Dissertation on

"A Secured certificate renewal scheme for constrained IoT devices"

Submitted in partial fulfilment of the requirements for the award of degree of

Bachelor of Technology
in
Computer Science & Engineering

UE19CS390B – Capstone Project Phase - 2

Submitted by:

Manideep P R	**PES1UG19CS258**
Nithesh A	**PES1UG19CS306**
Nagesh B S	**PES1UG20CS816**
Navya C	**PES1UG20CS818**

Under the guidance of

Prof. Vadiraja A
Associate Professor
PES University

August - December 2022

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100 Feet Ring Road, Bengaluru – 560 085, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)

100 Feet Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

‘A secured certificate renewal scheme for constrained IoT devices’

is a bonafied work carried out by

Manideep P R	**PES1UG19CS258**
Nithesh A	**PES1UG19CS306**
Nagesh B S	**PES1UG20CS816**
Navya C	**PES1UG20CS818**

in partial fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE19CS390B) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period August - December 2022. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 7th semester academic requirements in respect of project work.

Signature	Signature	Signature
Prof. Vadiraja A	Dr. Shylaja S.S.	Dr. B.K. Keshavan
Associate Professor	Chairperson	Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. ____________________________	____________________________
2. ____________________________	____________________________

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled **“A Secured certificate renewal scheme for constrained IoT devices”** has been carried out by us under the guidance of Associate Professor. Vadiraja A and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology** in **Computer Science and Engineering** of **PES University, Bengaluru** during the academic semester August - December 2022. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES1UG19CS258	**Manideep P R**
PES1UG19CS306	**Nithesh A**
PES1UG20CS816	**Nagesh B S**
PES1UG20CS818	**Navya C**

ACKNOWLEDGEMENT

We would like to express our gratitude to Prof. Vadiraja A, Department of Computer Science and Engineering, PES University, for his continuous guidance, assistance, and encouragement throughout the development of this UE19CS390B - Capstone Project Phase – 2.

We are grateful to the project coordinator, Prof. Mahesh H.B., for organizing, managing, and helping with the entire process.

We take this opportunity to thank Dr. Shylaja S.S. Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support we have received from the department. We would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

We are deeply grateful to Dr. M.R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J., Vice-Chancellor, PES University for providing us various opportunities and enlightenment every step of the way.

Finally, this project could not have been completed without the continual support and encouragement we have received from our family and friends.

ABSTRACT

In today's world, Internet of Things plays a major role in connecting billions of smart devices around the globe, which helps in exchanging data over the internet without manpower. As we see that IoT devices are deployed exponentially every year, providing security to these devices is need of the hour, as these devices transmit sensitive and secure data over the network. Prevention of these devices from various attacks such as Man in the middle attack, Botnets, Ransomware, Convergence, spoofing etc is the main goal of our project.

Our project (Secured certificate renewal scheme for constrained IoT devices) aims to provide security in the network of IoT devices. We provide this security by establishing a Key Distribution Centre (KDC) that generates certificates to the network's IoT devices for safe and secured communication and data exchange with other devices in the network. The proposed certificate is a novel certificate-based device access control method to mitigate the security issues in these devices, which not only prevents the device from various attacks but also preserves the integrity of the device in the IoT network.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1
INTRODUCTION

Security is the biggest concern in adopting Internet of things technology, with concerns that rapid development is happening without appropriate consideration of the profound security challenges involved and the regulatory changes that might be necessary. The rapid development of the Internet of Things (IoT) has allowed billions of devices to connect to the network. Due to too many connected devices and the limitation of communication security technology, various security issues gradually appear in the IoT.

Most of the technical security concerns are similar to those of conventional servers, workstations and smartphones. These concerns include using weak authentication, forgetting to change default credentials, unencrypted messages sent between devices, SQL injections, ,Man in the middle attack, spoofing and poor handling of security updates. However, many IoT devices have severe operational limitations on the computational power available to them. These constraints often make them unable to directly use basic security measures such as implementing firewalls or using strong cryptosystems to encrypt their communications with other devices - and the low price and consumer focus of many devices makes a robust security patching system uncommon.

Rather than conventional security vulnerabilities, fault injection attacks are on the rise and targeting IoT devices. A fault injection attack is a physical attack on a device to purposefully introduce faults in the system to change the intended behaviour. Faults might happen unintentionally by environmental noises and electromagnetic fields. There are ideas stemmed from control-flow integrity (CFI) to prevent fault injection attacks and system recovery to a healthy state before the fault.

IoT networks are deployed with many devices and can be used for various operations. These devices produce, collect and process huge amounts of data.

In today's world as more and more IoT devices are used, the transfer of data across the internet is vast and these devices are operated in an uncontrolled and hostile environment. Therefore, securing these devices should be a key feature of an IoT network and should be considered critical.

Most of the IoT devices are battery powered and have limited computational power, low memory processing cycles and low bandwidth. So, the risk of a hacker stealing personal information and hacking the device is very high.

The key security and privacy challenges in IoT devices are authentication, identification, and device heterogeneity. Integration, scalability, ethics communication mechanisms, commercial strategies, and surveillance are all major concerns.

CHAPTER 2
PROBLEM STATEMENT

Secured certificate renewal scheme for Constrained IoT devices.

The constrained devices in the IoT network are classified into 3 categories based on processing power, size of state and buffers and code complexity such as RAM and Flash. They are named Class 0, Class 1 and Class 2 devices. These devices are not capable of performing complex encryption and decryption quickly enough to be able to transmit data securely in real time.

Our project aims to secure Class 2 constrained devices by establishing a Key Distribution Centre that distributes secure keys to the network's IoT devices for safe and secure communication.

We also design a new certificate-based security scheme where we identify, authenticate, and provide security to the constrained IoT devices in the network by providing secure keys along with the certificate for encryption and decryption of data for communication. We also renew and revoke the certificates when required. We also identify the most efficient lightweight encryption algorithms suitable for constrained devices and implement them in our project.

CHAPTER 3
LITERATURE REVIEW

This chapter gives an overview of all the research papers we have referred to gain insights and knowledge which helped us for our project implementation. A detailed conclusion is drawn from each paper we have referred and in what way it has helped in our project implementation.

3.1 Security and Privacy Issues in IoT [1]

In this paper, the author talks about the various security and privacy issues in IoT devices. Firstly, this paper focuses on dissimilarities between IoT and Standard Internet and the deployment of both. It focuses that the Integrity and confidentiality of transmitted data must be maintained as well as the authentication of the objects is that aspects of IoT security and privacy. It talks about the various core technologies of IoT which includes Radio Frequency Identification known RFID, NFC which is called as Near Field Communication and, WSN which the full form of Wireless Sensor Networks. It also talks about the Architectures of IoT such as Business Layer, Application layer, Processing layer, Transport Layer and Perception Layer and their functionalities. It also talks about the major applications of IoT and their scope in today's world. The major portion of the paper discusses the Security and privacy needs of IoT along with the privacy enhancing technologies present and remedy of problems regarding security and privacy issues. The various security and privacy needs are User privacy, Access control, Identity Management, Secure data communication, Mobile Security and Secure Middleware are discussed. Some of the major security-related issues are Identification, Authentication, Data management, Heterogeneity etc.

Outcome:

This paper gives a clear picture of security and privacy issues in IoT which helps us in understanding our project even better and the challenges we must handle while implementing our project. Since providing security to IoT devices is a major functionality of our project, this paper helps us in understanding the issues and how to solve them.

3.2 Public Key Infrastructure – Overview [2]

This research paper on Public Key Infrastructure (PKI) gives us detailed information about the framework and working of the PKI. It also briefs about the encryption of the keys used in the framework. The PKI framework comprises security, operational policies, security services, and interoperability protocols that facilitate the administration of keys and certificates using public-key cryptography. Certification Authorities (CAs), Registration Authorities (RAs), and directory services, which can be used to form a hierarchy or chain of trust, are typically used to generate, distribute, and manage public keys and related certificates. Digital certificates that can be used to identify different entities can be implemented via CA, RA, and directory services. The goal of a public key infrastructure (PKI) architecture is to facilitate and support the secure exchange of data, credentials, and value (such as monetary instruments) in insecure settings like the Internet. A PKI allows for the creation of a trust hierarchy. One of the fundamental principles of a PKI is this. Formal trust mechanisms are required in Internet-based e-commerce to provide risk management controls. The role of the CA in a PKI can be used to demonstrate the concept of trust. In the Internet context, entities that are unfamiliar with each other lack adequate trust to conduct business, contractual, legal, or other forms of activities. This trust is established by utilizing a CA to create a PKI. Public key cryptography - The development, distribution, administration, and control of cryptographic keys are all PKI functions. Certificate issuance binds a public key to a person, company, or other entity, or to some other data, such as an email or a purchase order. Certificate validation verifies the existence of a trust relationship or binding, as well as the validity of a certificate for certain operations. Certificate revocation - Cancels a previously issued certificate and either publishes the cancellation to a Certificate Revocation List or activates the Online Certificate Status Protocol.

Outcome:
This paper gives a clear working of PKI which helps in the implementation part of our project. It generalizes the various stages involved in the PKI framework and the models included in it. Since generating certificates for the IoT nodes is the main agenda of our project, this paper has given a clear picture of the certificate implementation.

3.3 Efficient and Secure Group Key Management in IoT using Multistage Interconnected PUF [3]

This paper explains a group key management scheme based on a novel physically unclonable function (PUF) design: multistage interconnected PUF (MIPUF) to secure group communications in an energy-constrained environment. Physically unclonable function (PUF) can perform key management tasks such as key distribution, key storage, and rekeying securely and efficiently. PUFs are a popular type of low-power security primitive and have been proposed to be used in several key management subtasks in IoT settings. Multiple PUF architectures provide compatibility to secure authentication. This paper also gives a detailed study of Processing elements (PEs), Switching elements (SEs), multistage interconnection, and Processing network configuration. This study describes a multistage interconnected PUF (MIPUF) to secure group communications in an energy-constrained environment, which is based on a novel physically unclonable function (PUF) design. PUFs can execute key management activities such as key distribution, key storage, and rekeying in a secure and efficient manner. PUFs are a common low-power security primitive that has been proposed for usage in a variety of key management subtasks in IoT environments. Secure authentication is compatible with multiple PUF architectures. This paper also covers Processing elements (PEs), Switching elements (SEs), multistage connectivity, and Processing network setup in great depth. Multistage interconnection networks provide a balance between cost and configurability. A blocking multistage connection is the most cost-effective and provides sufficient configurability for MIPUF implementation. Because a connection between one free input and another free output is blocked by an existing connection in a network, a blocking multistage connection cannot realize all conceivable connections between inputs and outputs. A configuration vector controls signal flow between any two MIPUF nodes, protecting network configuration. Management of the Group: Key distribution, key storage, and rekeying are all part of group key management.

Outcome:
This paper proposes a group key management scheme in IoT consisting of key distribution, key storage and rekeying based on MIPUF. Security and overhead analysis on the scheme show that the design is not only secure against multiple attack methods but also low power.

3.4 Experimental Demonstration of Quantum Key Distribution (QKD) for Energy-Efficient Software-Defined Internet of Things [4]

This paper demonstrates an experimental integration between a Software-Defined IoT network of devices and a fibre-based QKD system, as a primary proof-of-concept implementation to showcase that QKD can serve a significantly increased number of IoT devices with an identical level of security while drastically improving energy savings for the IoT infrastructure. Quantum Key Distribution (QKD) technology has proved to be a powerful candidate for future-proof secure communications with recent field trials being successfully demonstrated in an exceedingly network scale3. QKD enables the sharing of a secure key between two remote entities while quantum physics prevents from measuring the encoded photons. Current implementations depend upon complex QKD systems 3 but the increased interest on chip based QKD4 may soon result in low-cost QKD devices and systems that may allow their massive deployment with IoT. In the Network and Application Domains, the core building block is the IoT platform which is the element retrieving and providing the information from the devices. to determine communication between the devices and therefore the IoT platform, a process is launched through the IoT provisioning application utilizing the SDN controller, to provision the devices supported by their MAC address 8.

When the SDN controller discovers the complete network, the appliance requests keys capable of the entire amount of the IoT devices. The QKD system generates 256 bit-keys after several steps, including error correction and privacy amplification happening within the ClChs. The keys are saved to the Key Buffer(KB)(1 Mbits), approximately 3900 256bitkeys, and are accessible from the Key Extraction Application which makes a call for participation over UDP to extract them. QKD serves both as an enhanced key source mechanism counting on quantum true random generation and as a secure mechanism to distribute the keys through the optical network to the IoT gateways and so to the IoT devices, thus completely replacing the embedded key generation processes otherwise present in current IoT networks.

To evaluate the advantages of the combination between QKD and also the IoT infrastructure, in addition because the impact to the networking performance, a series of experiments is conducted. We use the network emulator Mininet and therefore the IoT MQTT data protocol

Outcome:

This paper gives information on how the device key generation impacts energy consumption in IoT. This paper proposes a key generation methodology using QKD to increase the IoT device battery lifetime. The proposed scheme outperforms the DKG and presents an energy efficiency improvement of 18% without compromising the network reliability.

3.5 Matrix-based key management scheme for IoT networks [5]

In this paper, they proposed a brand-new lightweight Matrix-based key management protocol for the IoT network. The formal verification tool AVISPA has been utilized to test these security properties like authentication, integrity, and secrecy. Security and performance analysis showed that the proposed scheme protects users' sensitive data from several forms of attacks by achieving secure end-to-end communication and is suitable for resource-limited networks. The proposed scheme adopts the specification that consists of three main components: constrained nodes, gateway nodes, and remote server nodes The proposed scheme consists of 4 phases: initialization, key establishment, adding new nodes, and key refresh phase. In the initialization phase, all the subsequent information is preloaded into the node's memory

• A number that's unique within the network.

• A matrix M of order n, where the weather are generated randomly and are strictly positive, and n represents the quantity of nodes within the network during the deployment phase: M[i,j] > 0, with i,j = 1..n. This matrix will be stored within the flash RAM and hence are going to be erased from the node's memory later.

• A technique keyed-hash function Hash(message, k) that takes as input a message of arbitrary length message and a secret key k and compresses the message into a brief fixed length hash value, so it'll take up less space.

The key generation phase contains the two following steps:

a) Neighbour discovery.

As soon as the deployment is done, every node Ni identifies its direct neighbors (1-hop) by sending them a Hello message along with its identifier, a nonce and a hash value (digest), which is computed by the pre shared one way function that uses the diagonal element of the sender as secret value.

b) Computation of the secret values and pairwise keys.

During this step, each node computes its square matrix Mi, which is obtained from the initial one M by keeping only the elements M[k, l], where ∀ k, l ∈ vi.

Moreover, it computes a positive secret value Sij, which is equal to the absolute value of the determinant of the 2 by 2 matrix Mij.

- In adding new node phase:

To add a new node Nn to the network, the gateway node Ni that is close to this new node, randomly generates a new positive secret value Si and sends it to each of its neighbouring nodes encrypted with the corresponding pairwise key. Each node that receives this message and decrypts it with the appropriate symmetric key, extracts and saves the secret value, and encrypts a message in which it inserts the new secret value Si and transmits it to its direct neighbours except to the one who sent him the last message.

- In key refresh phase:

This phase tries to increase security by making the new keys completely different from the old ones, making cryptanalytic attacks more difficult. Indeed, if the same keys are held for an extended period of time, the attacker may be able to access them through traffic analysis. As a result, all keys should be changed. After the expiration time has passed, the gateway nodes start the refresh procedure on a regular basis.

Outcome:

This paper gives us information about the proposed lightweight matrix-based key management scheme for securing communications between IoT devices. Security analysis shows that the proposed scheme ensures the security goals like secrecy, integrity and authentication and can protect sensitive data from various types of attacks. In addition, the proposed system allows extensibility, scalability, resilience, authentication, and distribution.

3.6 Proposing an Encryption/ Decryption Scheme for IoT Communications using Binary-bit Sequence and Multistage Encryption [6]

The paper introduces a new symmetric cryptography technique that makes use of the user's symmetric key. Using the proposed approach, the same key will be utilised to encrypt the given data. This work proposes a new encryption/decryption technique that uses the XOR operation and is based on the ASCII, Binary-Bit sequence. It implies encryption at the compound stage, with the user providing the key. The original data will be encrypted at many stages in the proposed approach, with a key used for plaintext to cypher text encryption and the same key used for a cypher to plain text decryption. The suggested algorithm belongs to the Symmetric key algorithms category. For every encrypting or decrypting operation, data security becomes a critical consideration. Other considerations when constructing an encryption algorithm are space and time. Because our technique uses less memory, it has a high throughput. The affirmative portion of the proposed algorithm includes the fact that breaking the encrypting/decrypting procedures without knowing the exact secret key is not feasible, and so the proposed technique can be used in a variety of public applications in IoT communications to send data.

Outcome:
This paper gives a clear understanding of encryption and decryption scheme using Binary bit sequence and Multistage Encryption and helps us in the implementation part of our project. It helps us in selecting the best encryption/decryption scheme for the constrained devices in our project.

3.7 Internet of Things Security: Challenges and Key Issues [7]

The research in this paper is primarily concerned with the security of IoT technologies. IoT, for example, is vulnerable to a variety of threats, including DOS, password guessing, replay, and insider assaults. Because of the lack of standardization in the IoT, numerous topologies exist. The three-layer architecture consists of three layers, comprising perception, networking, and application layers. The five-layer architecture consists of five layers, including perception, networking, and application layers. The five-layer design, on the other hand, comprises processing and business levels in addition to the three previously mentioned. Perception, transport, processing, application, and business are the five levels. Perception, transport, and application layers have the same responsibilities as analogous layers in a three-layer design. IoT Security Services: Authorization, authentication, confidentiality, availability, integrity, and non-repudiation are all required security services for IoT solutions. Because IoT may access all a user's information, the user's private life must be protected against dangerous attacks. Unauthorized users should not be able to access the gadgets. As a result, before obtaining authorization, it is important to verify the user's identification. As a result, a user's identification can be verified in a variety of ways. Despite this, the most widely used authentication method is one that relies on the prior sharing of secrets, keys, or passwords. As a result, we'll go through the approaches used to reinforce authentication in the IoT context in this section.

Outcome:
This paper is mainly focused on the security of IoT technology. Such as, IoT suffers from several attacks, namely, DOS, password guessing, replay, and insider attacks.

3.8 Understanding Security Requirements and Challenges in Internet of Things (IoT): A Review [8]

In this paper, they categorize and explain the current state-of-the-art efforts in assuring security in the IoT network in this study. Efforts in privacy provisioning, a lightweight cryptographic framework, safe routing and forwarding, robustness and resilience management, denial of service detection, and insider attack detection are all covered in depth. Because the features of an IoT network differ from those of a traditional Internet network, privacy is critical. This paper identifies and discusses such concerns and requirements. In addition to privacy, lightweight cryptographic primitives that are suitable for IoT networks are necessary to provide network security. All the efforts in this direction have been collated, and next steps are being debated.

Outcome:
To preserve privacy, context-aware techniques and lightweight protocols are proposed and most lately virtualization techniques are used to maintain the integrity of the data. For lightweight

cryptographic primitives, novel solutions are required which should consume limited resources of an IoT mote.

3.9 A Physical-Layer Key Distribution Mechanism for IoT Networks. [9]

In this paper, A novel key distribution strategy for IoT networks is proposed in this paper. The suggested approach takes advantage of channel diversity to distribute encryption keys among network nodes. The proposed mechanism's fundamental uniqueness is that it ensures that multiple keys of varying lengths are distributed to all nodes at the same time.
The proposed approach ensures that all nodes in the network receive independent keys at the same time. The suggested technique assumes that the channel value is only known at the CE and the related node.

Outcome:
The proposed mechanism implies broadcasting random signals to the nodes from the central entity. Each node will decode the received signal by independent signal modulation type and order. The order of the modulation is selected based on the channel magnitude, and its corresponding constellation diagram is rotated by an angle equal to the channel phase. Simulation results demonstrate the high performance of the proposed scheme, robustness against channel estimation error, and immunity against intelligent eavesdroppers.

3.10 CoAP-Based Streaming Control for IoT Applications. [10]

In this paper, they present a CoAP-based streaming control (CoAP-SC) technique in this study, which is an extension of CoAP over UDP with error handling and flow control for increased throughput. The proposed CoAP-SC strategy takes into account the quantity of data messages in the sequence, the use of ACK messages, and the transmitting buffer size. Because the traditional CoAP approach does not account for error handling and flow restrictions for streaming transport, throughput performance suffers, especially in wireless sensor networks. If a message is lost in CoAP over UDP, for example, retransmission occurs after a timeout event, and therefore the error recovery technique may increase a substantial transmission delay. By utilizing TCP's fast retransmission, CoAP over TCP can quickly recover a lost packet, however, the TCP technique may add some overhead to the IoT context. Furthermore, CoAP over TCP inherits the complexity of TCP methods, which are incompatible with real-time streaming services in the IoT.

Outcome:
To overcome these problems, this paper proposes a CoAP-based streaming control (CoAPSC), which is an extension of CoAP over UDP with error handling and flow controls for throughput enhancement. The proposed scheme is designed by considering the sequence number of the data messages, the use of ACK messages, and the buffer size of the sending buffer.

3.11 Lightweight Security in IoT: A Survey. [11]

This study examines the many lightweight algorithms that can be utilized on such resource constrained devices. As the network's strength increases in terms of linked devices, it becomes increasingly vulnerable to external threats. Inadvertently, it allows access to/modification of data saved on the computer. To transmit/receive or access the data stored in the system, cryptographic algorithms can be used to overcome this challenge. Typical cryptographic algorithms necessitate a lot of work and resources in the end devices. Designing a cryptographic algorithm that is optimal for resource restricted contexts is a huge challenge. The National Institute of Standards and Technology (NIST) has authorised algorithms such as AES, SHA-2, and SHA-3, which are suitable for servers and desktop PCs but not for resource restricted situations. Authentication algorithms with state sizes of 512 and 1600 bits, such as SHA-2 and SHA-3, demand memory and space that are not available on resource restricted systems.

Outcome:
This research work has presented various lightweight cipher methods. Especially for resource-constrained devices, the main thrust area of research is towards providing the le sturdy and structured encryption techniques.

3.12 Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions. [12]

A Systematic Review The work conducted in this paper is a walkthrough from the first concept of IoT evolution and application to security challenges at various tiers, and eventually to various Intrusion Detection approaches. The scope of the research is limited to the issues provided by IoT technology's security requirements. The following are some of the study's significant contributions:

• The evolution of the Internet of Things, as well as its uses and difficulties, are discussed.
• Security problems at various IoT tiers are examined in detail.
• The risks and remedies related with DDoS assaults are thoroughly examined.
• A variety of anomaly detection strategies are evaluated and contrasted.
• A thorough examination of current Intrusion Detection System approaches is provided.
• Finally, research gaps identified through the survey are discussed, along with potential solutions.

In the literature, there are two key systems for preventing DDoS attacks: Intrusion Detection System (IDS) and Intrusion Prevention System (IPS) (IPS). In the review work that has been completed, the Intrusion Detection System has been examined, and various intrusion detection models have been reviewed. Intrusion Detection System categorization, multiple anomaly detection techniques, various Intrusion Detection System models based on datasets, various machine learning, and deep learning techniques for data pre-processing and malware detection were also explored. Finally, while assessing various intrusion detection approaches and future ambitions, a broader perspective was envisioned.

Outcome:
This paper gives us a walkthrough from the initial discussion of IoT evolution and application to security issues in different layers and finally to various Intrusion Detection techniques.

3.13 Proposing an Encryption/ Decryption Scheme for IoT Communications using Binary-bit Sequence and Multistage Encryption. [13]

This work proposes a new encryption/decryption technique that uses the XOR operation and is based on the ASCII, Binary-Bit sequence. It implies encryption at the compound stage, with the user providing the key. The original data will be encrypted at many stages in the proposed approach, with a key used for plaintext to cypher text encryption and the same key used for cypher to plain text decryption. The suggested algorithm belongs to the Symmetric key algorithms category. For every encrypting or decrypting operation, data security becomes a critical consideration. Other considerations when constructing an encryption algorithm are space and time. Because our technique uses less memory, it has a high throughput. The affirmative portion of the proposed algorithm is since cracking the encrypting/decrypting processes without knowing the exact secret key is not feasible, and thus the proposed algorithm can be used in a variety of public applications in Iot communications to send confidential data from one machine to another.

Outcome:
The paper presents a new symmetric cryptography algorithm and uses the symmetric key provided by the user. The same key will be used to encrypt the given data making use of the projected algorithm.

CHAPTER 4
PROJECT REQUIREMENT SPECIFICATION

This chapter elucidates the detailed requirements of the project and the tasks that are expected to be fulfilled at the end of it. The sections include a list of product features, divided into functional and nonfunctional requirements, along with constraints, expecte d risks, and possible outcomes and interactions.

4.1 Product features:

The major features and functionalities that our project is expected to deliver:

1. IoT device authentication
2. Establishing connection between node and Certificate Authority
3. Certificate Creation
4. Certificate Issue
5. Certificate renewal
6. Certificate Revocation
7. Sending Data

4.2 Functional Requirements:

4.2.1 IoT device authentication

In this feature the IoT device is verified by the Certificate Authority(CA) with appropriate credentials. Once the CA finds the device is legitimate, it establishes a secure connection with the node.

4.2.2 Establishing connection between node and Certificate Authority

In this feature a secure connection is established between the IoT node and the Certificate Authority by exchanging the public key of the IoT node and by the exchanging the cipher key and cipher iv of the Certificate Authority encrypted using the public key sent by the node.

4.2.3 Certificate creation:

In this feature the Certificate authority creates certificates and generates a pair of keys for encryption and decryption of data. The certificate creation process relies on an asymmetric encryption algorithm.

4.2.4 Certificate issue:

In this feature, the Certificate Authorities(CAs) issue the certificate to the IoT node on request. It signs the certificate with its own public key, and it is verified by the client before establishing a connection between the devices.

4.2.5 Certificate Renewal:

In this feature, the Certificate Authority will issue a new certificate with a new pair of keysafter the expiry of the issued certificate.

4.2.6 Certificate Revocation:

Before the certificate's scheduled expiration date, the process of invalidating a certificate when its private key is being compromised is known as Certificate Revocation.

4.3 Non-Functional Requirements:

4.3.1 Performance Requirement:

1. Ensure minimal response time for any certificate request.
2. Ensure that the certificate is issued to the requested IoT device.
3. Well-structured database for storage of issued certificates.

4.3.2 Safety Requirements:

1. Ensure that the sensitive data of the IoT devices is safeguarded and protected.
2. Ensure that the credentials of the IoT devices in the network are stored in a safem anner.

4.3.3 Security Requirements:

1. Ensure that the IoTdevices are safe and secure in the environment and are not attack
2. Ensure that the certificate and keys are issued to the IoT devices in a secure andrel iable channel.

CHAPTER5

SYSTEM REQUIREMENT SPECIFICATION

This chapter details the system requirements of the project.

5.1 Hardware platform:

Arduino (UNO)

Node MCU (ESP8266)

Temperature Sensor

5.2 Operating system and versions:

IoT Operating System 1: Embedded Linux.

IoT Operating System 2: Windows 10

Generally, IoT will run on Embedded Linux and Windows Platform

5.3 Software requirements:

Visual Studio
Arduino IDE
XAMPP Server
MySQL Database
AES Encryption
SHA1 Hashing algorithm
Cryptography modules

5.4 Assumptions:

We assume that our environment is prone to attacks. An attacker can intercept any message sent across the network. NIST standard length of key: Since 2015, NIST recommends a minimum of 2048-bit keys IoT nodes and gateway are physically safe. A Key Distribution Centre can issue many certificates for the devices connected in the network.

5.5 Constraints:

5.5.1 Class 0 IoT devices:

These devices have very less storage capacity in the order less than of 10kB RAM and less than100kB flash memory. Their processing capacity is also less compared to class 1 and class 2 IoT devices. They are less secure when communicating with other devices in the IoT network. These devices are pre-configured and are connected to the gateway and, proxies in the network.

5.5.2 Class 1 IoT devices:

These devices have better storage capacity in the order of 10kB RAM and 100kB flash memory. Their processing capacity is greater compared to class 0 devices. They are designed for Constrained Application Protocols. They cannot execute HTTP, TCP and TLS protocols.

5.5.3 Class 2 IoT devices:

These devices have better storage capacity in the order of 50kB RAM and 250kB flash memory. Their processing capacity is greater compared to class 0 devices and class 1 devices. They are designed for Constrained Application Protocols. They can execute HTTP, TCP and TLS protocols.

Implementation of our project scheme on class0 and class1 IoT devices is limited.

5.6 Risks:

- When the IoT node needs a new key or wants to renew the present key due to security reasons before the expiry of the present key, the KDC (Key Distribution Centre) must be ready to meet the requirements of the IoT node.
- Improper usage of keys under certain situations can make a hacker easier to crack it.
- The length of the kay should be appropriate so that the value of the data is protected.

CHAPTER 6

SYSTEM DESIGN

This chapter describes the design aspects of the project. High-level design, system perspectives, design of classes, and UML diagrams that have helped better clarity of execution of the problem statement are presented in detail.

6.1 High-Level System Design:

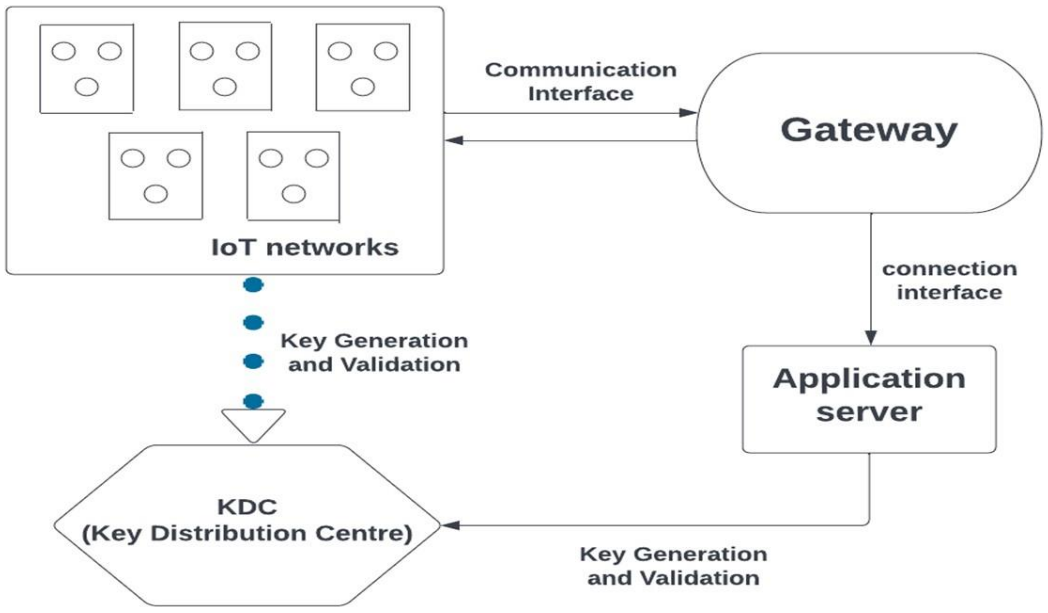

Communication Interface
Gateway
IoT networks
connection interface
Key Generation and Validation
Application server
KDC (Key Distribution Centre)
Key Generation and Validation

Figure 6.1.1: high-level design

IoT network:

This network consists of a network of IoT devices which are connected to the cloud via gateways.

Gateway:

A gateway is a physical device or virtual platform that connects sensors, IoT modules, and smart devices to the cloud.

Application Server:

Because they are responsible for providing commercial services to customers, application servers are sometimes seen as the most significant component of the IoT cloud. They must provide the necessary infrastructure and an adequate environment in which to operate many apps that follow certain application protocols. In the traditional cloud, application servers are typically based on HTTP. HTTP servers use Transmission Control Protocol (TCP) connections with clients to send and receive requests. Once connections have been established, an HTTP server can listen for requests from clients on specific ports and respond appropriately.

KDC:

A key distribution centre (KDC) is a server that is responsible for providing keys to the users in the network for encryption and decryption of data.

6.2 Certificate Creation:

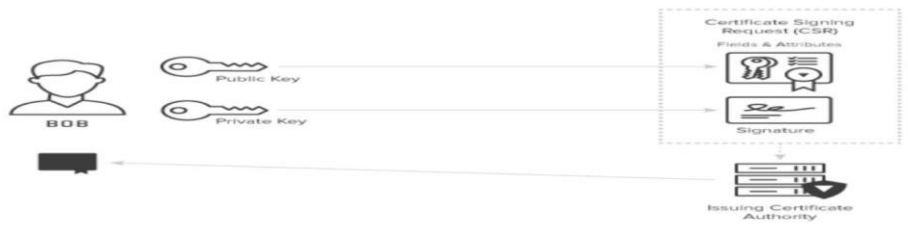

Certificate Signing Request (CSR)
Fields & Attributes
Public Key
BOB
Private Key
Signature
Issuing Certificate Authority

Figure 6.2.1: Digital Certificate Creation

The following tells how certificate is created, which mainly relies on asymmetric encryption:

Initially the private and public key are generated.

The validity for the certificate is set.

The certificate consists of version, serial, issuer, subject, signature algorithm(SHA256).

6.3 ER Diagram:

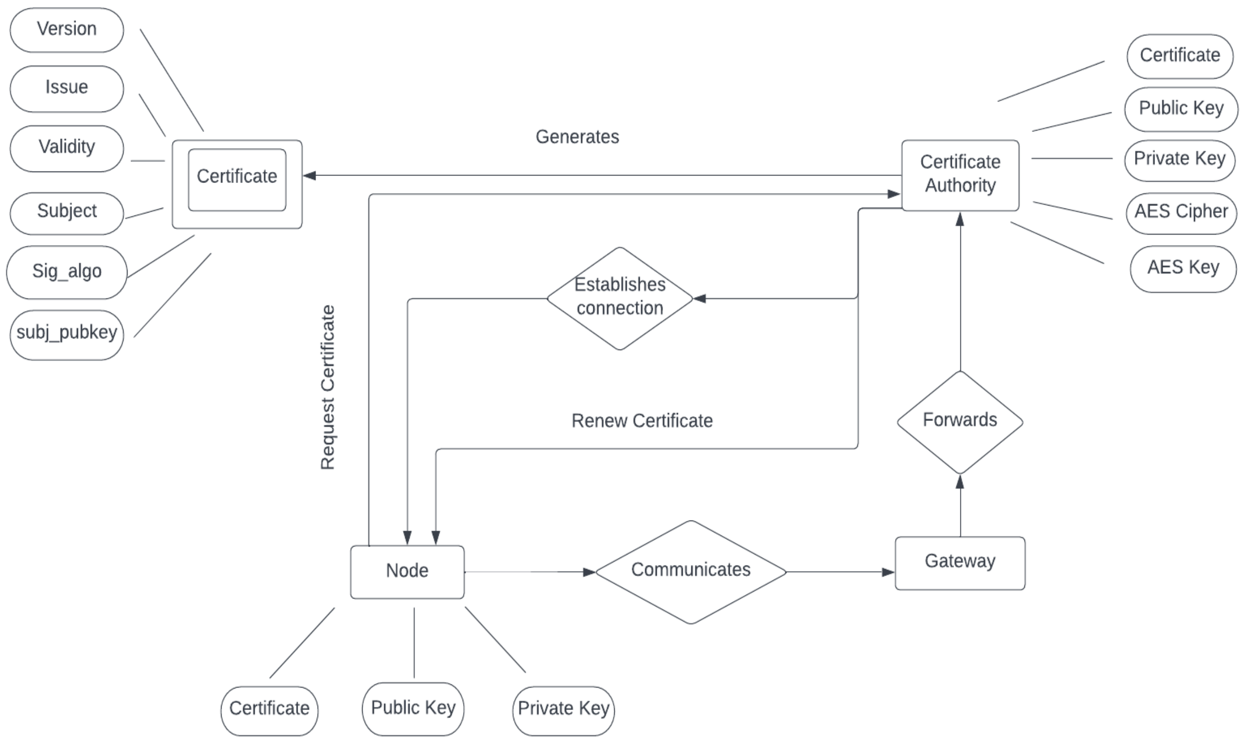


Figure 6.3.1: ER Diagram

6.4 Use Case Diagram:

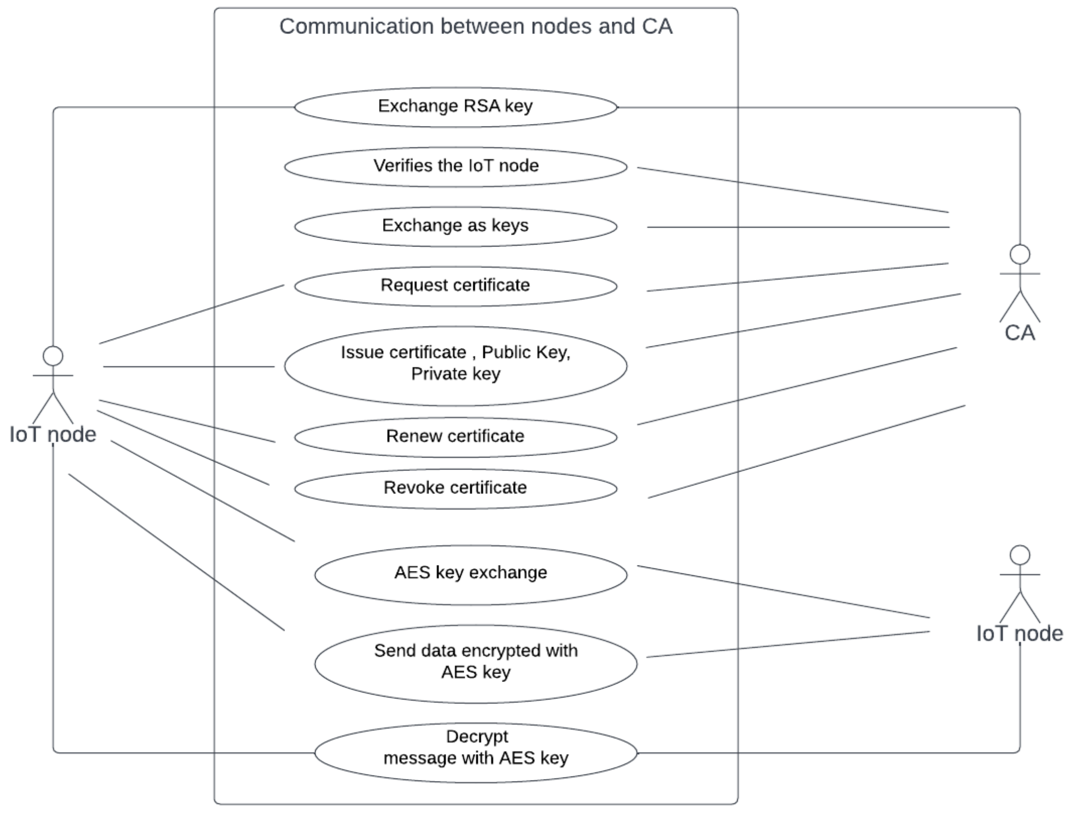


Figure 6.4.1: Use Case Diagram

When an IoT node is created in the network, it first establishes a secure connection with the Certificate Authority by exchanging its public key, AES cipher key and cipher iv created by the CA only if the IoT node is registered offline with the CA and it belongs to the network.

The certificate authority issues the certificate to the IoT node with certain validity.

When the certificate is set up for the IoT node, it can now communicate to other nodes in the network with the certificate issued by the certificate authority.

The certificate includes a private key, public key, cryptographic algorithm used for encryption and decryption, key size, and validity. The IoT nodes communicate with each other with their private and public keys.

Once the IoT node's certificate expires i.e., it reaches its validity. It requests for a new certificate from the certificate authority (CA) i.e., the Key Distribution Centre (KDC).

There is another situation where the IoT node can request a new certificate before its existing certificate reaches its validity. Reasons for this can be a disclosure of its existing certificate in the IoT network. So, when the KDC issues a new certificate to such IoT nodes. The existing one becomes inactive and dead. It will no longer be available. It communicates with the new certificate thereafter.

6.5 Master class diagram:

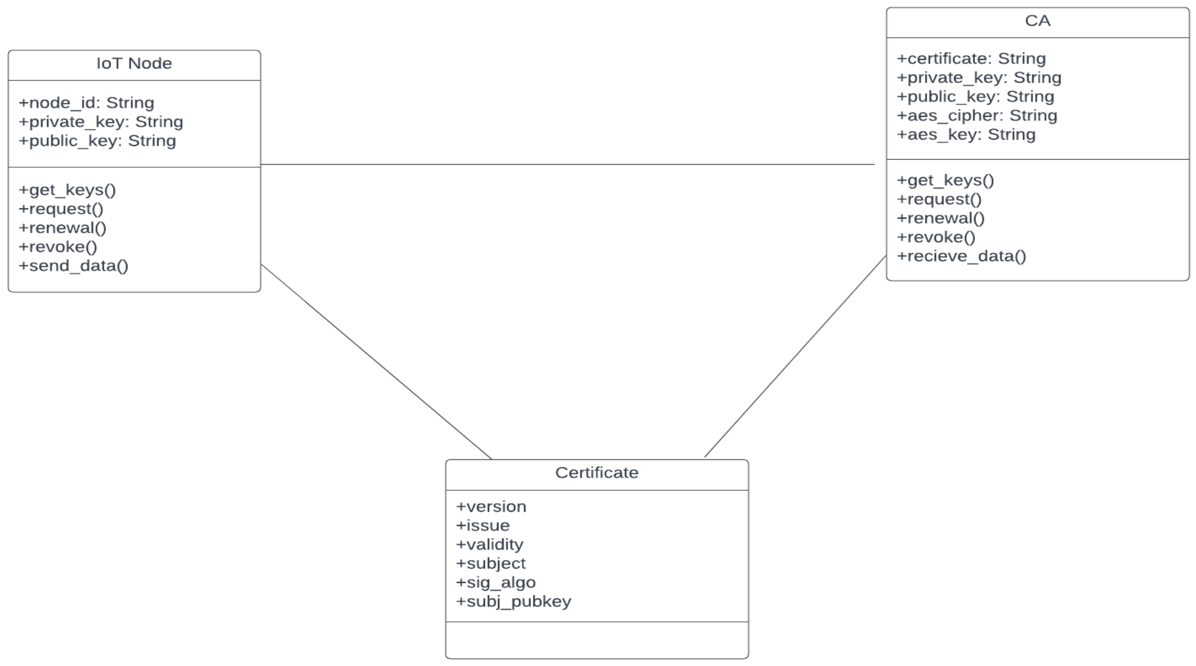


Figure 6.5.1: Master Class Diagram

IoT node:

It consists of basic information related to IoT nodes such as node Id, node name. It applies for certificates when it is newly created in the network.

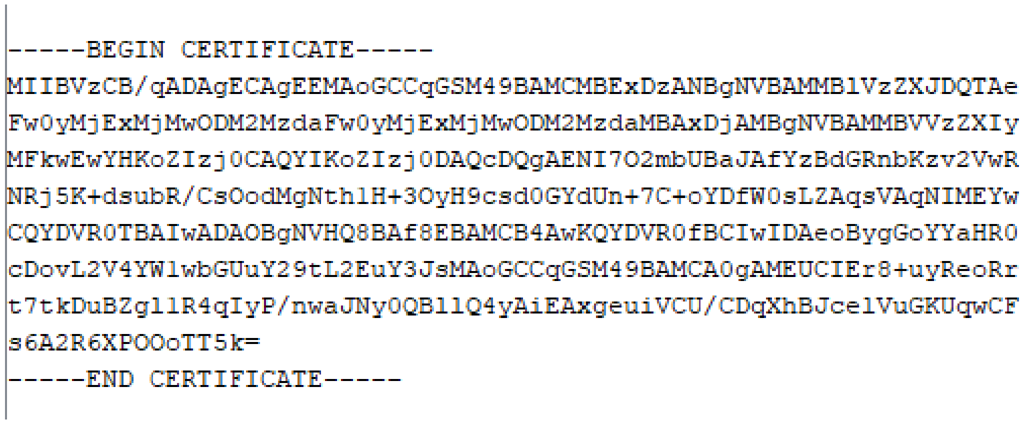
Certificate Authority:

Certificate Authority issues the certificate on request of the IoT node. It also renews and revokes the certificate on expiry or node’s request. All these tasks are done Key Distribution Centre (KDC)

Certificate:

Certificate consists of domain name, validity information, version, issue date, public key, encryption algorithm used.

6.6 Certificate format:



```
-----BEGIN CERTIFICATE-----
MIIBVzCB/qADAgECAgEEMAoGCCqGSM49BAMCMBExDzANBgNVBAMMB1VzZXJDQTAe
Fw0yMjExMjMwODM2MzdaFw0yMjExMjMwODM2MzdaMBAxDjAMBgNVBAMMBVVzZXIy
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAENI7O2mbUBaJAfYzBdGRnbKzv2VwR
NRj5K+dsubR/CsOodMgNth1H+3OyH9csd0GYdUn+7C+oYDfW0sLZAqsVAqNIMEYw
CQYDVR0TBAIwADAOBgNVHQ8BAf8EBAMCB4AwKQYDVR0fBCIwIDAeoBygGoYYaHR0
cDovL2V4YW1wbGUuY29tL2EuY3JsMAoGCCqGSM49BAMCA0gAMEUCIEr8+uyReoRr
t7tkDuBZg11R4qIyP/nwaJNy0QB11Q4yAiEAxgeuiVCU/CDqXhBJce1VuGKUqwCF
s6A2R6XPOOoTT5k=
-----END CERTIFICATE-----
```

Figure 6.6.1: Certificate format

The above image are the sample pictures of the certificate generated.

CHAPTER 7

PROPOSED METHODOLOGY AND ALGORITHMS

Encryption is a key feature for protecting and securing data. Encryption is an act of converting a confidential message to an encoded and unreadable format for high-security purposes. This unreadable text is known as ciphertext. Examples **of Encryption Algorithms are Advanced Encryption Standard (AES), Triple Data Encryption Standard (DES), Rivest–Shamir–Adleman (RSA) algorithm, Blowfish, Two fish, Secure-Socket Layer (SSL),** and so on.

7.1 Encryptions used:

1. AES encryption: The AES (Advanced Encryption Standard) is a symmetric encryption algorithm which uses the same key for encryption and decryption on either side of the communication. We use AES-128 which uses 128-bit key length to encrypt and decrypt the data. AES cipher key and cipher iv are generated by the certificate authority and exchanged with the IoT node for creating a secure channel between them.

2. SHA1(Secure Hash Algorithm): This algorithm takes the input and generates 160-bit hashed value. This method is used to hash the mac address of the IoT node. The hashed values and stored in Certificate Authority's database.

3. SHA256: This algorithm converts the input to a hash value of length 256 bits. This algorithm is used to encrypt the certificate generated by the Certificate Authority.

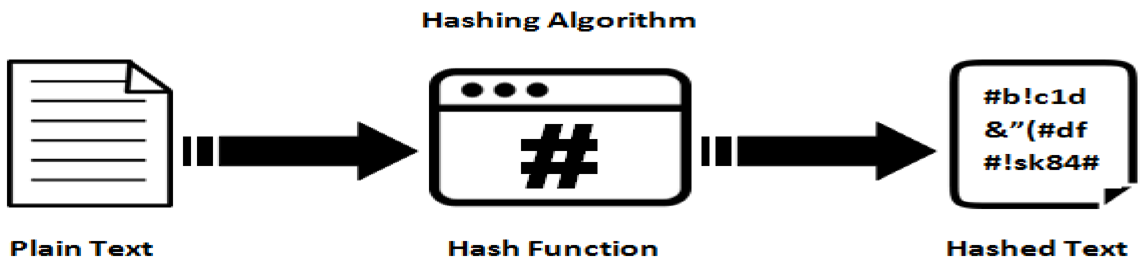


Figure 7.1 Hashing

CHAPTER 8

IMPLEMENTATION AND PSEUDOCODE

8.1 Implementation – Hardware Setup:

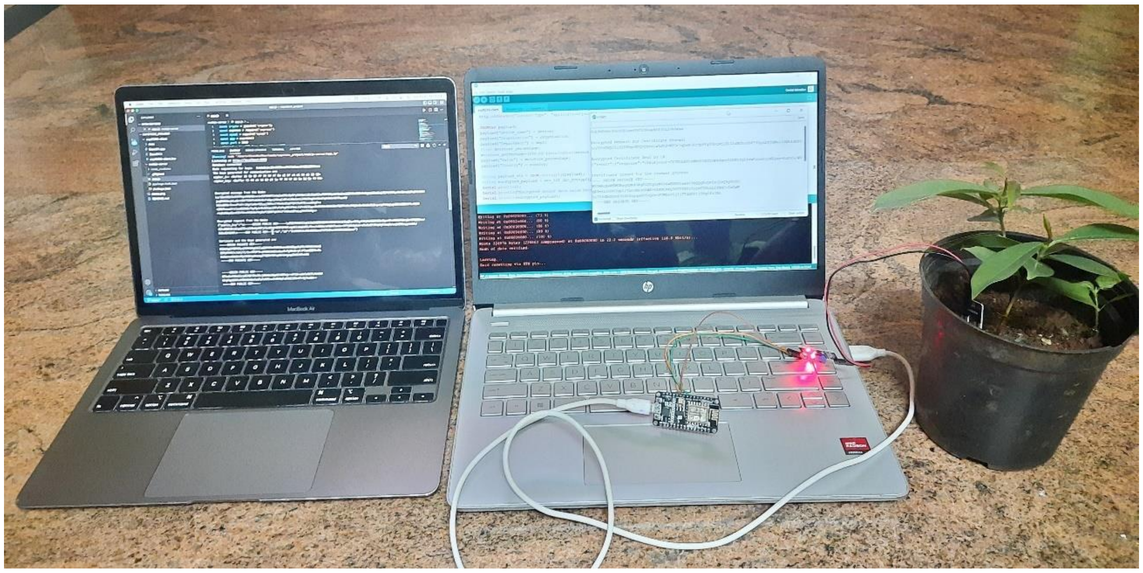


Fig 8.1.1: Hardware Setup

8.2 Client sending sensor data:

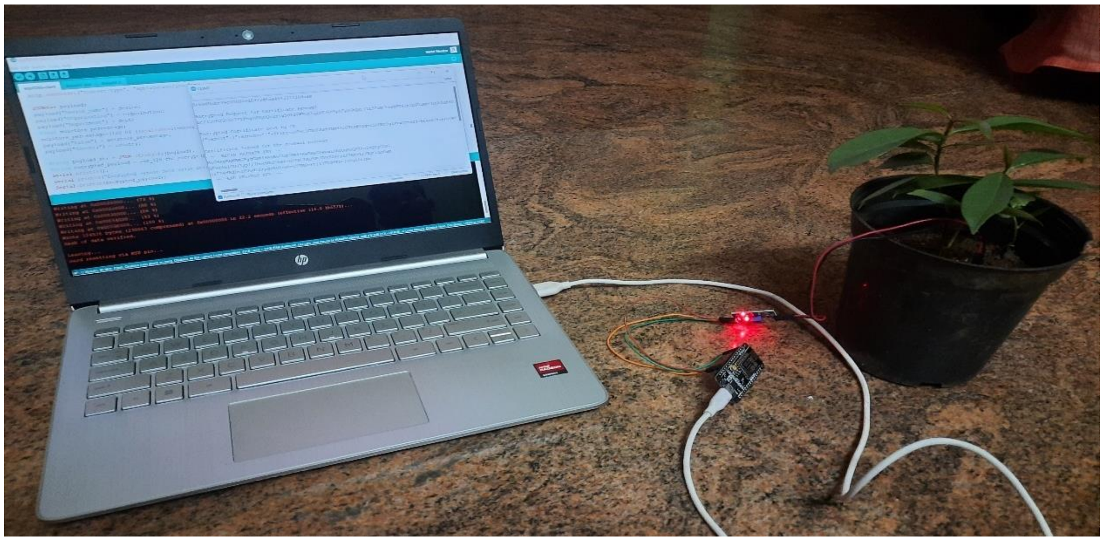


Fig 8.2.1: Client sending sensor data

8.3 Nodejs – Server code:

Certificate generation function:

```
let generate_certificate = function(date){
```

Generating key value pairs:

```
var kp = rs.KEYUTIL.generateKeypair("EC", "secp256r1");
var prv = kp.prvKeyObj;
var pub = kp.pubKeyObj;
var prvpem = rs.KEYUTIL.getPEM(prv, "PKCS8PRV");
var pubpem = rs.KEYUTIL.getPEM(pub, "PKCS8PUB");
date.setFullYear(date.getFullYear() - 1);
var x = new rs.KJUR.asn1.x509.Certificate({
  version: 3,
  serial: {int: 4},
  issuer: {str: "CA"},
  validity:  new Date(String(date)),
  subject: {str: "Certificate Authority"},
  sbjpubkey: pub, // can specify public key object or PEM string
  ext: [
    {extname: "basicConstraints", cA: false},
    {extname: "keyUsage", critical: true, names:["digitalSignature"]},
    {extname: "cRLDistributionPoints",
     array: [{fulluri: 'http://example.com/a.crl'}]}
  ],
  sigalg: "SHA256withECDSA",
  cakey: prv // can specify private key object or PEM string
});

// you can modify any fields until the certificate is signed.
x.params.subject = {str: "/CN=User2"};
const cert = x.getPEM();
return [prvpem , pubpem , cert , x.params.validity];
}
```

Creating connection to the database server:

```
const db = mysql.createConnection({
  host:'localhost',
  user:'root',
  password:'',
  database:'certificateauthority'
})

db.connect(err => {
  if (err) throw err;
  console.log("Database connected");
})
```

Get keys function:

```
app.post('/get_keys', (req, res) => {

  cipher_iv = crypto.randomBytes(16); // IV
  cipher_key = crypto.randomBytes(16);
  const public_key = req.body.public_key;
  var id = req.body.id;
  let sql = "select id from nodes where id = " + mysql.escape(id);
  db.query(sql,(err,result) => {
    if (err) throw err;
    if (result.length) {
      console.log("Node requesting for keys: " , id);
      console.log("The keys generated for communication are");
      console.log("cipher_iv:" , cipher_iv);
      console.log("cipher_key:" , cipher_key);
      console.log();
      console.log();
      let data = Buffer.concat([cipher_key, cipher_iv]);
      const resp = encrypt(public_key, data);
      res.send(JSON.stringify({key: resp}));
    }
    else{
      console.log("node id not verified");
      res.send(JSON.stringify({key: "Node not recognised"}));
    }
  })
})
```

Request function:

```
app.post('/request',(req,res) =>{
  console.log("Encrypted message from the Node:");
  console.log(req.body.request);
  console.log();
```

```
 const request = aesDecrypt(req.body.request);
 console.log("Decrypted request from the Node: ");
 console.log(request);
 console.log();
 const response_1 = JSON.parse(request);
 const public_key=response_1["public_key"];
 var id = response_1["id"];
 let sql = "select id from nodes where id = " + mysql.escape(id);
 db.query(sql,(err,result) => {
  if (err) throw err;
  if (result.length) {
   const [private_key,pub_key,certificate,validity]=generate_certificate(new Date());
   var insert_data = {id:id,private_key:
private_key,public_key:pub_key,certificate:certificate,validity:validity};
   values = [[String(id),String(private_key),String(pub_key),String(certificate),validity]];
   let sql_1 = "INSERT INTO issuedcertificates(id,privatekey,publickey,certificate,validity)
VALUES ?";
   db.query(sql_1,[values],(err,result_1) => {
    if (err) throw err;
    let data=JSON.stringify({private_key: private_key,public_key:pub_key,certificate:certificate});
    let start_2=Date.now();
    let enc_data=aesEncrypt(data);
    let finish_2=Date.now();
    console.log("Certicate and the Keys generated are");
    console.log(private_key);
    console.log();
    console.log(pub_key);
    console.log();
```

```
        console.log(certificate);
        res.send(JSON.stringify({result: {response: enc_data}}));
      })
    }
    else{
      console.log("node id not verified");
      res.send(JSON.stringify({key: "Node does not belong to the Network"}));
    }
  })
  })
```

Renew function:

```
app.post('/renew',(req,res) =>{
  let start=Date.now();
  const request = aesDecrypt(req.body.request);
  let finish=Date.now();
  console.log("Decrypted request from the Node: ");
  console.log(request);
  console.log();
  const response_1 = JSON.parse(request);
  const public_key=response_1["public_key"];
  const id=String(response_1["id"]);
  const certificate=String(response_1["certificate"]);
  //var value=[]
  let sql = "select id from nodes where id = " + mysql.escape(id);
  db.query(sql,(err,result) => {
    if (err) throw err;
    if (result.length) {
```

```
console.log("Node_id:" , id);
let sql_1 = "select * from issuedcertificates where certificate = " + mysql.escape(certificate);
db.query(sql_1,(err,result_1) => {
  if (err) throw err;
  Object.keys(result_1).forEach(function(key){
    var row = result_1[key];
    var cert=row.certificate;
    if(certificate==cert){
    expiry = new Date(row.validity);
    new_date=new Date();
    console.log(expiry);
    console.log(new_date);
    if(new_date>expiry){
      let start_1=Date.now();
      const [private_key,pub_key,certificate,validity]=generate_certificate(new Date());
      let finish_1=Date.now();
      var insert_data = {id:id,private_key:
private_key,public_key:pub_key,certificate:certificate,validity:validity};
      values =
[[String(id),String(private_key),String(pub_key),String(certificate),String(validity)]];
      let sql_1 = "INSERT INTO renewdcertificates(id,privatekey,publickey,certificate,validity)
VALUES ?";
      db.query(sql_1,[values],(err,result_3) => {
      if (err) throw err;
      console.log("New certificate are keys are issued");
      console.log(private_key);
      console.log();
      console.log(pub_key);
```

```
          console.log();
          console.log(certificate);
          let                                                    data=JSON.stringify({private_key:
private_key,public_key:pub_key,certificate:certificate});
          let enc_data=aesEncrypt(data);

          res.send(JSON.stringify({result: {response: enc_data}}));
          })
        }
        else{
          console.log("Certificate is still valid");
          res.send(JSON.stringify({result: {response: aesEncrypt("Valid")}}));
        }
      }
      else{
        console.log("Certicate not issued");
      }
      });
    })
  }
  else{
    console.log("node id not verified");
    res.send(JSON.stringify({key: "Node does not belong to the Network"}));
  }
 })
})
```

Revoke function:

```
app.post('/revoke',(req,res) =>{
  let start=Date.now();
  const request = aesDecrypt(req.body.request);
  let finish=Date.now();
  console.log("Decrypted request from ESP8266: ");
  console.log(request);
  const response_1 = JSON.parse(request);
  const public_key=response_1["public_key"];
  const id=String(response_1["id"]);
  const certificate=String(response_1["certificate"]);
  //var value=[]
  let sql = "select id from nodes where id = " + mysql.escape(id);
  db.query(sql,(err,result) => {
    if (err) throw err;
    console.log(result);
    if (result.length) {
      console.log("Node_id:" , id);
      let sql_1 = "select * from certificates where certificate = " + mysql.escape(certificate);
      db.query(sql_1,(err,result_1) => {
        if (err) throw err;
        Object.keys(result_1).forEach(function(key){
          var row = result_1[key];
          var cert=row.certificate;
          if(certificate==cert){
            console.log("The certificate has been revoked");
          }
          else{
```

```
          console.log("Certificate is not revoked and valid for communication");
        }
      })
      });
    }
  else{
    console.log("node id not verified");
    res.send(JSON.stringify({key: "Node does not belong to the Network"}));
  }
 })

})
```

Send data function:

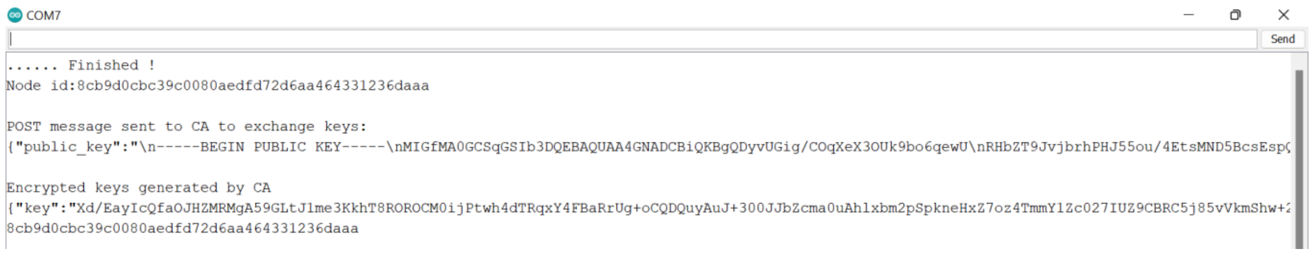
```
app.post('/send_data', (req, res) => {
  console.log("Encrypted data from Node using the certificate anf keys generated");
  console.log(req.body.request);
  console.log();
  const request = aesDecrypt(req.body.request);
  console.log("Decrypted data from Node: ");
  const msg=JSON.parse(request);
  console.log("Device id: ",msg["Device_name"]);
  console.log(msg["Organisation"]);
  console.log(msg["Department"]);
  console.log("Value:",msg["Value"]);
  console.log(msg["Country"]);
  // Do the login here.  Prepare the response
  let data = JSON.stringify({message:"value received"});
```

```
    res.send(JSON.stringify({result: {response: aesEncrypt(data)}}));
})
app.listen(3000, () => {
  console.log(`Listening at http://localhost:${port}`)
})
```

CHAPTER 9
RESULTS AND DISCUSSION

9.1 ESP8266 – Client:

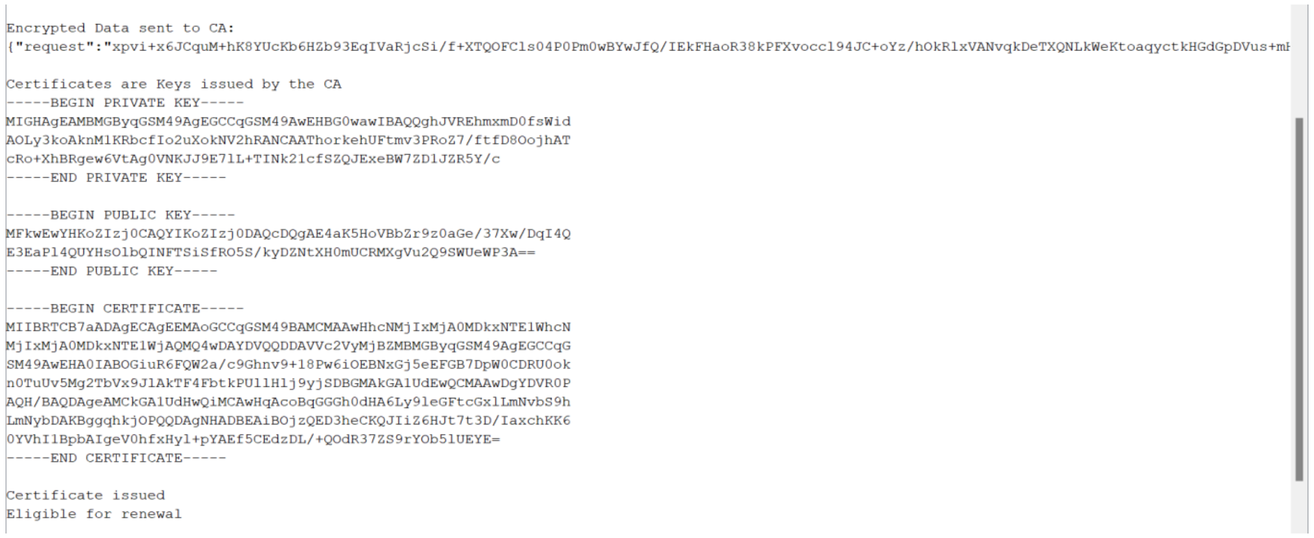
A secure connection is established between the IoT node and the Certificate Authority by exchanging the public key of the IoT node and by the exchanging the cipher key and cipher iv of the Certificate Authority encrypted using the public key sent by the node.


COM7

```
...... Finished !
Node id:8cb9d0cbc39c0080aedfd72d6aa464331236daaa

POST message sent to CA to exchange keys:
{"public_key":"\n-----BEGIN PUBLIC KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDyvUGig/COqXeX3OUk9bo6qewU\nRHbZT9JvjbrhPHJ55ou/4EtsMND5BcsEsp

Encrypted keys generated by CA
{"key":"Xd/EayIcQfaOJHZMRMgA59GLtJlme3KkhT8ROROCM0ijPtwh4dTRqxY4FBaRrUg+oCQDQuyAuJ+300JJbZcma0uAhlxbm2pSpkneHxZ7oz4TmmY1Zc027IUZ9CBRC5j85vVkmShw+
8cb9d0cbc39c0080aedfd72d6aa464331236daaa
```

Fig 9.1.1: Encrypted keys

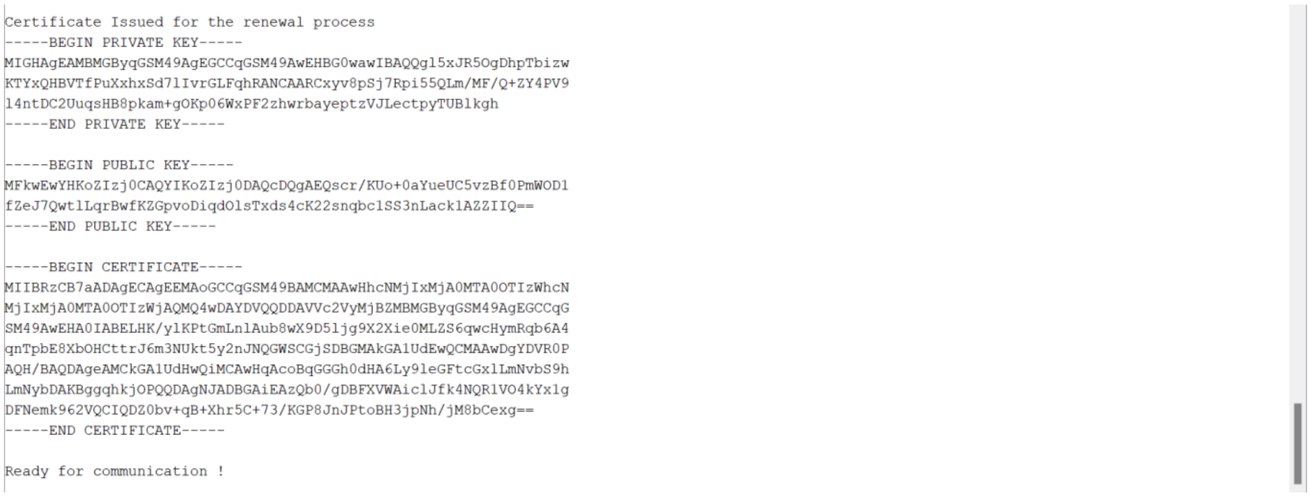
The Certificate Authorities (CAs) issues Certificate, Public key, Private Key to the IoT node on request.



```
Encrypted Data sent to CA:
{"request":"xpvi+x6JCquM+hK8YUcKb6HZb93EqIVaRjcSi/f+XTQOFCls04P0Pm0wBYwJfQ/IEkFHaoR38kPFXvoccl94JC+oYz/hOkRlxVANvqkDeTXQNLkWeKtoaqyctkHGdGpDVus+m

Certificates are Keys issued by the CA
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQghJVREhmxmD0fsWid
AOLy3koAknM1KRbcfIo2uXokNV2hRANCAAThorkehUFtmv3PRoZ7/ftfD8OojhAT
cRo+XhBRgew6VtAg0VNKJJ9E71L+TINk21cfSZQJExeBW7ZD1JZR5Y/c
-----END PRIVATE KEY-----

-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE4aK5HoVBbZr9z0aGe/37Xw/DqI4Q
E3EaPl4QUYHsOlbQINFTSiSfRO5S/kyDZNtXH0mUCRMXgVu2Q9SWUeWP3A==
-----END PUBLIC KEY-----

-----BEGIN CERTIFICATE-----
MIIBRTCB7aADAgECAgEEMAoGCCqGSM49BAMCMAAwHhcNMjIxMjA0MDkxNTE1WhcN
MjIxMjA0MDkxNTE1WjAQMQ4wDAYDVQQDDAVVc2VyMjBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABOGiuR6FQW2a/c9Ghnv9+18Pw6iOEBNxGj5eEFGB7DpW0CDRU0ok
n0TuUv5Mg2TbVx9JlAkTF4FbtkPUllHlj9yjSDBGMAkGA1UdEwQCMAAwDgYDVR0P
AQH/BAQDAgeAMCkGA1UdHwQiMCAwHqAcoBqGGh0dHA6Ly9leGFtcGxlLmNvbS9h
LmNybDAKBggqhkjOPQQDAgNHADBEAiBOjzQED3heCKQJIiZ6HJt7t3D/IaxchKK6
0YVhI1BpbAIgeV0hfxHyl+pYAEf5CEdzDL/+QOdR37ZS9rYOb5lUEYE=
-----END CERTIFICATE-----

Certificate issued
Eligible for renewal
```

Fig 9.1.2: Certificate issue

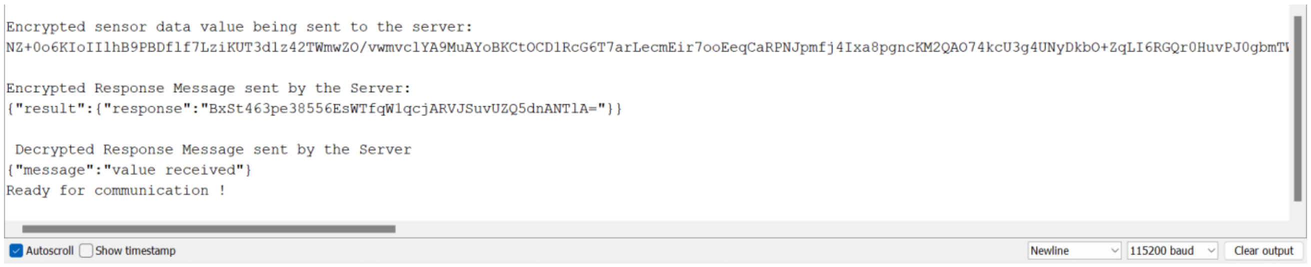
The Certificate Authority will issue a new certificate with a new pair of keysafter the expiry of the issued certificate.



```
Certificate Issued for the renewal process
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQgl5xJR5OgDhpTbizw
KTYxQHBVTfPuXxhxSd7lIvrGLFqhRANCAARCxyv8pSj7Rpi55QLm/MF/Q+ZY4PV9
l4ntDC2UuqsHB8pkam+gOKp06WxPF2zhwrbayeptzVJLectpyTUBlkgh
-----END PRIVATE KEY-----

-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQscr/KUo+0aYueUC5vzBf0PmWOD1
fZeJ7QwtlLqrBwfKZGpvoDiqdOlsTxds4cK22snqbc1SS3nLack1AZZIIQ==
-----END PUBLIC KEY-----

-----BEGIN CERTIFICATE-----
MIIBRzCB7aADAgECAgEEMAoGCCqGSM49BAMCMAAwHhcNMjIxMjA0MTA0OTIzWhcN
MjIxMjA0MTA0OTIzWjAQMQ4wDAYDVQQDDAVVc2VyMjBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABELHK/ylKPtGmLnlAub8wX9D5ljg9X2Xie0MLZS6qwcHymRqb6A4
qnTpbE8XbOHCttrJ6m3NUkt5y2nJNQGWSCGjSDBGMAkGA1UdEwQCMAAwDgYDVR0P
AQH/BAQDAgeAMCkGA1UdHwQiMCAwHqAcoBqGGGh0dHA6Ly9leGFtcGxlLmNvbS9h
LmNybDAKBggqhkjOPQQDAgNJADBGAiEAzQb0/gDBFXVWAiclJfk4NQR1VO4kYx1g
DFNemk962VQCIQDZ0bv+qB+Xhr5C+73/KGP8JnJPtoBH3jpNh/jM8bCexg==
-----END CERTIFICATE-----

Ready for communication !
```

Fig 9.1.3: Certificate renewal

The ESP8266 client sending data after successful certificate issued

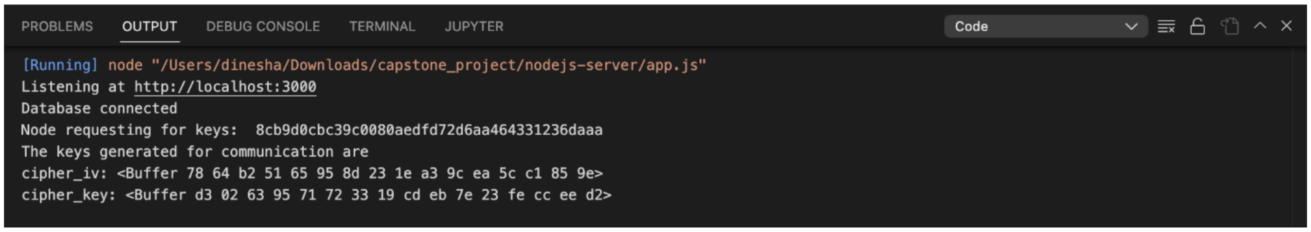


```
Encrypted sensor data value being sent to the server:
NZ+0o6KIoIIlhB9PBDflf7LziKUT3dlz42TWmwZO/vwmvclYA9MuAYoBKCtOCD1RcG6T7arLecmEir7ooEeqCaRPNJpmfj4Ixa8pgncKM2QAO74kcU3g4UNyDkbO+ZqLI6RGQr0HuvPJ0gbmT

Encrypted Response Message sent by the Server:
{"result":{"response":"BxSt463pe38556EsWTfqW1qcjARVJSuvUZQ5dnANTlA="}}

 Decrypted Response Message sent by the Server
{"message":"value received"}
Ready for communication !
```

Fig 9.1.4: Sending data

9.2 Nodejs- Server:

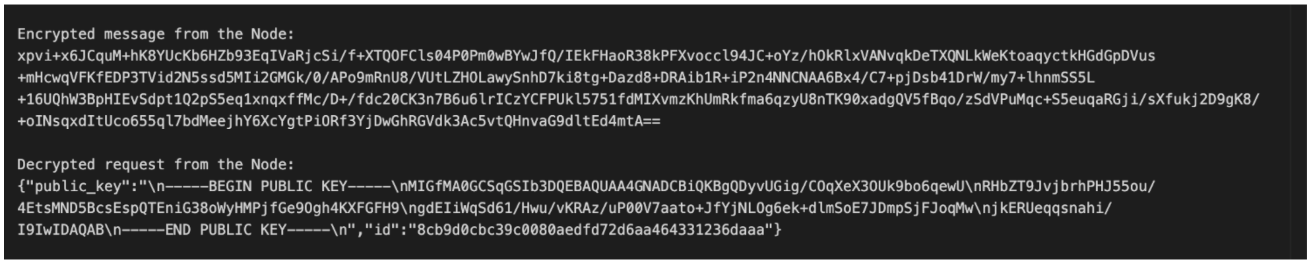
Keys generated by the CA to establish a secure connection with the node



```
[Running] node "/Users/dinesha/Downloads/capstone_project/nodejs-server/app.js"
Listening at http://localhost:3000
Database connected
Node requesting for keys:  8cb9d0cbc39c0080aedfd72d6aa464331236daaa
The keys generated for communication are
cipher_iv: <Buffer 78 64 b2 51 65 95 8d 23 1e a3 9c ea 5c c1 85 9e>
cipher_key: <Buffer d3 02 63 95 71 72 33 19 cd eb 7e 23 fe cc ee d2>
```

Fig 9.2.1: Key generated

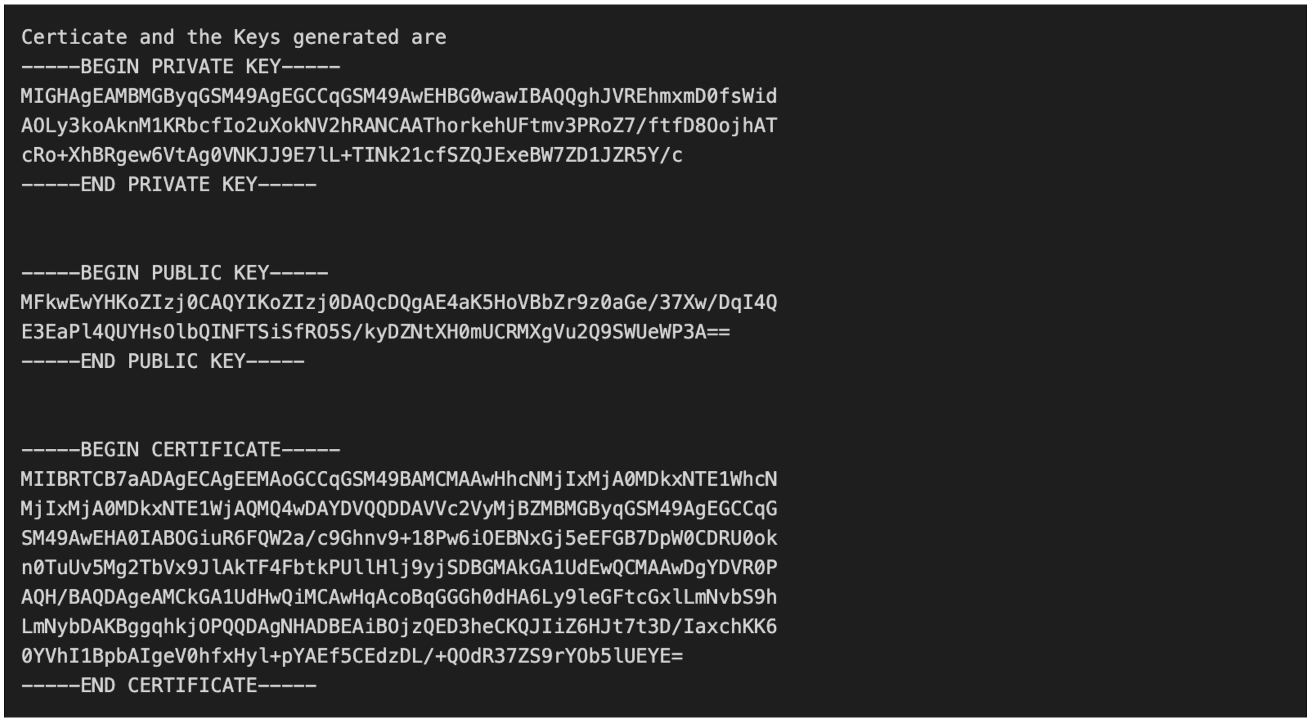
Encrypted message sent by the node during the certificate request process.



```
Encrypted message from the Node:
xpvi+x6JCquM+hK8YUcKb6HZb93EqIVaRjcSi/f+XTQOFCls04P0Pm0wBYwJfQ/IEkFHaoR38kPFXvoccl94JC+oYz/hOkRlxVANvqkDeTXQNLkWeKtoaqyctkHGdGpDVus
+mHcwqVFKfEDP3TVid2N5ssd5MIi2GMGk/0/APo9mRnU8/VUtLZHOLawySnhD7ki8tg+Dazd8+DRAib1R+iP2n4NNCNAA6Bx4/C7+pjDsb41DrW/my7+lhnmSS5L
+16UQhW3BpHIEvSdpt1Q2pS5eq1xnqxffMc/D+/fdc20CK3n7B6u6lrICzYCFPUkl5751fdMIXvmzKhUmRkfma6qzyU8nTK90xadgQV5fBqo/zSdVPuMqc+S5euqaRGji/sXfukj2D9gK8/
+oINsqxdItUco655ql7bdMeejhY6XcYgtPiORf3YjDwGhRGVdk3Ac5vtQHnvaG9dltEd4mtA==

Decrypted request from the Node:
{"public_key":"\n-----BEGIN PUBLIC KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDyvUGig/COqXeX3OUk9bo6qewU\nRHbZT9JvjbrhPHJ55ou/
4EtsMND5BcsEspQTEniG38oWyHMPjfGe9Ogh4KXFGFH9\ngdEIiWqSd61/Hwu/vKRAz/uP00V7aato+JfYjNLOg6ek+dlmSoE7JDmpSjFJoqMw\njkERUeqqsnahi/
I9IwIDAQAB\n-----END PUBLIC KEY-----\n","id":"8cb9d0cbc39c0080aedfd72d6aa464331236daaa"}
```

Fig 9.2.2: Encryption using keys

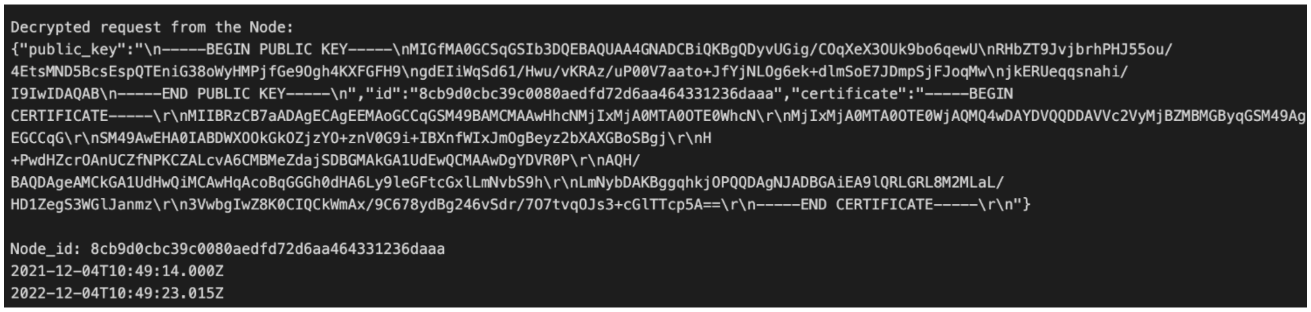
Certificate generated by the CA on certificate request by the node.



```
Certicate and the Keys generated are
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQghJVREhmxmD0fsWid
AOLy3koAknM1KRbcfIo2uXokNV2hRANCAAThorkehUFtmv3PRoZ7/ftfD8OojhAT
cRo+XhBRgew6VtAg0VNKJJ9E7lL+TINk21cfSZQJExeBW7ZD1JZR5Y/c
-----END PRIVATE KEY-----


-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAE4aK5HoVBbZr9z0aGe/37Xw/DqI4Q
E3EaPl4QUYHsOlbQINFTSiSfRO5S/kyDZNtXH0mUCRMXgVu2Q9SWUeWP3A==
-----END PUBLIC KEY-----


-----BEGIN CERTIFICATE-----
MIIBRTCB7aADAgECAgEEMAoGCCqGSM49BAMCMAAwHhcNMjIxMjA0MDkxNTE1WhcN
MjIxMjA0MDkxNTE1WjAQMQ4wDAYDVQQDDAVVc2VyMjBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABOGiuR6FQW2a/c9Ghnv9+18Pw6iOEBNxGj5eEFGB7DpW0CDRU0ok
n0TuUv5Mg2TbVx9JlAkTF4FbtkPUlLHlj9yjSDBGMAkGA1UdEwQCMAAwDgYDVR0P
AQH/BAQDAgeAMCkGA1UdHwQiMCAwHqAcoBqGGGh0dHA6Ly9leGFtcGxlLmNvbS9h
LmNybDAKBggqhkjOPQQDAgNHADBEAiBOjzQED3heCKQJIiZ6HJt7t3D/IaxchKK6
0YVhI1BpbAIgeV0hfxHyl+pYAEf5CEdzDL/+QOdR37ZS9rYOb5lUEYE=
-----END CERTIFICATE-----
```

Fig 9.2.3: Certificate generated at CA

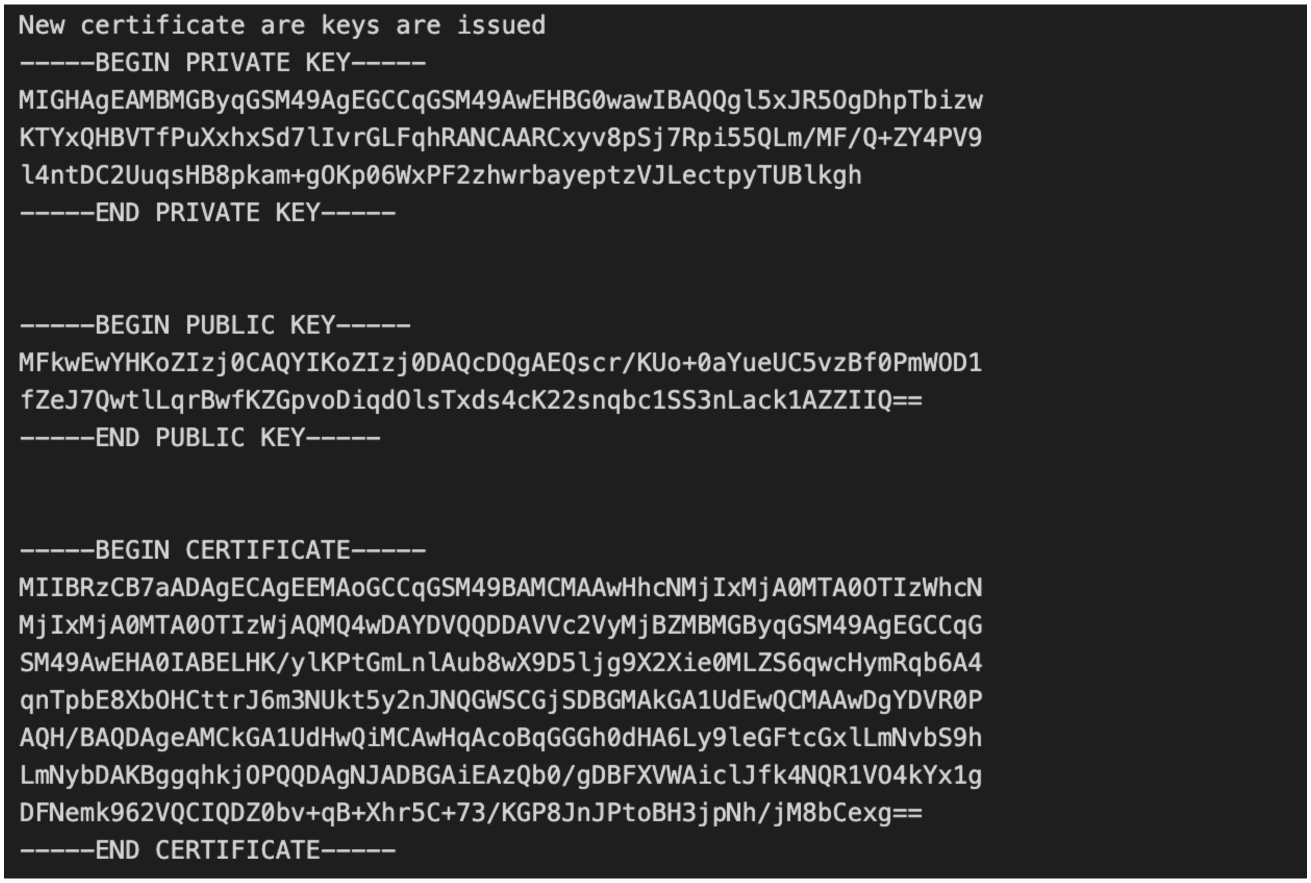
Certificate authority verifying the validity of the certificate during the certificate renewal process.



```
Decrypted request from the Node:
{"public_key":"\n-----BEGIN PUBLIC KEY-----\nMIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQDyvUGig/COqXeX3OUk9bo6qewU\nRHbZT9JvjbrhPHJ55ou/
4EtsMND5BcsEspQTEniG38oWyHMPjfGe9Ogh4KXFGFH9\ngdEIiWqSd61/Hwu/vKRAz/uP00V7aato+JfYjNLOg6ek+dlmSoE7JDmpSjFJoqMw\njkERUeqqsnahi/
I9IwIDAQAB\n-----END PUBLIC KEY-----\n","id":"8cb9d0cbc39c0080aedfd72d6aa464331236daaa","certificate":"-----BEGIN
CERTIFICATE-----\r\nMIIBRzCB7aADAgECAgEEMAoGCCqGSM49BAMCMAAwHhcNMjIxMjA0MTA0OTE0WhcN\r\nMjIxMjA0MTA0OTE0WjAQMQ4wDAYDVQQDDAVVc2VyMjBZMBMGByqGSM49Ag
EGCCqG\r\nSM49AwEHA0IABDWXOOkGkOZjzYO+znV0G9i+IBXnfWIxJmOgBeyz2bXAXGBoSBgj\r\nH
+PwdHZcrOAnUCZfNPKCZALcvA6CMBMeZdajSDBGMAkGA1UdEwQCMAAwDgYDVR0P\r\nAQH/
BAQDAgeAMCkGA1UdHwQiMCAwHqAcoBqGGGh0dHA6Ly9leGFtcGxlLmNvbS9h\r\nLmNybDAKBggqhkjOPQQDAgNJADBGAiEA9lQRLGRL8M2MLaL/
HD1ZegS3WGlJanmz\r\n3VwbgIwZ8K0CIQCkWmAx/9C678ydBg246vSdr/7O7tvqOJs3+cGlTTcp5A==\r\n-----END CERTIFICATE-----\r\n"}

Node_id: 8cb9d0cbc39c0080aedfd72d6aa464331236daaa
2021-12-04T10:49:14.000Z
2022-12-04T10:49:23.015Z
```

Fig 9.2.4: Certificate authority verifying validity

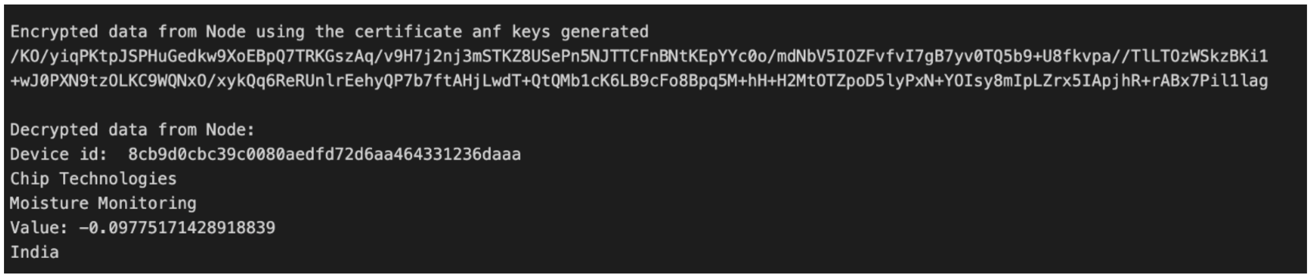
Certificate generated by the certificate authority during the renewal process.



```
New certificate are keys are issued
-----BEGIN PRIVATE KEY-----
MIGHAgEAMBMGByqGSM49AgEGCCqGSM49AwEHBG0wawIBAQQgl5xJR5OgDhpTbizw
KTYxQHBVTfPuXxhxSd7lIvrGLFqhRANCAARCxyv8pSj7Rpi55QLm/MF/Q+ZY4PV9
l4ntDC2UuqsHB8pkam+gOKp06WxPF2zhwrbayeptzVJLectpyTUBlkgh
-----END PRIVATE KEY-----


-----BEGIN PUBLIC KEY-----
MFkwEwYHKoZIzj0CAQYIKoZIzj0DAQcDQgAEQscr/KUo+0aYueUC5vzBf0PmWOD1
fZeJ7QwtlLqrBwfKZGpvoDiqdOlsTxds4cK22snqbc1SS3nLack1AZZIIQ==
-----END PUBLIC KEY-----


-----BEGIN CERTIFICATE-----
MIIBRzCB7aADAgECAgEEMAoGCCqGSM49BAMCMAAwHhcNMjIxMjA0MTA0OTIzWhcN
MjIxMjA0MTA0OTIzWjAQMQ4wDAYDVQQDDAVVc2VyMjBZMBMGByqGSM49AgEGCCqG
SM49AwEHA0IABELHK/ylKPtGmLnlAub8wX9D5ljg9X2Xie0MLZS6qwcHymRqb6A4
qnTpbE8XbOHCttrJ6m3NUkt5y2nJNQGWSCGjSDBGMAkGA1UdEwQCMAAwDgYDVR0P
AQH/BAQDAgeAMCkGA1UdHwQiMCAwHqAcoBqGGGh0dHA6Ly9leGFtcGxlLmNvbS9h
LmNybDAKBggqhkjOPQQDAgNJADBGAiEAzQb0/gDBFXVWAiclJfk4NQR1VO4kYx1g
DFNemk962VQCIQDZ0bv+qB+Xhr5C+73/KGP8JnJPtoBH3jpNh/jM8bCexg==
-----END CERTIFICATE-----
```

Fig 9.2.5: Renewed certificate

Data received at the sever using the keys generated by the certificate authority.

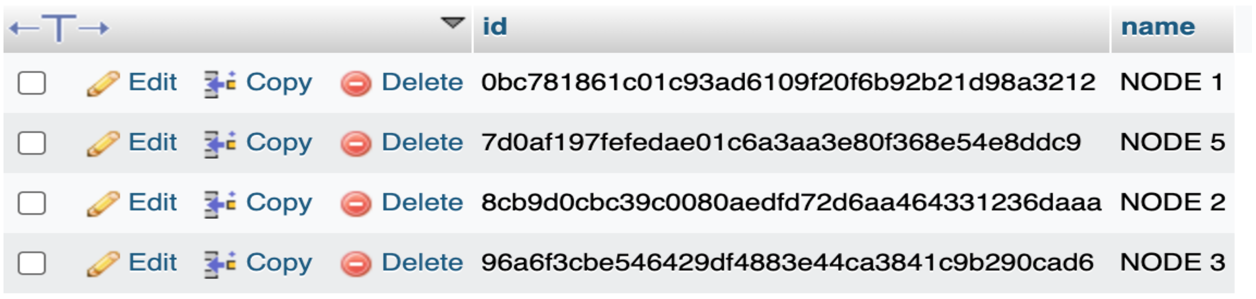


```
Encrypted data from Node using the certificate anf keys generated
/KO/yiqPKtpJSPHuGedkw9XoEBpQ7TRKGszAq/v9H7j2nj3mSTKZ8USePn5NJTTCFnBNtKEpYYc0o/mdNbV5IOZFvfvI7gB7yv0TQ5b9+U8fkvpa//TlLTOzWSkzBKi1
+wJ0PXN9tzOLKC9WQNxO/xykQq6ReRUnlrEehyQP7b7ftAHjLwdT+QtQMb1cK6LB9cFo8Bpq5M+hH+H2MtOTZpoD5lyPxN+YOIsy8mIpLZrx5IApjhR+rABx7Pil1lag

Decrypted data from Node:
Device id:  8cb9d0cbc39c0080aedfd72d6aa464331236daaa
Chip Technologies
Moisture Monitoring
Value: -0.09775171428918839
India
```

Fig 9.2.6: Data received at the sever sent by the Node MCU

9.3 Database:

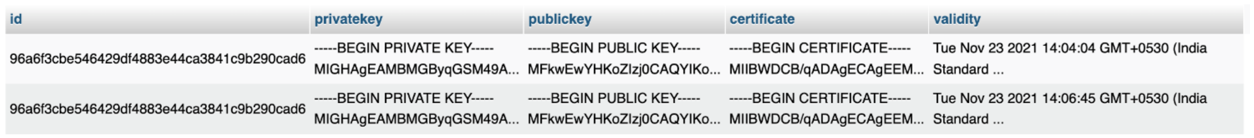
Nodes registered with the CA in the database.



			id	name
☐	Edit	Copy	Delete	0bc781861c01c93ad6109f20f6b92b21d98a3212	NODE 1
☐	Edit	Copy	Delete	7d0af197fefedae01c6a3aa3e80f368e54e8ddc9	NODE 5
☐	Edit	Copy	Delete	8cb9d0cbc39c0080aedfd72d6aa464331236daaa	NODE 2
☐	Edit	Copy	Delete	96a6f3cbe546429df4883e44ca3841c9b290cad6	NODE 3

Fig 9.3.1: Nodes registered with the CA in the database.

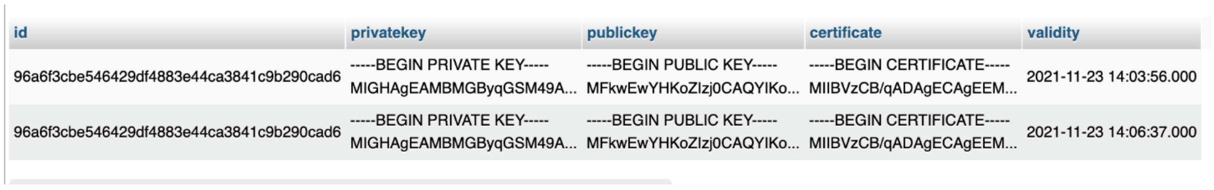
Issued certificates by the CA



id	privatekey	publickey	certificate	validity
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBWDCB/qADAgECAgEEM...	Tue Nov 23 2021 14:04:04 GMT+0530 (India Standard ...
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBWDCB/qADAgECAgEEM...	Tue Nov 23 2021 14:06:45 GMT+0530 (India Standard ...

Fig 9.3.2: Issued certificates by the CA

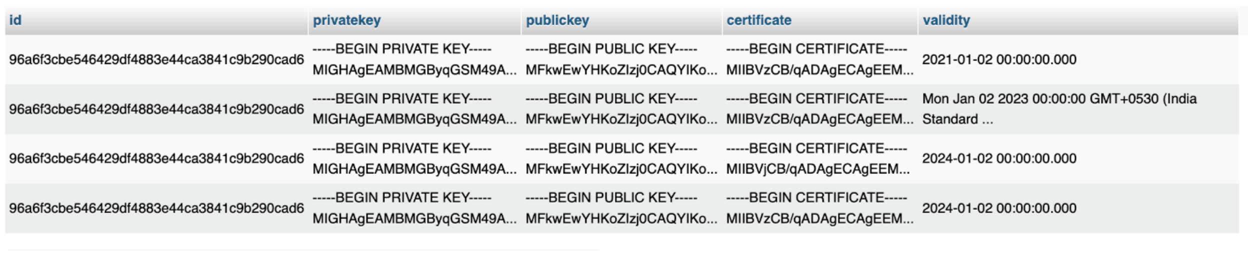
Renewed certificates by the CA



id	privatekey	publickey	certificate	validity
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVzCB/qADAgECAgEEM...	2021-11-23 14:03:56.000
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVzCB/qADAgECAgEEM...	2021-11-23 14:06:37.000

Fig 9.3.3: Renewed certificates by the CA

Revoked certificates by the CA



id	privatekey	publickey	certificate	validity
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVzCB/qADAgECAgEEM...	2021-01-02 00:00:00.000
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVzCB/qADAgECAgEEM...	Mon Jan 02 2023 00:00:00 GMT+0530 (India Standard ...
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVjCB/qADAgECAgEEM...	2024-01-02 00:00:00.000
96a6f3cbe546429df4883e44ca3841c9b290cad6	-----BEGIN PRIVATE KEY----- MIGHAgEAMBMGByqGSM49A...	-----BEGIN PUBLIC KEY----- MFkwEwYHKoZIzj0CAQYIKo...	-----BEGIN CERTIFICATE----- MIIBVzCB/qADAgECAgEEM...	2024-01-02 00:00:00.000

Fig 9.3.4: Revoked certificates by the CA

Execution time of AES 128 is lesser compared to AES 256 and AES 192

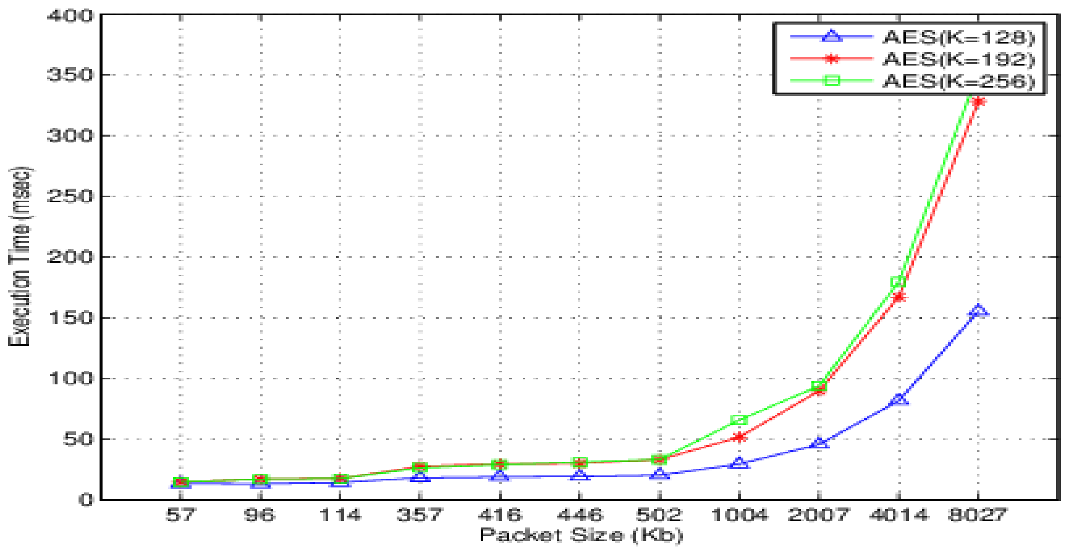


Fig 9.3.5: Execution time of AES 128 algorithm

Execution time of SHA 1 is lesser compared to SHA 256

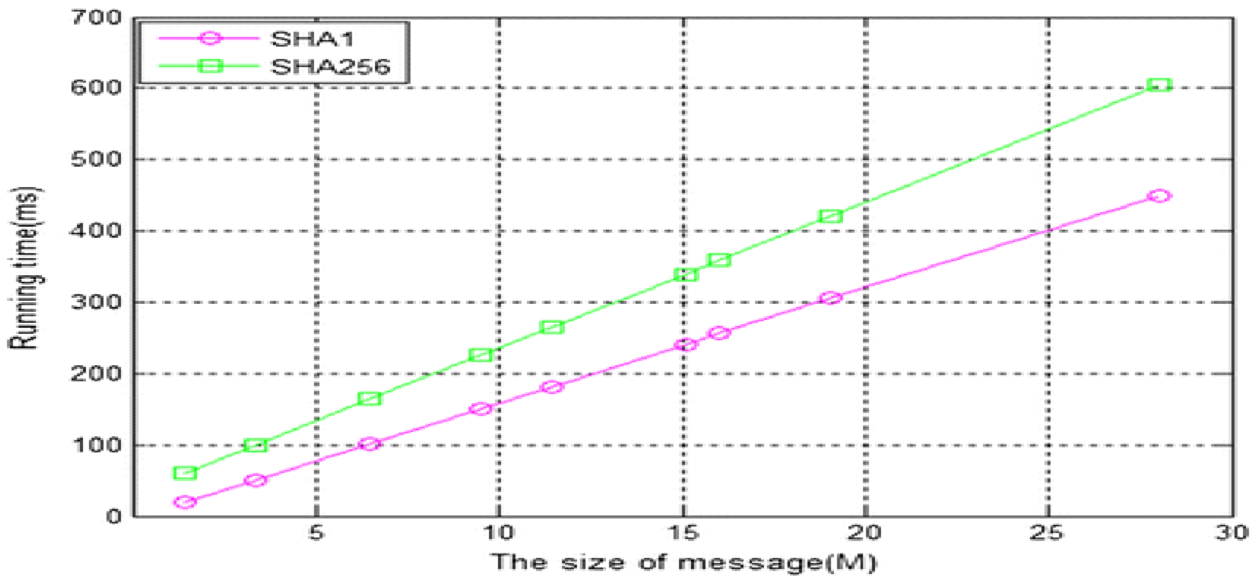


Fig 9.3.6: Execution time of SHA 1 algorithm

CHAPTER 10
CONCLUSION AND FUTURE WORK

The usage of constrained devices in the IoT network are growly rapidly in today's world and these devices share a lot of data over the internet. Therefore, the need to protect them is a major concern. Although many key exchange protocols are used, our methodology proposes a unique certificate renewal scheme for constrained devices in IoT network.

Our project aims to provide security to these devices by generating x509 certificate which uses EC secp256r1 algorithm to generate the keys. The certificate is signed with SHA256 algorithm to protect it from various attacks. The IoT nodes are ready for communication only when the certificate is valid and issued by the CA.

The IoT nodes can also renew its certificates by the certificate authority when its scheduled certificate's validity is expired. The certificate authority also has the privilege to revoke the certificates issued when it finds that the nodes keys are compromised or disclosed in the communication.

10.1 Summarize the key points.

Three main phases:

1. Generating the certificate
2. Renewing the certificate
3. Revoking the certificate

10.1 Provide a glimpse of Future work.

IEEE paperwork and Publication

Increase efficiency with respect to time and space complexity.

CHAPTER 11
REFERENCES / BIBLIOGRAPHY

[1] Rehman, A., Rehman, S.U., Khan, I.U., Moiz, M., & Hasan, S. (2016). Security and Privacy Issues in IoT. *Int. J. Commun. Networks Inf. Secur., 8*.

[2] Albarqi, Aysha & Alzaid, Ethar & Alghamdi, Fatimah & Asiri, Somaya & Kar,Jayaprakash. (2015). Public Key Infrastructure: A Survey. Journal of Information Security. 06. 31-37. 10.4236/jis.2015.61004.

[3] Gu, Hongxiang & Potkonjak, Miodrag. (2018). Efficient and Secure Group Key Management in IoT using Multistage Interconnected PUF. 1-6. 10.1145/3218603.3218646.

[4] A. Mavromatis, F. Ntavou, E. H. Salas, G. T. Kanellos, R. Nejabati and D. Simeonidou, "Experimental Demonstration of Quantum Key Distribution (QKD) for Energy-Efficient Software-Defined Internet of Things," 2018 European Conference on Optical Communication (ECOC), 2018, pp. 1-3, doi: 10.1109/ECOC.2018.8535267.

[5] Nafi, Mohammed & Bouzefrane, Samia & Omar, Mawloud. (2019). Matrix-Based Key Management Scheme for IoT networks. Ad Hoc Networks. 97. 102003. 10.1016/j.adhoc.2019.102003.

[6] I. Hussain, M. C. Negi and N. Pandey, "Proposing an Encryption/ Decryption Scheme for IoT Communications using Binary-bit Sequence and Multistage Encryption," 2018 7th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future

Directions) (ICRITO), 2018, pp. 709-713, doi: 10.1109/ICRITO.2018.8748293.

[7] Azrour, Mourade & Mabrouki, Jamal & Guezzaz, Azidine & Kanwal, Ambrina & Habib, Ullah & Khan,. (2021). Internet of Things Security: Challenges and Key Issues. Security and Communication Networks. 2021. 10.1155/2021/5533843.

[8] Idris Khan, Faraz & Hameed, Sufian. (2018). Understanding Security Requirements and Challenges in Internet of Things (IoTs): A Review.

[9] Alhasanat, M., Althunibat, S., Darabkh, K.A. *et al.*A Physical-Layer Key Distribution Mechanism for IoT Networks. *Mobile Netw Appl* 25, 173–178 (2020). https://doi.org/10.1007/s11036-019-01219-5

[10] Jung, J.-H.; Gohar, M.; Koh, S.-J. CoAP-Based Streaming Control for IoT Applications. Electronics 2020, 9, 1320. https://doi.org/10.3390/electronics9081320

[11] M. J. Dileep Kumar and P. Niranjan, "Lightweight Security In IoT: A Survey," 2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS), 2021, pp. 1366-1370, doi: 10.1109/ICAIS50930.2021.9395950.

[12] N. Mishra and S. Pandya, "Internet of Things Applications, Security Challenges, Attacks, Intrusion Detection, and Future Visions: A Systematic Review," in IEEE Access, vol. 9, pp. 59353-59377, 2021, doi: 10.1109/ACCESS.2021.3073408.

CHAPTER 12
APPENDIX

Certificate: It serves as proof of an endpoint's authenticity by securing authentication, encryption, and data integrity, and by protecting the device throughout its lifecycle.

Gateway: gateway is a physical device or virtual platform that connects sensors, IoT modules, and smart devices to the cloud.

KDC: A Key Distribution Centre (KDC) in cryptography is a system that is responsible for providing keys to the users in a network that shares sensitive or private data.

PKI: PKI performs encryption directly through the keys that it generates. It works by using two different cryptographic keys: a public key and a private key. Whether these keys are public or private, they encrypt and decrypt secure data.

NIST: The National Institute of Standards and Technology (NIST) is responsible for developing standards (Federal Information Processing Standards, or “FIPS”) and guidelines to protect non-national security federal information systems.

Trusted Execution Environment (TEE): A trusted execution environment (TEE) is a securearea of the main processor. It guarantees code and data loaded inside are protected with respect to confidentiality and integrity.

Trust Zone Operating System (TZOS) : Trusted operating systems split the services they offer (such as a file, print, or network access) into compartments, and allow only certain end-users, administrators, or applications into those areas.

FIDO2(Fast Identity Online): enables users to leverage common devices to easily authenticate to online services in both mobile and desktop environments (passwordless authentication).

The Rich Execution Environments (REEs), such as Android OS, cannot be fully audited andtrusted (due to their complexity). An isolated TEE can be used alongside the REE to implement security-sensitive functions. This makes it harder for an attacker to compromise these functions, as the attack surface is significantly reduced and is limited to communication with the TEE.

The Advanced Encryption Standard (AES): It is the most widely used symmetric block cipher. Galois Counter Mode (GCM) is a mode of operation for block ciphers that provides Authenticated Encryption. AES-GCM is a stream cipher that uses AES-CTR (Counter Mode) and the Galois Message Authentication Code (GMAC) internally.

HSM: A hardware security module (HSM) is a physical computing device that safeguards and manages digital keys, and performs encryption and decryption functions for digital signatures, strong authentication, and other cryptographic functions.