Enhancing IoMT Security: A Lightweight Anonymous Authentication System using Elliptic Curve Cryptography (ECC)

A Project Work Submitted in Partial Fulfillment of the requirements for the Degree of

Bachelor of Technology
in
Computer Science and Engineering

Submitted by

Sarbartha Gupta (202002021023)
Gaurav Das (202002021051)
Mandeep Kumar (202002022095)

Under the supervision of

Prof. Amitava Nag

Department of Computer Science and Engineering

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(A Centrally Funded Institute under Ministry of HRD, Govt. of India)

BODOLAND TERRITORIAL AREAS DISTRICTS :: KOKRAJHAR :: ASSAM :: 783370

Website: www.cit.ac.in

June 2024

Department of Computer Science and Engineering

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(An Autonomous Institute under MHRD)

Kokrajhar – 783370, BTAD, Assam, India

CERTIFICATE OF APPROVAL

This is to certify that the work embodied in this project entitled **Enhancing IoMT Security: A Lightweight Anonymous Authentication System using Elliptic Curve Cryptography (ECC)** submitted by Sarbartha Gupta, Gaurav Das and Mandeep Kumar to the Department of **Computer Science & Engineering**, is carried out under our direct supervisions and guidance.

The project work has been prepared as per the regulations of Central Institute of Technology and I strongly recommend that this project work be accepted in partial fulfillment of the requirement for the degree of B.Tech.

Supervisor

Dr. Amitava Nag
Professor, Dept. of CSE

Countersigned by

Dr. Amitava Nag
HoD
Dept. of CSE

Department of Computer Science and Engineering

केन्द्रीय प्रौद्योगिकी संस्थान कोकराझार

CENTRAL INSTITUTE OF TECHNOLOGY KOKRAJHAR

(An Autonomous Institute under MHRD)

Kokrajhar – 783370, BTAD, Assam, India

CERTIFICATE BY THE BOARD OF EXAMINERS

This is to certify that the work entitled **Enhancing IoMT Security: A Lightweight Anonymous Authentication System using Elliptic Curve Cryptography (ECC)** submitted by Sarbartha Gupta,Gaurav Das and Mandeep kumar to the Department of **Computer Science & Engineering**, of Central Institute of Technology, Kokrajhar has been examined and evaluated.

The project work has been prepared as per the regulations of Central Institute of Technology and qualifies to be accepted in partial fulfillment of the requirement for the degree of B. Tech.

Project Co-ordinator

Mr. Mithun Karmakar
Assistant Professor
Dept. of CSE

External Examiner

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to our guide, **Dr. Amitava Nag** for his valuable guidance, consistent encouragement, personal care and timely help which provided us with an excellent atmosphere for doing our project. In spite of the busy schedule, he has extended his cheerful and cordial support to us, without which we could not have completed our project work.

We express our heartfelt thanks to our Head of the Department, **Dr. Amitava Nag** who has been actively involved and very influential from the start till the completion of our project.

We would also like to thank all the teaching and non-teaching staffs of the Computer Science Engineering Department for their constant support and Encouragement.
Last but not the least it is our pleasure to acknowledge the support and wishes of our friends and well-wishers, both in academic and non-academic spheres.

Date:

Sarbartha Gupta
(202002021023)

Gaurav Das
(202002021051)

Mandeep Kumar
(202002022095)

ABSTRACT

Internet of Things (IoT) is the next phase of the digital revolution. Recent advancements in this field, along with cryptography techniques, have opened new opportunities for various IoT based applications, including digital health, smart hospitals, automated pathology labs, and so forth where the data is collected by resource-contrained IoT devices to monitor the patients' health condition in real time. However, this also leads to security vulnerabilities where an unauthorized user can access health-related information or gain access to IoT sensor nodes that are integrated with medical equipment and attached to the patient's body, resulting in unthinkable outcomes. Hence, encryption and authentication of data have become a major priority to address in order to preserve integrity and privacy. To deal with this security-related issues, our project presents an Elliptic Curve Cryptography based user authentication and access control protocol. The proposed scheme establishes a secure session for the legitimate user to access data and prohibits unauthorized users from gaining access to the IoT-Sensor nodes and it's data. In particular, we propose a new lightweight and time-efficient ciphering algorithm for resource-constrained IoT devices. Furthermore, we perform a hardware implementation of the registration and authentication phase of the proposed method with python using a Raspberry Pi 3 Model and temperature sensor as an IoT device to show that proposed method can be executed in significantly less time than most traditional security protocols.

List of Tables

List of Figures

Contents

Chapter 1

Introduction

This chapter introduces the concept of cryptography, briefly described about IoT (Internet of things) and few of its many applications and also discussed about the role of IoT in healthcare as well as the security issues and challenges it faces and our motivation for undertaking this project as well as our objectives and contribution for the project.

1.1 Cryptography

Cryptography is the practice of safeguarding information and communication using codes and ciphers [4]. It entails turning ordinary text into a coded message that is only readable by the designated receiver. Cryptography is classified into three types: symmetric, hash functions, and asymmetric. Cryptography is used for different purposes, including computer passwords, digital currencies, secure web browsing, cryptocurrency, and authentication.

1.2 Internet of things(IoT)

The Internet of Things (IoT) is a collection of networked physical devices, vehicles, buildings, and other items that are outfitted with sensors, software, and network connectivity. These devices can gather and exchange data, allowing them to communicate with one another and other systems over the internet [9]. The primary goal of IoT is to enable seamless connectivity and data sharing between devices, resulting in enhanced efficiency, automation, and convenience in many aspects of our daily lives [13].

1.2.1 Applications of IoT

- **IoT in Agriculture:**

  IoT enables indoor planting by monitoring and managing micro-climate conditions, leading to increased productivity [29]. For outdoor planting, IoT-enabled devices can monitor soil moisture and nutrients, as well as weather data, to better operate smart irrigation and fertilizer systems. If the sprinkler systems dispense water only when needed, this prevents wasting a valuable resource.

- **IoT in Healthcare:**

  IoT gadgets for private citizens include wearables and smart home equipment that make living easier [1]. Fitbits, cellphones, Apple watches, and health monitors are all examples of wearable gear. These technologies enhance entertainment, network connectivity, health, and fitness.Smart homes automate tasks such as setting environmental controls to ensure that your home is as comfortable as possible when you arrive. Dinners that require an oven or a crockpot can be started remotely, ensuring that the meal is ready when you arrive. Security is also made more accessible, with the user able to manage appliances and lights remotely, as well as triggering a smart lock to allow the proper personnel to enter the house even if they don't have a key.

- **IoT in Insurance**

  Even the insurance business can benefit from the Internet of Things explosion. Insurance firms can offer policyholders discounts on IoT wearables like Fitbit. Fitness tracking allows the insurer to offer customized plans and encourage better behaviors, which benefits everyone in the long term, both the insurer and the consumer.

- **IoT in Traffic Monitoring:**

  The Internet of Things, a major contributor to the concept of smart cities, can help control automobile traffic in large cities. Using mobile phones as sensors to collect and share data from our automobiles using applications such as Google Maps or Waze, which are both examples of IoT [11]. It provides information on the traffic conditions on various routes, the projected arrival time, and the distance from the destination, all while contributing to traffic monitoring.

1.3 IoT in Healthcare:

Wearable IoT devices enable hospitals to monitor patients' health at home, minimizing hospital stays and delivering real-time information that could save lives (Wei, 2012). Smart beds in hospitals keep workers aware of their availability, reducing wait times for open space. Putting IoT sensors on vital equipment reduces breakdowns and improves reliability, which can mean the difference between life and death. Elderly care has gotten substantially more comfortable using IoT. In addition to the applications stated above, real-time home monitoring sensors can detect whether a patient has fallen or is having a heart attack.

1.3.1 Security issues and challenges

Medical IoT devices are built for convenience, however because to their simplicity, most do not offer encryption. This means that any time a medical IoT device connects to a hospital network or a healthcare database, it runs the danger of interception or penetration.

- **Patient Confidentiality Risks:**
  Data movement increases the risk of inappropriate [30] access.Even tiny leaks can reveal critical patient information, potentially leading to misuse or public disclosure.

- **Data Breach Incidents:**
  Unfortunately, the healthcare sector has been a target for data breaches .When integrating IoT, the risk is compounded by the increased number of endpoints, each of which is a possible weak link [22]. History has proven that attackers frequently seek out the weakest link in a system, and IoT provides a larger surface to investigate.

- **Risks of Device Hacking:**
  More devices mean more entry points for those with malevolent intent [24]. Each device, whether a heart monitor or a smart inhaler, can be a possible entry point for hackers [25]. When health-related data is at danger, breaches can have serious implications.

- **Insufficient Security Protocols:**
  With so much at stake, one would expect our security systems to be rock solid, right? Surprisingly, this isn't always the case [16] [23]. Many IoT devices in healthcare are installed with default settings or obsolete software, leaving them susceptible.

- **Lack of Standardization:**
  There is no single playbook for IoT in healthcare applications. Manufacturers have their methods, whereas hospitals have theirs [17]. This lack of standardization can result in holes in healthcare security. What are those gaps? They are what hackers live for.

The report is organized as follows: Chapter 2 discusses the related work and Literary survey. Chapter 3 explains the use and advantages of ECC as well as how it works along with briefly describing AES, ASCON and our Security goals. The proposed architecture and the algorithms used along with a detailed explanation of mutual authentication, session key establishment and access control is unraveled in Chapter 4. Chapter 5 introduces the hardware components and discusses the implemented test bed. Chapter 6 presents the results, performance and comparative analysis as well as covers the formal and informal security analysis. The report concludes in Chapter 7 and highlights the future scope.

1.4 Motivation

IoT networks are historically more susceptible to cyber-attacks due to the use of vulnerable wireless mediums for communication, lack of robust security protocols for the resource-constrained environment, and inadequate cyber intelligence amongst the medical fraternity. Due to the rapid evolution of telemedicine practices and digital healthcare that have made their inception, the IT industry did not get ample time to review the security vulnerabilities and privacy loopholes that exist within the deployed IoT networks. The cyber threats to these IoT networks, especially the hospital network, can not only damage hospital property but also breach private and sensitive

information which can potentially endanger the lives of the patients. There are various types of attacks that can impact the hospital network, some of the most widely used attacks are Replay attack, Man-in-the-middle attack, Impersonation attack, privileged insider attack, etc. In most traditional or existing schemes, either some of the mentioned attacks are not prevented or it is prevented but with a high computational and communicational cost which is not effective in IoMT's resource-constrained environment. Therefore, IoT networks' security and privacy have become of the utmost importance with the added condition of being resource efficient.The proposed scheme will follow steps to ensure the security and privacy of data. First, the Gateway registers the sensor nodes followed by the registration of users to ensure security. Then User and IoT sensor nodes mutually verify each other before initiating every new session, followed by the exchange of information. This scheme will not only be lightweight and time-efficient but it will also enhance the security of the system as a whole.

1.5 Objectives

1. To design a system with enhanced security where communication between user, gateway, and sensor happens through insecure or public channels.
2. To develop a lightweight, anonymous, time, and energy-efficient Elliptic Curve Cryptography user authentication scheme for IoT-based healthcare applications.
3. To implement the proposed system in a Hardware test bed and assess its performance and security in real-time.

1.6 Contribution

1. The proposed system's communication takes place in public channels and is safe against various attacks such as Impersonation attack, Man-in-the-middle attack, replay attack, Denial of service attack, etc.
2. The proposed system is lightweight, energy, and time-efficient. It also exhibits most of the essential security properties such as anonymity, privacy, and untraceability.
3. The proposed system has been implemented and tested in real-time on a Hardware test bed.

Chapter 2

Related Works

This chapter discusses the various related works that helped to establish the foundation of our proposed scheme and also the vulnerabilities of those works.

Challa et al [7] proposed an ECC-based user authentication and key agreement scheme; but Jia et al. [18] found the scheme [7] unsafe against impersonation attacks. It is also noticed that the computation and communication cost of [7] is also huge. Chanda et al [8] proposed a variation of Elliptic Curve (ECMQV) based encryption and authentication scheme; however its computational cost was a bit high nor did it provide anonymity . A scheme that depends on IoT-based cloud architectures' authentication is proposed by Zhou et al. [34] . This approach however is also vulnerable against Man-in-the-middle attack, privileged-insider attack, replay attack as well as impersonation attacks. Devi et al [12] presented a message authentication code for IoMT security but it came at a high communicational cost including vulnerability to impersonation and privileged insider attack. Manasha et al [27] proposed a three factor authentication leveraging the secretive nature of ECC; however it has few drawbacks in computational cost as well as vulnerability against offline password guessing attacks [30]. Sharma and Kalra [28] introduced a lightweight user authentication protocol; this approach however is insecure against privileged inside attacks.

Meanwhile Gope and Sikdar [15] designed a lightweight and privacy preserving two factor authentication but their scheme needed more computational cost. A scheme based on lightweight authentication and key management that is effective on IoT applications was presented by Wazid et al. [33]. This scheme uses a one-way cryptographic hash function, bitwise XOR to provide lightweight and secure communication. Masud et al [20] proposed a lightweight user authentication scheme with anonymity preserving property, but it is not practical enough to be used in real world scenarios. Braeken et al [6] proposed a PUF based authentication protocol but it is unable to provide protection against Man-in-the-middle attack.

An ECC and symmetric key encryption based is presented by Sadhukhan et al. [26] despite its robustness against man-in-the-middle attack and replay attack, it was vulnerable against user impersonation and privileged insider attack. It also failed to incorporate sensor addition. Chatterjee et al. [10] proposed an ID based public cryptography; however due to having bilinear pairing it is computationally heavy and inherently complex. An identity based scheme was introduced by Turkanovi et al [31] although this technique is vulnerable to sensor node spoofing

and impersonation attack. An authentication scheme was developed by Shuai et al [29] based on public key cryptosystem, this approach protects against man-in-the-middle attack and replay attack, nevertheless it fails to protect against privileged insider and impersonation attacks. Ali et al [1] developed an improved elliptic curve cryptography(ECC) based scheme, but not only did it have a delayed authentication phase it is also not robust against DoS and replay attacks.

By looking at the table below, it is clear that traditional schemes are insufficient in providing appropriate security or confidentiality. Some are either prone to multiple attacks or don't provide secrecy benefits such as anonymity, untraceability etc. Even if some of them manage to provide the above mentioned benefits, it comes at either a high computational or communicational cost.

Ref.	Threats	NASP	Computational cost	Communicational cost	Year
[33]	2	C	High	Low	2019
[28]	1, 5	C	Medium	High	2019
[20]	2	C	Medium	Medium	2021
[27]	5	C	Medium	Low	2022
[12]	1,2,3	A,C	Medium	High	2023
[8]	1	C	Medium	*	2024

Table 2.1: Literature Review Reference Summary

Acronynms: (*): Not performed, **NASP**:Non-accomplished security properties, **1**: Impersonation, **2**: Privileged insider, **3**: Offline password guessing, **4**: Man-in-the-middle, **5**: Replay, **A**: Untraceability, **B**:Mutual authentication, **C**: Anonymity.

Chapter 3

Preliminaries

This chapter discusses what Elliptic Curve Cryptography (ECC) is, how it works , what are its advantages and how to encrypt and decrypt using ECC. We have also discussed Advanced Encryption Standard(AES), ASCON and mentioned about security goals.

3.1 Elliptic Curve Cryptography

Elliptic curve cryptography (ECC) is a novel public-key encryption technique that takes advantage of the special mathematical characteristics of elliptic curves. ECC uses much smaller key sizes to achieve equivalent security compared to more conventional cryptography systems like RSA. This results in enhanced device performance on platforms with limited resources, faster processing, and lower bandwidth usage. ECC is extremely resistant to decryption attempts [2], even by powerful computers, because its security depends on intricate mathematical operations involving points on these special curves [19]. Therefore, ECC is essential for protecting sensitive data in a variety of applications, such as encrypted private data, secure communication protocols, and digital signatures for blockchain transactions [32]. ECC is an essential tool for data protection in today's increasingly digital world because of its efficiency and resilience.

3.1.1 Advantages of ECC:

- Smaller key sizes: Compared to RSA, ECC can achieve similar security levels with much smaller key sizes. This translates to faster encryption and decryption processes and requires less storage space.
- Efficiency: Due to smaller key sizes, ECC is more efficient for resource-constrained devices like mobile phones and smart cards.

3.1.2 Working of ECC

ECC Key Exchange

- **Global Public Elements**
  - $Eq(a, b)$: Elliptic curve with parameters a, b and q (where q is a prime number or an integer of the form 2^n)
  - G: Point on the curve/elliptic curve whose order is a large value of n

- **User A Key Generation**
  - Select Private Key n_A where $n_A < n$
  - Calculate Public key P_A where $P_A = n_A \times G$
- **User B Key Generation**
  - Select Private Key n_B where $n_B < n$
  - Calculate Public key P_B where $P_B = n_B \times G$
- **Calculation of Secret Key by User A**
  - $K_A = K = n_A \times P_B$
- **Calculation of Secret Key by User B**
  - $K_B = K = n_B \times P_A$

ECC Encryption

- Let the message be M.
- Encode this message M into a point on the elliptic curve.
- Let the point be P_m.
- The point P_m is encrypted now.
- For encryption, choose a random positive integer K.
- The Cipher Point will be $C_m = (KG, P_m + KP_B)$ (For encryption, the public key of B is used).
- The point C_m will be sent to the receiver.

ECC Decryption

- For decryption, multiply the first point/coordinate in the pair with the receiver's secret key, i.e., $KG \times n_B$ (For decryption, the private key of B is used).
- Then subtract it from the second point/coordinate in the pair, i.e., $P_m + KP_B - (KG \times n_B)$.
- But we know $P_B = n_B \times G$, so,
- $P_m + KP_B - KP_B = P_m$ (Original Point).
- So, the receiver gets the same point.

3.2 ASCON

ASCON is a promising algorithm for lightweight cryptography, addressing the security needs of resource-constrained devices in the ever-growing world of IoT. It offers a balance between security and efficiency, making it a valuable tool for protecting data in connected devices

- **Two Main Functions:** ASCON comes in two flavors:
  - **Authenticated Encryption (AEAD):** This encrypts data to ensure confidentiality and also includes a tag to verify the data's integrity during transmission. This is crucial for IoT devices where data security is important.
  - **Hashing:**This function takes an arbitrary amount of data and generates a fixed-size fingerprint. It's useful for data verification and integrity checks.

3.3 XOR

XOR is a fundamental encryption method in cryptography. It is a versatile tool that offers a simple but effective way to scramble and unscramble data with the right key. Two of the major reasons it is widely used are its reversibility and secrecy, which make it simple and effective for cryptographic operations. It works by combining a message (plaintext) with a secret key (often a random number) using the XOR operation. This operation scrambles the data, making it unreadable.

3.4 Security and Other Goals

The proposed scheme exhibits prominent security properties. We adopted the security properties from:

1. **Two-way authentication and key establishment:** With sensitive data and life-saving machines (e.g., ventilators) at stake, the hospital network demands a two-way authentication system (mutual authentication) and key agreement between user devices and sensor nodes to establish a secure connection.
2. **Data Confidentiality:** In a hospital network, data is the lifeblood. Every byte, from user details (e.g., user IDs, passwords, etc.) to encryption keys, must flow confidentially, like a doctor whispering a diagnosis in a patient's ear. Breaches can be fatal and can scatter sensitive information like IDs, passwords, and encryption keys, which can endanger patients' lives, damage trust, and tarnish the hospital's reputation.
3. **Message Integrity and Freshness:** In the hospital network, the medical data should be kept updated with accurate information. Tampering with medical data can be catastrophic. To safeguard lives and reputations, secure e-healthcare protocols are essential to ensure that data remains untainted (fresh) and authentic.
4. **Anonymity of Identity:** Intruders are always looking for loopholes and important data like passwords, user IDs, etc. Concealing the network device identities during message exchange can prevent credential interception and potential MITM/impersonation attacks. Therefore, it is imperative to implement protocols that verify access without revealing device identities, thwarting attempts to steal credentials for malicious activities.

5. **Lightweightness:** IoT nodes are resource-constrained, i.e., the nodes can execute only limited computations. Therefore, the security protocols must be constructed with lightweight cryptography primitives (e.g., hash and bitwise XOR) to extend devices' active lifetime.

6. **Insecure Channel Communication:** In real-world scenarios, the data is sent through an insecure channel. Cryptography techniques like ECC (Elliptic Curve Cryptography) are ideal for securing communication in public channels, protecting sensitive medical records using complex mathematical curves that create public and private keys so entangled that eavesdroppers would need to solve impossible equations to decipher the message.

3.5 Some Important Terms

- **Hash Function:** In cryptography, hash functions act like digital fingerprints that can turn any data into a unique, fixed-size code. This code is used to safeguard data integrity (detecting alterations) and verify message authenticity (ensuring the sender and content are legit), all in just a few lines of code. It is like a cryptographic secret handshake, ensuring only the right parties whisper in the digital shadows.

- **Nonce:** In cryptography, a nonce is used like a random temporary value sprinkled into messages. Each unique nonce adds freshness, like a constantly changing password, keeping communication secure and bad guys guessing.

Notation	Description
U_{ID}, S_{ID}	User identity, Sensor node (SN) identity
P_{WD}	Device password set by user
R_{req}	Registration request
$\mathrm{R}_{SG}, R_{SN}, K$	Random secret generated by gateway, Sensor Node and device
h(.), $\|\|$	Hash function, concatenation operator
$\oplus$, SK	Bitwise XOR operation, Session key
$\mathrm{S}_{TID}, U_{TID}$	Temporary identity of sensor node and user (device)
$\mathrm{N}_{D}, N_{G}, N_{S}$	Nonce generated by device, gateway, and sensor node respectively
P_m	Point on the elliptic curve
C_m	Cipher point
IN	Information or data

Table 3.1: Notations and Descriptions

Chapter 4

Proposed Architecture

This chapter introduces our system model, authentication architecture and discusses the various algorithms used in our project along with a detailed explaination of our methodology containing user registration, sensor registration and Mutual Authentication between user, gateway and Sensor node.

4.1 Assumptions

We make the following assumptions:

- The IoT sensor node, user device, and gateway can all carry out cryptographic activities.
- The gateway is connected to the main power supply and has no computational or storage limitations, while the user device is a regular computational device and the IoT sensor nodes are resource-constrained.
- The combination of a temperature sensor with a raspberry pi is known as an IoT sensor node.
- A public (insecure) channel is used to transmit the messages after and during user registration, but the communication channel during used sensor node registration is secure.

4.2 System Model

As shown in Fig. 4.1, the proposed scheme is based on smart hospital and integrated IoT sensor nodes. At first, the sensor node is registered at the gateway through a secure channel using lightweight XOR followed by the registration of the user at the gateway using Elliptic Curve Cryptography through an insecure or public channel. Due to the nature of the proposed scheme's security measures, it is mandatory to register the sensor before the user. After the registration phase is over, we move on to the authentication phase which takes place through an insecure or public channel.

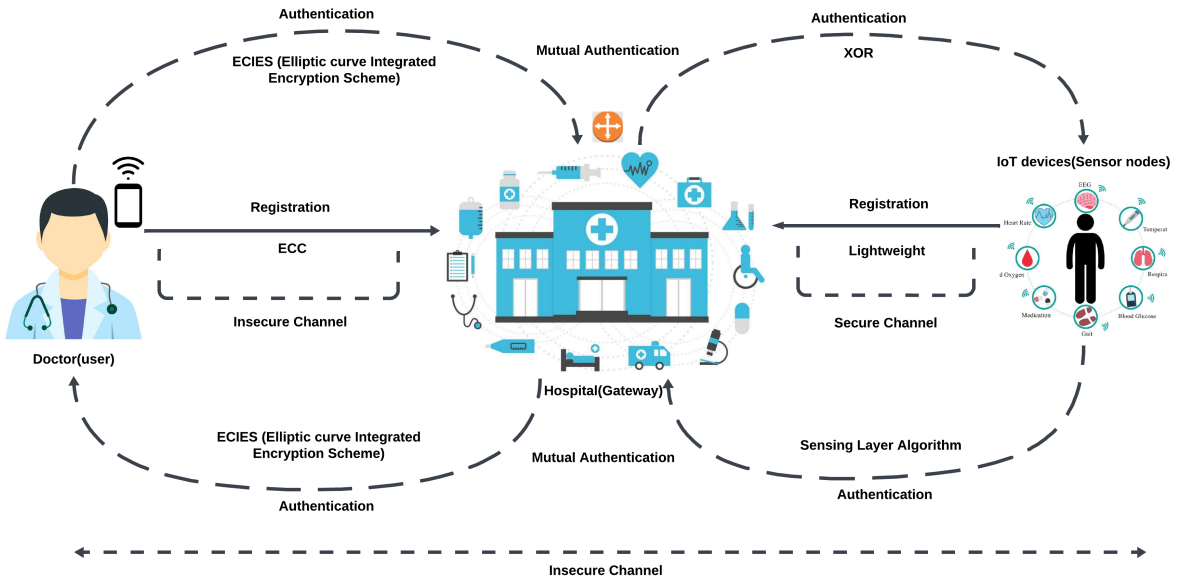

Figure 4.1: System Model for Proposed Scheme

4.3 Authentication Architecture

The Fig. 4.2 shown below is the architecture of the authentication phase of our project. In this phase, the user attempts to access the sensor's data by inputting their ID, password, and choosing the desired sensor. The request is encrypted (Algorithm 1) and sent to the gateway using ECIES where it is decrypted (Algorithm 2), verified, and authenticated. The gateway then generates the session key and encrypts (Algorithm 3) it along with the Sensor ID and nonce using XOR operation and sends the resultant intermediate value to the sensor node.

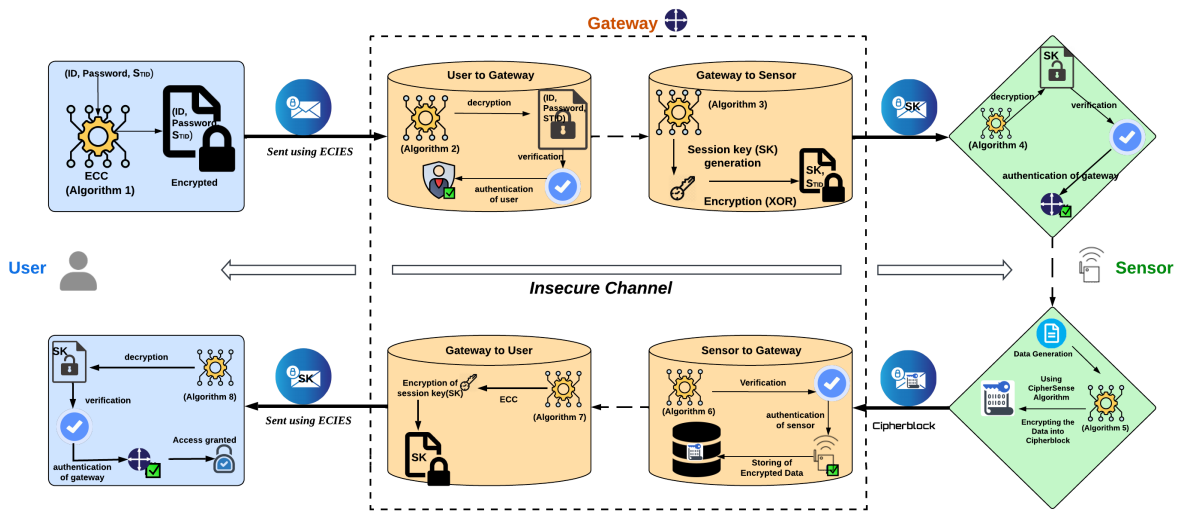

Figure 4.2: The architecture of the authentication phase

The sensor upon receiving the message, decrypts (Algorithm 4) it, verifies the contents followed by authentication of the gateway. Afterwards, the sensor node encrypts (Algorithm 5) the data with the received session key using our newly proposed lightweight algorithm called

CipherSense, where the data is converted into ciphered blocks. The cipher block is sent to the gateway, which verifies and authenticates the sensor node (Algorithm 6) and stores the encrypted data. The gateway then retrieves the session key from its database, encrypts (Algorithm 7) and sends it using ECIES to the user. The user decrypts (Algorithm 8) the message, verifies its contents, and authenticates the gateway, which indicates the success of mutual authentication. Now that the user has received the session key, it has been granted access to decrypt and view the stored encrypted data.

4.4 Algorithms Used

The number and types of algorithms used during the authentication phase are shown below in Fig. 4.3 along with a brief explanation of these algorithms.

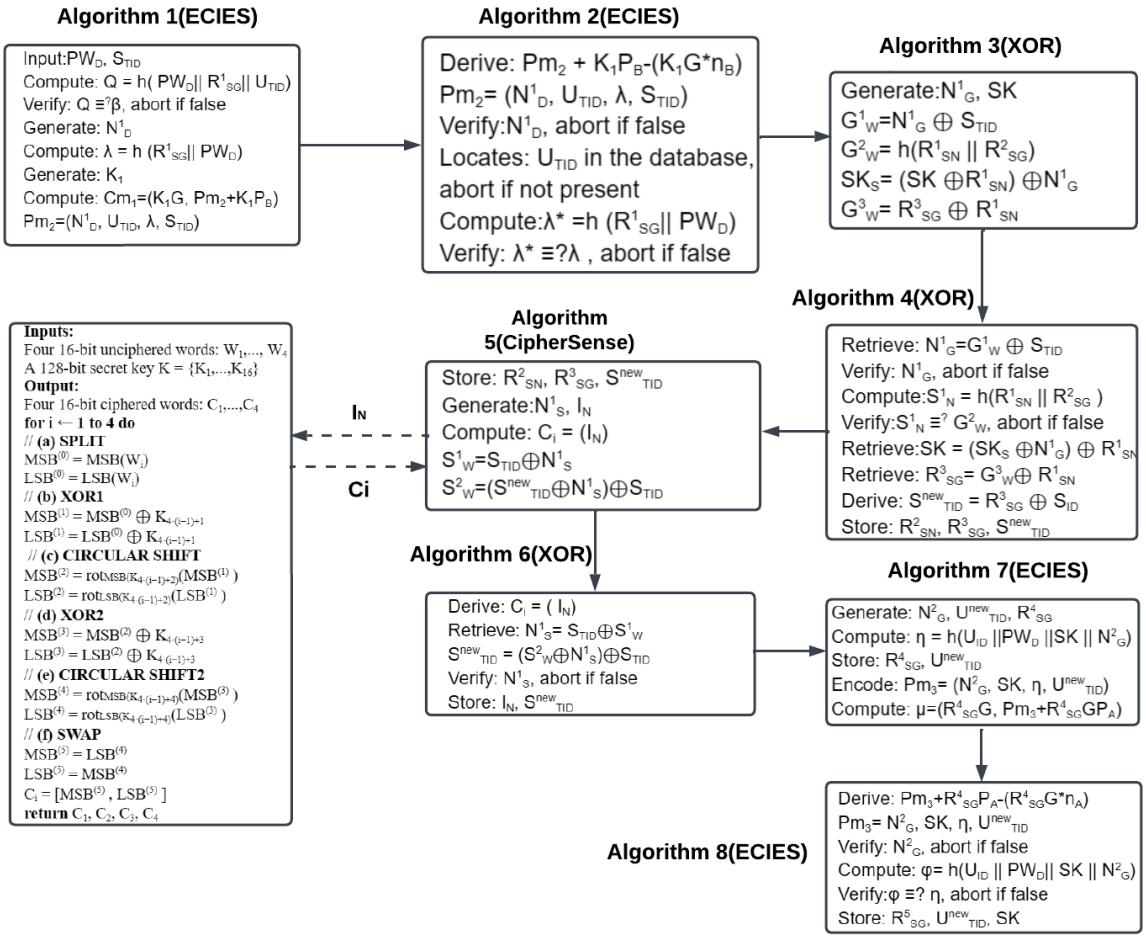

Figure 4.3: Different types of algorithms used during the authentication phase.

- **Algorithm 1 (ECIES):** Elliptic Curve Integrated Encryption Scheme, also known as ECIES, is a hybrid encryption algorithm that combines the benefits of elliptic curve cryptography (ECC) and symmetric-key cryptography.Here we used ECC to encrypt the contents of the message and sent it to the gateway using ECIES.Since practically ECC can't be directly used to send encrypted messages, we had to use a hybrid scheme like ECIES to send the encrypted message.

- **Algorithm 2 (ECIES):** Similar to Algorithm 1, we used ECIES here, but instead of using it for encryption, we used it for decryption.After the verification and authentication of the user are completed, the next algorithm, i.e., Algorithm 3, starts working.
- **Algorithm 3 (XOR):** The algorithm used here is called XOR. It is a Boolean logic operation that is used to compare two input bits and generate one output bit; if the bits are the same, then the result is 0. XOR is an encryption method widely used in cryptography as a standalone algorithm as well as part of a complex algorithm.The reason we used XOR here is because of its lightweight nature and the security it provides against brute force attacks.
- **Algorithm 4 (XOR):** As mentioned earlier in Algorithm 3, we used XOR here as well, but unlike in the previous algorithm, where we used XOR for encryption, here we used it for the purpose of decryption.So the encryption part of the XOR operation is done in Algorithm 3 and the decryption here.This not only secures the contents of the message but is also lightweight, which is necessary because of our resource constraint sensor node.
- **Algorithm 5 (CipherSense):** CipherSense is a simple symmetric and lightweight encryption algorithm specifically suited for WSNs. It uses a 128-bit key to encrypt and decrypt data blocks.The algorithm can be thought of as a simplified version of AES and ASCON-128, developed with the main aim of reducing encryption time in a resource-constrained environment.The algorithm has been implemented in Raspberry PiOS and tested with Raspberry Pi 3.

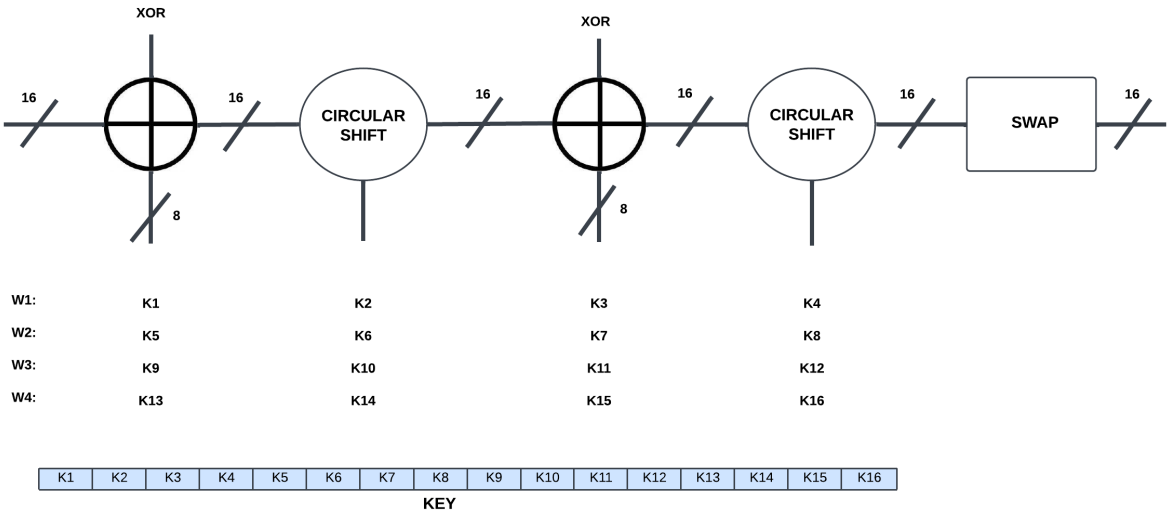

Figure 4.4: A simplified block diagram of CipherSense algorithm

A simplified block diagram of our proposed lightweight algorithm is shown above in Fig. 4.4. The 128-bit secret key is logically decomposed into 16 subkeys of 8 bits each, i.e., K1,...,K16, which are used to encrypt blocks of data. More precisely, we assume that the data is logically decomposed into blocks of four words of 16 bits each, i.e., W1,..., W4. Each block is then processed in six consecutive steps to obtain an enciphered block, i.e., C1,...,C4. In every step of our proposed algorithm, a 16-bit word is transformed by a function $f()$ to obtain another 16-bit word.

CipherSense Algorithm: Encryption Algorithm for the Sensor Side

Inputs:

- Four 16-bit unciphered words: $W1, ..., W4$
- A 128-bit secret key $K = \{K1, ..., K16\}$

Output:

- Four 16-bit ciphered words: $C1, ..., C4$

1. **for** $i \leftarrow 1$ to 4 **do**
   (a) SPLIT
      - $MSB(0) = MSB(Wi)$
      - $LSB(0) = LSB(Wi)$

   (b) XOR1
      - $MSB(1) = MSB(0) \oplus K4(i-1)+1$
      - $LSB(1) = LSB(0) \oplus K4(i-1)+1$

   (c) CIRCULAR SHIFT
      - $MSB(2) = rotMSB(K4(i-1)+2)(MSB(1))$
      - $LSB(2) = rotLSB(K4(i-1)+2)(LSB(1))$

   (d) XOR2
      - $MSB(3) = MSB(2) \oplus K4(i-1)+3$
      - $LSB(3) = LSB(2) \oplus K4(i-1)+3$

   (e) CIRCULAR SHIFT
      - $MSB(4) = rotMSB(K4(i-1)+4)(MSB(3))$
      - $LSB(4) = rotLSB(K4(i-1)+4)(LSB(3))$

   (f) SWAP
      - $MSB(5) = LSB(4)$
      - $LSB(5) = MSB(4)$

   (g) $Ci = [MSB(5), LSB(5)]$
2. **return** $C1, C2, C3, C4$

Henceforward, we use the notation $[MSB(k+1), LSB(k+1)] = f([MSB(k), LSB(k)])$ to indicate that in the step k the most significant byte (MSB) and least significant byte (LSB) are transformed by the function f(). Moreover, we indicate with rotn() a cyclic shift of n bits. Basically, the algorithm consists of six steps, i.e., (a)–(f), and, according to the previous notation, can be summarized as follows.

I. SPLIT: Each input word is logically split into two parts corresponding to the MSB and LSB, i.e., [MSB(0), LSB(0)] = [MSB(Wi), LSB(Wi)].

II. XOR1: The word [MSB(0), LSB(0)] is XORed with subkeys $K_{4(i-1)+1}$.

III. CIRCULAR SHIFT: The subkey $K_{4(i-1)+2}$ is split into two parts, [MSB($K_{4(i-1)+2}$), LSB($K_{4(i-1)+2}$)] which are used to rotate the word [MSB(1), LSB(1)].

IV. XOR2: Another XOR operation is performed so that bytes MSB(3) and LSB(3) are both XORed with the subkey $K_{4(i-1)+3}$.

V. CIRCULAR SHIFT2: The subkey $K_{4(i-1)+4}$ is split into two parts, [MSB($K_{4(i-1)+4}$), LSB($K_{4(i-1)+4}$)] which are used to rotate the word [MSB(1), LSB(1)].

VI. SWAP: Finally, the ciphered word C_i is obtained by swapping least and MSBs obtained in the previous step, i.e., C_i = [MSB(5), LSB(5)] = [LSB(4), MSB(4)].

- **Algorithm 6 (XOR):** Here we have used the XOR operation again to retrieve the nonce and new temporary sensor id sent from the sensor node, but the received encrypted data is not decrypted and instead stored directly in the gateway's database.
- **Algorithm 7 (ECIES):** As mentioned in Algorithms 1 & 2, we use ECIES here to encrypt the session key along with the nonce and new temporary Id of the user and send it to the user.
- **Algorithm 8 (ECIES):** Finally, at the user's end, we again use the ECIES algorithm to decrypt, verify, and authenticate the encrypted message sent from the gateway. This also means that the user has now been granted access to decrypt and view the data.

4.5 Methodology

4.5.1 Sensor node registration phase

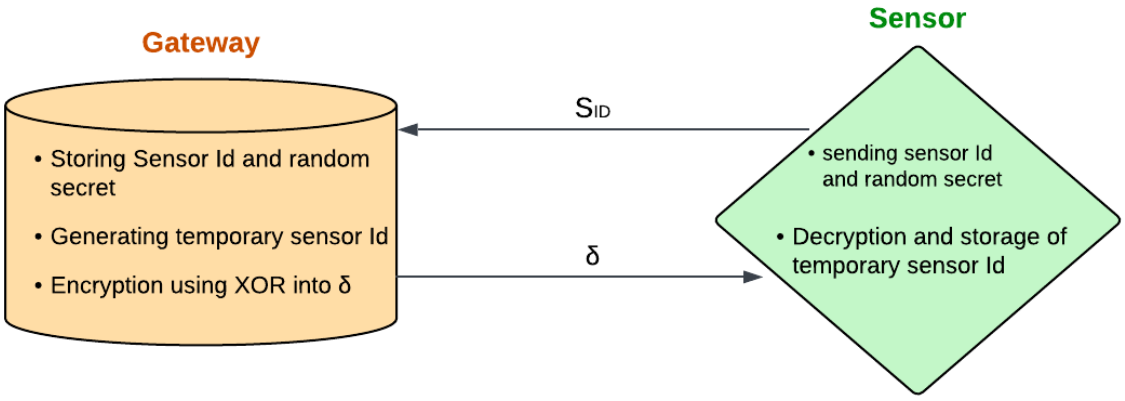

Figure 4.5: Sensor node registration phase (Illustration)

The sensor node has to register itself in the gateway before being able to deliver real-time information of the patient to the user. Fig.4.5 and Fig.4.6 illustrate the sensor node registration phase.

Step 1: The sensor node creates the random secret value R_{SN}^1 after retrieving the identity information S_{ID} from its processor. The sensor node then concatenates S_{ID} and R_{SN}^1 and transmits it over the secure channel to the gateway.

Step 2: The gateway saves the S_{ID}, R^1_{SN}, upon reception. The new random secret value R^2_{SG} is then generated by the gateway. After calculating $\delta = (S_{\text{ID}} \oplus R^2_{\text{SG}}) \oplus R^1_{\text{SN}}$ temporary identity and $S_{\text{TID}} = R^2_{\text{SG}} \oplus S_{\text{ID}}$, the gateway stores R^2_{SG} and S_{TID} in its memory. Ultimately, the sensor node receives δ from the gateway via a secure channel.

Step 3: Step 3: After receiving a message from the gateway, the sensor node calculates $R^{2*}_{SG} = (\delta \oplus R^1_{SN}) \oplus S_{ID}$. Finally, the sensor node saves the values R^1_{SN}, R^{2*}_{SG} and S_{ID} and obtains its temporary identity from $R^{2*}_{SG} \oplus \text{S}_{\text{ID}}$.

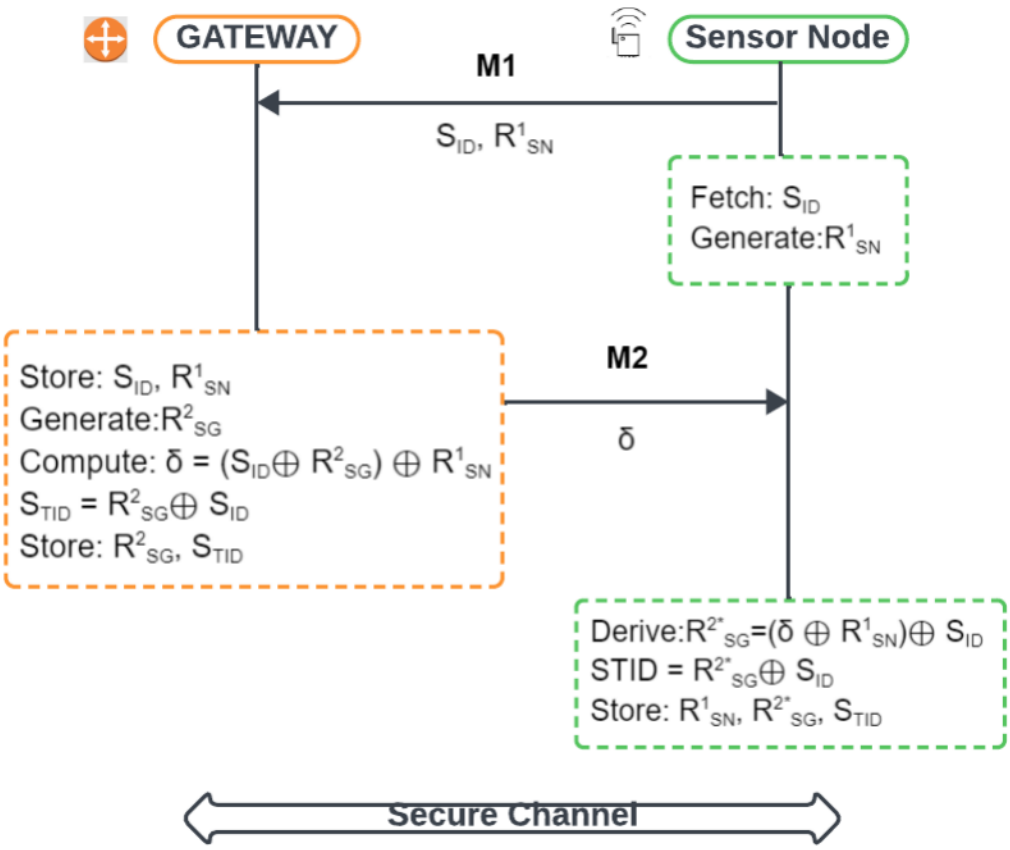

Figure 4.6: Sensor node registration phase (Algorithm)

4.5.2 User registration phase

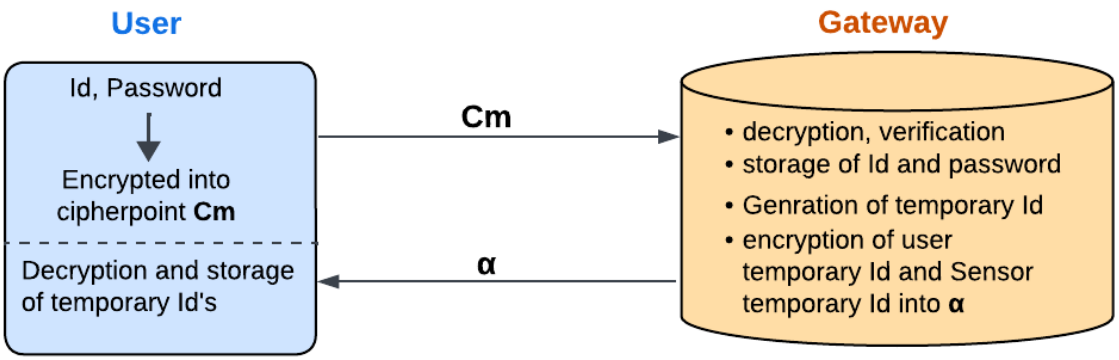

Figure 4.7: User registration phase (Illustration)

To access real-time patient health data, a user must register their device with the gateway. The user registration phase's steps are shown in Fig. 4.7 and Fig. 4.8

Step1: The device creates K and computes $C_m = (K.G, P_m + K.P_B)$ where $P_m = (U_{\text{ID}}, \text{P}_{\text{ID}})$ and transmits request R_{req} when the user inputs their user id (U_{ID}) and password (P_{WD}). The request is sent to the gateway over the insecure channel.

Step2: Using the private key from $P_m + K.PB - (K.G.n_B)$, the gateway extracts the user's U_{ID} and P_{WD} and saves it for further use. After generating R^1_{SG} (random secret generation) and the user's temporary ID (U_{TID}), the gateway computes $\alpha = (R^1_{SG}G, Pm_1 + R^1_{SG}G.P_A)$, where Pm_1 is equal to $(R^1_{SG}, U_{TID}, S_{TID})$. Subsequently, the gateway transmits α to the user via the unsecure channel after storing the values for R^1_{SG} and U_{TID} internally.

Step3 After obtaining the value α from the gateway, the user uses $Pm_1 + R^1_{SG}.P_A - (R^1_{SG}G.n_A)$ to obtain R^1_{SG} and the temporary user identity (U_{TID}) by utilizing its private key. R^1_{SG} and U_{TID} values are kept in storage. Next, for future reference, the user computes and saves $\beta = h(P_{WD}||R^1_{SG}||U_{TID})$.

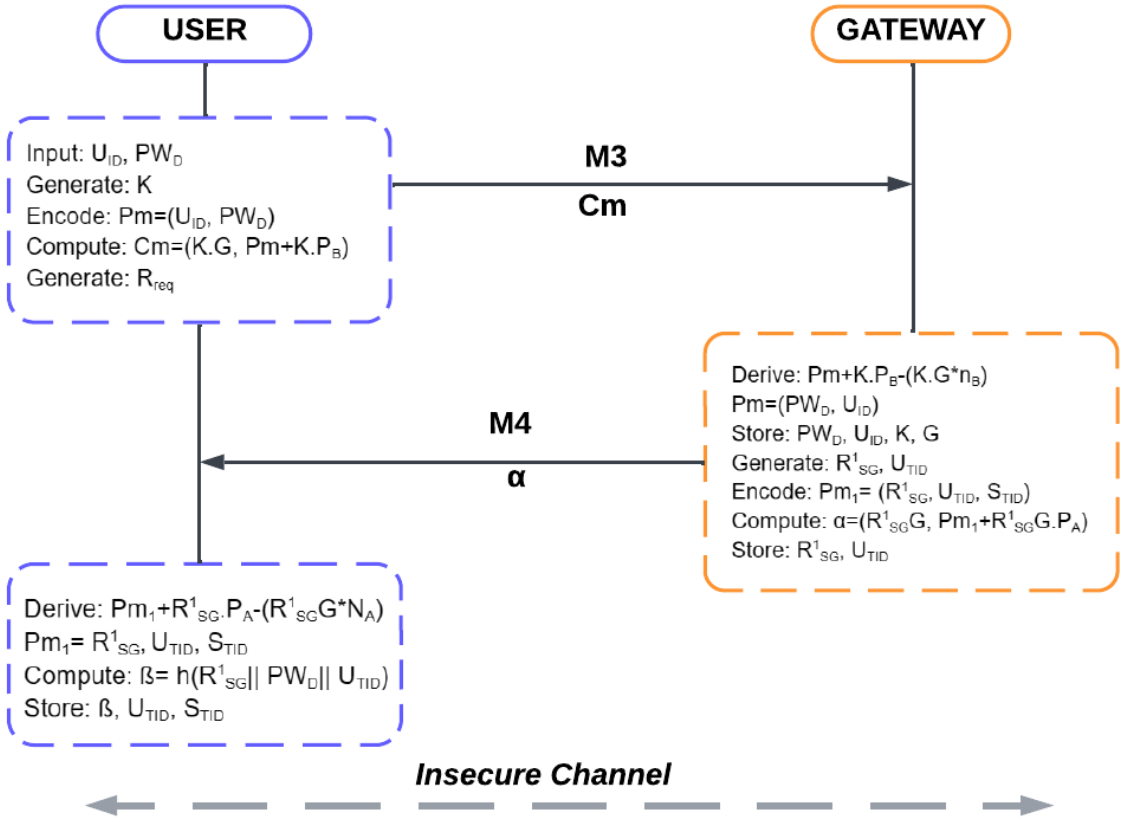

Figure 4.8: User registration phase(Algorithm)

4.5.3 Mutual authentication and key agreement phase

Consider a User who intends to observe instantaneous information of a particular IoT sensor node embedded on the medical appliance. But before the exchange of information, the User and sensor node verify the authenticity of each other. Fig. 5.5 and Fig. 5.6 show the complete process of mutual authentication and key agreement.

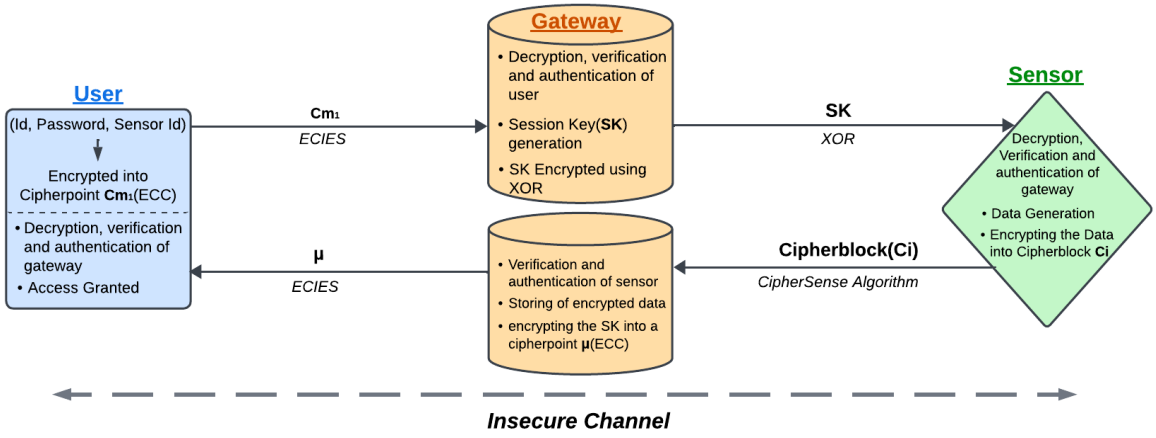

Figure 4.9: Mutual authentication and key agreement phase (Illustration)

User to Gateway

Step1: After entering the device's password PW_D, the user calculates $Q = h(PW_D||R^1_{SG}||U_{TID})$. Currently, the device belonging to the user confirms Q in relation to β, which was saved in the device during the registration stage. The device proceeds with the operation if Q and β have equal values; otherwise, it terminates. The User's device then computes $\lambda = h(R^1_{SG}||PW_D)$ after generating the nonce N^1_D. Eventually, it produces K_1 once more and calculates Cm_1 = $(K_1G, Pm_2 + K_1P_B)$, where $Pm_2 = (N^1_D, U_{TID}, \lambda)$. The message Cm_1 is sent to the gateway by the user's device.

Gateway to Sensor

Step2: Using the private key from $Pm_2 + K_1P_B - (K_1G.n_B)$, the gateway first determines the values N^1_D, U_{TID}, and λ after receiving information from the user. Next, the gateway verifies that the nonce N^1_D is current. If fresh detected, the process proceeds; if not, it terminates. The process is either continued or canceled depending on whether the U_{TID} can be found in the database and compared with the received ones. After that, the gateway creates λ^* by computing the hash of $h(R^1_{SG}||PW_D)$. The λ is now taken out of the gateway's database and compared to the computed λ^*. Identity leads to the user's successful gateway authentication. After generating the nonce N^1_G and computing the XOR of N^1_G and S_{TID}, the gateway saves the result in G^1_W. The gateway then forms G^2_W by calculating $h(R^1_{SN}||R^2_{SG})$. Additionally, the gateway creates the sensor node's session key and encloses it covertly inside $SK_S = (SK \oplus R^1_{SN}) \oplus N^1_G$. The $(G^3_W = R^3_{SG} \oplus R^1_{SN})$ is calculated and stored by the gateway. The information G^1_W, G^2_W, U_{TID}, SK_S, and G^3_W is now sent by the gateway to the sensor node via a public channel.

Sensor to Gateway

Step3: The sensor node takes the nonce N^1_G from $G^1_W \oplus S_{TID}$ upon receiving and confirms the freshness. If a fresh nonce is found, the process continues; if not, it ends. After that, the sensor creates the S^1_N by computing $h(R^1_{SN}||R^2_{SG})$. The sensor checks to see if S^1_N and G^2_W are the same; identical values show that the gateway's authentication at the sensor node was successful. Session key (SK) is retrieved by the sensor node from $(SK_S \oplus N^1_G) \oplus R^1_{SN}$.The nonce N^1_S and Data or Information In are generated by the sensor node. Along with retrieving the R^2_{SG} from $G^3_W \oplus R^1_{SN}$, the sensor node also calculates the new temporary ID of the sensor node $S^{new}_{TID} = R^3_{SG} \oplus S_{ID}$ and saves the R^2_{SN}, R^3_{SG}, and S^{new}_{TID} . Finally the sensor encrypts the data with the received session key(SK) using our newly proposed lightweight Cipher Sense algorithm that converts the data into an encrypted CipherBlock(C_i) and sends Ci, N^1_S to the gateway.

Gateway to User

Step4: Once the gateway receives the message from the sensor node, it extracts the nonce N^1_S and S^{new}_{TID}. Next, it is confirmed that the nonce is current. If confirmed, it also shows that the sensor node and gateway's mutual authentication procedure was successful. The user computes $\eta = h(U_{ID}||PW_D||SK||N^2_G)$ in the following step, in which the gateway creates nonce N^2_G and retrieves the session key SK from its storage. The user device U^{new}_{TID} and

R^4_{SG} then generate a new temporary identity through the gateway. The gateway computes $\mu = (R^4_{SG}G, Pm_3 + R^4_{SG}GP_A)$, where $Pm_3 = (N^2_G, SK, \eta, U^{new}_{TID})$, and saves R^4_{SG} and U^{new}_{TID}. Ultimately, the gateway uses ECIES to send the message μ over an insecure public channel.

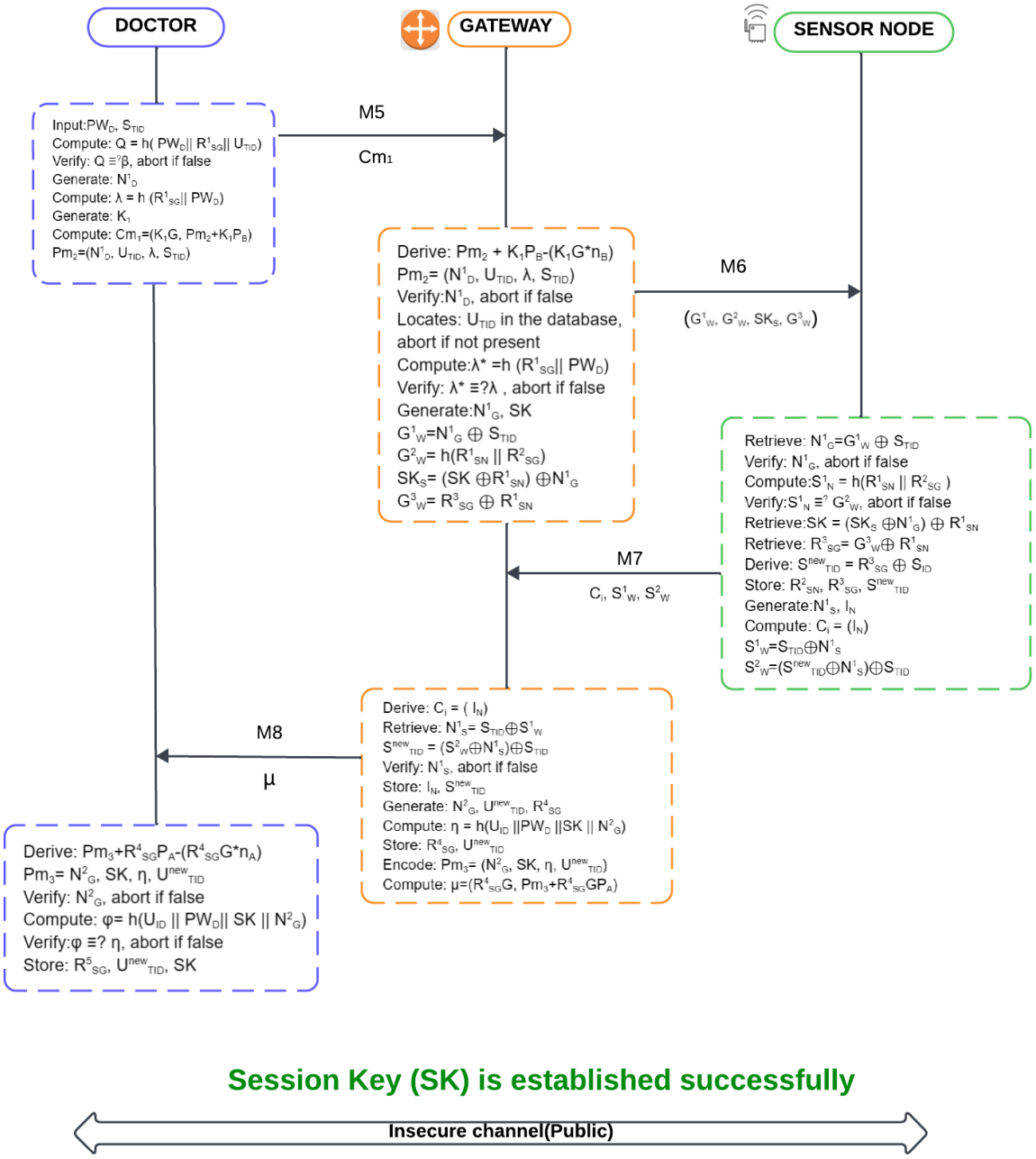

Figure 4.10: Mutual authentication and key agreement phase (Algorithm)

User Access Granted

Step5: Step 5: The user device uses the private key from $Pm_3 + R^4_{SG}P_A - (R^4_{SG}G.n_A)$ to get the values N^2_G, SK, η, and U^{new}_{TID}. The gateway then confirms that the nonce N^2_G is fresh; if it is, the operation proceeds; if not, it is terminated. The gateway computes $h(U_{ID}||PW_D||SK||N^2_G)$ in addition, storing the outcome in φ. Next, φ and η are contrasted; comparable values signify the

successful completion of mutual authentication between the user and the gateway. Additionally, it shows that the user has been authorized access to the encrypted data, which is accessible with the session key (SK) that they have been given. The values R_{SG}^{4} and U_{TID}^{new} are then stored by the device in its memory.
Finally, between the sensor node and the user device, the secret session key (SK) is created efficiently.

Chapter 5

Hardware implementation

This chapter introduces the hardware components used in our project and discusses the implemented test bed.

5.1 Hardware Components

The hardware configuration is made up of various integrated components that allow for secure communication, data processing, and visualization [14] [21]. The key components are the Raspberry Pi 3 Model B as the edge node, the Acer Nitro 5 laptop as the central gateway, the ASUS TUF Gaming laptop as the user device, and the HP V202b monitor for display purpose and some connecting wires.

1. Raspberry Pi 3 Model B

The Raspberry Pi is a microprocessor and a compact, play card-sized single-board computer that has provisions just like a desktop computer. In our project, we have used the Raspberry Pi 3 Model B, which features a quad-core 64-bit ARM Cortex-A53 processor running at 1.2GHz along with 1GB of LPDDR2 SDRAM memory. This compact-size device is used as an edge node in our project as a go-between the sensors and the gateway. Its specifications, like the ARM processor, allow it to efficiently handle sensor data processing, encryption, and communication tasks. An SD card (16 GB) inserted in the Raspberry Pi is installed with the Raspberry Pi OS (formerly known as Raspbian).

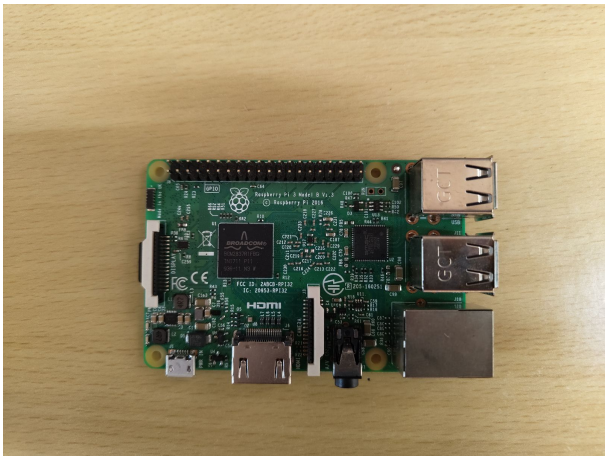

Figure 5.1: Raspberry Pi 3 Model B

2. DHT11 Sensor

The DHT11 is a resource-constrained sensor that measures temperature and humidity digitally and is widely used in various IoT-based applications. This small-sized sensor includes a capacitive humidity sensor and a thermistor.

Specifications:

- Temperature range: 0°C to 50°C (32°F to 122°F)
- Humidity range: 20% to 90% relative humidity
- Temperature accuracy: ±2°C
- Humidity accuracy: ±5% RH
- Sampling rate: 1 Hz (once per second)
- Operating voltage: 3.3V to 5.5V
- Digital output with single-wire serial interface

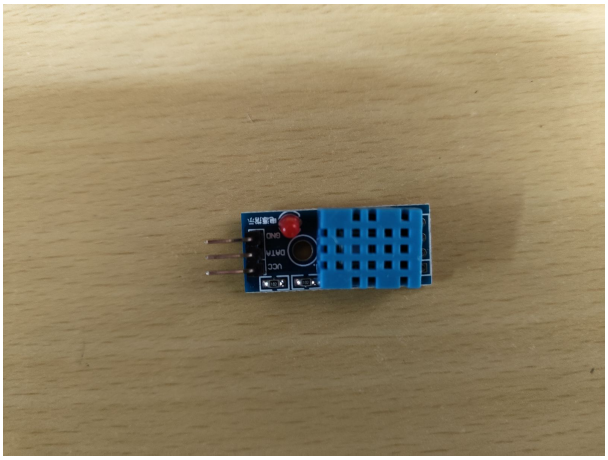

Figure 5.2: DHT11 Sensor

3. Laptops

a. Acer Nitro 5 (Gateway)

The Acer Nitro 5 laptop is used as the central gateway in our project, responsible for managing the registration, encryption, decryption, and authentication processes required for secure communication with sensors and users.

Specifications:

- Processor: Intel Core i5 10th Generation
- RAM: Up to 32GB DDR4 RAM
- Graphics: NVIDIA GeForce GTX
- Display: 15.6-inch Full HD IPS
- Storage: High-speed 256 GB SSD and 1 TB HDD
- OS: Windows 11

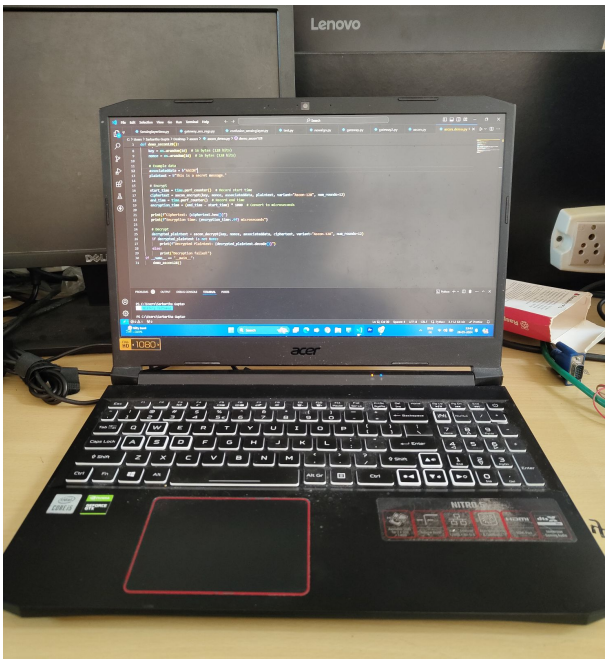

Figure 5.3: Acer Nitro 5 (Gateway)

b. ASUS TUF F-15 (User)

The ASUS TUF Gaming laptop is used as the user device. Users authenticate themselves with the gateway (Acer Nitro 5) using secure encryption techniques. Once authenticated, users can access and decrypt the encrypted sensor data stored on the gateway, enabling them to analyze or monitor the collected data securely.

Specifications:

- Processor: Intel Core i5 10th Generation
- RAM: Up to 32GB DDR4 RAM

- Graphics: NVIDIA GeForce GTX
- Display: 15.6-inch Full HD IPS
- Storage: High-speed 512 GB SSD
- OS: Windows 11

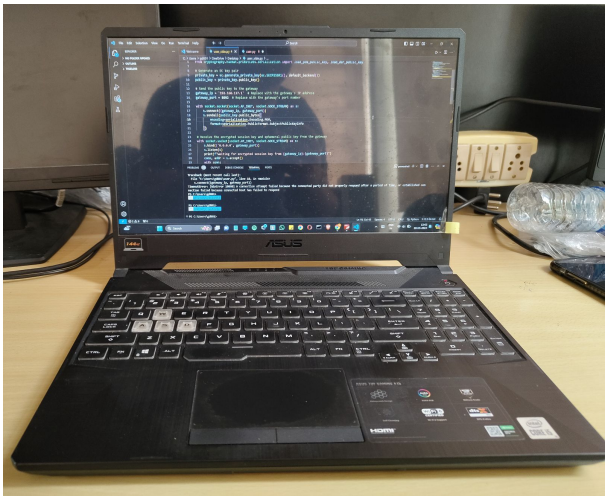

Figure 5.4: Asus F-15 (User Device)

4. Monitor

In our setup, the HP V202b monitor is linked to the Raspberry Pi 3 Model B, which functions as an edge node for processing and transferring sensor data. The project's procedures and data can be seen on the Raspberry Pi by directly connecting its HDMI output to the monitor's HDMI input connection.

Specifications:

- Size of Display: 19.5 inches
- Resolution: 1600 x 900 pixels
- Aspect Ratio: 16:9
- Brightness: 200 cd/m^2
- Contrast Ratio: 600:1
- Response Time: 5 ms (gray to gray)
- Input Ports: VGA
- Stand Adjustments: Tilt (-5° to +22°)

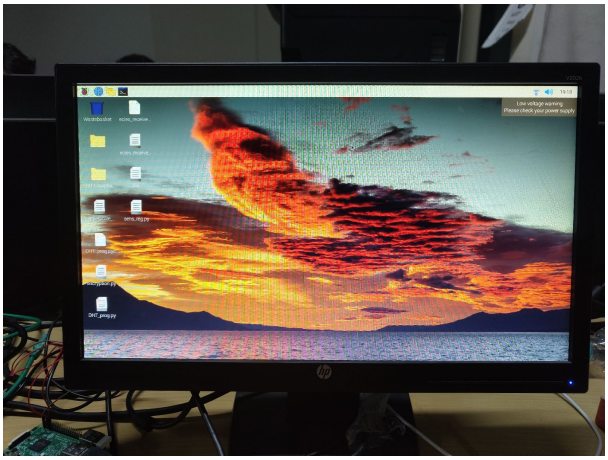

Figure 5.5: Monitor

5. Connecting Wires

a. Male and Female Wires

To link different components in our project, we use male and female wires. These cables come in various colors for convenient identification and include a conductive core encased in insulation.

b. HDMI to VGA Converter

It is used to connect the Raspberry Pi 3 Model B with Monitor.

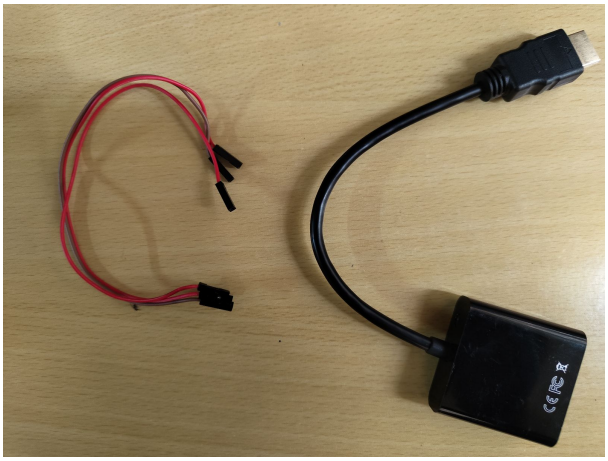

Figure 5.6: Connecting Wires

5.2 Hardware Implementation Setup

The DHT11 sensor uses a single-wire bi-directional serial interface to connect with the Raspberry Pi. Three connections are needed for the sensor: data (DATA), ground (GND), and power (VCC). It is to be noted that the DHT11 sensor's DATA pin needs to be linked to a GPIO pin. The majority of the GPIO pins on the Raspberry Pi 3 Model B feature integrated pull-up and

pull-down resistors that may be turned on or off via software.

In summary, the DHT11 sensor and the Raspberry Pi 3 Model B often link to the following:

- DHT11 VCC pin → Raspberry Pi 3.3V pin (Pin 1)
- DHT11 GND pin → Raspberry Pi Ground pin (e.g., Pin 6, 9, 14, 20, 25, 30, 34, or 39)
- DHT11 DATA pin → Raspberry Pi GPIO 4 pin (Pin 7)

Then, an HP V202b 19.5-inch LED-backlit LCD display is connected to the Raspberry Pi in order to visualize the project's procedures and data. The HDMI to VGA converter is used to make it very easy to connect the Raspberry Pi's digital HDMI output to the monitor's analog VGA input, allowing real-time sensor data, encryption procedures, and communication status to be displayed.
The Acer Nitro 5 laptop, configured as the central gateway, and the ASUS TUF F-15 laptop, serving as the user device, are wirelessly linked to the Raspberry Pi and the HP V202b monitor. This wireless connectivity enables secure communication between the edge node (Raspberry Pi), the central gateway, and the user device, allowing encrypted sensor data and authentication to be transmitted.
By connecting these hardware components via wired and wireless connections, the setup creates a secure and efficient communication framework for collecting, processing, transmitting, and visualizing sensor data while maintaining data confidentiality, integrity, and authentication throughout the process.

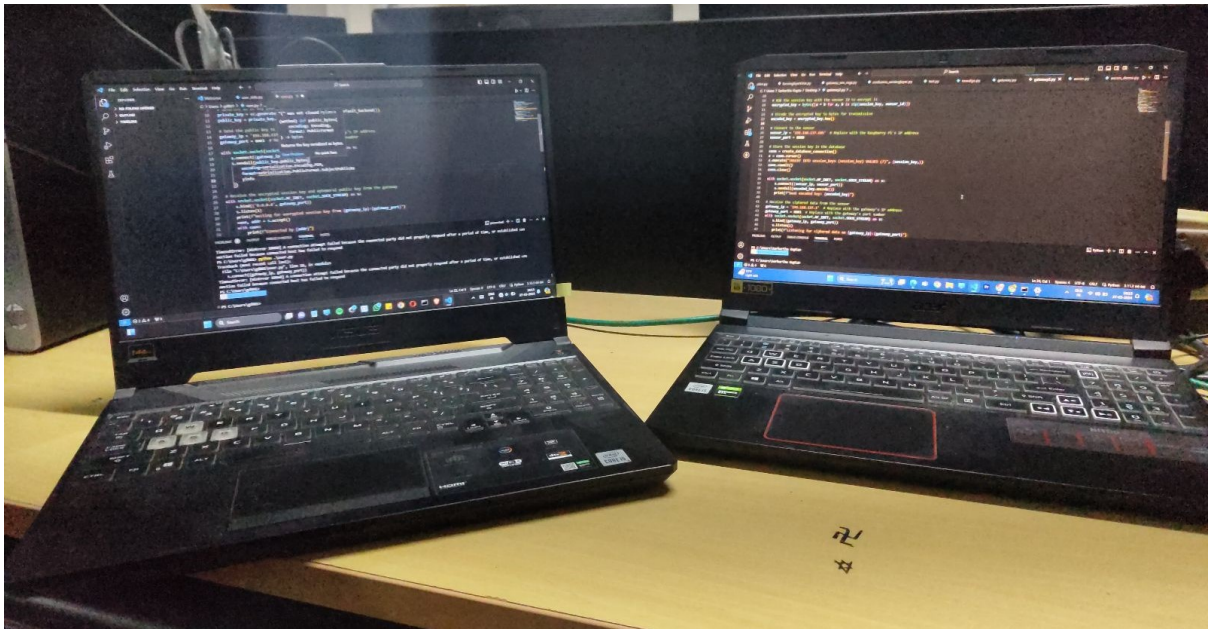

Figure 5.7: Hardware Implementation 1

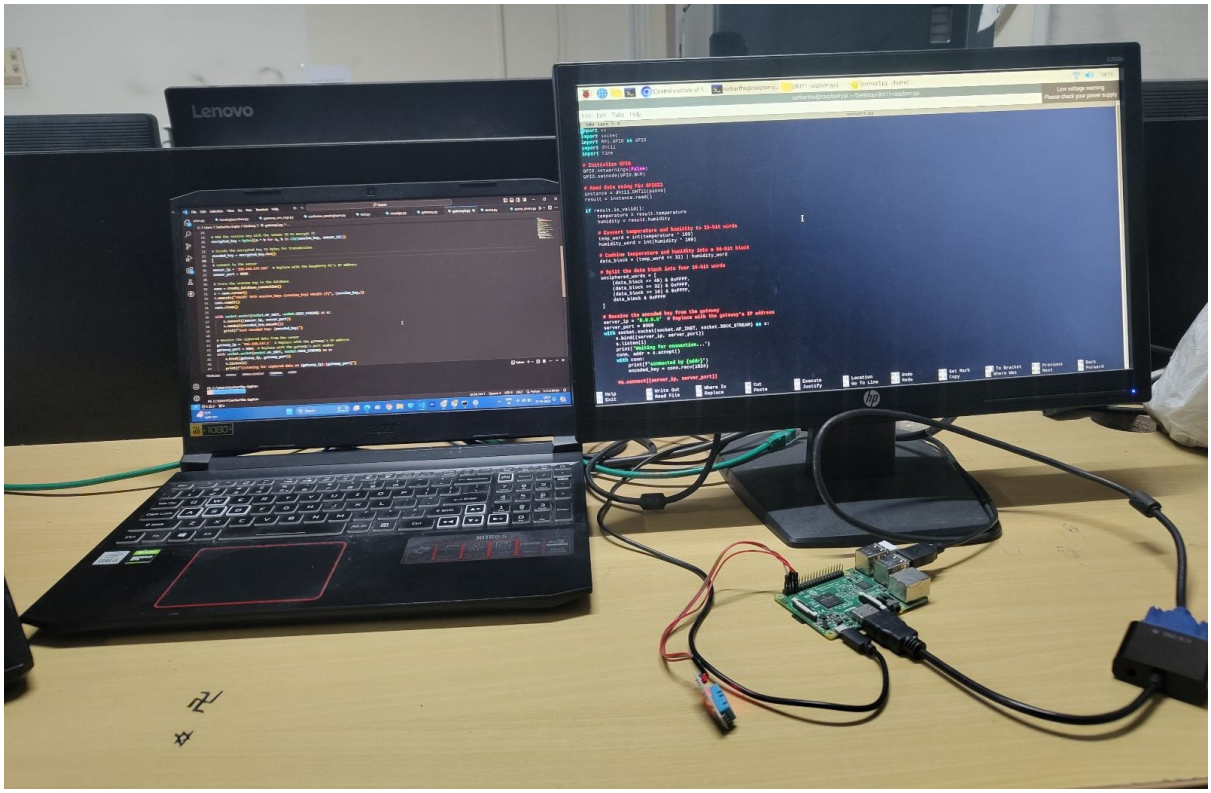

Figure 5.8: Hardware Implementation 2

Chapter 6

Results and Discussion

6.1 Results of Hardware implementation

1. **Sensor Registration Outcome:**
   Firstly, the sensors start the registration process by sending their unique sensor identities (IDs) to the gateway. The gateway, upon receiving the sensor IDs, stores them in its database and then generates pseudonymous IDs (temporary IDs) for each sensor. Next, the gateway sent these temporary IDs back to the respective sensors, along with feedback stating that the registration process was successful. This procedure makes sure that the actual sensor IDs remain anonymous, enhancing the privacy and security of the sensors.

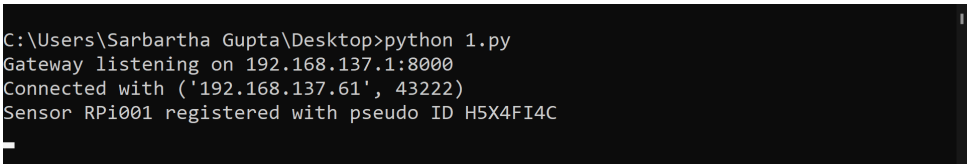

Figure 6.1: Sensor Registration at Gateway

2. **User Registration Outcome:**
   Here, users give their chosen ID and password, which are encrypted using the Elliptic Curve Integrated Encryption Scheme (ECIES) before being transmitted to the gateway. The ECIES ensures that the user credentials are securely encrypted during transmission, protecting them from potential unauthorized access.

   The gateway, upon receiving the encrypted user credentials, decrypts them and stores the credentials in its database for future purposes. The gateway then sends a confirmation message and also includes the temporary pseudonymous IDs of the previously registered sensors back to the user, indicating successful registration. This step enables users to establish a secure connection with the registered sensors through the gateway while preserving the anonymity of the sensor identities.

Figure 6.2: User Registration at Gateway

3. **Authentication of Users at the Gateway:**
   After the initial registration phase, users proceed to authenticate themselves with the gateway to gain authorized access. Users give their credentials and send them again using the Elliptic Curve Integrated Encryption Scheme (ECIES) through an insecure channel to the gateway, which decrypts the credentials and verifies the received credentials against the user information stored in its database during the registration phase. After successful authentication, the gateway will generate a random session key and a unique nonce (a number used once) for the current session. The session key and the associated nonce are then stored in the gateway's database, enabling secure communication between the authenticated user and the registered sensors.

4. **Authentication of Gateway at the Sensor Node:**
   Subsequent to user authentication, the gateway performs an XOR encryption operation, combining the previously generated session key with the pseudonymous sensor ID. The resulting encrypted data, along with a nonce and a timestamp, is transmitted to the respective sensor. The sensor decrypts the key using the XOR operation and verifies if the sensor ID contained within it is correct or not. At the same time, the sensor checks the freshness of the received nonce and timestamp, making sure that the session key is recent and has not been replayed from a previous session.

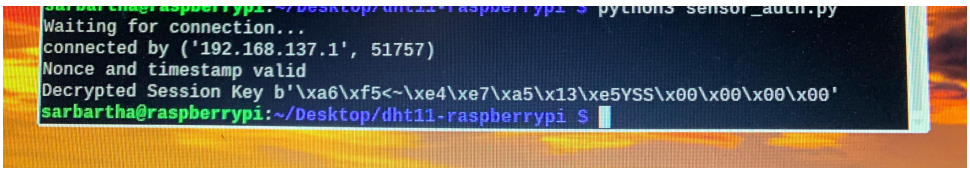

Figure 6.3: Gateway authentication at sensor node

5. **Outcome of Data Encryption:**
   On the sensor side, the data collected from sensors (particularly temperature sensors in our case) undergoes encryption before being transmitted to the gateway. The encryption process takes place at the sensor level, employing our newly proposed algorithm specifically designed for lightweight encryption purposes. The data is then subjected to the encryption algorithm, resulting in ciphered blocks of data.The ciphered blocks, along with the associated nonce, are then sent to the gateway.The nonce is included to serve as a countermeasure against potential replay attacks.

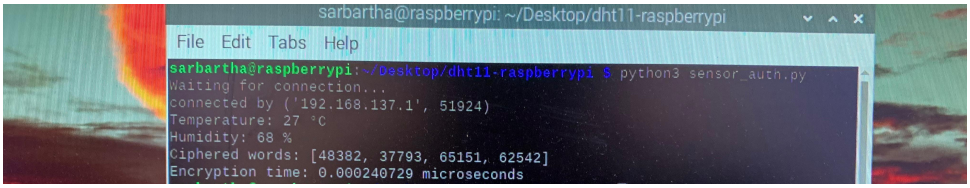

Figure 6.4: Ciphered Words

6. **Sensor Authentication Outcome**

   The gateway receives the ciphered data blocks and the nonce from the sensor side and starts the authentication process in order to verify the integrity and freshness of the received information. If the nonce is found to be valid and fresh, the gateway then proceeds to store the ciphered data blocks in its database. This step ensures that the encrypted sensor data is securely retained for further processing or analysis.

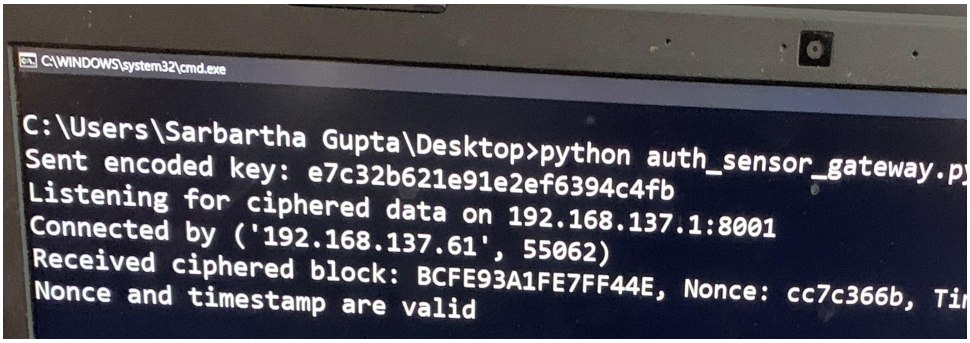

Figure 6.5: Sensor authentication at gateway

7. **Mutual Authentication and Access Granted:**

   The session key, which was previously generated by the gateway during the user authentication phase (Step 3) is securely stored in the database, is now transmitted to the authenticated user. This session key serves as a shared secret between the user and the gateway, enabling secure communication and data access.

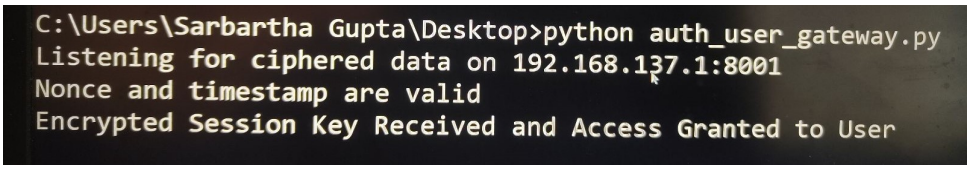

Figure 6.6: User Access Granted

6.2 Computational Cost

Phases	User Device	Gateway	Sensor Side	Total
P1	CCh	-	-	CCh
P2	-	-	-	-
P3	3*CCh	3*CCh	CCh	7*CCh
Ps	4*CCh	3*CCh	CCh	8*CCh

Table 6.1: Computational Cost of our proposed scheme

Scheme	User device	Gateway	Sensor Node	Total Cost
[34]	10*CCh	7*CCh	19*CCh	36*CCh
[28]	11*CCh	7*CCh	5*CCh	23*CCh
[33]	9*CCh	15*CCh	8*CCh	32*CCh
[20]	4*CCh	3*CCh	2*CCh	9*CCh
Ps	4*CCh	3*CCh	CCh	8*CCh

Table 6.2: Comparison of Computation Cost

Acronyms:- CCh: Computational Cost Hash, P1: User Registration, P2: Sensor Node Registration, P3: Mutual Authentication and Key Agreement, Ps: Proposed Scheme

6.3 Encryption Time of CipherSense:

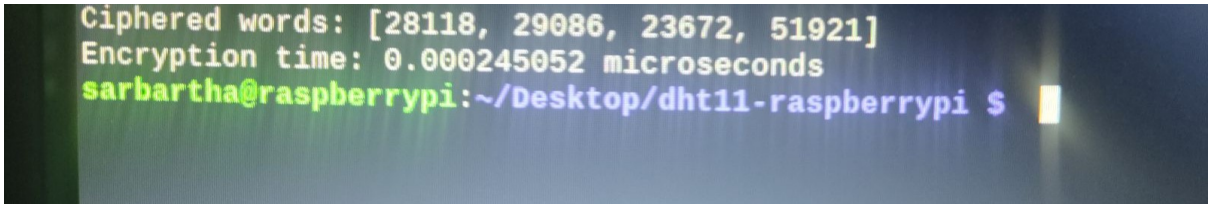

Figure 6.7: Encryption Time

However the execution time of a whole single session is estimated to be 50-60 seconds.

6.4 Commutational Cost

The communication cost refers to the quantity of data or information that must be communicated between various entities in a network or distributed system, such as computers, servers, or devices. The communication cost is often calculated in terms of the number of bits or bytes that must be communicated over the network or communication channel. It can have a considerable

impact on the overall performance, efficiency, and scalability of a system, especially in cases where huge amounts of data must be transported or when network bandwidth is limited.

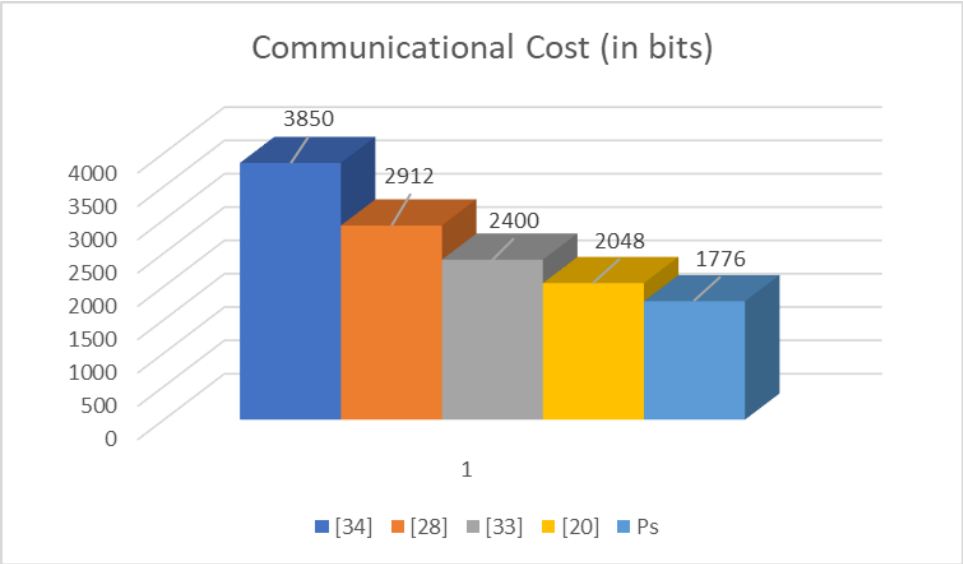

Figure 6.8: Communicational Cost

6.5 Messages Exchanged

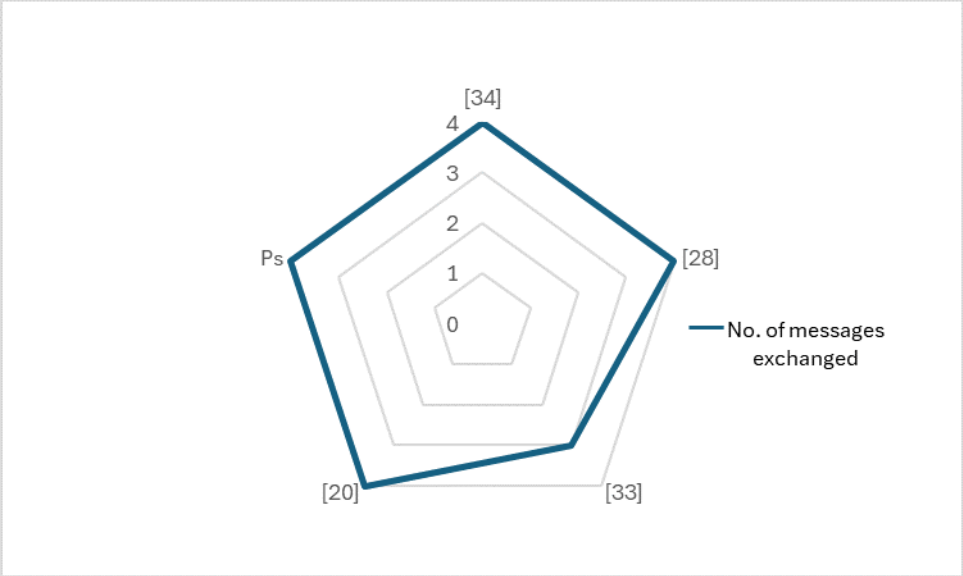

Figure 6.9: Number of messages exchanged by the proposed and conventional protocols during key establishment phase

6.6 Comparisons with Existing algorithms

The below comparison table is intended to provide a detailed overview of the performance and security characteristics of various encryption methods.

Algorithms	Key Size (bits)	Data Size (bytes)	Encryption time (microseconds)	Confusion Score	Diffusion Score	No. of XOR Operations performed
ASCON-128	128	64	0.3 - 1	200	23	9
AES-128	128	64	1100.1 - 1400.1	214	25	11
SensingLayer [5]	128	64	9.0 - 10.1	256	12	2
CipherSense	128	64	0.00024 - 0.00027	190	7	2

Table 6.3: Comparisons With different algorithms

6.7 Security Analysis:

6.7.1 Informal

1. **Protection against Replay Attacks:**
   Proof. As an illustration, let's say the attacker intercepted the message $M5 = Cm1$ and replayed it to the gateway. The gateway determines the values and verifies the freshness of the nonce N_{D}^1 upon reception. The gateway ends the session because the replayed message contains an outdated nonce. This same procedure is applied to the other messages ($M6$, $M7$, and $M8$). Furthermore, since the nonce N_{D}^1 is covertly wrapped in Pm_2, which is already encoded in $Cm1$, the adversary cannot alter it. In a similar vein, the nonce in messages $M6$, $M7$, and $M8$ is shielded against alteration. The suggested procedure is hence immune to replay assaults.

2. **Robust against Man-in-the-Middle (MITM) Attack:**
   Proof. Assume that the MITM attack is carried out by an attacker between the user device and the gateway. The attacker attempted to alter the message $M5 = Cm1$ in order to deceive other parties involved. The nature of ECC encryption means that the attacker cannot obtain the values without a private key. Moreover, even if the attacker manages to obtain the value, the collision-resistant nature of hash functions prevents the opponent from changing $\lambda = (R_{SG}^1 || PW_D)$. As a result, the attacker is unable to change the $M5$ data. $M6$, $M7$, and $M8$ are similarly shielded from changes.

3. **Resilient from Impersonation Attacks:**
   Proof. Since the adversary lacks access to confidential information about the entities, they are unable to misuse or misrepresent the supplied protocol. Assume that the attacker intercepted the message $M6 = G_W^1, G_W^2, U_{TID}, SK_S$, and G_W^3 and is attempting to obtain the user's and sensor node's secret credentials. The attacker is thwarted, though, because $G_W^1 = N_G^1 \oplus S_{TID}$ cannot be accessed in plain text, $G_W^1 = h(R_{SN}^1 \parallel (R_{SG}^2)$ can be accessed as a message digest, U_{TID} is the user device's temporary identification, and SK_S and G_W^3 are also determined via XORing. Because of the collision-resistant nature of hash functions, the attacker is unable to extract the original data from the message. Hence, attackers and malicious nodes do not have any identity or even secret information.information to prove legitimate users and sensor nodes, respectively. Therefore, impersonation is not possible in the proposed protocol. Even the other exchanged message M7 is non-vulnerable.

4. **Resistant to Password Guessing and Eavesdropping Attack:**

Proof. Suppose the adversary wants the user's password, to gain it the adversary decides to eavesdrop on $M5$ and $M8$ exchange messages. Unfortunately, neither the contents of $Cm1$ nor μ are in plaintext, even the ID and passwords are temporary. Not to mention the nature of ECC encryption, any type of guessing attack is rendered invalid.

5. **Prevents Privileged Insider Attacks:**

   Proof. Let's take an example, imagine the hacker has privileged access to the system and decides to obtain the user's real identity and password. Since we use temporary IDs and the real IDs are not stored, the hacker fails to obtain the real identity and password of the user.

6. **Attain Identity Anonymity and Untraceability:**

   Proof. Imagine that a hacker intercepts messages $M5$ and $M6$. Because temporary identities ($U_{TID} and S_{TID}$) are utilized instead of actual identities ($U_{ID} and S_{ID}$), the attacker is unable to get the identity information even after successfully acquiring it. As a result, the conversation stays private. Additionally, each session's temporary identities are different ($U_{TID}^{new}, S_{TID}^{new}$). The attacker is unable to follow the message's path as a result.

7. **Session Key Agreement and its Security:**

   Proof. In order to protect the interactions, the suggested technique allows the user and the IoT node to exchange the session keys in real-time. By serving as a middleman, the gateway facilitates their key exchange. As an example, the gateway in Figure 4 creates keys for each party following a successful review of their authenticity. The gateway transmits message $M6$ to the sensor node, enclosing the sensor node (SK) session key within $SK_S = (SK \oplus R_{SN}^1) \oplus N_G^1$. Next, the gateway delivers the user message $M8$ with the session key (SK) included in $Pm_3 = (N_G^2, SK, \eta, U_{TID}^{new})$. This key can be used by both parties to secure their communications. Notably, the keys are not transferred in plain-text, thus even though they are shared across public, insecure channels, the attacker cannot steal them. The adversary is unable to obtain the keys from messages $M6$ and $M8$ since they are unaware of R_{SN}^1, N_G^1, PW_D, and N_G^2.

6.7.2 Formal

To test its formal security analysis of our CipherSense Encryption algorithm, we tested it on a trusted tool called the "Automated Validation of Internet Security Protocols and Applications (AVISPA)" [3] . There are 4 types of backends in AVISPA i.e; "on-the-fly model-checker (OFMC)", "SAT (Boolean satisfiability problem) based model checker (SATMC)", "tree automata-based on automatic approximations for the analysis of security protocols (TA4SP)" and "constraint-logic based attack searcher (CL-AtSe)". We have to give input in "High-Level Protocol Specification Language (HLPSL)" in AVISPA which converts it into the "Intermediate Format (IF)" with the HLPSL2IF. IF is fed into one backend of AVISPA which generate the "Output Format (OF)". The OF states three outcomes, in particular, i.e., "safe", "unsafe", or "inconclusive". Fig:- 6.7.2 and Fig:- 6.10 illustrates the simulation results of OFMC. It is apparent from the simulation results that the OFMC backend visited 52 nodes with a depth of

6 plies amd states that the encryption protocol is safe from cyber threats.

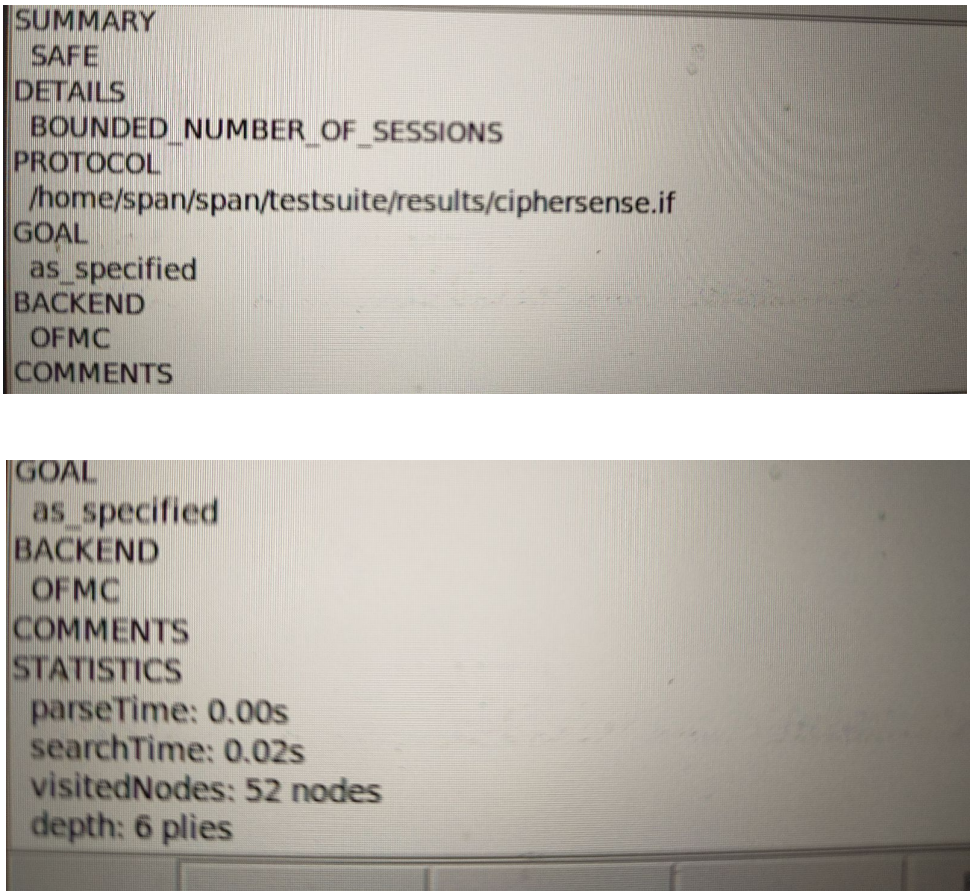

Figure 6.10: Simulation results of OFMC backend of AVISPA

Analysis and comparison of the proposed protocol based on protection against attacks and Security goals

Attack and Security Properties	[20]	[8]	[12]	[27]	[28]	[33]	Ps
Identity anonymity of user device	✓	✗	✗	✓	✗	✓	✓
Identity anonymity of sensor node	✓	✗	✗	✓	✗	✓	✓
Privileged insider attack	✓	✗	✗	✓	✗	✗	✓
Offline password guessing attack	✗	✗	✗	✓	✗	✓	✓
Denial of service attack	✗	✓	✓	✗	✓	✓	✓
User impersonation attack	✓	✗	✓	✓	✓	✗	✓
Man-in-the-middle attack	✓	✗	✗	✓	✓	✓	✓
Mutual authentication	✓	✓	✓	✓	✓	✓	✓
Session key agreement	✓	✓	✓	✓	✓	✓	✓
Untraceability	✓	✓	✗	✓	✓	✓	✓
IoT sensor node impersonation attack	✓	✗	✓	✓	✓	✓	✓
Replay attack	✓	✓	✓	✓	✓	✓	✓

Table 6.4: Comparison of Attack and Security Properties

Chapter 7

Conclusion and Future Scope

7.1 Conclusion

The Elliptic Curve Cryptography (ECC)-based user authentication strategy intended for Internet of Things (IoT)-based healthcare systems holds great potential for protecting patient data and deterring cyberattacks. The method makes use of ECC's advantages such as smaller key sizes and quicker computation on resource-constrained Internet of Things devices to offer dependable user authentication in low-resource public channels. The system provides a multi-layered security architecture for the sharing of healthcare data and includes essential components including mutual authentication, session key formation, allowing user access, and defense against various cyberattacks. Moreover, the protocol is suitable for usage with low-power devices commonly seen in the medical Internet of things due to its lightweight architecture. Further research may look into combining various advanced cryptography techniques, tailoring the protocol for specific healthcare contexts, and conducting extensive vulnerability assessments to ensure long-term resistance against evolving threats in order to provide even more security. All things considered, ECC-based user identification presents a viable way forward for securing healthcare communication in the ever-expanding and networked realm of the Internet of Things.

7.2 Future Scope

1. Integrate the proposed system with web or mobile application.
2. Testing cyber attacks in real time
3. Use of blockchain for enhancing security.

Bibliography

[1] Zeeshan Ali et al. “A robust authentication and access control protocol for securing wireless healthcare sensor networks”. In: *Journal of Information Security and Applications* 52 (2020), p. 102502.

[2] Karrar Taher R Aljamaly and Ruma Kareem K Ajeena. “The elliptic scalar multiplication graph and its application in elliptic curve cryptography”. In: *Journal of Discrete Mathematical Sciences and Cryptography* 24.6 (2021), pp. 1793–1807.

[3] Alessandro Armando et al. “The AVISPA tool for the automated validation of internet security protocols and applications”. In: *Computer Aided Verification: 17th International Conference, CAV 2005, Edinburgh, Scotland, UK, July 6-10, 2005. Proceedings 17*. Springer. 2005, pp. 281–285.

[4] Luke Beckwith et al. “A high-performance hardware implementation of the less digital signature scheme”. In: *International Conference on Post-Quantum Cryptography*. Springer. 2023, pp. 57–90.

[5] Zakaria Benomar et al. “A Fog-based Architecture for Latency-sensitive Monitoring Applications in Industrial Internet of Things”. In: *IEEE Internet of Things Journal* 10.3 (2021), pp. 1908–1918.

[6] An Braeken. “PUF based authentication protocol for IoT”. In: *Symmetry* 10.8 (2018), p. 352.

[7] Sravani Challa et al. “Secure signature-based authenticated key establishment scheme for future IoT applications”. In: *Ieee Access* 5 (2017), pp. 3028–3043.

[8] Susovan Chanda et al. “An Elliptic Curve Menezes–Qu–Vanston-Based Authentication and Encryption Protocol for IoT”. In: *Wireless Communications and Mobile Computing* 2024 (2024).

[9] Chin-Chen Chang and Hai-Duong Le. “A provably secure, efficient, and flexible authentication scheme for ad hoc wireless sensor networks”. In: *IEEE Transactions on wireless communications* 15.1 (2015), pp. 357–366.

[10] Urbi Chatterjee, Rajat Subhra Chakraborty, and Debdeep Mukhopadhyay. “A PUF-based secure communication protocol for IoT”. In: *ACM Transactions on Embedded Computing Systems (TECS)* 16.3 (2017), pp. 1–25.

[11] Duc-Thuan Dam et al. “A survey of post-quantum cryptography: Start of a new race”. In: *Cryptography* 7.3 (2023), p. 40.

[12] S Siamala Devi et al. “Paillier Cryptography Based Message Authentication Code for IoMT Security.” In: *Comput. Syst. Sci. Eng.* 44.3 (2023), pp. 2209–2223.

[13] Sumit Singh Dhanda, Brahmjit Singh, and Poonam Jindal. “Lightweight cryptography: a solution to secure IoT”. In: *Wireless Personal Communications* 112.3 (2020), pp. 1947–1980.

[14] Mohamed Gafsi et al. “Hardware implementation of a strong pseudorandom number generator based block-cipher system for color image encryption and decryption”. In: *International Journal of Circuit Theory and Applications* 51.1 (2023), pp. 410–436.

[15] Prosanta Gope and Biplab Sikdar. “Lightweight and privacy-preserving two-factor authentication scheme for IoT devices”. In: *IEEE Internet of Things Journal* 6.1 (2018), pp. 580–589.

[16] Nilupulee A Gunathilake, Ahmed Al-Dubai, and William J Buchana. “Recent advances and trends in lightweight cryptography for IoT security”. In: *2020 16th International Conference on Network and Service Management (CNSM)*. IEEE. 2020, pp. 1–5.

[17] M Shamim Hossain. “Cloud-supported cyber–physical localization framework for patients monitoring”. In: *IEEE Systems Journal* 11.1 (2015), pp. 118–127.

[18] Xiaoying Jia et al. “Signature-based three-factor authenticated key exchange for internet of things applications”. In: *Multimedia Tools and Applications* 77 (2018), pp. 18355–18382.

[19] Suman Majumder et al. “ECC-CoAP: Elliptic curve cryptography based constraint application protocol for internet of things”. In: *Wireless Personal Communications* 116.3 (2021), pp. 1867–1896.

[20] Mehedi Masud et al. “Lightweight and anonymity-preserving user authentication scheme for IoT-based healthcare”. In: *IEEE Internet of Things Journal* 9.4 (2021), pp. 2649–2656.

[21] Pietro Nannipieri et al. “Hardware design of an advanced-feature cryptographic tile within the european processor initiative”. In: *IEEE Transactions on Computers* (2023).

[22] Priyansi Parida et al. “Image encryption and authentication with elliptic curve cryptography and multidimensional chaotic maps”. In: *IEEE Access* 9 (2021), pp. 76191–76204.

[23] Rosheen Qazi et al. “Security protocol using elliptic curve cryptography algorithm for wireless sensor networks”. In: *Journal of Ambient Intelligence and Humanized Computing* 12 (2021), pp. 547–566.

[24] Muhammad Rana, Quazi Mamun, and Rafiqul Islam. “Lightweight cryptography in IoT networks: A survey”. In: *Future Generation Computer Systems* 129 (2022), pp. 77–89.

[25] Vidya Rao and KV Prema. “A review on lightweight cryptography for Internet-of-Things based applications”. In: *Journal of Ambient Intelligence and Humanized Computing* 12.9 (2021), pp. 8835–8857.

[26] Dipanwita Sadhukhan et al. “A lightweight remote user authentication scheme for IoT communication using elliptic curve cryptography”. In: *The Journal of Supercomputing* 77.2 (2021), pp. 1114–1151.

[27] Manasha Saqib, Bhat Jasra, and Ayaz Hassan Moon. “A lightweight three factor authentication framework for IoT based critical applications”. In: *Journal of King Saud University-Computer and Information Sciences* 34.9 (2022), pp. 6925–6937.

[28] Geeta Sharma and Sheetal Kalra. “A lightweight user authentication scheme for cloud-IoT based healthcare services”. In: *Iranian Journal of Science and Technology, Transactions of Electrical Engineering* 43 (2019), pp. 619–636.

[29] Mengxia Shuai et al. “A secure authentication scheme with forward secrecy for industrial internet of things using Rabin cryptosystem”. In: *Computer Communications* 160 (2020), pp. 215–227.

[30] Vishal A Thakor, Mohammad Abdur Razzaque, and Muhammad RA Khandaker. “Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison and research opportunities”. In: *IEEE Access* 9 (2021), pp. 28177–28193.

[31] Muhamed Turkanović, Boštjan Brumen, and Marko Hölbl. “A novel user authentication and key agreement scheme for heterogeneous ad hoc wireless sensor networks, based on the Internet of Things notion”. In: *Ad Hoc Networks* 20 (2014), pp. 96–112.

[32] Shamsher Ullah et al. “Elliptic Curve Cryptography; Applications, challenges, recent advances, and future trends: A comprehensive survey”. In: *Computer Science Review* 47 (2023), p. 100530.

[33] Mohammad Wazid et al. “LDAKM-EIoT: Lightweight device authentication and key management mechanism for edge-based IoT deployment”. In: *Sensors* 19.24 (2019), p. 5539.

[34] Lu Zhou et al. “Lightweight IoT-based authentication scheme in cloud computing circumstance”. In: *Future generation computer systems* 91 (2019), pp. 244–251.