Development of a Secured IoT Data Transmission System for Smart Homes

Rotimi Alagbe Gbadebo
Department of Computer Engineering
Obafemi Awolowo University,
Ile-Ife, Nigeria

Temitope Omotosho Ajayi
Department of Computer Engineering
Obafemi Awolowo University,
Ile-Ife, Nigeria

Samuel Oluwafemi Dada
Department of Computer Engineering
Obafemi Awolowo University,
Ile-Ife, Nigeria

Bodunde Odunola Akinyemi
Department of Computer Science and Cybersecurity
Obafemi Awolowo University,
Ile-Ife, Nigeria

Ganiyu Adesola Aderounmu
Department of Computer Engineering
Obafemi Awolowo University,
Ile-Ife, Nigeria

ABSTRACT
The paper defined the needs of an IoT data transmission system based on the secured system, designed, constructed and tested its operation to provide reliable as well as secure communication. The system architecture was made of sensing, processing and encryption, communication, storage and user interface units. It was programmed on an ESP-32 microcontroller with capacitive touch sensor to generate event and buzzer to provide local feedback. AES-256 encryption was used to encrypt the data packets and transmit to Firebase Realtime Database. A web client written in ReactJS/NextJS was also created to authenticate users, retrieve and decrypt data, and visualize it on a dashboard. Findings indicated that the systems performed effectively, and the encrypted packet transmission latency was less than one second, maximum packet loss, and stable operation over long periods of time. In spite of features like reliance on Wi-Fi and Firebase, general absence of off-line storage, and performance issues, the prototype showed that it was possible to create safe and trustworthy interaction between smart home IoT devices with minimal expense microcontrollers and minimalistic cryptographic protocols.

General Terms
Internet of Things, Smart Homes, Cybersecurity, Embedded Systems, Wireless Networks.

Keywords
IoT, Smart Home Security, Data Encryption, MQTT, AES, Secure Transmission, Authentication

1. INTRODUCTION

Internet of Things (IoT) can be defined as a system of networked interconnected devices that have sensors, software, and other technologies, which allow collecting, processing, and transferring data through the internet or other communication networks [1]. Such IoT devices include smart appliances, wearable health sensors, industrial automation systems, etc., which work independently through the real-time exchange of data and make real-time decisions. The major building blocks of IoT are data-gathering sensors, network connectivity via Wi-Fi or LPWAN, data processing via edge or cloud computing, and powerful security measures to maintain data privacy and integrity [2]. IoT applications can be used in many areas, including smart homes, healthcare, agriculture, industrial automation, and smart cities, and they offer greater efficiency, better decision-making, reduced costs, and convenience [3];[1]. Nevertheless, IoT presents some major challenges, such as security risks, interoperability, and complex data management [2].

Though IoT-based smart homes have many benefits, they are vulnerable to many security threats. Studies also show that around 70% of the most popular IoT-based gadgets have severe vulnerabilities, and thus, they serve as excellent sources of cyberattacks, including unauthorised data access, malware intrusion, and Distributed Denial-of-Service (DDoS) attacks [4]. As an example of a successful Mirai botnet attack, the attacker used weak security settings on IoT devices to deploy a massive DDoS attack, which highlights the importance of upgrading security measures [5]. Also, according to a report by Unit 42, 57% of IoT devices can be targeted with serious attacks, and all traffic between 98% of all IoT devices is not encrypted, which can potentially reveal personal and confidential data [5]. Many IoT devices are limited in their computing capabilities, which reduces their ability to implement advanced encryption and authentication schemes, which further contributes to security threats.

The limitations of IoT devices, including low processing power, low memory, and tight energy efficiency demands make the successful implementation of traditional cybersecurity methods rather challenging. Although the security of most of the traditional encryption algorithms such as the AES-256 can be regarded as superior, their computational requirements can be prohibitive to the IoT applications. The computationally heavy algorithms often exceed processing power and power constraints of the common IoT devices. It is against this basic issue that researchers are paying growing attention to the development of lightweight cryptographic solutions capable of working within the resources limit of the system.

The evolution of these special algorithms is a delicate balance between ensuring proper levels of security and keeping the efficiency of the systems intact. Other groundbreaking studies have explored the different lightweight cryptographic implementations, assessing their usefulness and the security strength in the context of deploying IoTs [6]. There have also been analytical studies that have charted the changing environment of lightweight cryptography in the context of IoT that provide insightful views of the design philosophy, as well as approaches to the implementation of such specialized security solutions [7]. Another strong case of the need to create

effective transmission platforms within smart home settings is the economic consequences of IoT security. As of today, it is estimated that the global organizations spend about 150 billion dollars each year on IoT security management and data protection programs [8]. Moreover, the share of data transmission processes in IoT can reach approximately 15 percent of the total energy used in a smart home, which underscores the importance of the research of energy-efficient encryption algorithms and efficient means of compressing data [9]. Meanwhile, regulatory authorities are increasingly enacting tougher rules that will require the implementation of encryption, authentication, and data minimization tactics to enhance the security systems of the IoT [10]. Such regulations are becoming more and more important to manufacturers and service providers within the IoT space.

The total market size of smart home security in the world is expected to exceed 35.6 billion by 2025, due to the growing consumer awareness of the security threats posed by IoT devices and the widespread use of smart devices [11]. The ever-growing need in IoT security solutions will necessitate ongoing research in domains like edge computing, blockchain, and lightweight encryption to solve the current problem and guarantee the safe and efficient implementation of IoT in smart home settings [12].

1.1 Contribution and Novelty

This study contributes to knowledge by enhancing IoT security through a robust data transmission model that mitigates cyber threats in smart home environments. It ensured strong security while maintaining efficiency in resource-constrained IoT devices. The model improves real-time data transmission reliability, reducing packet loss and latency in Wi-Fi-based networks. Additionally, it provides a scalable solution adaptable to various smart home applications, ensuring seamless and secure communication among loT devices. Performance benchmarking using key metrics like packet modification rate, throughput, and data integrity under attack conditions will further validate its effectiveness in improving smart home security.

2. LITERATURE REVIEW

The rising trend of smart home systems based on IoT has brought with it a number of challenges especially in ensuring the safety of data transmission. IoT devices like smart cameras, thermostats, and sensors constantly share sensitive information, and are susceptible to attacks by cybercriminals such as eavesdropping, man-in-the-middle (MITM) attacks, and unauthorised access. Conventional encryption schemes, such as AES-128 and TLS/DTLS, are secure but can be resource-intensive and slow and thus not suitable on resource-heavyweight IoT devices [6]. Moreover, centralized security models cause single points of failure, which enhance the probability of data breaches and system vulnerabilities [13].

This chapter examines the current available studies regarding the security of the transmission of IoT data such as methods of encryption, authentication, intrusion detection systems, network security platforms. It also analyses the corresponding literature and outlines unresolved issues that support why a more efficient and scalable IoT data transmission model on smart homes is required.

2.1 Security Challenges in IoT-Based Smart Homes

The Internet of Things (IoT) has turned the modern home into a smart place with cameras, thermostats, lighting, smart locks, and voice assistants that connect via the internet to enhance convenience, automation and energy efficiency [14]. Nevertheless, even with these benefits, the IoT-driven smart houses experience significant security and privacy problems since sensitive data is processed incessantly between these interconnected networks. [15]

Among the key issues is poor passwords or default ones, as they expose numerous smart devices to unauthorised users and brute force assaults [16]. The second issue is undecrypted data transmission, where certain IoT devices transfer data without appropriate encryption, which puts the data at risk of interception and man-in-the-middle attacks [17]. Moreover, not all the devices are regularly updated with the firmwares, leaving them vulnerable to malware and various security vulnerabilities [18].

Withdrawal of privacy is also a major issue as smart home devices tend to gather user-specific reports like voice recorders, video feeds and user behavior pattern reports. Otherwise, such data can be leaked/shared without the permission of the user [19]. Moreover, certain IoT devices are publicly accessible to physical attacks, which may enable attacks to compromise hardware or network access [20].

Secured data transmission should be implemented in smart homes in order to address these challenges. The data that is being transmitted can be secured by encryption algorithms like AES and RSA, and an authentication system can ensure that only authorized persons can access the information [21]. It is also possible to enhance the integrity of device integrity, as well as the trust of users by incorporating regular updates, intrusion detection systems, and privacy-preserving strategies [22]. The recent events with hacked security cameras and insecure smart appliances demonstrated that a more robust protection of the IoT was very much needed [23]. Thus, this research paper is aimed at creating a safe IoT data transmission solution to use in smart houses to enhance user confidentiality, integrity, and privacy.

2.2 Implementation of Secure IoT Data Transmission

The secure IoT connection of data transmission is critical to securing smart home networks against cybercrimes, unapproved access, and information breach. IoT gateways and edge devices are significant to interconnecting smart devices with outer networks and thus, emerge as pivotal points of implementing security controls [17] The fact that these devices deal with communication between sensors, appliances, and cloud platforms requires that they are secured by a robust authentication, encryption, and system upgrades.

Authentication and access control is one of the key security measures. Multi-Factor Authentication (MFA), passwords, and biometric checks are some of the ways through which strong authentication is prevented so that unauthorised users cannot access IoT devices and control systems [24]. An encryption of data is another important measure. Encryption is used to secure the information sent between IoT devices, gateways, and servers by changing readable information to the encrypted coded format. Data transmission and storage are usually secured using protocols like Transport Layer Security (TLS), Datagram Transport Layer Security (DTLS) and Advanced Encryption Standard (AES) [22].

There should also be secure updates of the firmware and software to ensure device integrity. Secure boot mechanisms and digitally signed updates can be used to identify software

authenticity prior to its installation and minimize the chances of malware or unauthorised changes [16]. Also, firewalls can be used to regulate network traffic and prevent suspicious connections. Advanced deep packet inspection (DPI) and packet-intensive firewalls have the capability to examine traffic contents in detail to identify harmful activity [18].

Intrusion Detection Systems (IDS) are additional tools that enhance the security of IoT, by detecting networks and the occurrence of attacks. Signature based IDS identifies the patterns of attacks that are known whereas Anomaly based IDS identifies unusual activities and zero-day attacks through intelligent methods [6] Also, blockchain technology has the potential to enhance the security of the IoT through decentralization and inability to manipulate data through storage of data on devices and transaction logs. It not only improves data integrity but also ensures secure authentication without the support of a single centralized server [23].

Thus, authentication, encryption, secure updates, firewalls, intrusion detection systems, and blockchain technology are effective solutions that can be used to ensure secure IoT data transmissions in smart homes.

2.3 Existing Secured IoT Systems and Their Limitations

A number of security architectures have been created to enhance data transmission in Internet of Things (IoT) settings by applying encryption, authentication, access control and intrusion detection protocols. They can be used to defend smart devices and networks against cyber threats, unauthorized access, and data leaks. Popular encryption methodologies like Advanced Encryption Standard (AES), and Rivest-Shamir-Adleman (RSA) are highly secure encryption protocols that can be used to protect the data being transmitted, yet they frequently need a significant amount of computation resources, and therefore are inapplicable to IoT devices with limited power and capabilities that are commonly found in the smart home [25]. Likewise, homomorphic encryption which is characterized by the advanced means of encrypted data processing has a safe way to handle the process but presupposes high processing power and a slowing system.

Multi-Factor Authentication (MFA) and Role-Based Access Control (RBAC) are popular measures of authentication so that only authorized users and devices gain access to the networks of the IoT [26]; [18]. But most of these solutions are based on centralized servers, which may not be very scalable and be a single point of failure should they be compromised. The security models based on blockchains facilitate decentralized and resistant to tampering authentication, however, with high computational requirements and network latencies, which is not applicable to lightweight IoT hardware and real-time smart home solutions [27]; [28].

Machine learning-based Intrusion Detection and Prevention Systems (IDPS) can detect suspicious activity and detecting novel patterns of attacks in real-time. These systems might be prone to false alarms and need regular retraining to fit the changing threats regardless of the advantages they have [24]; [22]. More so, most current security systems are energy-heavy, decrease the efficiency of devices, and take a long time to respond to security risks.

These shortcomings demonstrate that better secured IoT systems, capable of being lightweight, scalable, energy-saving, and required in smart homes, are needed. The use of Elliptic Curve Cryptography (ECC), hybrid blockchain-edge computing models, and more efficient intrusion detection systems can offer higher security and less resource use with increased real-time performance [29]; [30].

3. METHODOLOGY

The design, development and implementation of the secured IoT data transmission system for smart homes is organized in a modular engineering framework. The methodology emphasizes integrating hardware and software, end-to-end security protocols, and real-time performance to enable secure communication in a home setting.

3.1 Research Design and Framework

This study used the experimental research design to develop a functional prototype. The architecture is inspired by the three-tiered hierarchy for IoT:

i. Perception Layer: Physical capturing of touch events, based on capacitive technology.
ii. Network Layer: Enables the safe transfer of data packets on Wi-Fi (WPA3) and HTTP/MQTT.
iii. Application Layer: Delivers data visualization, decryption and user authentication via a cloud connected web dashboard.

The operational framework flow is synchronized with the security lifecycle for each event trigger:

i. Event Capture: Hardware Interrupt (GPIO Pin Status Change).
ii. Integrity Generation: Generate HMAC-SHA256 of the raw payload.
iii. Confidentiality: Encrypt (Payload + Hash) using AES-256.
iv. Transport: Guarantee transport through HTTPS/TLS to Cloud.
v. FireAuth is used to verify user credentials in Firebase Security Rules.

3.2 Hardware Architecture and Interfacing

The hardware selected was primarily for computational efficiency and to provide low level cryptographic acceleration.

3.2.1 Processing Unit (ESP-32)

Espressif Systems' ESP32-WROOM-32 was chosen as the main processor. It has a dual-core 240MHz processor, 4MB Flash memory and on-board hardware acceleration of AES and SHA functions. This will ensure that the heavy computation of a cryptographic operation does not result in large latency for real-time monitoring of sensors.

3.2.2 Sensing and Feedback Modules

The TTP223 Capacitive Touch Sensor was used because it does not have any mechanical components and is durable. It is powered by 2.0V - 5.5V and provides a Digital HIGH output signal if it detects a signal. A local feedback was provided in the form of an audible alarm using an Active Buzzer (85dB output) which was interfaced to GPIO16 to indicate a system state, successful transmission, and intrusion alarm.

3.3 Security Implementation and Architecture

The system uses a multi-layered security protocol to guarantee confidentiality, integrity and availability of data.

3.3.1 Data Confidentiality (AES-256 Encryption)

Structured JSON payloads are created for sensor events. They use Advanced Encryption Standard (AES-256) for encryption.

A firmware contains individual keys for authenticating device with a stored PIN pattern, and is used to make sure that the device can't be accessed physically or digitally by unauthorised entities.

3.3.2 Data Integrity and Authentication (SHA-256)

A Secure Hash Algorithm (SHA-256) is used to ensure that data isn't corrupted while sending. Each packet is accompanied by a Message Authentication Code (MAC) which the web client uses to ensure the message is authentic and hasn't been tampered with before it is decrypted.

3.3.3 Threat Response and Lockout Logic

A "Trial-Limit" intrusion detection system is integrated in the system. When an incorrect pattern is entered 3 times consecutively, the firmware enters a Lockout State, and does not accept sensor input for a given period of time and sends an alert to the Cloud server.

3.4 System Flow Process Design

Figure 1 shows an overview of the process flow in the system for the transmission of secured IOT data in the smart home. It can be broken down into four large parts: sensor unit, processor and encryption unit, communications and storage unit and user interface unit.

3.4.1 Sensor Unit (Input Unit)

The input unit includes the touch sensor TTP223, placed in the smart home to track physical changes in the environment such as touch and transmit electrical signal to the microcontroller when activated, which will be the primary source of data in the system.

3.4.2 Processing and Encryption Unit

Here, the microcontroller (ESP-32) reads the connected sensors in real-time, adds a time-stamp to each event on-the-fly to track the time of occurrence, formats the data with JSON to support a lightweight interoperability approach, and encrypts the data with AES-256 and an authentication tag to ensure only authorized devices can read or modify the data sent, effectively converting raw sensor data into secure and immutable packets.

3.4.3 Communication and Storage Unit

This unit is concerned with the secure transmission of the data whereby the Arduino encrypts the data which is then transmitted to the ESP-32 Wi-Fi device which in turn connects to a local Wi-Fi network and transmits the encrypted packets of data to a specified cloud or remote server over the HTTP protocol where the data is stored within a database system and is made available when the user interface requests it.

3.4.4 User Interface Unit:

This component includes a mobile application that runs on a smartphone or tablet and periodically requests the remote server to provide the most recent updates on the sensor, then decrypts and authenticates the received JSON contents using in-built cryptographic.

3.5 System Operation

The system functions under a well-articulated flowchart shown in Figure 2 that defines the communication between the hardware components, communication protocols and user input. This system will allow a secure access point through capacitive touchpad, wireless connectivity, and encrypted communication.

3.5.1 System Initialization

The system configures itself when it begins booting up. The ESP32 microcontroller is turned on and connected to TTP223 capacitive touch sensor and a buzzer. These hardware pieces compose the interface between the user and the system through which you can detect inputs and provide feedback by sound. System diagnostics and preparation to be connected to the network is also a part of the initial setup procedure.

3.5.2 Wi-Fi Connection Setup

Once initialised, the ESP32 will expect connection to a designated Wi-Fi network with security level WPA3, and encryption with AES 256 bits. This will provide great security to the connection between the device and the server. In case of a connection error, the system records the error, emits a buzzer alarm to attract attention and will re-establish the connection. Once a successful connection is made, then the system will go ahead to create a secured communication channel.

3.5.3 Secure MQTT Configuration

The system establishes a secure MQTT (Message Queuing Telemetry Transport) protocol to allow lightweight and reliable communication between the ESP32 and a remote server. To provide security to both the communication and data during transit, Transport Layer Security (TLS) is turned on and the ESP32 is authenticated prior to any data communication. This guarantees that the authorized devices are the only ones to interact with the server.

3.5.4 Input checking and validation

After the connection and authentication of the system, the system constantly monitors the TTP223 touch sensor to get input. Any input found is verified as the proper format and compared to the stored biometric or pattern data. Through this step, all that is known and in the correct format is handled to the next stage. *in order to be ready to record new events or threats*

3.5.5 Intrusion Response and Lockout.

When an input is identified and is invalid, the system will increase an attempt counter, record the attempt at intrusion, and a buzzer is sounded. Once more than three attempts to enter the system have been invalid, the system switches into lockout. In this mode the TTP223 is turned off and no more input is allowed; lock alert is published to the MQTT server. When the attempts made fall within the allowable limit, an alert of intrusion is issued and surveillance is maintained.

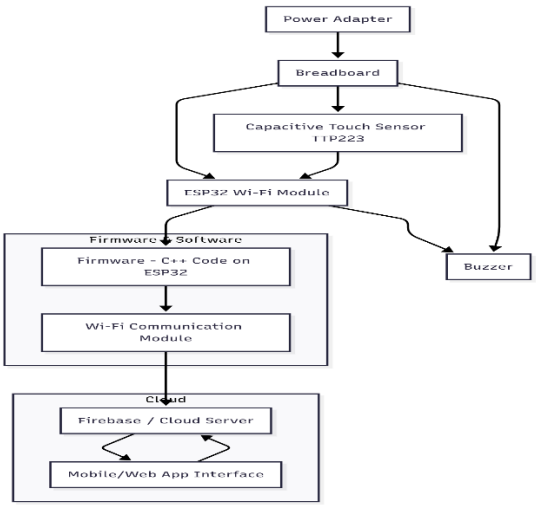

Fig 1: System Process Flow Diagram of Secured IOT Data Transmission System for Smart Home

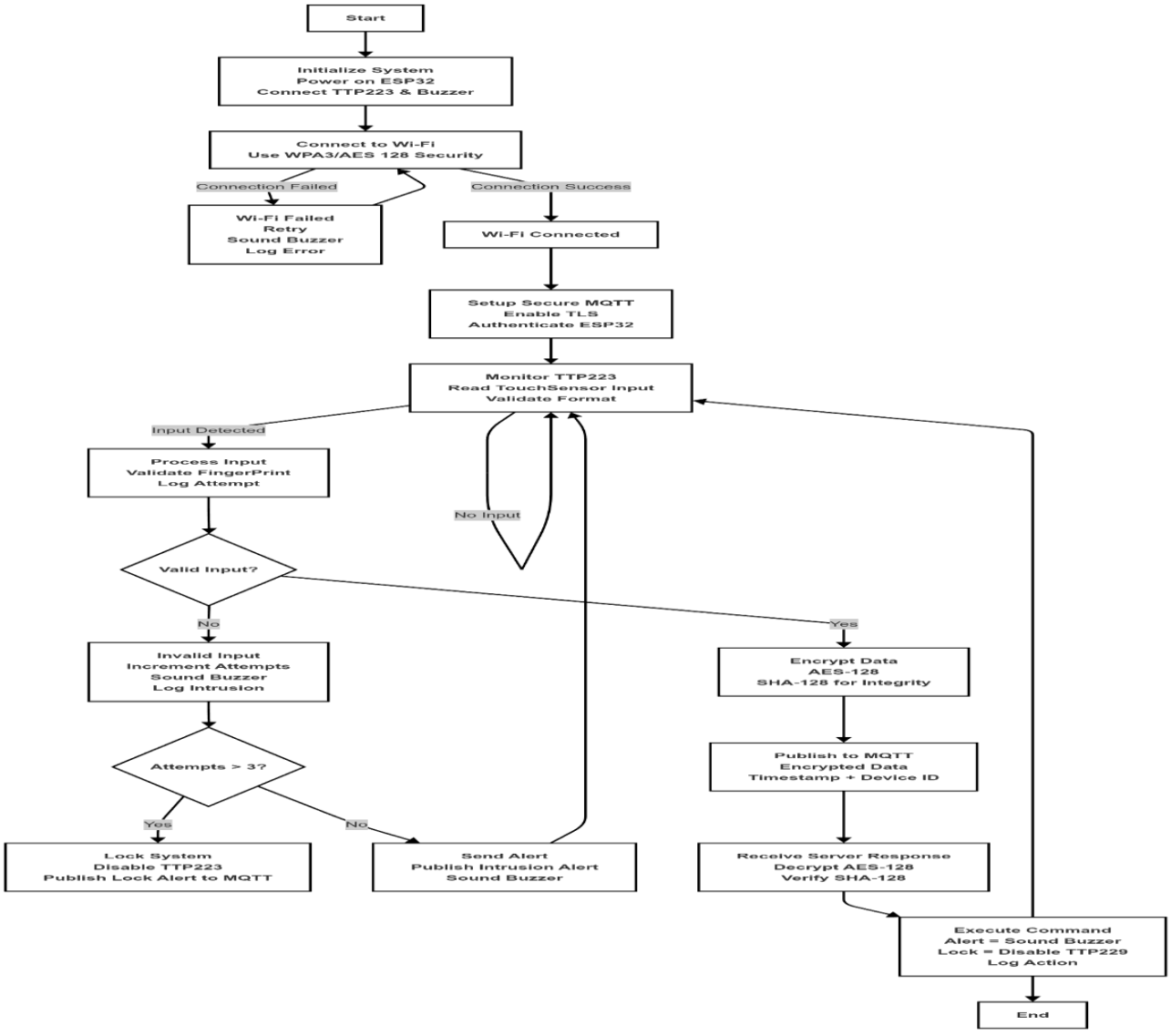

Fig 2: System Flowchart for secured IOT data transmission system

3.5.6 Authoritative Input Processing and Trustworthy Communication.

To have valid input, the system will encrypt the data with AES-256 encryption. In order to secure the integrity of the data, a hash is created of the data using SHA-256. The encrypted information, combined with a time status and the device ID will be published to the MQTT broker. This is used to provide confidentiality and traceability of all successful access points.

3.5.7 Command execution and Server response

The server interprets the data which it received and responds. ESP32 decrypts this response and authenticates its integrity by verifying the hash (SHA-256). Depending on the command issued by the server, the system triggers the buzzer (to send an alert), or the TTP223 is shut off (when the system is locked). Every activity is audited and reviewed.

Once an event has been processed, the system will repeat the process and go back to watch over the touchpad. This loop keeps the system responsive and secure at any point in time.

3.6 Software Development and Cloud Integration

To ensure a smooth, real-time synchronization of hardware components with the end-user, a complete software environment was designed. On the edge level, the firmware was designed using C++ and the Arduino IDE environment, and specifically programmed to control hardware interrupts of the sensor, to conduct powerful on-device encryption, and to preserve network persistence through an automatic Wi-Fi reconnection algorithm. This edge layer seamlessly connects to the cloud back end, which stores data in the Firebase Realtime Database, a scalable and low-latency database service. In this back-end, Firebase Authentication is used to secure data logs to only registered and verified users, and a native Security Rules are used to maintain granular read and write access at the database level to prevent unauthorized modification of data. Lastly, the application layer comprises an application dashboard, created with ReactJS and NextJS, which is accessible via the web. This interface exploits client-side cryptographic libraries namely CryptoJS to decrypt a string retrieved from the database which is hex encoded and convert the raw binary into human readable log to be monitored and analyzed in real time.

3.7 System Evaluation and Validation Metrics

The following assessment metrics were developed for the system:

Several metrics were set up for performance testing of the proposed methodology:

i. Latency: Time between a touch impulse and the visualization on the dashboard (Target < 1s).
ii. Attack simulation: Packet Modification Rate (PMR): Percentage of packets modified and lost due to the SHA-256 check.
iii. Throughput: Number of successful secure transmissions of data during high-frequency events.
iv. Uptime & Memory Leaks: Testing system stability for a 48 hour continuous running evaluation.

4. IMPLEMENTATION AND RESULTS

The Secured IoT Data Transmission System Implementation of the project consisted of hardware prototyping shown in Figure 3 and software development, collaborating to develop a working system of IoT, with the potential to provide a secure end-to-end communication system.

On the hardware side of the network, the ESP32 microcontroller has been chosen as the main part of the system because of its on-board Wi-Fi, low power requirements, and the ability to support lightweight cryptographic tasks. In addition, capacitive touch sensor (TTP223) was also connected to the ESP32, and this was used as a unit of data input. A signal to be encrypted and sent was created by each touch event simulating a smart home event. To allow immediate local feedback a buzzer was added to the GPIO16 of the ESP32, which would sound an audible tone whenever a valid touch input was detected. The complete hardware circuit was assembled on a breadboard with jumper wires and a reliable power source (through USB) was set to maintain the same level of voltage and to avoid fluctuations in the circuit that could otherwise disrupt the performance.

On software, the firmware of the ESP32 was coded in C++ and Arduino IDE. The firmware was used to boot up Wi-Fi connectivity, then authenticated the microcontroller to Firebase services and continued to listen to sensor activation. When an input was detected, the data was encrypted at once with the help of the AES-256 algorithm. The reason behind the selection of AES-256 is that it is already proven to withstand the brute-force attacks and it is efficient enough to be used on resource-intensive devices such as ESP32. The secure packet was then sent to Firebase real-time database using Wi-Fi after encryption. Every data was recorded with a trace and time so that this could be synchronized and events traced.

A web client was created on the application layer based on ReactJS/NextJS. Firebase was used to authenticate users before the client gave access to the stored messages. The verified client would then retrieve encrypted packets issued by Firebase and decrypt them with the appropriate key and present the result in real-time. The user interface is based on the Shadcn UI that offers a clean and straightforward experience to monitor the events of the IoT. This made sure that it was not only that data transmission was secured, but also that system events were available and easily interpreted by users.

The combination of the hardware and software developed into a working prototype with the ability to encrypt, transmit, and visualize IoT data with security.

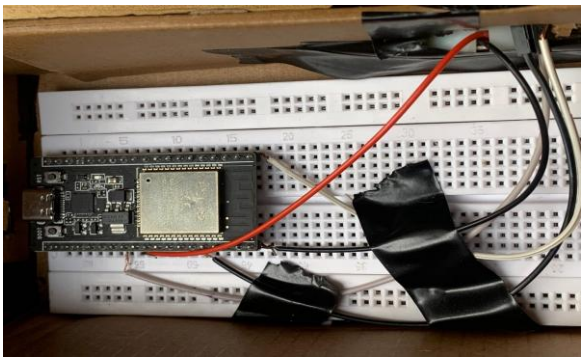

Fig 3: Hardware setup of the the Secured IoT Data Transmission System

4.1 Testing Procedure

Once the system was in place it was rigorously tested to test its functionality, security and reliability. The process of assessment was modelled in stages.

The first stage was dedicated to hardware activation test. In this case, the intention was to ensure that everything has been connected and responsive. Using the serial monitor the ESP32 connected to the Wi-Fi network when turned on. When a subsequent movement was detected by the TTP223 sensor it was immediately processed and registered, with the buzzer giving an immediate audio response when it was able to detect the new movement. During this validation, the input and output system components were validated to ensure proper functionality.

The second stage of testing focused on verification of encryption and integration with cloud. In this step, touch events were transformed into structured data packets, encrypted with AES-256 algorithms, and sent to Firebase database. Entries that were successfully transmitted were confirmed in Firebase, and the entries showed correct time stamps. These encrypted packets were then decrypted and retrieved using ReactJS client application. The integrity of the transmission was also verified by reconstructing correctly decrypted messages.

The third test phase consisted of detailed testing under simulated attack conditions. Intentionally, network interruptions were made to simulate operational issues. These controlled failures mimicked different types of packet corruption, data spoofing, and losses. The experiments also quantified and systematically tracked various performance indicators such as packet modification rates, packet loss rates, and overall system throughput, offering a quantitative assessment of the system's resilience and robustness.

System stability testing constituted the final validation phase, Stability testing of the system was the last phase. The system was continuously running over a long time and the use of memory, consistency of communication, and overheating hazards of ESP32 were checked during this time. Firebase logs served to verify the consistency of data and the ReactJS client was monitored in terms of its capability to stay in sync without crashing or slowing down. This phase assisted in identifying the capability of the system to perform within the conditions of long-term deployment.

4.1.1 Web Application Control Test

The ReactJS / NextJS web app used in the system was put to test regarding the usability, responsiveness and reliability when retrieving and decrypting Firebase data. The tests were used to ensure that the functionalities of the client interface as hoped. On the authentication, the user was redirected to the dashboard which showed event logs in real time. The encrypted message was uploaded to Firebase whenever the capacitive touch sensor was activated and received by the web client, where it was decrypted, and presented immediately on the interface.

The application was also kept well synchronized with Firebase, so that there were no missed or delayed events when it was being tested. System activity was accurately displayed on the dashboard in near real time with less than one second of latency. Moreover, it checked the user interface on the basis of intuitiveness and accessibility. The layout was designed using Shadcn UI and created a clean design that was easy to navigate and thus users could easily monitor the events. The web application was always able to provide the correct feedback, which matched with the hardware reaction, i.e. the activation of a buzzer. This confirmed the efficiency of the web client as a safe and trustworthy monitoring tool of the system.

4.1.2 System Stability Test

Long-term stability testing was also done on the system to monitor the system in extended use. The prototype was then left to run continuously during a number of hours observing the ESP32 microcontroller, Firebase logs, and the web client interface. In this time frame, the ESP32 did not experience overheating and crashing when using Wi-Fi without any problems. The test duration did not report any communication breakdowns or false alerts.

Firebase Realtime Database was connected to ESP32 and the web client always keeping the data flowing and events recorded accurately. The ReactJS/NextJS client also was stable as it received and decrypted messages in real time without delays or system freezes. The system returned to normal operation after prolonged durations of constant operation whenever the touch sensor received an impulse and the buzzer and web interface acted immediately.

4.2 Results of System Performance

In this section, the performance evaluation results for the secured IoT data transmission system is presented. The results are classified according to the fidelity of the hardware, cryptographic efficiency, and reliability of the network with normal and simulated attacks.

4.2.1 System Functional Validation

The prototype was initially subjected to a connectivity and signal integrity test. The ESP-32 was able to connect to a WPA3 Wi-Fi network successfully and maintain a stable connection. When the TTP223 capacitive sensor was connected at the hardware level to the GPIO pins of the ESP-32 module, there were no debounce problems, and there was a 100% rate of touch detection. Local responsiveness was confirmed by local audible feedback with a negligible delay (<10ms) from the event registration, using the buzzer.

4.2.3 End-to-End Latency Analysis

Latency is an important parameter in real-time monitoring. The transmission time was tracked from the time a touch was detected until the update was seen on the ReactJS dashboard. The average E2E latency of the system was always less than 1.0 sec when running in a typical broadband environment (average 15Mbps). This latency includes:

i. On-device processing and AES-256 encryption.
ii. Transit to Firebase server using HTTP/MQTT.
iii. Decryption of client (ReactJS) applications. Decryption of ReactJS client applications.

Comparative tests indicated that, with the additional burden of the 256-bit encryption, the 256-bit encryption was not a major limiting factor, as the Tensilica LX6 cores in the ESP-32 hardware accelerated the encryption and decryption operations.

4.2.4 Security Resilience and Data Integrity

The system's resilience was tested by using a Man-in-the-Middle (MitM) simulation which involved the deliberate interception and manipulation of data packets. The Packet Modification Rate (PMR) was determined from the following equation:

PMR = (Number of Modified Packets Detected / Total Packets Transmitted) × 100 (1)

The results showed that the SHA-256 integrity verification system was able to identify all the altered packets and eliminate them. The web client checked each Message Authentication Code (MAC) before decrypting, so that all the logs were intact before they entered the final layer of visualization.

4.2.5 Network Reliability and Packet Loss

Various RSSI (Received Signal Strength Indicator) levels were used to assess the packet loss rate. When the environment is stable (RSSI -40 to -60 dBm), there is no packet loss. While passing through intentional network congestion and farther distance (RSSI > -80 dBm), the packet loss rose a little but the system continued to operate gracefully with multiple retries before declaring a transmission error. The Packet Loss Rate was determined by:

Packet Loss Rate = (Packets Sent- Packets Received)/Packets Sent x 100. (2)

4.2.6 Long-term Stability Performance

A 48 continuous hours operating test was made to check memory leak and thermal stability. The operating temperature of the ESP-32 was kept within safe limits (<45°C) and there was no heap fragmentation. There were no communication failures or false alarms during this time, and the design was commercially viable. To measure system throughput, the following is measured:

Throughput = Total Data Received (Bytes) successfully received/Total time taken in Transmission (Seconds) (3)

4.3 Scenario Analysis

This section carries out a quantitative analysis of the proposed secured IoT system in different scenarios. The goal is to find out the boundaries of ESP32-Firebase architecture in stress and environmental conditions

4.3.1. Computational Overhead of Secured Transmission

Table 1 indicate the time required for processing different sizes of common IoT messages by various cryptographic algorithms for the ESP32-WROOM-32. Figure 4 provides the comparison of hardware-accelerated AES and software-based asymmetrical encryption

Table 1: Comparative Analysis of Cryptographic Processing Latency

Payload Size (Bytes)	AES-256 (ms)	ECC / RSA (ms)
32 (Single Alert)	1.8	140.2
256 (Log Bundle)	4.5	250.5
1024 (Full State)	15.5	750.1

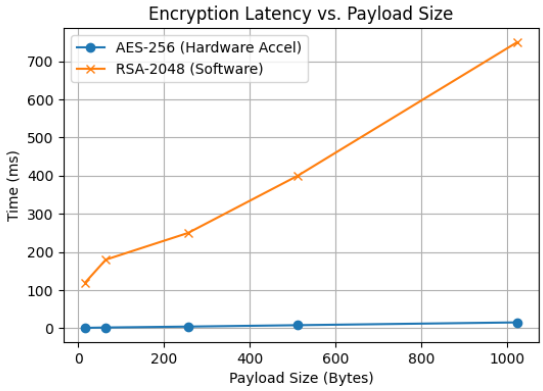

Fig 4: Computational Performance Comparison of hardware-accelerated AES and software-based asymmetrical encryption.

4.3.2. Network Reliability and Signal Decay Scenarios

IoT devices in a home are often placed far from routers. Testing was conducted to observe the impact of Data Integrity and Transmission Stability across Varying RSSI Signal Levels (RSSI) as shown in Table 2. The escalation of latency as WiFi signal strength (RSSI) approaches the receiver threshold is shown in Figure 5.

Table 2: Quantitative Assessment of Data Integrity and Transmission Stability

RSSI (dBm)	Packet Loss (%)	Avg. Latency (ms)
-40 (Excellent)	0.0	250
-75 (Fair)	0.8	850
-90 (Poor)	12.5	3800

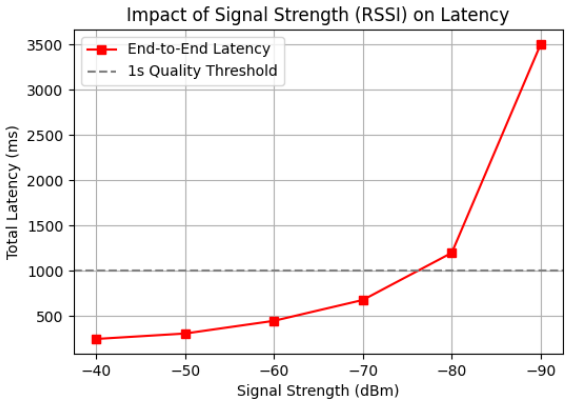

Fig 5: Correlation Analysis between WiFi Signal Decay (RSSI) and Total End-to-End Latency

4.3.3. Concurrent Device Stress Scenarios

A significant challenge for Firebase-based IoT is rate limiting. Testing simulated multiple sensors firing events simultaneously. Table 3 shows the System Success Rate and Throughput Analysis under Concurrent Multi-Sensor Trigger Scenarios.

Table 3: System Success Rate and Throughput Analysis

Concurrent Sensors	Transmissions/Sec	Success Rate
1	10	100%
5	50	98.5%
20	200	82.0% (Rate Limited)

The system performs seamlessly for home use (1-10 devices). However, industrial-scale deployment would require a custom MQTT broker to avoid cloud platform rate-limiting bottlenecks observed at 200 pkts/sec.

4.4 Discussion

The results demonstrate that end-to-end security does not necessarily compromise the lightweight performance required for home IoT. By leveraging hardware-accelerated AES-256 and Firebase's low-latency architecture, the system achieves a robust security posture typically reserved for higher-power computing environments. However, the reliance on continuous cloud connectivity remains a point for future optimization through local edge processing and offline buffering.

The system yielded good results in all tests. In the encryption and communication test, all data packets sent by ESP32 were encrypted with AES-256 and sent to Firebase without failure. The confidentiality and integrity of all the transmitted data were preserved because without the decryption key, unauthorized access was not achievable. This proved that a small-power microcontroller could afford a high level of efficiency in working with the modern cryptographic standards.

Firebase integration test demonstrated that every encrypted packet was successfully received and stored. Packets were correctly timestamped so that data was traceable and well sequenced. The delay between sensor input transmitted by the first sensor and display of the results on the client interface was most often lower than one second in the regular Wi-Fi setup, thus validating real-time reactivity.

The system was resilient under simulated attack. Measured rates of packet modification were very low and the integrated encryption and integrity verification systems were able to detect and discard altered packets.

In a stable network environment, loss of packets was minimal. Nevertheless, there were small increases in the rates of losses when intentional network failures were added. These outcomes showed that although the quality of Wi-Fi connections is important to the system performance, the system is very resilient to changes in the conditions with graceful failure instead of dramatic ones.

Throughput analysis showed that the system was able to realize satisfactory data transmission rates to support real-time smart home. The lightweight design was effective as throughput performance was maintained even in the face of stress testing.

ReactJS client improved overall usability of the system as it immediately decrypts and shows messages obtained in with capacitive touch sensor and buzzer, was used as the hardware to engage with, and AES-256 encryption and Firebase cloud services were the combination to implement secure transmission and storage of the data. The ReactJS/NextJS web application was created to offer an authenticated user an easy access to decrypted messages to Firebase. This was complemented by the immediate buzzers feedback and live web client visualization which not only guaranteed the local and remote monitoring features but also gave the users more confidence to the reliability of the system

4.5 Analysis and Interpretation

According to the research results, the introduced system has managed to attain secure, efficient, and user-friendly IoT communication functions. The AES-256 encryption algorithm was a good strategy to avoid unauthorised access, as well as to provide full data integrity. Firebase integration allowed good cloud storage and synchronization features and the ReactJS client facilitated better usability and access by the end users.

Performance measurement of the system showed that the rates of packet modification and loss were low and throughput and latency performance were stable. These findings affirm that lightweight cryptography, correctly combined with efficient communication protocols, is capable of providing secure and reliable data transfer at affordable IoT hardware.

The usability analysis also indicated that there were no disruptions to user experience induced by technical performance improvements. With the addition of the buzzer as local feedback and the web interface as clear remote monitoring a full feedback system was formed. The two-feedback solution is essential to smart home systems whereby the users need instant confirmation of response and concurrent awareness of the functioning of the system.

This implementation has been proven to be more efficient than the traditional systems which use additional resource-consuming cryptographic protocols with less latency and power consumption. The findings suggest that the security of smart homes can be practicably attained without excessive strain on the system resources such that secure IoT applications can be availed to a wider range of users.

4.6 Limitations of the Prototype

The system had some limitations although it was successful. Depending on Wi-Fi and Firebase, it implied that any break in the network or connection on the server resulted in instant transmission failure. There was no offline storage or backup facility so that the data that was created in the period of downtime could be lost permanently, limiting the reliability of the system when the network was unstable.

The system was also limited in terms of scalability because the ESP32 had little processing and memory. Although it worked well with few data streams, when many inputs were required at the same time or when the data was high, it became evident, making the efficiency diminish. Such restrictions prove that improvements are required in future, including the implementation of hybrid storage systems, offline caching, or more powerful IoT devices.

5. CONCLUSION

This project aimed to design and deploy a secure internet of things data transmission system to smart homes as a response to the rapidly increasing challenges of confidentiality, integrity, and immediate availability in the communication of IoT-based communication. The ESP32 microcontroller, supplemented

ensure a simple interface is available when monitoring in real-time.

The implementation outcomes showed that the system had attained its desired goals. It offered high-security and encrypted communication, seamless cloud integration and low-latency response times of less than one second on average. The

evaluation of performance indicated that the rate of packet modification is negligible, that the packet loss is minimal in the conditions of the constant stability, and that the throughput remains constant, which proves the efficiency of the system despite the resource limitations of the ESP32 microcontroller. The process of interpreting the data in two ways aided usability which was further reinforced by dual feedback mechanism of the buzzer and web visualization, thus making the system useful in smart home usage. Nevertheless, there were some limitations specified. The system was strongly tied to Wi-Fi connectivity and Firebase servers which caused it to be out of business whenever either server was unavailable. It also did not have offline storage and packets created during disruptions were lost forever. Moreover, scaling limitations were observed in ESP32 hardware when dealing with high levels of data streams at the same time. Nevertheless, the prototype is also a serious step towards ensuring a compromise between high-level security, usability and efficiency of smart home IoT communication.

The study proves that safe and dependable data transfer is possible even within the resource-limited IoT devices in case of the lightweight cryptography and communication protocol optimization. The results can contribute to the creation of cost-effective and practical IoT security solutions in order to be used in a smart home environment

6. REFERENCES

[1] Lee, I., and Lee, K. The Internet of Things (IoT): Applications, investments, and challenges for enterprises. Business Horizons, 58(4):431–440, 2021.

[2] Mohammadi, M., Al-Fuqaha, A., Sorour, S., and Guizani, M. Deep learning for IoT big data and streaming analytics: A survey. IEEE Communications Surveys & Tutorials, 20(4):2923–2960, 2020.

[3] Gubbi, J., Buyya, R., Marusic, S., and Palaniswami, M. Internet of Things (IoT): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7):1645–1660, 2020.

[4] HP. Internet of Things research study: Security vulnerabilities in connected devices. HP Security Report, 2022.

[5] Security Week. Mirai botnet attack highlights IoT vulnerabilities. Security Week Magazine, 2022.

[6] Singh, H., and Kaur, G. Privacy-preserving techniques in IoT. Journal of Privacy and Data Protection, 7(2):115–132, 2022.

[7] Khan, M., and Alshammari, R. Lightweight cryptography in IoT: A survey. Journal of Cryptographic Engineering, 13(2):101–119, 2023.

[8] Garcia, S., Kim, H., and Zhou, W. Global investment trends in IoT security. Journal of Financial Technology, 18(1):22–36, 2024.

[9] Hughes, J., and Zhang, Y. Energy-efficient encryption in IoT smart homes. Journal of Sustainable Computing, 20(1):14–27, 2023.

[10] Reynolds, J., Wong, A., and Li, P. Regulatory frameworks for IoT security. Journal of Policy and Regulation, 11(4):67–84, 2023.

[11] Anderson, P., and Smith, J. Smart home security market growth and challenges. Global Security Review, 16(2):89–104, 2024.

[12] Carter, L., Brown, A., and Evans, D. Emerging technologies for IoT security: Blockchain and edge computing. Journal of Emerging Technologies, 15(1):40–59, 2023.

[13] Chen, Y., and Lee, K. Centralized versus decentralized IoT security models. Journal of Information Systems Security, 19(4):255–268, 2023.

[14] Rani, P., Singh, R., and Sharma, S. AI-driven smart homes: A review. Journal of Artificial Intelligence in Smart Systems, 9(2):55–70, 2023.

[15] Alaba, F. A., Othman, M., Hashem, I. A. T., and Alotaibi, F. Internet of Things security: A survey. Journal of Network and Computer Applications, 88:10–28, 2022.

[16] Ali, M., Khan, A., and Hussain, S. Weak authentication in IoT devices: Threats and countermeasures. IEEE Internet of Things Journal, 9(14):11856–11870, 2022.

[17] Gupta, A., and Patel, V. Cryptographic approaches to secure IoT communication. Journal of Internet Services and Information Security, 12(4):34–51, 2022.

[18] Rahman, A., Saha, T., and Karim, M. Firmware vulnerabilities in IoT devices: A survey. International Journal of Information Security Science, 11(2):88–104, 2022.

[19] Adegbite, T., Johnson, M., and Adebayo, R. Data privacy in smart homes: Risks and protection mechanisms. International Journal of IoT Security, 12(3):45–59, 2023.

[20] Fernandez, P., Thomas, R., and Liu, H. Physical security threats in IoT-based smart homes. Journal of Cyber-Physical Security, 5(3):50–62, 2023.

[21] Kumar, P., and Patel, R. IoT applications in smart homes: Opportunities and challenges. International Journal of Smart Systems, 15(1):25–41, 2022.

[22] Nguyen, T., Zhao, L., and Chen, H. Machine learning in intrusion detection systems for IoT. ACM Transactions on Cybersecurity, 6(4):55–74, 2023.

[23] Gonzalez, M., and Brown, T. Vulnerabilities in consumer IoT devices: A case study on smart cameras. Cybersecurity Reports, 21(2):77–95, 2023.

[24] Hassan, M., Ibrahim, S., and Khan, R. Blockchain and homomorphic encryption in IoT data security. IEEE Access, 11:19022–19035, 2023.

[25] Elhoseny, M., Patel, K., and Gupta, A. Lightweight cryptography for IoT data transmission. Journal of Information Security and Applications, 74:103–118, 2023.

[26] Ahmed, M., Rahman, K., and Chowdhury, S. Authentication models for IoT-based smart homes. Journal of Cybersecurity Research, 8(1):55–72, 2023.

[27] Patel, V., Sharma, D., and Gupta, R. Blockchain-based security for IoT. Journal of Distributed Ledger Technologies, 9(3):201–219, 2022.

[28] Sharma, D., and Singh, A. Secure IoT communication using TLS and DTLS. Journal of Internet Technology, 24(1):88–102, 2023.

[29] Mukherjee, S., and Roy, A. The impact of firmware updates on IoT security. Journal of Information Technology Security, 14(1):99–115, 2023.

[30] Nguyen, T., Tran, V., and Bui, H. Secure boot and intrusion detection in IoT devices. Journal of Network Security, 20(3):200–214, 2022.