multiple processing units, optimizing key generation, encryption, or decryption operations for maximum efficiency. Thirdly, leveraging hardware acceleration technologies such as cryptographic coprocessors or dedicated hardware modules can significantly enhance processing speed by offloading cryptographic tasks from the main CPU. Additionally, asynchronous processing techniques can decouple encryption and decryption tasks from synchronous processing pipelines, allowing cryptographic operations to proceed independently of other system tasks. Lastly, efficient key management practices, including streamlined key generation, distribution, and storage mechanisms, can minimize computational overhead and ensure efficient utilization of cryptographic resources throughout the encryption and decryption processes. By implementing these strategies tailored to the proposed hybrid encryption approach, one can effectively mitigate the potential decrease in processing speed while maintaining robust security for sensitive data transmission scenarios.

4.8 Proposed System Applications

The proposed hybrid encryption approach holds significant potential for a wide range of applications across various sectors. In particular, it can be tailored to meet the security needs of sensitive data transmission and communication in critical domains such as healthcare, finance, government, and defense. For instance, in healthcare, where patient data privacy is paramount, the hybrid encryption technique can ensure the confidentiality and integrity of medical records during transmission between healthcare providers

Table 7 State of art comparison table

S.No	Technique	Advantages	Limitations
1	Proposed hybrid encryption	Enhanced security due to hybrid encryption Combining strengths of symmetric and asymmetric	Requires computation for both AES and RSA operations Key management complexity Increased computational overhead
2	AES	Fast encryption and decryption Widely adopted and standardized	Requires secure key management Vulnerable to brute-force attacks without adequate key length
3	RSA	Strong security due to asymmetric natureof encryption Suitable for key management	Vulnerable to chosen plaintext attacks

and institutions. Similarly, in financial transactions, the hybrid approach can bolster the security of online banking and payment systems, safeguarding sensitive financial information against unauthorized access and fraud. Moreover, in government and defense sectors, where classified information is exchanged, the hybrid encryption method can enhance the resilience of communication networks, protecting national security interests from cyber threats and espionage. Overall, the proposed hybrid encryption technique presents versatile applications across diverse sectors, offering a robust solution for securing data transmission in sensitive environments.

4.9 Proposed System Limitations

Computational Overhead Implementing a hybrid encryption scheme requires additional.

Computational resources compared to using a single encryption algorithm. This can lead to increased processing time and resource utilization, especially in resource-constrained environments such as IoT devices or embedded systems.

Key Management Complexity Managing both symmetric and asymmetric keys in a hybrid encryption system can introduce complexity, especially in large-scale deployments. Proper key generation, distribution, and storage mechanisms are crucial to maintaining the security of the system, which may pose challenges in terms of administration and maintenance.

Integration Challenges Integrating multiple encryption algorithms into existing systems or protocols may require substantial modifications and testing to ensure compatibility and interoperability.

Potential Attack Vectors Hybrid encryption systems may introduce new attack vectors that exploit weaknesses in either the symmetric or asymmetric encryption components. For example, an attacker may attempt to compromise the asymmetric key exchange process to gain access to the symmetric session key.

Key Size and Strength The security of the hybrid encryption system relies on the strength of both the symmetric and asymmetric encryption algorithms as well as the length of the encryption keys used. Inadequate key sizes or weaker encryption algorithms could undermine the overall security of the system.

5 Conclusion

A novel hybrid two-phase cryptographic technique has been developed to ensure robust security in wireless sensor networks (WSNs). The approach involves six key steps aimed at enhancing the security of communication within WSNs. First, the plaintext is divided into two parts, plaintext1 and plaintext2, simplifying the subsequent cryptographic processes. AES encryption is then applied to plaintext1, leveraging the strength of symmetric encryption to protect the data, while the remaining part of the plaintext, plaintext2, undergoes RSA encryption, utilizing asymmetric encryption for added security. In the second phase, ciphertext1 is decrypted into plaintext1 using the AES decryption process, while simultaneously ciphertext2 is decrypted into plaintext2 using the RSA decryption process. Finally, the plaintext1 and plaintext2 are combined to produce the original plaintext, ensuring secure transmission without any security breaches. This two-phase hybrid approach, combining symmetric and asymmetric cryptographic techniques, effectively safeguards communication in WSNs against various attacks. However, to further enhance security, future iterations of the scheme may incorporate additional hybrid cryptographic algorithms with substantial computational complexity, bolstering the resilience of WSNs against emerging security threats and ensuring the integrity, confidentiality, and availability of transmitted data. By integrating Artificial Intelligence (AI) with cryptographic techniques, future research endeavors can advance the state-of-the-art in WSN security, ensuring robust protection against evolving cyber threats.

Author Contributions All authors contributed to the study conception,design and implementation. All authors read and approved the final manuscript.

Funding The authors received no specific funding for this study.

Data Availability Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Code Availability Code availability is not applicable for this article.

Declarations

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

References

1. Akyildiz, I. F., Su, W., Sankarasubramaniam, Y., & Cayirci, E. (2002). Wireless sensor networks: a survey. *Computer Networks, 38*(4), 393–422.
2. Panda, M. (2014). Security in wireless sensor networks using cryptographic techniques. *Am. J. Eng. Res. (AJER), 3*(1), 50–56.
3. Anusuya Devi, V., Kalaivani, V.: Hybrid cryptosystem in wireless body area networks using message authentication code and modified and enhanced lattice-based cryptography (MAC-MELBC) in healthcare applications, Concurrency and Computation: Practice and Experience,Willey, vol. 33, no. 9, pp. 1–13, 2020.
4. Mallikarjunaswamy, N. J., Prasanna, K., & LathaYadav, T. R. (2020). Secure reprogramming exploitation m-RSA technique in wireless sensor networks. *Int. J. Eng. Appl. Sci. Technol., 5*(6), 188–192.
5. Sampradeepraj, T., & Raja, S. P. (2020). On improving reliability in multicast routing protocol for wireless sensor network. *Inf. Technol. Control, 49*(2), 260–273.
6. Kavin, B. P., & Ganapathy, S. (2019). A novel M-commerce data security mechanism using elliptic curve cryptography. *Int. J. Innov. Technol. Explor. Eng. (IJITEE), 8*(10), 847–851.
7. Kavin, B. P., Ganapathy, S., Kanimozhi, U., & Kannan, A. (2020). An enhanced security framework for secured data storage and communications in cloud using ECC, access control and LDSA. *Wireless Personal Communications, 115*(1), 1107–1135.
8. Sreevidya, B., Rajesh, M., & Mamatha, T. M. (2018). Design and development of an enhanced security scheme using RSA for preventing false data injection in wireless sensor networks. *Ambient Commun. Comput. Syst., 69*(6), 225–236.
9. MajidAlshammari, R., & Elleithy, K. M. (2018). Efficient and secure key distribution protocol for wireless sensor networks. *Sensors, 18*(10), 3569.
10. Patil, S., & Patil, P. (2014). Designing a model for energy efficient and secured data communication using RSA algorithm in wireless sensor networks. *Int. J. Sci. Res. (IJSR), 3*(7), 1–5.
11. Asaad, M. S., & Croock, M. S. (2021). Adaptive security approach for wireless sensor network using RSA algorithm. *Indones. J. Electr. Eng. Comput. Sci., 22*(1), 361–368.
12. Gulen, U., Alkhodary, A., & Baktir, S. (2019). Implementing RSA for wireless sensor nodes. *Sensors, 19*(13), 1–15.
13. Kavan, N., Premananda, B. S.: Implementation of simplified AES algorithm for wireless sensor nodes on FPGA, Int. J. Sci. Eng. Res., 4(8), 2013.

14. Thangarajan, S., & Bhaaskarana, V. S. K. (2018). High speed and low power implementation of AES for wireless sensor networks. *Procedia Comput. Sci., 14*(3), 736–743.
15. Dragomir, D., Panait, C.: Performance and energy consumption analysis of AES in wireless sensor networks. In: International conference on innovative network systems and applications, 461, 2016
16. Gupta, M., & Sinha, A. (2021). Enhanced-AES encryption mechanism with S-box splitting for wireless sensor networks. *International Journal of Information Technology, 93*(1), 933–941.
17. Shin, S., Won, K., & Shin, S. (2020). Size efficient preprocessed symmetric rsa for wireless body area network. *ACM SIGAPP Appl. Comput. Rev., 20*(1), 15–23.
18. Ramadan, M., Liao, Y., Li, F., Zhou, S., & Abdalla, H. (2019). IBEET-RSA: identity-based encryption with equality test over Rsa for wireless body area networks. *Mobile Netw. Appl., 25*, 223–233.
19. Sangwan, A., & Bhattacharya, P. P. (2018). A hybrid cryptography and authentication based security model for clustered WBAN. *J. Mech. Continua Math. Sci., 13*(1), 34–54.
20. Singla, R., & Kaur, N. (2018). Compressed and secure energy efficient routing protocol for WBAN. *International Journal of Computational Science and Engineering, 6*(7), 2347–2693.
21. Shin, S. Choi, S., Won, W., Shin, S.: Preprocessed symmetric RSA authentication for wireless body area networks in space. In: proceedings of the conference on research in adaptive and convergent systems, 230–235, 2019.
22. Satapathy, P., Pandey, N. Khatri, S. K.: NFC Car Keys By Using RSA Cryptography In WSN Security. In: Proceedings of the Third International Conference on Electronics Communication and Aerospace Technology, 143–147, 2019.
23. Aruna Deepthi, S., Aruna, V., & Leelavathi, R. (2022). Image transmission using leach and security using RSA in wireless sensor networks. *Advances in Intelligent Systems and Computing, 14*(20), 31–51.
24. Leelavathi, G., Shaila, K., Venugopal, K.R.: Design and implementation of montgomery multipliers in RSA cryptography for wireless sensor networks, In: proceedings of first international conference on smart system, innovations and computing, 565–574, 2018.
25. Surekha, J., & Anita Madona, M. (2015). Analysis of RSA and ELGAMAL algorithm for wireless sensor network. *Int. J. Comput. Tech., 2*(4), 25–31.
26. Nasralla, M. M. (2021). Sustainable virtual reality patient rehabilitation systems with IoT sensors using virtual smart cities". *Sustainability, 13*(9), 1–15.
27. Naresh, V. S., Nasralla, M. M., Reddi, S., & García-Magariño, I. (2022). Quantum Diffie-Hellman extended to dynamic quantum group key agreement for e-healthcare multi-agent systems in smart cities". *Sensors, 3940*(20), 3940.
28. Sobnath, D., Rehman, IU., Nasralla, MM.: "Smart cities to improve mobility and quality of life of the visually impaired, EAI/Springer Innovations in Communication and Computing(Book series), 3–28, 2019
29. Kumar, S. A., García-Magariño, I., Nasralla, M. M., & Nazir, S. (2021). Agent-based simulators for empowering patients in self-care programs using mobile agents with machine learning. *Mobile Inf. Syst., 2021*, 1–10.
30. AltafKhan, M., Nasralla, M. M., Umar, M. M., Ghani-Ur-Rehman, S. K., & Choudary, N. (2022). An efficient multilevel probabilistic model for abnormal traffic detection in wireless sensor networks. *Sensors, 410*, 1–22.
31. Kumar, M., Walia, G. K., Singh, H. S., & Gill, S. S. (2023). AI-based sustainable and intelligent offloading framework for IIoT in collaborative cloud-fog environments. *IEEE Transactions on Consumer Electronics, 70*, 1414–1422.
32. Chakraborty, A., Kumar, M., & Chaurasia, N. (2023). Secure framework for IoT applications using deep learning in fog computing. *J. Inf. Secur. Appl., 77*, 103569.
33. Kumar, P., Chauhan, N., Kumar, M., & Awasthi, L. K. (2021). Clustering based opportunistic traffic offloading technique for device-to-device communication. *International Journal of Systems Assurance Engineering and Management, 3*, 827–839.
34. Om Prakash, S., Mohit K.: Machine learning based theoretical and experimental analysis of DDoS attacks in cloud computing. In: 2023 international conference on device intelligence, computing and communication technologies, (DICCT), 2023
35. Samriya, J. K., Kumar, M., & Gill, S. S. (2023). Secured data offloading using reinforcement learning and Markov decision process in mobile edge computing. *International Journal of Network Management, 33*(5), e2243.

V. Anusuya Devi received her B.E (Computer Science and Engineering) degree from Amrita Institute of Technology and Science, Coimbatore (Anna University, Chennai) in the year 2006. She received her Master of Computer Science and Engineering Degree from Anna University in the year 2010. She received Ph.D Degree under the Faculty of Information and Communication Engineering (Research area: Network Security) from Anna University, Chennai in the year 2022. She has more than 15 years of teaching experience and she has published many papers in the reputed national and international journals and conferences. Her area of specialization includes Cryptography and Network Security, Wireless Body Area Networks and Wireless Networks

T. Sampradeepraj received his BE in Computer Science and Engineering from the BharathiyarsUniversity, Tamilnadu, India in 2002, ME in Wireless Technologies from the Thiagarajar College of Engineering, Anna University, Tamilnadu, India in 2005 and his PhD from the ManonmaniamSundarnar University, Tirunelveli. His current research interests include wireless sensor networks, network coding and network security. He published many papers in International Journals and International conferences .