Al-Quds Open University

Faculty of Graduate Studies and Scientific Research

Master of Information Technology

Securing Communication Standards of IoT-based Smart Irrigation Systems in Palestine

تأمين معايير الاتّصالات لأنظمة الرّي الذّكيّة المعتمدة على إنترنت الأشياء في فلسطين

THESIS

by

Samar Younis AlShwieki

Student ID: 0330012310213

Supervisor

Dr. Eng. Samer Jaloudi

Submitted in Partial Fulfillment of the Requirements
For the Master of Information Technology Degree at the Faculty of Graduate Studies

Ramallah, Palestine

November, 2025

جامعة القدس المفتوحة

عمادة الدراسات العليا والبحث العلمي

ماجستير تكنولوجيا المعلومات

تأمين معايير الاتّصالات لأنظمة الرّي الذّكيّة المعتمدة على إنترنت الأشياء في فلسطين

رسالة ماجستير

إعداد الطالبة

سمر يونس الشويكي

الرقم الجامعي :0330012310213

إشراف

د. م. سامر جالودي

قدمت هذه الرسالة استكمالا لمتطلبات الحصول على درجة الماجستير لتكنولوجيا المعلومات

في عمادة الدراسات العليا والبحث العلمي

رام الله، فلسطين

تشرين الثاني ، 2025

Examination Committee Page

The committee for Samar Younis AlShwieki certifies that this is the approved version of the following thesis and is acceptable in quality and form for publication in paper and in digital formats:

Securing Communication Standards of IoT-based Smart Irrigation Systems in Palestine

Committee Members:

Committee Supervisor: Dr.Eng. Samer Husni Jaloudi – Al- Quds Open University
Signature: [illegible]
Date: 01/02/2026

Committee First Member: Dr. Mamoun Abu Helou – Al- Istiqlal University
Signature: 1/2/2026 [illegible]
Date: 01/02/2026

Committee Second Member: Dr. Eng. Mohammed I. Y. Ikhlayel– Al- Quds Open University
Signature: 31.01.2026 [illegible]
Date: 01/02/2026

Al-Quds Open University

2025

Declaration

I, Samar Younis Alshwieki, hereby declare that the work presented in this thesis has not been submitted for any other degree or professional qualification, and that it is the result of my own independent work.

Signed: [illegible]

Date:05/01/2026

Abstract

Securing Communication Standards of IoT-based Smart Irrigation Systems in Palestine

The agricultural sector is one of the most vital sectors in Palestine, where water resource management faces increasing challenges due to water scarcity, limited infrastructure, and the continued use of inefficient traditional irrigation methods. In recent years, Internet of Things-based smart irrigation systems have emerged as an effective solution to improve water use efficiency. However, these systems are exposed to growing security threats, particularly at the communication level, especially in environments that rely on low-power wireless networks.

This thesis aims to secure communication standards for IoT-based smart irrigation systems in Palestine by designing and implementing a secure and flexible communication framework that considers the specific characteristics of the Palestinian environment, including the geographical distribution of agricultural lands, variable network coverage, and limited energy resources. A practical smart irrigation system was designed and implemented based on LoRa technology, integrating multiple security mechanisms such as data encryption, secure key management using hardware-based secure elements, and built-in LoRa protection features, such as message headers, to mitigate cyber-attacks, including replay attacks.

In addition, an alternative system based on the MQTT protocol was designed to operate when direct Internet connectivity is available, enhance system flexibility, and compare different communication standards in terms of performance and security. When deployed over the TLS protocol, MQTT provides encrypted communication, authentication, and data integrity with low latency, making it suitable for connecting smart irrigation systems to cloud servers and data analytics platforms.

The results indicate that combining long-range low-power communication technologies such as LoRa with Internet-based protocols like MQTT improves the reliability and security of smart irrigation systems while maintaining efficient energy consumption. This hybrid approach enhances water resource management and supports data-driven decision-making in Palestine and agricultural sustainability.

Keywords: Internet of Things, smart irrigation systems, LoRa, MQTT, communications security, Palestine.

الملخص

تأمين معايير الاتصالات لأنظمة الري الذكية المعتمدة على إنترنت الأشياء في فلسطين

يُعدّ قطاع الزراعة من القطاعات الحيوية في فلسطين، حيث تواجه إدارة الموارد المائية تحديات متزايدة نتيجة محدودية المياه والاعتماد على أساليب ري تقليدية غير فعّالة. وقد أدى التوسع في استخدام أنظمة الري الذكية المعتمدة على إنترنت الأشياء إلى تحسين كفاءة استخدام المياه، إلا أن هذه الأنظمة تواجه مخاطر أمنية متزايدة، خصوصًا على مستوى معايير الاتصالات في البيئات التي تعتمد على شبكات لاسلكية منخفضة الاستهلاك للطاقة وبنى تحتية محدودة.

تهدف هذه الرسالة إلى تأمين معايير الاتصالات لأنظمة الري الذكية المعتمدة على إنترنت الأشياء في فلسطين، من خلال تصميم وتنفيذ نظام اتصالات آمن ومرن يراعي خصوصية البيئة الفلسطينية من حيث التوزع الجغرافي للأراضي الزراعية، وتفاوت جودة التغطية الشبكية، ومحدودية مصادر الطاقة. وقد تم في هذا العمل تصميم وتنفيذ مشروع عملي لنظام ري ذكي يعتمد على تقنية LoRa، مع دمج آليات أمنية متعددة شملت تشفير البيانات، وإدارة المفاتيح داخل وحدات آمنة (SE/HSM)، والاستفادة من آليات الحماية المدمجة في تقنية LoRa مثل مقدمة الرسالة، للحد من هجمات إعادة الإرسال وغيرها.

كما تم تصميم نظام بديل يعتمد على بروتوكول MQTT ليعمل عند توفر اتصال مباشر بالإنترنت بهدف دراسة مرونة النظام ومقارنة فعالية معايير الاتصالات المختلفة من حيث الأداء والأمن. وقد أتاح استخدام MQTT فوق بروتوكول TLS مستوى مناسبًا من الأمان مع زمن استجابة منخفض، مما يجعله خيارًا ملائمًا لربط أنظمة الري الذكية بالخوادم السحابية.

أظهرت نتائج الدراسة أن استخدام تقنيات الاتصال بعيدة المدى ومنخفضة الاستهلاك للطاقة مثل LoRa، وبروتوكولات الإنترنت مثل MQTT، يسهم في تحسين موثوقية وأمن أنظمة الري الذكية، ويعزز كفاءة إدارة الموارد المائية، بما يدعم استدامة الزراعة الذكية في فلسطين.

الكلمات المفتاحية: إنترنت الأشياء، أنظمة الري الذكية، LoRa، MQTT، أمن الاتصالات، فلسطين.

Associated Publications

Acknowledgements

In the name of God, the Most Gracious, the Most Merciful

Praise be to God who enabled me to complete this work and overcome the difficulties I encountered.

Peace and blessings be upon the Messenger of God (Muhammad) and all of his prophets and messengers who taught us, through their patience and steadfastness in calling to Islam, how to be persistent and patient in achieving goals and objectives.

Thanks, and appreciation to my family and friends who provided me with support and assistance.

Many special thanks to my colleague (**Mohammed Zaatara**) for helping me complete this work.

Samar, Al-Shweiki

November 2025

Dedication

To the teacher of humanity, Muhammad, may the best prayers and peace be upon him.

To those who have left us, and their good impact still accompanies me and illuminates my path.

To those who taught me that success is not given away, but rather taken away by effort.

To my teachers who were my torches of guidance.

To my friends who shared with me the trouble of research and moments of achievement.

To those around me who still provide me with strength and support.

To my ambitious soul that did not know the impossible.

To every dream I endured until it became a reality.

I dedicate this message to you, as it is not the end of the road, but the beginning of a bigger dream.

Samar Younis AlShwieki
November 2025

Table of Contents

Examination Committee Page iii
Declaration iv
Abstract v
الملخص vi
Associated Publications vii
Acknowledgements viii
Dedication ix
Table of Contents x
List of Figures xvii
List of Tables xix
List of Abbreviations xxi
Chapter 1: Introduction 1
1.1 Overview and Background 1
1.2 Motivation 6
1.3 Problem Statement 7
1.4 Research Objectives 7
1.5 Thesis Contribution to the Field 8
1.6 Thesis Outline 8
1.7 Conclusion 9
Chapter 2: Literature Review 10
2.1 Introduction 10
2.2 Previous Studies 10
2.2.1 Agriculture 10
2.2.1.1 Smart Irrigation Systems 10
2.2.1.2 Agriculture in Palestine 11
2.2.1.3 The Impact of Climate Change on Agriculture 12
2.2.2 Security of IoT Communication 13
2.2.2.1 Security and Privacy 13
2.2.2.2 Kyber 15
2.2.2.3 LoRa / LoRaWAN 16
2.2.2.4 IoT Protocols 17
2.2.2.5 IoT Standards 18

2.3 Literature Review Summary Table 18
2.4 Conclusion to This Chapter 20
Chapter 3: IoT Protocols and Data Security Mechanisms 21
3.1 Introduction 21
3.2 Protocols Proposed for Agricultural IoT: 21
3.2.1 Low Power Wireless Networks Protocols 21
3.2.1.1 LoRaWAN 21
3.2.1.1.1 Overview 21
3.2.1.1.2 Configurations 22
3.2.1.1.3 LoRa / LoRaWAN 22
3.2.2 IP-based Networks IoT Protocols 23
3.2.2.1 MQTT 23
3.2.2.2 CoAP 24
3.2.2.3 AMQP 25
3.2.2.4 DDS 26
3.2.2.5 OPC UA 26
3.2.3 Comparisons : 28
3.2.3.1 Comparison between IP-based Networks IoT Protocols: 28
3.2.3.2 Comparison between MQTT, CoAP, and OPC UA Performance 30
3.2.4 Security Protocols and Encryption Technologies 32
3.3 Securing Methods: 33
3.3.1 Attacks faced by the Agricultural Internet of Things : 33
3.3.1.1 Denial-of-Service (DoS) Attacks 33
3.3.1.2 MITM 34
3.3.1.3 Firmware Exploits 34
3.3.1.4 Protocol Attacks 35
3.3.1.5 Data Attacks 35
3.3.2 Securing Techniques Proposed for Agricultural IoT: 37
3.3.2.1 Authentication: 37
3.3.2.2 Cryptographic Algorithms 38
3.3.2.3 Blockchain Technology 38
3.3.2.4 Secure Communication Protocols 39
3.4 Cryptographic Algorithms 40
3.4.1 Classic Encryption Algorithms : 40

3.4.1.1 Symmetric Encryption Algorithms 40
3.4.1.2 Asymmetric Encryption Algorithms 42
3.4.1.3 Hybrid Encryption Algorithms 44
3.4.1.4 Comparison Between the Classical Encryption Algorithms 46
3.4.2 Post- quantum cryptography (considered as extended to classical) : 47
3.4.2.1 Comparison: 48
3.4.2.2 Kyber (Key Encapsulation Mechanism – KEM) 49
3.4.2.2.1 Overview 49
3.4.2.2.2 Kyber Standard Characteristics 50
3.4.2.2.3 Kyber Encapsulation/Decapsulation Mechanism 51
3.4.2.2.4 Random Function 52
3.4.2.3 Comparison 53
3.5 Conclusion 54
Chapter 4: System Design 55
4.1 Introduction 55
4.2 System Basic Information 55
4.2.1 Fields Used in Different Forms: 55
4.2.2 Variable Size 56
4.2.3 Performance Metrics 56
4.3 MQTT- Based Systems 58
4.3.1 Model 1: General Broker without Authentication 58
4.3.2 Model 2: General Broker with Authentication 58
4.3.3 Model 3: General TLS - MQTT Broker with Authentication 59
4.3.4 Models 4: General TLS - MQTT Broker via Cloud 59
4.3.5 Proposed MQTT-Based System 60
4.3.5.1 System Architecture 61
4.3.5.2 System Workflow: 62
4.3.5.3 System Stages 63
4.3.5.3.1 Stage1: Collecting Data from Sensors 63
4.3.5.3.2 Stage2: Sending Data to Cloud and Distributing It via MQTT 63
4.3.5.3.3 Stage3: Analyzing Data & Making Decision 65
4.4 LoRa - Based Systems 66
4.4.1 Model 1: (P2P) Transmission without Authentication 66
4.4.2 Model 2: (P2P) Transmission with Authentication 66

4.4.3 Model 3: Point-to-Multipoint Transmission 66
4.4.4 Proposed LoRa-Based System 67
4.4.4.1 System architecture 68
4.4.4.2 LoRa Configurations 69
4.4.4.2.1 LoRa Standards 69
4.4.4.2.2 LoRa Pins 70
4.4.4.3 System Workflow: 71
4.4.4.3.1 Sending Pk 72
4.4.4.3.2 Receiving PK 72
4.4.4.3.3 Sending Announce 72
4.4.4.3.4 Sending Encrypted Moisture 73
4.4.4.3.5 Verifying Data and Decision 73
4.5 Conclusion to This Chapter 73
Chapter 5: Results and Discussion 74
5.1 Introduction 74
5.2 MQTT- Based Systems 74
5.2.1 System Variables Size 74
5.2.2 System Performance Metrics Results (Data) 75
5.2.3 Performance Metrics Results (Time) 76
5.2.4 Performance Metrics Results (CPU Usage) 77
5.2.5 Performance Metrics Results (Energy Consumption) 78
5.2.6 Comparisons 79
5.2.6.1 Other models (Variable Size) 79
5.2.6.2 Other models (Performance Metrics) 81
5.2.6.3 Comparison with Another Paper's Results (CPU Usage, Response Time) 83
5.2.6.4 Comparison with Another Paper's Results (Time) 84
5.3 LoRa - Based Systems 85
5.3.1 System Variables Size 85
5.3.2 System Performance Metrics Results (Data) 86
5.3.3 Performance Metrics Results (Time) 87
5.3.4 Performance Metrics Results (CPU Usage) 88
5.3.5 Performance Metrics Results (Energy Consumption) 89
5.3.6 Comparisons 91

5.3.6.1 Comparison with other models Variable Size 91
5.3.6.2 Other models (Performance Metrics) 92
5.3.6.3 Comparison with Another Papers' Results (Tran. Rate, Res. Time): 93
5.3.6.4 Comparison with Another Papers' Results Using LoRa (Metrics) 94
5.3.6.5 Comparison with Kyber Handshake Time in Lora Systems 95
5.4 Final Metrics Analysis 96
5.5 Conclusion to This Chapter 97
Chapter 6: System Security Analysis 98
6.1 Introduction 98
6.2 MQTT-Based System 98
6.2.1 MQTT Based-System Security Analysis: 98
6.2.1.1 Transmitting / Receiving via TLS-Cloud 99
6.2.1.2 The Kyber handshake 99
6.2.1.3 AES-GCM Encryption 100
6.2.1.4 HMAC - Verify the Authenticity of the Content 100
6.2.1.5 Verifications 101
6.2.1.5.1 Verify Message Length: Check message length 101
6.2.1.5.2 Verify HMAC: Applied Integration and Source Verification 102
6.2.1.5.3 Verify Counter: Prevent Retransmission (Replay Protection) 102
6.2.1.5.4 Verify AES-GCM & Tag 102
6.2.1.6 Decrypt: Data recovery after verifications 103
6.2.1.7 Deliver Data to Application: Achieving system goals in the application 103
6.2.2 MQTT Based-System vs Attacks 104
6.2.2.1 TLS-Cloud: 104
6.2.2.2 Kyber Handshake (CRYSTALS-Kyber / ML-KEM) 105
6.2.2.3 AES\GCM Encryption: 106
6.2.2.4 Message-Length Verification: 106
6.2.2.5 HMAC (Account and Verification): 107
6.2.2.6 Counter Verification 108
6.2.2.7 Tag Verification in AES-GCM 108
6.3 LoRa -Based System 109
6.3.1 Security Analysis: 109
6.3.1.1 Secure Key Management 109

6.3.1.1.1 Separate Key (pk, sk) 110
6.3.1.1.2 Special Key to Node 110
6.3.1.1.3 Protect Keys In SE/HSM 110
6.3.1.2 Trigger Gateway 111
6.3.1.3 Kyber Handshake (explained before) 111
6.3.1.4 AES/GCM Encryption (explained before) 111
6.3.1.5 Verification (explained before) 111
6.3.1.6 Decrypt Message (explained before) 111
6.3.1.7 Deliver Date to Application (explained before) 111
6.3.2 System vs Attacks 112
6.3.2.1 Secure Key Management 112
6.3.2.1.1 Key Mining Attacks: 112
6.3.2.1.2 Key Reuse Attacks: 112
6.3.2.1.3 Impersonation Attacks: 112
6.3.2.2 Trigger Gateway 113
6.3.2.2.1 Unauthorized Command Injection 113
6.3.2.2.2 Unauthorized Remote Control / Remote Takeover 113
6.3.2.1 Kyber Handshake (explained before) 113
6.3.2.2 AES/GCM Encryption (explained before) 113
6.3.2.3 Verification (explained before) 113
6.3.2.4 Decrypt Message/ hmac Foregery (explained before) 113
6.3.2.5 Deliver Data to Application (explained before) 113
6.4 Security Test Results 114
6.4.2 Replay Attack 114
6.4.2.1 Capture a Message Sent from the Broker 114
6.4.2.2 System Verification for First Sending 114
6.4.2.3 Resending Message 115
6.4.2.4 System Verification after Resending 115
6.4.3 Tampering Attack (Test1) 116
6.4.3.1 Capture a Message Sent from the Broker 116
6.4.3.2 System Verification for First Sending 116
6.4.3.3 Resending Message 117
6.4.3.4 System Verification after Resending 117
6.4.4 Tampering Attack (Test2) 118

6.4.4.1 Capture a Message Sent from the Broker 118
6.4.4.2 System Verification for First sending 118
6.4.4.3 Resending Message 118
6.4.4.4 System Verification after Resending 119
6.5 Full Security Analysis 120
6.6 Conclusion 123
Chapter 7: Conclusion and Future Work 124
References 126
Appendix A: System Parts 135
Appendix B: MQTT-Based Systems Methods Results 137
Appendix C: LoRa-Based Systems Methods Results 141
Author’s Biographical Sketch 144

List of Figures

Figure 1.1: Earth Surface Parts 1
Figure 1.2: IoT Communication Standards 4
Figure 3.1: MQTT Architecture 24
Figure 3.2: Agricultural IoT Protocols 27
Figure 3.3: Agricultural IoT Attacks 36
Figure 3.4: IoT Authentication Tools 37
Figure 3.5: Key Encapsulation / Decapsulation Mechanism 51
Figure 4.1: General Broker with Authentication 58
Figure 4.2: MQTT-Based System Architecture 61
Figure 4.3: MQTT-Based System Workflow 62
Figure 4.4: System Decision with LDR Value > 500 63
Figure 4.5: System Decision with LDR Value < 500 63
Figure 4.6: Check Connection 63
Figure 4.7: Sending PK 64
Figure 4.8: Receiving PK 64
Figure 4.9: Receiving CT 65
Figure 4.10: Decrypting Moisture 65
Figure 4.11: LoRa-Based System Architecture 68
Figure 4.12: LoRa Standards 69
Figure 4.13: LoRa Pins 70
Figure 4.14: LoRa-Based System Workflow 71
Figure 4.15: Sending PK 72
Figure 4.16: Receiving PK 72
Figure 4.17: Sending Announce 72
Figure 4.18: Sending Encrypted Moisture 73
Figure 4.19: Verifying & Decision 73
Figure 5.1: Performance Metrics Results (Time) 76
Figure 5.2: Performance Metrics Results (CPU Usage) 77
Figure 5.3: Performance Metrics Results (Energy Consumption) 78
Figure 5.4: Comparison with other models in Variable Size 79
Figure 5.5: Metrics Comparison (CPU Usage, Response Time) 83
Figure 5.6: Time Comparison (Key Generation, Encryption, Decryption) 84

Figure 5.7: System Metrics (Time) 87
Figure 5.8: System Metrics (CPU Usage Percent) 88
Figure 5.9: System Metrics (Energy Consumption) 89
Figure 5.10: Variable Size Comparison 91
Figure 5.11: Comparison with Another Papers' Results in Lora Systems (Tran. Rate, Res. Time) 93
Figure 5.12: Comparison with Another Papers Results in Lora Systems 94
Figure 5.13: Handshake Time Comparison 95
Figure 6.1: Secure MQTT connection with Kyber 98
Figure 6.2: Cloud Authentication 99
Figure 6.3: HMAC Fields 100
Figure 6.4: System Verifications 101
Figure 6.5: MQTT Based-System Vs. Attacks 104
Figure 6.6: Secure LoRa System 109
Figure 6.7: LoRa Based-System vs Attacks 112
Figure 6.8: Capture a Message 114
Figure 6.9: System Verifications for First Sending 114
Figure 6.10: Resending Message 115
Figure 6.11: System Verifications for Resent Message 115
Figure 6.12: Capture a Message 116
Figure 6.13: System Verifications for First Sending 116
Figure 6.14: Resending Message 117
Figure 6.15: System Verifications after Resent Message 117
Figure 6.16: Capture a Message 118
Figure 6.17: System Verifications for First Sending 118
Figure 6.18: Resending Message 118
Figure 6.19: System Verifications for Resent Message 119
Figure 6.20: MQTT-Based Systems vs. LoRa-Based Systems Encryption Steps 120
Figure 6.21: System Position 122

List of Tables

Table 2-1: Literature Review Summary Table 19
Table 3-1: LoRa Configurations 22
Table 3-2: Comparison Between IP-based Networks IoT Protocols 28
Table 3-3: MQTT, CoAP, and OPC UA Performance Comparison 30
Table 3-4: Appropriate Encryption Method For Each Protocol 32
Table 3-5: Agricultural IoT Security Techniques Summary 40
Table 3-6: Comparison Between Symmetric Encryption Algorithms 41
Table 3-7: Comparison Between Asymmetric Encryption Algorithms 43
Table 3-8: Combination Between Symmetric and Asymmetric Encryption Algorithms44
Table 3-9: Comparison Between the Classical Encryption Algorithms 46
Table 3-10: Comparison Between Post Quantum Cryptography and Classical Cryptography 48
Table 3-11: Kyber Standard Characteristics 50
Table 3-12: Comparison Between AES, ECC, and Kyber 53
Table 5-1: MQTT-Based System Variables Size 74
Table 5-2: MQTT-Based System Performance Metrics Results (Data per Byte) 75
Table 5-3: MQTT-Based System Performance Metrics Results (Time per Milli Second) 76
Table 5-4: MQTT-Based System Performance Metrics Results (CPU Percentage Usage) 77
Table 5-5: MQTT-Based System Performance Metrics Results (Energy Consumption per milli Joul) 78
Table 5-6: Comparison of Variable Size Results in Different MQTT-Based Models .. 80
Table 5-7: Comparison of Performance Metrics Results in Different MQTT-Based Models 81
Table 5-8: MQTT- Based System Comparison with Another Papers Results 83
Table 5-9: Time Comparison (Key Generation, Encryption, Decryption) with Another Paper Results with Kyber Standards 84
Table 5-10: LoRa-Based System Variables Size 85
Table 5-11: LoRa-Based System Performance Metrics Results (Data per Byte) 86
Table 5-12: LoRa -Based System Performance Metrics Results (Time per Milli Second) 87

Table 5-13: LoRa -Based System Performance Metrics Results (CPU Percentage Usage) 89
Table 5-14: LoRa -Based System Performance Metrics Results (Energy Consumption per milli Joul) 90
Table 5-15: Comparison of Variable Size Results in Different LoRa-Based Models ... 91
Table 5-16: Comparison of Performance Metrics Results in LoRa-Based Models 92
Table 5-17: Comparison of LoRa-Based Systems with Other Papers' Results 94
Table 5-18: Comparison with Another Papers’ Results in LoRa-Based Systems 95
Table 5-19: Comparison with Kyber Handshake Values in LoRa-Based Systems 96

List of Abbreviations

Abbreviation	Full Form
ABP	Activation by Personalization (LoRaWAN mode)
AC	Alternating Current
AES	Advanced Encryption Standard
AES-GCM	Advanced Encryption Standard — Galois/Counter Mode
AMQP	Advanced Message Queuing Protocol
API	Application Programming Interface
ARP	Address Resolution Protocol
ASCON	Authenticated Encryption Scheme (NIST finalist)
BB84	Bennett and Brassard 1984 Quantum Key Distribution Protocol
CoAP	Constrained Application Protocol
CoRe	Constrained RESTful Environments
CPU	Central Processing Unit
CPU%	Central Processing Unit Utilization Percentage
CRC	Cyclic Redundancy Check
CSV	Comma-Separated Values
DA	Data Access
DC	Direct Current
DCPS	Data-Centric Publish-Subscribe
DDS	Data Distribution Service
DDOS	Distributed Denial of Service
DES	Data Encryption Standard
DH	Diffie-Hellman (Key Exchange)
DNS	Domain Name System
DOS	Denial of Service

DTLS	Datagram Transport Layer Security
E91	Ekert 1991 Quantum Key Distribution Protocol
ECC	Elliptic Curve Cryptography
ESP32	Espressif Systems Programmable 32-bit Microcontroller
ESP-IDF	Espressif IoT Development Framework
EU	European Union
FAO	Food and Agriculture Organization
FPGA	Field Programmable Gate Array
GDP	Gross Domestic Product
HA	Historical Access
HID	Human Interface Device
HMAC	Hash-based Message Authentication Code
HTML	Hyper-Text Markup Language
HTTP	Hyper-Text Transfer Protocol
HTTPS	Hyper-Text Transfer Protocol Secure
IBM	International Business Machines
ID	Identifier
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IIoT	Industrial Internet of Things
IoT	Internet of Things
IoT-GW	Internet of Things Gateway
ISA	International Society of Automation
JSON	JavaScript Object Notation

KEM	Key Encapsulation Mechanism
LoRa	Long Range
LoRa-GW	LoRa Gateway
LoRaWAN	Long Range Wide Area Network
MAC	Media Access Control
MITM	Man in the Middle
MKP	Multiple Knapsack Problem
MLWE	Module Learning with Errors
MQTT	Message Queuing Telemetry Transport
MQTT-SN	Message Queuing Telemetry Transport for Sensor Networks
NIST	National Institute of Standards and Technology
N/V	Not Valid
NVS	Non-Volatile Storage
OASIS	Organization for the Advancement of Structured Information Standards
OMG	Object Management Group
OPC UA	Open Platform Communications Unified Architecture
PC	Personal Computer
PKE	Public Key Encryption
PKI	Public Key Infrastructure
PRG	Program
PQC	Post-Quantum Cryptography
QoS	Quality of Service
QKD	Quantum Key Distribution
RAM	Random Access Memory
RC4	Rivest Cipher 4

REST	Representational State Transfer
RSA	Rivest–Shamir–Adleman
RTPS	Real-Time Publish-Subscribe
SHA	Secure Hash Algorithm
SOAP	Simple Object Access Protocol
SSL	Secure Sockets Layer
TCP/IP	Transmission Control Protocol / Internet Protocol
TDES	Triple Data Encryption Standard
TLS	Transport Layer Security
TLSv1.3	Transport Layer Security Version 1.3
UDP	User Datagram Protocol
UV	Unified View
WEKA	Waikato Environment for Knowledge Analysis
Wi Fi	Wireless Fidelity
WPAN	Wireless Personal Area Network
XML	eXtensible Markup Language

Chapter 1: Introduction

1.1 Overview and Background

The professions a person works in throughout their life are diverse: city dwellers work in government jobs, private companies, and factories, while Bedouins tend to graze and migrate, and most rural people work in agriculture and in caring for animals such as sheep, cows, and other animals.

Agriculture, with its two branches (plant and animal production), is considered the backbone of life on Earth because it provides essential needs for living organisms, such as food, medicine, and raw materials. It also contributes to providing a beautiful view through plants and flowers in their various forms. Plants also purify the air and provide oxygen.

Agriculture is defined as the process of producing food such as grains, fiber, fruits, vegetables, and animal feed. It also includes raising livestock and pets such as cows. Agriculture also includes the production of commodities such as flowers, nursery plants, timber, leather, fertilizers, fibers (such as cotton and wool), fuels (such as biodiesel), and medicines (Kerr Casper, 2007).

Because of its vital benefits, agriculture plays a significant role in contributing to national income. As Banerjee (2022) states, "*The agriculture sector is the backbone of an economy, which provides the basic ingredients to mankind and now raw materials for industrialisation, the lesson learnt from the economic experience of many developing countries tell us that agriculture prosperity greatly led to economic development. The leading developed countries in today's world were once primarily agricultural countries, though emerging economics still control agriculture and contribute mainly to national income. still 28 percent of the national income comes from this sector in India.*"

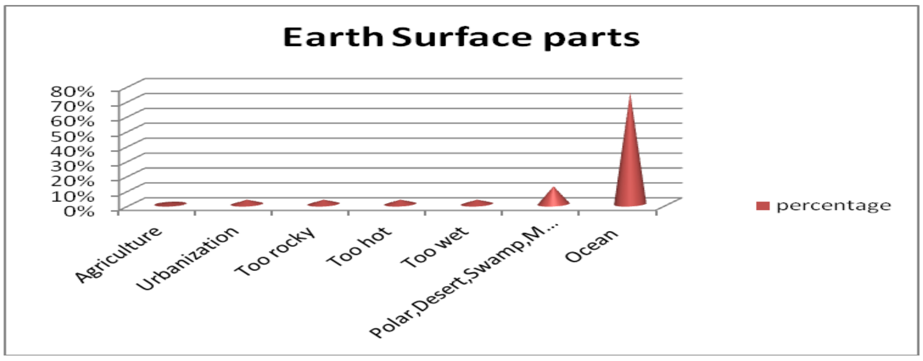


Figure 1.1: Earth Surface Parts

The above figure shows the percentage of arable land, which does not exceed 1% of the Earth’s surface area. (Kerr Casper, 2007).

Therefore, researchers have studied the prospects related to agriculture and ways to develop it. Among these studies are:

Agriculture is exposed to many threats that prevent it from contributing to national income and raising individuals’ standard of living, including urban sprawl that destroys agricultural lands by converting them into residential or commercial buildings, agricultural pests, soil erosion, and others.

One of the most dangerous of these threats is global warming, which is defined as: the rise in the average temperature of the Earth's surface as a result of an increase in the percentage of a group of gases such as carbon dioxide emissions resulting from burning fossil fuels in factories or cars, or from deforestation that absorbs these gases, which traps heat that would leak from the Earth, which affects the climate and causes changes in the weather. (Hegde, 2021)

According to FAO statistics between 2007 and 2022, agricultural losses accounted for an average of 23% of total losses in all sectors, and more than 65% of losses resulting from drought were in this sector. While it accounted for 20% of losses related to natural disasters such as floods and others in the agricultural sector, the average total agricultural losses from disasters reported at the Sendai Framework Observatory was US$13 billion annually, most of which was attributable to floods (16%), fires and wildfires (13%), and drought (12%). Figures from disaster needs assessment surveys are likely to be significantly lower than reality due to limitations and delays in data reporting. (FAO, 2023)

Global warming and fluctuations have significant impacts on agriculture and are considered among the most serious threats, as they cause water scarcity and land drought, leading to agricultural failure and crop loss.

Therefore, researchers have studied the prospects related to this phenomenon. Among these studies is a study that summarized the effect of the phenomenon on agriculture as follows:" it could be concluded that increments in temperature increment crop-breath rates; lessen crop length, the quantity of grains shaped, and crop yield; repress sucrose osmosis in grains; influence the endurance and dissemination of nuisance populaces; rush supplement mineralization in soil; decline manure use effectiveness; and increment vanishing " (Saini & Bhatt, 2020)

As in the world, Palestine's agriculture sector holds economic and political significance. According to statistics from the Palestinian Central Bureau of Statistics, the sector employs approximately 11.5% of the workforce. Agricultural income also represents 5.6% of GDP and 21% of total exports. The great importance of agriculture in Palestine lies in the fact that it plays a pivotal role in protecting land from confiscation and Israeli settlements. (Palková et al., 2022)

Environmental and political challenges are the main factors threatening Palestine's agriculture sector. However, the effects of these issues are intensified when examined in the context of climate change. In this perspective, the quantity and quality of water have been steadily declining in recent years. Moreover, the scarcity of land resources, rapid population growth, pollution of aquifers and marine environment, desertification, and land degradation are challenges. With regional climatic change, such as changes in rainfall quantity and distribution, and increased seasonal temperature fluctuations, environmental problems are exacerbated. Consequently, the negative impact on the agriculture sector may be attributed to crop destruction, reduced water availability, and biodiversity loss. Thus, it can negatively affect the natural control of agricultural pests and delay the onset of growing seasons. So, farmers have used IoT technology to remotely operate pumps, irrigate farmland, and protect it from drought and desertification.

The Internet of Things is defined as a network of integrated elements of devices, sensors, machines, and software across the Internet environment, to communicate, exchange information, and interact with each other. (Quy et al ,2022)

Many studies discussed the benefits of implementing the Internet of Things and big data in agriculture, and pointed to the challenges - most notably security technologies - that must be overcome to accelerate the deployment of the Internet of Things in smart agriculture so that it becomes accessible to the majority of farmers, including small and medium-sized farm owners. It will enhance productivity, provide clean and environmentally friendly food, support food traceability, and reduce human labor, improving production efficiency. (Quy et. al, 2022)

Communications security remains the most important issue that accompanies any data transfer, so it is essential to protect IoT systems against malicious attacks. Therefore, many studies have addressed this topic, including the study (Rehman, Singh & Manickam, 2020) that explores current IoT security solutions that can be used in an IoT environment based on specific security characteristics. Most of these solutions have been shown to focus specifically on designing lightweight systems for wearable IoT devices with limited resources, while addressing issues of authentication, confidentiality, and trust in IoT systems. The results of this study show that no current security system offers a comprehensive solution, as each IoT system has its own advantages and limitations. With the advent of the Fourth Industrial Revolution, future civilization will revolve around Internet of Things systems. Therefore, researchers and developers need to focus on designing IoT systems that are robust against attack methods at all three IoT layers: the physical layer, the network layer, and the application layer. In this way, we can achieve a sustainable IoT ecosystem.

Securing communication standards for IoT-based smart irrigation systems is critical for numerous reasons related to efficiency, security, and sustainability, especially in light of the growing challenges facing the agricultural sector, where numerous benefits are being achieved: protecting transmitted and received data, preventing electronic breaches, compliance with legislation and standards adopted at work, and many more.

What increases the importance of the matter is the diversity of these standards in different branches, as shown in the attached image

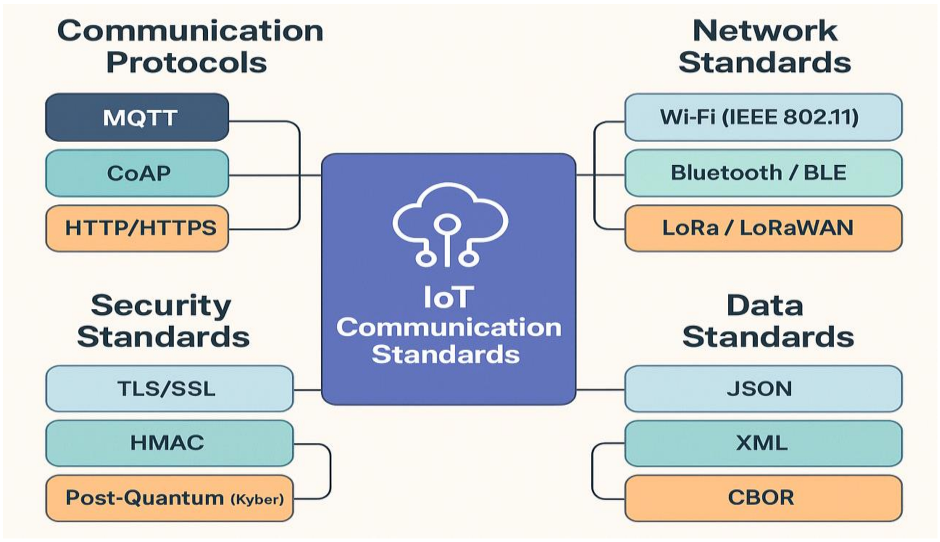


Figure 1.2: IoT Communication Standards

Looking at the picture, we find that the term "Internet of Things communications standards" refers to the rules, protocols, and technical specifications that determine how smart devices communicate with each other and with different networks. The dark blue color used in the central box (IoT Communication Standards) symbolizes stability and reliability, and expresses the general framework or reference layer within which all other standards are integrated. The blue-gray color used in Communication Protocols combines the MQTT, CoAP, and HTTP/HTTPS protocols, and indicates the message-exchange layer and communication between nodes. This color is usually used to represent logical operations and organized communication. The MQTT protocol color has been highlighted as a darker shade because it is the most common and widely used. The cyan-green color used with Network Standards (Wi-Fi, Bluetooth/BLE, LoRa/LoRaWAN) symbolizes the wireless networking layer and reflects the concept of communication and diffusion across distances. The light green color used in Security Standards, which combines TLS/SSL, HMAC, and Post-Quantum (Kyber), indicates security, protection, and reliability. The bright orange color used in Data Standards, which combines JSON, XML, and CBOR, symbolizes processing, flexibility, and formula compatibility, and highlights the role of these standards in representing data and exchanging it efficiently between different systems. These standards aim to ensure compatibility, security, efficiency, and stability of communication between various IoT devices, such as sensors, actuators, wearable devices, and control systems. To ensure these standards, we use multiple technologies: encryption, authentication, and protocols that support communications security.

Many research papers have discussed this topic and explained the use of these technologies, and other research papers have discussed attacks on communications systems during their operation, such as MITM attacks and others. Another set of papers reviewed effective security methods in confronting these attacks to ensure the highest level of security, confidentiality, and credibility in communications systems. It is worth noting that cyber-attacks are not limited to wired communications but extend to wireless communications, as is the case with LoRaWAN network communications. Wireless attacks may be easier because wireless signals spread through the air and are easier to pick up if they do not receive the necessary protection. Therefore, the relevant institutions have worked to provide multiple protection measures to ensure obtaining the maximum possible level of protection that allows data to be transferred in complete confidentiality, ensuring the friendship, integrity, and confidentiality necessary for information security in various networks.

Despite numerous studies examining IoT connectivity standards and the mechanisms for applying these technologies in smart agriculture, and the interest of several researchers in improving the efficiency of irrigation and water management systems using protocols such as MQTT (Message Queuing Telemetry Transport, a lightweight messaging protocol), LoRaWAN (Long Range Wide Area Network, a wireless IoT protocol), and CoAP (Constrained Application Protocol, a specialized protocol for simple devices with limited resources), most of this research has focused on operational and technical aspects without paying sufficient attention to the level of cybersecurity in resource-limited agricultural environments such as Palestine. The literature also lacks integrated models that combine low-power, long-range communication with advanced quantum computing-resistant encryption mechanisms such as the Kyber algorithm (a cryptographic algorithm designed to resist quantum attacks).

This study addresses the lack of a comprehensive security framework for smart irrigation systems that balances performance, energy efficiency, and safety within Palestinian agricultural contexts. It aims to develop and validate an integrated security approach using lightweight protocols and modern encryption to ensure robust data protection and reliable communication.

1.2 Motivation

Based on the great role that agriculture plays in human life, many motivations emerge to increase agricultural production and preserve plant wealth, including:

1: Digital transformation in agriculture: The spread of smart irrigation systems based on the Internet of Things has expanded, and securing communication through them has become a necessary need to protect sensitive data and information that these systems use to achieve efficiency and effectiveness.

2: The importance of protecting sensitive data: smart irrigation systems contain vital information such as weather, soil, and water data, and as cyber threats increase, any breach or manipulation of such data can result in significant economic and agricultural losses.

3: Ensuring the sustainability of water resources in Palestine: The effectiveness of smart irrigation systems depends on accurate and timely data transmission; thus, securing communication enhances their ability to improve the sustainable use of water resources.

4: Improving user confidence: Enhancing safety in smart irrigation systems increases the confidence of farmers and users to adopt modern technology and take advantage of its advantages, thereby increasing the spread and use of smart farming technologies.

5: Provide sophisticated and flexible security solutions: The research seeks to provide sophisticated security solutions that meet users' needs and keep pace with technological advances that contribute to improving the overall performance of smart irrigation systems.

6: Contributing to the Palestinian agricultural economy: By ensuring communication in smart irrigation systems, agricultural productivity can be improved and losses reduced, contributing to the strengthening of the agricultural economy and food security

1.3 Problem Statement

In recent years, IoT-based smart irrigation systems have become a key part of agriculture's digital transformation. These systems provide effective water resource management solutions by accurately collecting and analyzing data to make improved irrigation decisions. However, the interconnected nature of these systems makes them vulnerable to multiple security risks, such as penetration, eavesdropping, and data manipulation.

Many of these systems worldwide, in general, and in Palestine in particular, lack robust safety protocols to protect their communications from cyber threats, putting sensitive data, such as soil and weather information, and biosensor data, at risk, which may negatively affect the efficiency and effectiveness of these systems.

Therefore, the problem is how to secure IoT-based smart irrigation system connections to ensure data protection and improve the sustainability of water resources by developing effective security strategies that address potential cyber threats without negatively affecting the overall performance of the system. **These challenges include:**

1. How can we design security solutions to improve connectivity security in smart irrigation systems?

2. How can we balance performance and security without negatively affecting the system's efficiency?

1.4 Research Objectives

This research establishes a secure and reliable communication system for IoT-based smart irrigation systems and raises their effectiveness through various objectives, including:

1: Develop security solutions that enhance communication standards' security, including encryption, authentication, and data integrity.

2: Encouraging farmers to trust modern technology to increase agricultural production by providing security guarantees and promoting the adoption of smart irrigation systems in Palestine.

3: Improving the reliability of smart irrigation systems, supporting local agriculture by enhancing water efficiency, and protecting agricultural operations from cyber threats is key to achieving agricultural sustainability.

1.5 Thesis Contribution to the Field

This study contributes to enhancing network and communications security in smart irrigation systems based on Internet of Things technologies. The main goal is to ensure the secure exchange of information between different devices within the system. The mission aims to provide a solid foundation for protecting smart irrigation systems and enhancing their stability and operational efficiency against contemporary security challenges. This research has the potential to contribute to more sustainable and secure protection of agriculture and water resources by expanding the use of Internet of Things technologies that ensure the immediate operation of water sources, and creating a starting point for future research in this field.

1.6 Thesis Outline

Following the introduction in Chapter One, this thesis presents and discusses several topics in the subsequent chapters. Chapter Two (Literature Review) reviews a range of previous studies related to the core themes of the thesis, including irrigation systems, data security methods, and other relevant areas. Chapter Three (IoT Protocols and Data Security Mechanisms) outlines general data security methodologies, identifies the risks facing information systems, and explores strategies to mitigate them. Chapter Four (System Design) describes a set of experimental systems that were tested, analyses their results, and determines the most effective design for the proposed methods. Chapter Five (Results and Discussion) presents and evaluates the outcomes of each proposed system. Chapter Six (Security Analysis) examines the level of security achieved by the system and its resistance to electronic attacks. Finally, Chapter Seven provides the conclusion, highlights key challenges, and offers recommended future directions.

1.7 Conclusion

This chapter established the background, motivation, and objectives of the study. It highlighted the importance of securing communication in IoT-based smart irrigation systems within the Palestinian context and outlined the main research questions and objectives. The following chapter reviews the existing literature on IoT applications in agriculture, communication protocols, and security challenges, forming the theoretical foundation of this work.

Chapter 2: Literature Review

2.1 Introduction

The Science of the Internet of Things constitutes a broad horizon for knowledge, experimentation and data transfer, and the security of information transmitted through it is one of the most important issues that scientists have sought to reach in the best possible ways, so many scientists have sought to study this science and how to secure the information transmitted through it, so this chapter sheds light on a group of studies that have sought to research aspects of this science.

2.2 Previous Studies

This section aims to present previous studies in the field of Agriculture and the field of securing communication when using Internet of Things technology.

2.2.1 Agriculture

2.2.1.1 Smart Irrigation Systems

The papers **(**Muthuramalingam et al., 2023) (Karpagam et al., 2020) (Ullah et al., 2021) (Sowmyashree & Swaraj, 2020) (Islam et al., 2020) (Thaher & Ishaq, 2020) (Moussa, Nataraj & Seeralan, 2021) (Nigussie et al., 2020) (Mojarad et al., 2021) aim to design and implement a smart irrigation system based on the Internet of Things to increase water use efficiency in agriculture and reduce waste. It shows that this system can provide an effective and sustainable solution to traditional irrigation problems in agriculture. To improve water-use efficiency and increase productivity, sensors in the system collect data and send it to the control unit, which processes it and decides whether to operate the pumps. The system sends data to cloud servers for later use and processing. This system can contribute to significant economic and environmental benefits; however, it faces several challenges, including cost, maintenance, and reliance on wireless communication, which is affected by various atmospheric influences.
(Ahmadi et al., 2024; Quy et al., 2022; Ingole et al., 2024; Blessy & Kumar, 2021) The papers aim to present and analyze strategies to comprehensively improve smart agriculture monitoring through connectivity management and a dynamic device clustering strategy in agricultural Internet of Things (IoT) networks. These strategies enhance the efficiency and stability of agricultural IoT networks, resulting in increased productivity and reduced costs. Despite the challenges related to cost and integration,

the potential benefits make the adoption of these strategies an important step toward achieving smart and sustainable agriculture.

(Froiz-Míguez et al., 2020) The main objective of these papers is to design and implement an intelligent irrigation system based on LoRa and LoRaWAN technologies to improve water use efficiency in agriculture and to apply edge computing to analyze data within the system. Sensor nodes installed in agricultural fields collect environmental data such as soil moisture and temperature. This data is then transmitted via LoRa and LoRaWAN technology to fog computing gateways. It processes the data locally to make immediate decisions on the need for irrigation. The processed data is then sent to cloud servers for later storage and analysis. Farmers can monitor the data and make decisions based on the received analyses via available applications. The proposed system is water-efficient, has a low response time, and saves energy. Despite the challenges related to cost and maintenance, the opportunities provided by improving agricultural productivity and resource sustainability make this system a promising solution for modern agriculture.

(Ahmed et al., 2022; Massaoudi, Berguiga, & Harchay, 2022). These studies examine the application of contemporary technologies, including artificial intelligence, blockchain, and the Internet of Things, to climate-smart agriculture, emphasizing fundamental concepts, challenges, and opportunities. Despite the many challenges, such as cost, the need for regulation, privacy policies, and the importance of training workers and familiarizing them with the different branches of the system, the opportunities available make the adoption of these technologies an important step towards achieving sustainable and more productive agriculture in the face of climate change. The role of the Internet of Things is focused on collecting environmental and agricultural data through smart sensors, which improves the management of resources such as water and fertilizers.

2.2.1.2 Agriculture in Palestine

Several studies have examined agriculture in Palestine from various perspectives. For example, Salem, Yihdego, and Mohammed (2021) investigated the status of irrigation water sources in the occupied Palestinian territory, with a particular focus on freshwater and reused treated water. The paper also presented a range of challenges facing Palestinian land in providing water for agriculture due to restrictions on access to freshwater, as well as the effects of Israel's occupation and associated water policies.

The paper also reviewed the current status of freshwater resources in the occupied Palestinian territory and the potential for reused treated water to be used for irrigation as an alternative to addressing water shortages and challenges.

Palková et al. (2022). This study aims to explore the role of education in agriculture 4.0, a concept that includes the application of modern technologies such as artificial intelligence, the Internet of Things, robotics, and graphic analysis in the agricultural sector. The application of this concept has been examined in some EU and Palestinian states to identify differences, opportunities, and challenges faced by these States in the application of Agriculture 4.0. Agriculture 4.0 refers to the fourth industrial revolution in agriculture, which relies on the use of modern technology to improve agricultural productivity, reduce losses, and achieve greater sustainability. The paper also reviews the importance of modernizing agricultural education systems to keep pace with this technical revolution in agriculture.

Abuzir (2017). This study aimed to identify the main factors affecting vegetable production in Palestine through the use of the WEKA data exploration tool. Data exploration techniques have been employed to analyse a wide range of agricultural data in order to identify the key factors influencing agricultural production in the Palestinian vegetable sector. It also reviews a range of challenges facing agriculture, including water scarcity, climate change, and political conflicts that limit access to agricultural land and water.

Abuzir, Awad, and Khdair (2021). This study aimed to design and develop an online market information system to support Palestinian farmers in accessing essential agricultural market information. The system is an interactive tool that seeks to facilitate farmers' access to accurate and up-to-date data on prices, supply, and demand, thereby contributing to improving their business decisions and increasing their economic efficiency. The paper also discusses the system's challenges of limited Internet access in some rural areas of Palestine, as well as the lack of technological awareness among some farmers.

2.2.1.3 The Impact of Climate Change on Agriculture

Numerous studies have addressed the effects of global warming and climate change on agriculture, including those by (Iqbal et al. 2024); (Dubovitski et al. 2021); (Saini & Bhatt, 2020); (Ozdemir, 2021); (Faradiba,2024); (Mbaraka, 2023); (Gulzar, Abbas & Waqas, 2020); (Jabal, Khayyun & Alwan, 2022) and (Liaqat et al., 2022) review the effects of global warming on agriculture, discussing how climate change affects crop yields, food security, and sustainable agriculture. The objective of these studies is to conduct a thorough examination of the obstacles confronting agriculture due to global warming and climate change, investigate strategies to enhance agricultural sustainability, and assess the impacts of climate change on agriculture, including insufficient rainfall and drought resulting from rising temperatures. Finally, by analyzing previous studies in the fields of smart agriculture, the Internet of Things, and communications security, it becomes clear that there is a significant accumulation of knowledge in the development of smart irrigation systems using lightweight communication protocols such as MQTT, CoAP, and LoRaWAN. Numerous studies, including Muthuramalingam et al. (2023), Karpagam and Merlin (2020), and Ullah et al. (2021), have focused on improving water-use efficiency and automating irrigation processes through sensors and remote control. These approaches have contributed to enhancing water resource management and reducing waste. However, most of these studies were limited to operational aspects and did not address in depth the cyber threats that smart irrigation networks could face or the risks of hacking and data loss.

2.2.2 Security of IoT Communication

2.2.2.1 Security and Privacy

Has et al. (2024). This paper reviews and analyzes efficient data management techniques for agricultural Internet of Things (IoT) applications, focusing on three main axes: data compression, security, and analysis of the MQTT protocol. The paper suggests that the use of data compression and security techniques, in addition to the MQTT protocol, can significantly improve data management in agricultural IoT applications. Despite the challenges, achieving a balance between performance and security can lead to more efficient and reliable systems. There is a continuing need for research and development to improve the integration of these technologies and expand their use in diverse agricultural environments.

(Yousefi & Jamei, 2023; Gupta, Vanjale & Rasal, 2020). The papers review and analyze encryption algorithms that can improve the security of devices and data in IoT networks, focusing on the challenges and advantages of each algorithm. offers insights into the effective application of and the importance of developing new encryption algorithms and more effective key management that suit diverse IoT environments, pointing to the ongoing need for research and development to achieve robust and effective security solutions that meet resource constraints and practical challenges in this field. It indicates that achieving a balance between security and performance is key to ensuring the effectiveness of encryption applications in the Internet of Things.

(Tawalbeh & Quwaider, 2021) (Qureshi et al., 2022). These research papers present a model based on Internet of Things and cloud computing technologies to improve the safety and efficiency of agricultural operations, while clarifying the benefits and ways to address the challenges related to applying this model. The paper presents an integrated model based on IoT and cloud computing technologies to improve the safety and intelligence of the agricultural environment. The model highlights the benefits of improved resource efficiency, increased safety, and intelligent data analysis. Despite the challenges associated with cost and data management, the model offers a promising future vision for transforming traditional agriculture into smart and sustainable agriculture using modern technologies. The research indicates the ongoing need for research and development to improve current technologies and ensure their effective integration with the diverse agricultural environment.

(Saideh, Jamont, and Vercouter, 2024); (Rahaman et al., 2024; (Ferrag et al., 2020), and (Vangala et al., 2020). These studies explore methods of enhancing IoT security using authentication factors based on data collected from sensors, with a focus on achieving more secure and effective authentication in IoT environments. This approach offers significant benefits to security and user experience but requires addressing challenges related to privacy, data management, and integration. The paper calls for continued research and development to improve authentication methods and ensure effective security in diverse IoT environments.

(Zhang et al., 2020); (Rao & Deebak, 2022); **(**Toh, 2020); (Qian et al., 2024); (Amjath & Senthooran, 2020); (Samawi, 2021). The papers explore security and privacy issues in smart cities and industries by adopting advanced technologies and integrated security solutions. It presents multiple solutions to improve security and protect privacy. Smart environments face challenges related to integration, big data management, and cost, but using technologies such as encryption, access control systems, data analytics, and blockchain, security can be greatly improved. The paper calls for increased research and development in the field of security to ensure the success of smart cities and future industries.

2.2.2.2 Kyber

The following studies focus on the use of Kyber, as the study (Kim & Seo, 2025) presents a physical implementation of KEM-MQTT using Kyber-512 on resource-constrained AVR devices. Achieves performance improvements (KeyGen, Encapsulation, Decapsulation) of up to 81–85% compared to the reference implementation. While the study (Kaganurmath, Cholli & Anala, 2025) presents a lightweight key exchange protocol using Kyber + BLAKE2s + ChaCha20 within MQTT on Contiki OS and Cooja Simulator. The study (Moon et al., 2025) addresses the vulnerability of traditional security protocols to quantum computing and proposes solutions based on quantum-resistant algorithms, such as Kyber, to secure communications in edge (Edge) and fog (Fog) networks.

The study by (Segatz & Al Hafiz 2025) provides an effective implementation of the quantum computing-resistant CRYSTALS-Kyber algorithm on the ESP32 controller common in IoT applications, by splitting the algorithm to take advantage of the device's dual-core structure and using built-in cryptographic processors (SHA and AES) to improve the performance of key generation, packaging, and unpacking operations. The study shows that taking advantage of these features leads to a significant acceleration in implementation compared to the basic design. However, the study remains limited in that it focuses on performance improvements for custom implementation without a comprehensive assessment of how these improvements impact security against practical attacks or application testing within real IoT network scenarios, limiting the generalizability of the results to real multi-condition environments.

A study by Khalil & Manaa (2025) presented a two-stage security framework for Fog-IoT networks to enhance protection against quantum attacks. The second stage of the framework relies on the CRYSTALS-Kyber algorithm to ensure encryption resistant to future attacks, along with the use of Chaotic Elliptic-Curve Diffie-Hellman in the first stage to securely exchange keys. The framework provides robust security with applicability in resource-constrained IoT environments. However, the study remains limited in that it did not provide a comprehensive assessment of performance when using different protocols, such as MQTT or LoRaWAN, which reduces the generalizability of the results to large-scale practical applications.

2.2.2.3 LoRa / LoRaWAN

(Fragkopoulos et al. 2023). The study evaluates the performance of LoRaWAN networks in rural and peri-urban environments, focusing on technical factors affecting communication quality. Although the study was important in highlighting the impact of the environment and network layout on performance, it focused on technical and physical aspects without addressing security aspects or the impact of encryption mechanisms on network efficiency.

(Yassir et al. 2024). The study analyzed LoRaWAN performance in the context of IoT applications, but it was limited to a limited operating environment and did not study the effect of node density or different operating environments. The two studies also did not provide adaptive solutions that combine improved performance with ensuring data security in resource-limited environments.

(Ehsan et al. 2025). The study evaluates quantum computing-resistant key encapsulation algorithms, specifically Kyber and NTRU, in the context of resource-limited IoT environments, focusing on performance indicators such as execution time, memory, and power consumption, and bandwidth efficiency. While it did not explicitly address the measurement of central processing unit (CPU usage), it did not test the algorithms within real low-rate communication scenarios such as LoRaWAN networks, which limits the generalizability of its results to some IoT applications with strict limitations.

The study (Mahdi & Abdullah, 2025) proposes a hybrid cryptographic framework that combines the Kyber-512 algorithm and ASCON to improve communication security in resource-constrained IoT environments. The framework covers the stages of key generation, encryption, secure transmission, and decryption with computational efficiency suitable for devices with limited resources. However, the study relies solely on a simulated test environment without experimental implementation on a real IoT network, and it does not fully address practical attack resistance analysis, limiting the generalizability of its results to large-scale real-world applications.

(Commey et al. 2025). The study analyzes the performance of quantum computing-resistant encryption algorithms on resource-constrained consumer electronics devices, comparing them with algorithms using traditional methods in terms of execution time, key size, and memory impact. Although the study is valuable in providing a quantitative view of the performance of post-quantum algorithms, it did not test them within real ongoing deployment scenarios, nor did it evaluate their impact on low-power communication protocols such as LoRaWAN, which limits the generalizability of the results to practical IoT applications.

2.2.2.4 IoT Protocols

(Tightiz & Yang, 2020); (Choudhary & Meen, 2022); (Qays et al., 2023); (Brahmananda & Vairagade, 2020); (Mansour et al.,2023). These studies discuss the various IoT protocols and differences between them in terms of accuracy, efficiency, applications, and security issues related to them. It also demonstrates the importance of the Internet of Things in everyday life by connecting devices and designing smart systems capable of collecting and analyzing data. It also discusses the concept of smart grids and their role in improving and increasing the efficiency of energy systems that rely on IoT technology to monitor and control them.

(Seoane et al. 2021). The study evaluates the performance of the MQTT and CoAP protocols in IoT environments with support for security mechanisms by analyzing the impact of applying security protocols on key performance indicators such as response time, resource consumption, and communication efficiency. The study aimed to highlight the trade-off between security level and performance in common IoT protocols and showed that adding security mechanisms directly affects communication efficiency, especially in resource-limited environments. However, the study is limited to traditional encryption algorithms and does not address encryption techniques resistant to quantum computing, nor has it tested performance in low-rate and power networks such as LoRaWAN, which limits the suitability of its results for future secure IoT systems.

(Mishra et al. 2021). The study evaluates the performance of MQTT brokers under stress through indicators such as processor usage and response time, to measure the brokers' ability to handle high loads. However, the study was limited to traditional network infrastructure and did not test performance on low-resource devices or with encryption techniques resistant to quantum computing, limiting the generalizability of the results to modern, secure IoT applications.

2.2.2.5 IoT Standards

(Lee et al., 2021). The study is a comprehensive survey of operating and manual standards in the IoT environment, where it discusses the most important standards to be observed as well as the most important challenges and gaps to be addressed to ensure a secure IoT environment.

The paper (Prince et al., 2024) discusses how to enhance the security of Internet of Things (IoT) devices against cyberattacks using IEEE standards and deep learning techniques. The study focuses on integrating IEEE standards with deep learning algorithms to effectively detect and respond to threats. The paper also reviews the challenges associated with securing IoT devices and proposes AI-based solutions to improve cybersecurity.

(Gebremichael et al., 2020). The study offers security requirements in the current IoT industrial environment and the extent to which it can find compatibility between the current communication protocols and platforms with current industrial standards, as well as the challenges to be addressed to ensure a safe and reliable industrial environment in the future, where The paper (Manda,2021) reviews the general frameworks for telecommunications operators to ensure security in the IoT environment, where it can be considered an important reference for enhancing the security of networks and devices during the continuous expansion of IoT technologies, and provides practical guidance for designing and implementing effective and integrated security frameworks.

Finally, studies such as (Kim & Seo, 2025) and (Kaganurmath, Cholli & Anala, 2025), although they provided advanced approaches using algorithms resistant to quantum computing (Kyber), remained within the framework of theoretical simulation without integrating them into field communication systems.

2.3 Literature Review Summary Table

The following table summarizes the most relevant studies related to IoT-based smart irrigation and secure communication frameworks. The review reveals a growing interest in combining lightweight protocols with advanced cryptography to enhance system reliability and security.

Table 2-1: Literature Review Summary Table

Study	**Focus / Objective of Study**	**Main Findings**	**Limitations**	**Relation to Current Study**
(Amjath & Senthoora, 2020)	Using information hiding technology (Steganography) to hide data exchanged between IoT nodes	Integrating Steganography into IoT communications can add a layer of security	Did not include real-world tests	Securing IoT systems
(Seoane et al.,2021)	Evaluate the performance of CoAP and MQTT protocols in IoT environments with support for security mechanisms	Adding security layers significantly affects response time and resource consumption	Did not include real-world tests	Provides a basis for understanding the impact of security on the performance of IoT protocols
Ahmed et al. (2022)	Exploring the use of smart technologies, blockchain, and the IoT to improve productivity and sustainability	Enhanced smart agriculture by improving resource management,	Did not include real-world tests	Securing IoT systems in agriculture
(Aslan et al., 2023)	A comprehensive review of cybersecurity-related issues	Reviews security issues in digital systems	Relies on previous references and does not provide practical results	Securing IoT systems in agriculture
(Ingole. & Padole, 2024)	Design and implement an IoT-based agricultural intelligence platform	Developed a practical model of a system based on sensors connected to IoT to collect and analyse data to determine irrigation times	Lacks a strong focus on security aspects and security performance analysis	Securing IoT systems in agriculture.
(Kim & Seo, 2025)	Improving the performance of the MQTT protocol to resist quantum attacks	The possibility of integrating post-quantum encryption mechanisms with the MQTT protocol	Framework of theoretical simulation does not provide practical quantitative results	Securing IoT systems using post-quantum encryption mechanisms

From this table and the above analysis, it becomes clear that there is a clear research gap represented by the absence of an integrated application framework that combines low-power, long-range communication protocols with advanced modern encryption mechanisms with the aim of achieving a balance between performance, reliability, and cybersecurity. Most previous research efforts have focused on developing smart agricultural systems from an engineering or operational perspective, without building a comprehensive security system capable of protecting communications and data in realistic agricultural environments. They have also not been tested in complex environmental and economic contexts, such as the Palestinian situation, which suffers from scarce water resources and a weak communications infrastructure. Therefore, this study aims to address this shortcoming by designing and implementing an integrated security system for smart irrigation systems in Palestine, relying on integrating IoT protocols such as MQTT with encryption algorithms such as AES, TLS, and Kyber, to build a practical model that achieves effective data protection with high performance and energy efficiency.

2.4 Conclusion

This chapter provided an in-depth review of previous studies on IoT applications in agriculture, communication protocols such as MQTT, CoAP, and LoRaWAN, and related cybersecurity approaches. The analysis identified a clear research gap in secure, long-range, and energy-efficient communication frameworks for Palestinian smart irrigation. The next chapter provides an overview of the basic communication protocols used in IoT environments, such as MQTT, CoAP, and LoRaWAN, and discusses the most common data security mechanisms to protect information exchange. It also identifies the main types of attacks targeting IoT-based agricultural systems.

Chapter 3: IoT Protocols and Data Security Mechanisms

3.1 Introduction

Smart agriculture based on the Internet of Things (IoT), particularly smart irrigation systems, is among the most sustainable and efficient solutions for optimizing water resource management. However, the deployment of IoT technologies introduces significant cybersecurity challenges that must be addressed to ensure the confidentiality, integrity, and availability of transmitted data. This chapter provides an overview of the primary communication protocols employed in IoT environments, such as MQTT, CoAP, and LoRaWAN, and discusses the most common data security mechanisms used to safeguard information exchange. It also outlines the main types of attacks targeting IoT-based agricultural systems, highlighting the need for robust, lightweight, and energy-efficient security frameworks to protect smart irrigation infrastructure.

3.2 Protocols Proposed for Agricultural IoT:

3.2.1 Low Power Wireless Networks Protocols

3.2.1.1 LoRaWAN

3.2.1.1.1 Overview

Long Range Wide Area Network (LoRaWAN) technology was established in 2009 and works on the principle of sending wireless signals with low-power data packets over a long distance between receiving nodes. It uses LoRaWAN, a MAC layer-based cloud networking protocol designed for battery-powered wireless devices, and is implemented as a star network that supports two-way communication between nodes and the gateway. It adopts Chirp Spread Spectrum (CSS) technology to adopt its own modulation technology by using chirp pulses to encode information via radio waves. The LoRaWAN protocol essentially acts as a protocol for routing the network layer between LoRaWAN gateways and LoRa devices. LoRaWAN also manages the data rate, communication frequencies, and transmission power of all network nodes, which are asynchronous and transmit data on demand. LoRaWAN enables sensor nodes to send information as small, low-power packets to a gate several kilometers away—up to 5 km in cities and 15 km in open or rural places. away in a single hop, and this technology is robust against interference. (Froiz-Míguez et al., 2020) (Almuhaya et al.,2022)

3.2.1.1.2 Configurations

It should be noted that the LoRa protocol has basic settings that must be adhered to, which are: (Yassir et al., 2024)

Table 3-1: LoRa Configurations

Configuration	Value	Use
Frequency	915 MHz	According to the region
Bandwidth	125 Khz	Balance between range and data rate
Spreading Factor	7-12	Increasing the factor improves the range and reduces the data rate
Coding Rate	4/5 – 4/8	Improve error resistance
Tx Power	14 dB – 20 dB	Depends on the distance and the surrounding environment
Transmission Mode	LoRa	Provides a longer range and better interference resistance

3.2.1.1.3 LoRa / LoRaWAN

LoRa represents the physical layer in the communication system and is responsible for the wireless transmission and reception process using Chirp Spread Spectrum (CSS) technology, which enables long-range transmission with low power consumption. This layer defines signal characteristics such as transmission frequency, propagation rate (Spreading Factor), bandwidth (Bandwidth), and error correction rate (Coding Rate), making it the physical basis for the operation of systems based on LPWAN technology. It is considered a modulation technique (Singh et al., 2024)

LoRaWAN is a network protocol that operates on top of the LoRa physical layer, defining communication and management mechanisms between edge devices (End Devices), gateways (Gateways), and network servers (Network Servers). LoRaWAN provides functions such as dual-encryption data security (Network & Application Keys), network join management (Join Procedure), and adaptive transmission rate control (Adaptive Data Rate). Therefore, LoRa can be considered the physical infrastructure, while LoRaWAN is the protocol system that regulates communication and ensures security and reliability across the network. (Jouhari et al ,2023)

3.2.2 IP-based Networks IoT Protocols

3.2.2.1 MQTT

Message Queuing Telemetry Transport (MQTT) is a protocol created by the Organization for the Advancement of Structured Information Standards (OASIS) (Memon et al., 2024).

MQTT is a lightweight machine-to-machine communication protocol that implements a publish-subscribe architecture. In MQTT, multiple clients-either publishers or subscribers- interact with a central message broker. If a client wants to send a message, they post it under a specific thread on the message broker. The message broker forwards this message to all customers who previously subscribed to this topic. There is no direct communication between clients, eliminating coupling in time, space, and synchronization. For each message published, the publisher can specify a Quality of Service (QoS) level. This property determines the effort to ensure that data reaches the medium, which in turn forwards it to all potential subscribers. (Prantl & Krupitzer, 2021)

MQTT, being a binary protocol, typically uses a fixed two-byte header and can accommodate message payloads of various sizes, ranging from small data packets to a maximum of 256 MB. MQTT supports three different levels of service quality:

Quality of Service 0 (Maximum Delivery): Messages are delivered at most once, which means that the sender does not expect any acknowledgment or confirmation from the receiver. This level of QoS provides the lowest cost and highest message transmission rate, but does not guarantee delivery and does not store sent messages for retransmission when needed.

• QoS1 (maximum delivery): This level of QoS ensures that messages are delivered at least once to the receiver. The sender sends the message and expects to receive an acknowledgment (ACK) from the receiver. If no acknowledgment is received within a specified period of time, the message is retransmitted. QoS level 1 ensures that each message is delivered at least once; however, this may lead to duplicate messages at the receiver if the acknowledgement is lost or delayed.

• QoS2 (precise delivery): This is the highest level of QoS in the MQTT protocol and provides the strongest guarantee of message delivery. Messages are delivered exactly once to the recipient. This level of QoS involves a four-step handshake process: the sender sends the message to the receiver and waits for a notification (PUBREC). When the message is received, the receiver sends a notification (PUBREC) to the sender.

The sender sends a PUBREL notification to the receiver, indicating that it has received a PUBREC notification. Finally, the receiver gets a PUBCOMP notification, completing the process. **(**Has et al., 2024)

The following figure explains how the MQTT protocol works

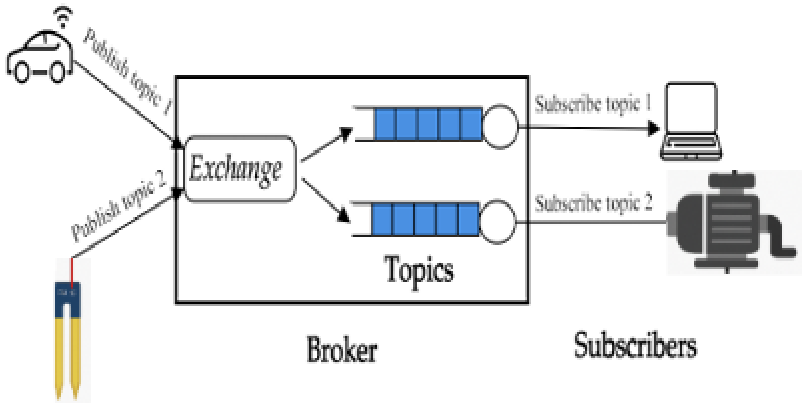


Figure 3.1: MQTT Architecture.

The figure shows the "Publish/Subscribe" communication model used in Internet of Things (IoT) systems, in which the sender of data (publisher) is separated from the recipient (subscriber) by an intermediary known as the "broker." On the left, we find publishers' devices, such as a car and a sensor, each of which sends data in the form of specific "topics" (such as: Topic 1 and Topic 2). The broker (Broker) receives these messages from publishers via a unit known as Exchange, which directs messages to the appropriate topic channels (Topic Queues) based on the topic title. On the right, there are subscribers' devices (Subscribers), such as a computer and a phone, each of which subscribes to a specific topic to receive its messages (Subscribe topic 1 and Subscribe topic 2). This model helps separate the sender from the receiver, enabling scalability and flexibility in systems, and is widely used in IoT applications such as smart agriculture.

3.2.2.2 CoAP

Constrained Application Protocol (CoAP) is the Internet protocol standardized by the Internet Engineering Task Force (IETF) Constrained Resource Environments (CoRE) working group. (Bayılmış et al., 2022)

The CoAP protocol uses two message modes to communicate between request and response, which are "proceed" and "discrete." The main difference between these two modes lies in the time and manner of response. In direct client-server communication, "proceed" means that the server sends its response message immediately after receiving the request. In this case, the server response comes with an acknowledgement (ACK) message. While the discrete mode is used in indirect communication between the client and the server, the response is sent in a separate message from the acknowledgment, and it may take some time for the server to provide it. The CoAP protocol provides two types of messages to ensure the reliability and repeatability of messages. The two types of messages are "confirmable" and "non-confirmable." These two messages are used, respectively, for reliable and unreliable transmission. The use of a confirmable message requires the use of an acknowledgement message to confirm the arrival of the message, while the use of a non-confirmable message does not require the use of an acknowledgement message. One of the most important advantages of the CoAP protocol is that it can be used with devices that are restricted in interaction between devices, and it provides a fast connection thanks to sending small data packets over the UDP layer. This protocol cannot be used in asynchronous communications because it does not support the publisher-subscriber architecture, nor does it support streaming. Customers cannot use a topic to send and reply to messages. (Hmissi & Ouni, 2022)

3.2.2.3 AMQP

Advanced Message Queuing Protocol (AMQP) is a lightweight M2M protocol. It is both an ISO and an OASIS standard protocol. (Qays et al., 2023).

It is characterized by its capability in orienting messages, queuing, switching, security, reliability, and privacy. The AMQP protocol follows the architecture of the publish/subscribe approach. The principal benefit of using AMQP consists of the store-and-forward element that guarantees reliability and trustworthiness, although it can involve possible network disruptions. This protocol maintains reliable communication through message delivery and ensures delivery primitives involving at-most-once, at-least-once, and exactly once. It needs a trustworthy transport protocol that explains its use of the Transmission Control Protocol (TCP) for message exchange. (Hassan et al., 2020)

The AMQP standard defines the encoding of primitive and composite data types, a complex transport mechanism, and a messaging system (that is, the structure and semantics of the messages carried by the transport mechanism). Furthermore, the AMQP specification defines how TLS and SASL can be used to authenticate and encrypt communications. Encryption enables the secure representation of binary data and clarifies data types to support application requirements. The transfer mechanism enables multiple parallel communication sessions to be opened within a single AMQP connection between two processes. Any number of one-way links can be established within a single session. This link is used for tire transport. AMQP can hash a large message, moving it with subsequent frames. (Wytrebowicz, Cabaj & Krawiec, 2021)

3.2.2.4 DDS

Data Distribution Service (DDS) is an open standard middleware designed by OMG that provides real-time communication through the publish–subscribe message pattern. (Wytrebowicz, Cabaj & Krawiec, 2021)

DDS is used for real-time and industrial M2M communications, running over both TCP and UDP. DDS supports broker less architecture, using a publish-subscribe model for interaction between entities without a broker. The tasks of a broker are handled by Data Writers (DW) and Data Readers (DR). The main advantages of the DDS protocol are that the data usage is fundamentally anonymous, since the publishers do not inquire about who consumes their data, and the probability of system failure is limited (the system is more reliable) because there is no single point (no broker) of failure for the entire system. The most remarkable disadvantage of DDS is that it is designed for industrial applications (IIoT) with considerable hardware resources. This makes the implementation for constrained devices that need a lightweight protocol even harder. Another disadvantage is the increased communication workload caused by data publishing, even if there are no interested subscribers (Hmissi & Ouni, 2022).

3.2.2.5 OPC UA

Open Platform Communications (OPC) Unified Architecture (UA) is a machine-to-machine communication protocol for industrial automation developed by the OPC Foundation. (Kirdan et al., al,2023)

OPC is a protocol standardized by IEC 62541 with a server-client architecture, the result of the automation sector collaborating in providing device information to the application without access to the device model and performance based on the Microsoft Component Object Model in 1994. OPC aims to maintain interoperability with other operating systems through a unified architecture called OPC UA. The OPC UA security architecture equips the authentication, authorization, confidentiality, integrity, auditability, availability, transport, communication, and application layers. While TLS ensures encryption for the HTTP protocol at the transport layer, the communication layer provides the secret messages and integrity. The application layer handles user authentication and authorization during the session. OPC UA also supports the publish-subscribe model to implement fewer session transactions using UDP packets. This architecture is more standardized than others, so it can be applied to all types of IoT applications because it includes the features required for applications developed in the IoT field. (Tightiz &Yang ,2020). The following figure represents a summary of the characteristics of the agricultural IoT protocols:

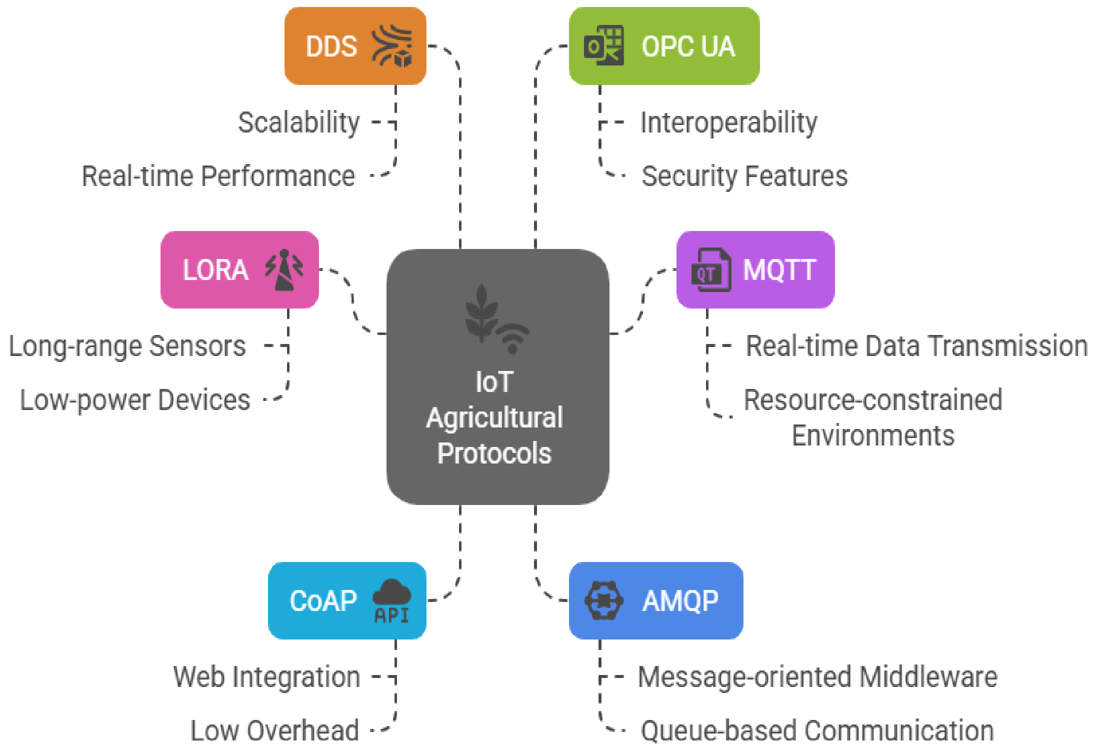


Figure 3.2: Agricultural IoT Protocols

3.2.3 Comparisons :

3.2.3.1 Comparison between IP-based Networks IoT Protocols:

The following table presents a comparison of IP-based Networks IoT Protocols based on a set of characteristics that aid in selecting the most suitable protocol for the system. (Qays et. al,2022), (Bayılmış et al ,2022), (Kirdan et al,2023)

Table 3-2: Comparison Between IP-based Networks IoT Protocols

Feature/ protocol	**MQTT**	**COAP**	**AMQP**	**DDS**	**OPC UA**
Quality Level	Three: QOS0,1,2	Tow: CON or NON	Yes	Yes	No
Data Security	TLS, SSL	DTLS	TLS, SSL	SSL, DTLS	SSL
Transport Layer	TCP	UDP	TCP	TCP, UDP	TCP
Complexity	Low	Low	Low	High	High
Message Prioritization	No	Yes	Yes	Yes	No
Message Pattern	Pub-Sub	Req-Res, Pub-Sub	Req-Res, Pub-Sub	Pub-Sub	Req-Res, Pub-Sub Push-Pull
Extensibility	Yes	Yes	Yes	Yes	Yes
Real-time	No	No	No	Yes	Yes
Port	1883,8883	5683	5671/ 5672	7400, 7410, 7411, 54461, 59992, 59668	4840
Message Size	Up to 256 Mb	Usually 1024 Kb or less	Undefined	Undefined	Undefined
Architecture	Client/Brok er/ Subscriber	Client/Serv er	Client/Broke r/ Subscriber, Client/ Server	Client/ Subscriber	Publish/ Subscribe Client/Server

The table represents a comparison of several communication protocols-MQTT, CoAP, AMQP, DDS, and OPC UA-in several respects. The MQTT, CoAP, and AMQP protocols are lightweight due to their low complexity, making them best for IoT applications, while the DDS and OPC UA protocols are highly complex, but they provide advanced functionality for large-scale or industrial systems. MQTT offers three levels of quality of service, providing secure message delivery, while CoAP, based on its UDP-based transmission and prioritization capabilities, is ideal for limited devices. Both AMQP and DDS prioritize messages and support strong security over TLS/SSL, while DDS offers additional flexibility thanks to its support for TCP and UDP. The protocols listed in the table provide different types of messages: MQTT and OPC UA use the publish-send method, CoAP and AMQP support the request-response (Req-Res) and publish-send methods, and DDS combines push and pull operations alongside publish-send. All previous protocols are scalable and support security, while OPC UA lacks quality of service and prioritization, making it less flexible in some applications. We conclude from the above that the choice of protocol depends on the applications used, such as lightweight processes (MQTT or CoAP), enterprise messaging (AMQP), or advanced real-time systems (DDS).

The table lists a set of technical terms that play a prominent role in achieving the best results in terms of performance, security, and reliability. One such term is CON/NON, which operates within the CoAP protocol, where CON (confirmable) means that a message needs confirmation to ensure it arrives, while NON (non-confirmable) is sent without waiting for a response and is used in applications that prefer speed over reliability, such as sending fast sensor data.

The security terms TLS (Transport Layer Security) and SSL (Secure Sockets Layer) are encryption protocols that typically work with TCP to ensure data confidentiality and integrity during transmission, while DTLS (Datagram TLS) is the modified version to work with UDP, where the same security is available with less consumption and resources, which suits devices with limited capabilities. IoT protocols show clear differences when compared in terms of real-time, virtual port, message size, and topology. In terms of immediate support, DDS and the newer OPC UA protocol (when combined with TSN) are best suited for industrial systems that require an accurate and immediate response, unlike protocols such as AMQP, MQTT, and CoAP that lack a strict time guarantee. As for the default port, the ports used in previous protocols differ, which may affect configuration flexibility and network security. In terms of message size, protocols such as DDS, OPC UA, and AMQP support large sizes, making them

suitable for complex industrial applications, while protocols such as CoAP and MQTT restrict size to provide efficiency on low-power devices.
The architecture also differs: protocols such as MQTT and AMQP often rely on a client/intermediary/subscriber model, while CoAP, DDS, and OPC support UA; AMQP shares client/server models, and OPC UA also uses a deployment/subscription model while providing great flexibility in integrating distributed systems. Finally, TCP is a reliable transmission protocol that delivers data in the correct order and is used in protocols such as MQTT and AMQP. While UDP is an unreliable but fast and lightweight protocol, it is used in CoAP and DDS when speed is more important than reliability. These three characteristics play a clear role in selecting the most appropriate protocol for the system's nature. TCP and TLS-based protocols are best for applications that require security and accuracy, while UDP, DTLS, and NON are preferred in lightweight and high-speed systems such as sensor networks.

3.2.3.2 Comparison between MQTT, CoAP, and OPC UA Performance

The following table provides a comprehensive comparison between MQTT, CoAP, and OPC UA protocols in terms of response time, power consumption, and packet loss rate. (Silva et al.2022)

Table 3-3: MQTT, CoAP, and OPC UA Performance Comparison

Protocol	**Response Time (ms)**	**Energy Consumption (mJ per message)**	**Packet Loss Rate (%)**	**Notes / Context**
MQTT	50 – 200	5 – 15	1 – 5	Lightweight publish/subscribe protocol; performs well in low-bandwidth networks; depends on QoS level (0,1,2).
CoAP	20 – 100	2 – 8	2 – 6	Designed for constrained devices; UDP-based; lower latency than MQTT for local networks.
OPC UA	100 – 300	10 – 25	1 – 4	Industrial IoT protocol; robust messaging with transaction support; latency depends on network and security mode.

The table reflects a performance comparison between three common protocols in the Internet of Things (IoT) environment, namely MQTT, CoAP, and OPC UA, in terms of response time, power consumption, and packet loss rate. Analysis shows that CoAP has the lowest response time (20–100 ms), making it suitable for restricted devices and local environments with limited resources due to its reliance on UDP and lightweight message exchange. On the other hand, the MQTT protocol exhibits an average response time (50–200 ms), with an average power consumption of between 5 and 15 mJ per message, and its performance depends largely on the level of quality of service (QoS) used, making it suitable for low-bandwidth networks and flexible in deployment/subscription patterns. The OPC UA protocol, designed for industrial environments, is characterized by high flexibility and reliability in exchanging messages while supporting transactions, but it records the highest response time (100–300 milliseconds) and energy consumption (10–25 millijoules), and its performance depends largely on network conditions and the way the connection is secured. As for the packet loss rate, the results showed that all protocols maintain low rates ranging from 1% to 6%, with slight variation depending on the nature of the network and the size of the messages, indicating that these protocols are able to ensure acceptable reliability in data transmission in the IoT environment. In general, the table reflects the required balance between time efficiency, power consumption, and reliability when choosing the appropriate protocol depending on application requirements and hardware limitations.

3.2.4 Security Protocols and Encryption Technologies

This section includes a table that links each protocol to the best encryption type that suits it. (Mrabet et al,2020)

Table 3-4: Appropriate Encryption Method For Each Protocol

Security Protocol	**Encryption Type**	**Cryptographic Algorithms Used**	**Purpose / Usage**
LoRaWAN	Symmetric + Asymmetric	AES-128 (CTR, CMAC)	Ensures confidentiality, integrity, and authentication between end devices and network/application servers.
MQTT over TLS	Symmetric + Asymmetric	AES-256-GCM, ChaCha20-Poly1305, RSA, ECDSA	Secures IoT message exchange via brokers; provides encryption, integrity, and mutual authentication.
CoAP over DTLS	Symmetric + Asymmetric	AES-128-GCM, RSA, ECC	Protects lightweight IoT communications; ensures confidentiality and integrity in constrained networks.
HTTPS (TLS)	Symmetric + Asymmetric	AES-256-GCM, ChaCha20-Poly1305, RSA, ECDHE	Provides secure communication between clients and web/cloud servers; ensures confidentiality and authentication.
Zigbee 3.0	Symmetric	AES-128 (CCM)	Protects home automation networks, ensuring data confidentiality and authentication.
Bluetooth Low Energy (BLE)	Symmetric	AES-128 (CCM)	Secures device pairing and data transfer in personal area networks.
Sigfox	Symmetric	AES-128 (ECB)	Provides data encryption and authentication in ultra-narrowband IoT systems.
5G IoT	Symmetric + Asymmetric	AES-256-GCM, ChaCha20-Poly1305, ECC	Protects data and signalling in modern cellular IoT networks; supports high-speed secure communication

The table presents a comprehensive overview of the primary security protocols employed in Internet of Things (IoT) systems, highlighting the type of encryption, specific cryptographic algorithms, and their intended purposes. A combination of symmetric and asymmetric encryption is predominantly used for protocols that require robust security across networked and cloud-connected devices, such as LoRaWAN, MQTT over TLS, CoAP over DTLS, HTTPS, and 5G IoT. In these cases, asymmetric algorithms like RSA, ECDSA, and ECC facilitate secure key exchange and authentication, while symmetric algorithms such as AES (in CTR, GCM, or CCM modes) and ChaCha20-Poly1305 provide efficient and fast encryption of data payloads. In contrast, protocols designed for constrained or local networks, including Zigbee 3.0, Bluetooth Low Energy (BLE), and Sigfox, primarily rely on symmetric encryption alone, reflecting a balance between security and computational efficiency in resource-limited environments. AES-128 emerges as the standard symmetric cipher in most lightweight protocols, whereas higher-strength symmetric encryption (AES-256) is preferred in high-speed or cloud-connected systems to ensure stronger confidentiality.

Overall, the table underscores the critical role of selecting appropriate cryptographic strategies tailored to the protocol's operational context, balancing the need for data confidentiality, integrity, and authentication, and the computational and energy constraints inherent to IoT devices. This layered approach to encryption design ensures that IoT systems maintain security across diverse communication scenarios without compromising performance.

3.3 Securing Methods:

In this section, we review cyberattacks against the agricultural Internet of Things (IoT), and then we review a range of the most important communication security techniques used in this field.

3.3.1 Attacks faced by the Agricultural Internet of Things :

Data in the agricultural IoT sector is under attack, like other IoT sectors. We will mention in this item the most famous cyber-attacks on the agricultural IoT, such as:

3.3.1.1 Denial-of-Service (DoS) Attacks

A Denial of Service (DoS) attack is defined as a cyberattack originating from the internet in order to render a server inaccessible to legitimate users. (Gupta et al., 2020)

Attacks have emerged as one of the most prevalent and disruptive threats in the realm of cybersecurity. These attacks aim to flood the network or application with many requests to make the network service, application, or website unavailable to its original users. As businesses and organizations increasingly rely on digital platforms for their operations, the impact of DoS attacks can be devastating, resulting in financial losses, damaged reputations, and loss of customer trust. The sophistication of these attacks has grown significantly over the years, leading to more complex variants, including Distributed Denial of Service (DDoS) attacks, where multiple compromised systems are used to launch an assault on a single target. The rise of the Internet of Things (IoT) and cloud computing has exacerbated the threat landscape, providing attackers with a broader attack surface. The increasing number of connected devices often lacks adequate security measures, making them easy targets for exploitation. Consequently, cybercriminals can leverage these devices to orchestrate large-scale DDoS attacks, resulting in overwhelming traffic that can cripple even the most robust systems. (Qureshi,2024)

3.3.1.2 MITM

A Man-In-The-Middle (MITM) attack is the oldest type of cyberattack, in which a malicious person is secretly involved in a communication between two parties. This attack allows the victim's data to be read and modified. The attacker achieves this by establishing a pseudo-secret connection between his computers and the victim's devices. In Wi-Fi zones, attackers direct network traffic to pass through themselves. Thus, people's traffic on that network begins to flow through the attacker. An attacker who hijacks this traffic can get personal data or passwords from here. (Aslan et al., 2023)
Man-in-the-middle attacks include different techniques, depending on the threat model. For example, Secure Sockets Layer (SSL) hijacking is an attack in which a man in the middle intercepts a request from the client to the server and then goes on to establish an encrypted connection. (Wlazlo et al., 2021)

3.3.1.3 Firmware Exploits

Regarding IoT device assaults in smart cities, firmware exploits pose a serious risk. "Firmware" is the term for the low-level software that provides the necessary operating system and functionality integrated into the hardware of IoT devices. Manufacturers of IoT devices should use tamper-resistant bootloader designs, frequent firmware upgrades, and secure firmware development procedures to reduce firmware attacks. To lower the possibility of successful firmware-based attacks, users should be urged to keep their IoT devices updated with the most recent firmware updates (Alotaibi et al., 2025).

Examples of this type include fake firmware updates that were previously used in the medical field. Fake firmware in medical devices poses many threats to healthcare devices, as an attacker can access devices and manipulate applications using fake copies of them. Counterfeit firmware is produced and distributed in a way that appears to be authentic. (Newaz et al., 2021)

3.3.1.4 Protocol Attacks

Protocol attacks target the channels and communication protocols used by IoT devices to communicate with one another. Attackers can interfere with, alter, or intercept the data flow by breaching these channels, which could result in several harmful effects. Attackers use sniffing as a method to listen in on network activity. Attackers can use this technique to intercept and examine data sent back and forth between IoT devices. Data interception during transmission is one of the security hazards that may result from this (Alotaibi et al., 2025).

These attacks exploit weaknesses in network protocols to disrupt their services. These attacks affect servers, firewalls, and load balancers, which drain the resources of this network. An example of these attacks is Ping of Death, as these attacks send huge data to the target, which exceeds the buffer capacity and disrupts the system. The job. (Qureshi, 2024)

3.3.1.5 Data Attacks

The term “data attack” in the context of the IoT refers to a variety of malicious actions intended to jeopardize the availability, confidentiality, integrity, and general security of data inside IoT systems. The IoT is a network of interconnected smart devices that gather, share, and analyze data, making them vulnerable to various data-related attacks. Some common data attacks in IoT include

Data Interception and Eavesdropping: Interception attacks are considered a development in prefix hijacking techniques, unlike traditional hijackings that only redirect data traffic, while interception attacks work to divert data traffic and then redirect it, ensuring that data traffic continues towards its intended target, maintaining network connectivity and functions while monitoring or manipulating data. The interception attack is characterized by the attacker's ability to intervene as a secret intermediary in the communication path between the source and its intended destination. (Scott, Johnstone & Szewczyk, 2024)

Data Tampering: The data exchanged between IoT devices and servers is subject to alteration by malicious actors. In this case, sensor readings or any other type of data transfer may need to be changed. An example of this is manipulating temperature sensor readings in an industrial environment, causing wrong decisions to be made based on unreliable data. (Sasi et al., al,2024).

Data Falsification: A data falsification attack within the context of the IoT pertains to a malicious act in which an attacker deliberately modifies or manipulates data inside an IoT system to deceive, disrupt, or compromise its functioning, integrity, or security. Attackers introduce fabricated information into the system by capitalizing on weaknesses or actively changing device inputs. This phenomenon can potentially result in erroneous decision-making processes founded upon facts of questionable reliability. In this attack, an attacker can inject falsified data instead of actual data using the captured node to transmit fake data to IoT applications. Upon receiving erroneous data, an affected IoT "application" could yield malicious commands to execute erroneous services, resulting in the degradation of IoT systems. (Rao& Deebak, 2022).

The following figure represents a summary of the characteristics of the agricultural IoT attacks:

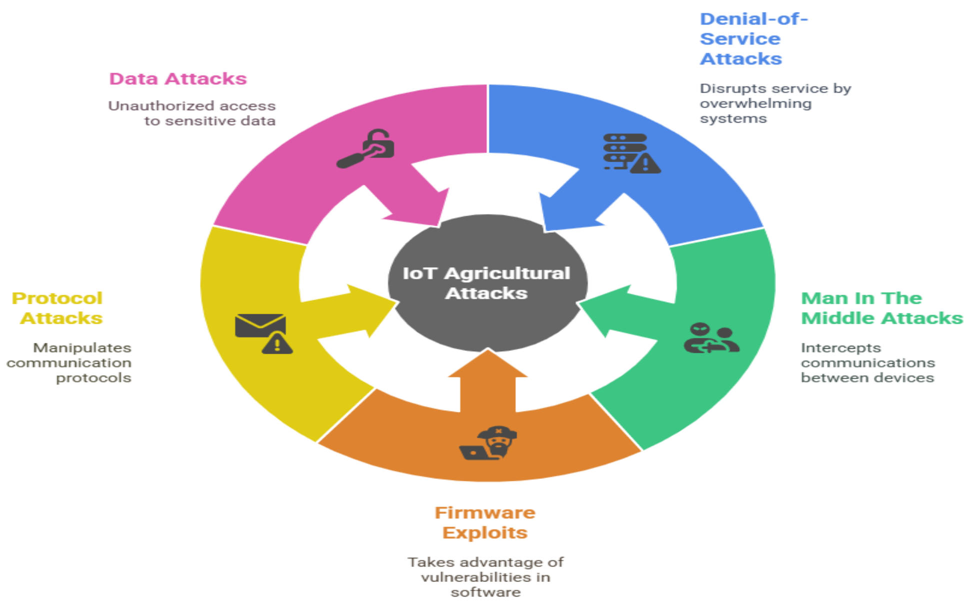

Data Attacks
Unauthorized access to sensitive data
Denial-of-Service Attacks
Disrupts service by overwhelming systems
IoT Agricultural Attacks
Protocol Attacks
Manipulates communication protocols
Man In The Middle Attacks
Intercepts communications between devices
Firmware Exploits
Takes advantage of vulnerabilities in software

Figure 3.3: Agricultural IoT Attacks

3.3.2 Securing Techniques Proposed for Agricultural IoT:

To preserve the transferred data, secure and effective security methods must be used. In this part, we review the most important IoT data security technologies.

3.3.2.1 Authentication:

To preserve the transferred data, secure and effective security methods must be used. In this part, we review the most important IoT data security technologies.

Authentication is the initial step in establishing a secure system, protecting all devices from both external and internal threats. All devices participating in the network must be able to identify and measure the validity of data sent or received by any of the participants in the IoT environment. The participant must possess sufficient computing power and the ability to decide whether to accept or reject incoming data, and this must be done in a highly secure manner while maintaining the privacy of the network and the rest of the participants (Brahmananda & Vairagade, 2020).

During the past decade, much research has been conducted to address security challenges in the Internet of Things environment and develop methods of protection from attacks. Authentication has been a key focus in these studies, as strong authentication mechanisms work to reduce the risks of unauthorized access to Internet of Things networks, as attackers can impersonate other objects, eavesdrop on sensitive information, and destroy trust relationships between them. Due to the limited computing power and power resources of IoT devices, it is difficult to implement complex authentication protocols, and the diversity of devices and communication protocols in IoT networks adds another level of authentication complexity. (Saideh, Jamont & Vercouter, 2024) The following figure shows tools and techniques for IoT authentication in smart cities. (Alotaibi, Aldawghan & Aljughaiman,2025)

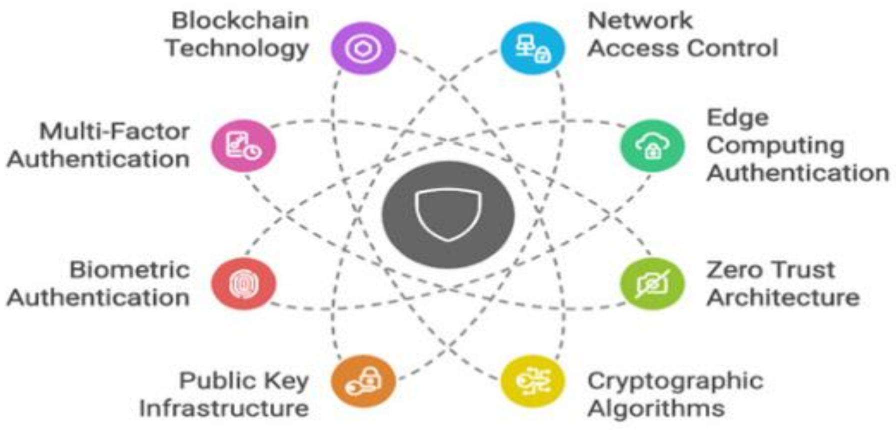


Figure 3.4: IoT Authentication Tools

3.3.2.2 Cryptographic Algorithms

Cryptography: A term derived from two Greek words, "kryptos," meaning "hide," and "graphein," meaning "write," it is the art and science of hiding data from unauthorized users during storage and transmission. The main goal of encryption technology is to convert data into an unreadable form so that only an authorized user can access it, and in no case should an attacker have any opportunity to access the database server or hide the data. Encryption is carried out in two different stages: encoding and decoding. Both encryption and decryption are important aspects of security. In the encryption process, the plaintext, or secret message, is converted into a foreign, or encrypted, message known as a ciphertext with the help of the secret key on the sender's side. At the receiver, upon decryption, the ciphertext is returned to its original form using the same secret key used in the encryption process. (Kumar et al., 2021)

With the increasing use of data sharing and online communication, securing data from cyberattacks has become crucial. Nowadays, providing data confidentiality and privacy represents a major challenge for cybersecurity researchers and specialists. Data confidentiality means protecting it from unauthorized access or theft. This can be achieved with the help of encryption by encrypting and decrypting data. Encryption aims to secure important data or documents on the hard disk or when transferred over unsecured communication channels. (Alenezi, Alabdulrazzaq & Mohammad, 2020)

Many methods of cryptographic algorithms have been produced, which will be discussed later.

3.3.2.3 Blockchain Technology

A blockchain is a distributed database that stores an ever-increasing list of organized records, called blocks, that are linked together using encryption. Each block includes several fields: an encrypted portion of the previous block, a time setting, and data for the elements within the block. These blocks are used to record transactions in a decentralized public digital ledger. These blocks are considered interconnected, as records cannot be changed without changing the compatibility of all subsequent blocks and networks. (Guru et al., 2023)

Blockchain technology can be effectively applied in almost all areas of IoT, including IoT-based green agriculture. Blockchain technology is applied in the Internet of Things to provide privacy, where it is used to share encrypted data. Therefore, blockchain can be used in IoT-based green agriculture as a distributed digital ledger containing all messages. This distributed ledger is replicated and stored in various IoT nodes in the agriculture sensor layer. **(**Ferrag et al., 2020)

3.3.2.4 Secure Communication Protocols

The research community and industry now have to face a set of challenges in designing and deploying network security protocols caused by standardization and operation of the Internet of Things (IoT) network package. The low power consumption and limited computational capabilities of many IoT devices make designing and integrating these protocols difficult. There are currently many well-established and new security systems in the Internet of Things to protect application traffic. There is also a new set of object security protocols that aim to save energy consumption and improve encryption systems, and developers can also use well-known transport layer security protocols: Transport Layer Security (TLS) and Datagram Transport Layer Security (DTLS), which are considered mature protocols and enjoy widespread support. **(**Claeys et al., 2021)

The TLS protocol provides communications security by using a transport-level encryption package over the TCP/IP stack. This package consists of encryption algorithms that enable key exchange and encryption. TLS applies transport layer encryption between two directly connected devices. Using TLS with MQTT requires a reliable broker, since the broker can access all messages. This is because when used with MQTT, it does not guarantee comprehensive encryption between publishers and subscribers, but rather guarantees encryption only between the publisher and the broker or the broker and the subscriber. TLS can prevent data, such as usernames and passwords, from being transmitted in plain text, making it difficult to hack IoT devices. These applications often overlook MQTT's three Quality of Service (QoS) levels, completely ignoring different network conditions. (Prantl & Krupitzer, 2021)

The following table represents a summary of the agricultural IoT security techniques:

Table 3-5: Agricultural IoT Security Techniques Summary

Security Technique	Effect on Agricultural IoT
Secure Communication Protocols	Ensure safe data transmission between IoT devices, reducing risks of interception and data breaches. Examples include TLS, DTLS
Blockchain Technology	Enhances data integrity and transparency by creating a tamper-proof record of transactions. Useful for securing sensor data, supply chains, and automated smart contracts.
Cryptographic Algorithms	Provide data encryption, preventing unauthorized access. Algorithms like AES, RSA, ECC, and SHA ensure the confidentiality and integrity of farm data.
Authentication Protocols	Verify device identities and prevent unauthorized access. Examples include OAuth, Kerberos, and Zero Trust models, helping protect smart farming networks from cyber threats.

3.4 Cryptographic Algorithms

3.4.1 Classic Encryption Algorithms :

Classical cryptography relies primarily on the complexity of mathematical operations. The security of classical encryption methods depends on the computational challenge of analyzing large amounts of data (Tripathi et al., 2024).

3.4.1.1 Symmetric Encryption Algorithms

Symmetric key technology is common because encryption and decryption are done using the same key, called the private key, also known as the secret key. We need a secure channel to share this private key between the sender and the receiver. Symmetric key encryption algorithms have two different types based on input data: block encryption and stream encryption. In block cipher systems, data is processed or encoded on a fixed-length bit set called a block, while stream cipher systems operate on a data stream where data is processed on a stream of bits. (Alenezi et al, 2020)

The following table shows some types of symmetric encryption with a comparison between their different characteristics (More,2025)

Table 3-6: Comparison Between Symmetric Encryption Algorithms

Feature/key	DES	TDES	AES	BLOWFISH	RC4
Block Size/bit	64	64	128	64	N/A
Key Size/bit	56	168	128,192,256	32-448	40-2048
Algorithm Structure	Fiestel Net-work	Fiestel Net-work	Substitution Permutation Network	Fiestel Net-work	Stream Cipher
Rounds	16	48	9,11,13	16	N/V
Attacks	Brute Force	Theoretically Possible	Side Channel	Not Yet	Key Scheduling

The table shows a comparison between five popular encryption algorithms, which are DES, TDES, AES, Blowfish, and RC4, in terms of a set of important characteristics that determine the security and efficiency of each algorithm. The comparison begins with block size, as DES, TDES, and Blowfish use a 64-bit block, while AES uses a larger 128-bit block, which provides higher security. RC4 is a stream encryption algorithm and does not rely on dividing data into blocks. As for key size, DES uses a short key (56 bits), which makes it weak, while TDES uses 168 bits, and AES provides three options (128, 192, and 256 bits), which makes it more flexible and secure, and Blowfish supports a wide range. Between 32 and 448 bits, RC4 supports up to 2048 bits.

Regarding the algorithm structure, DES, TDES, and Blowfish rely on the Feistel Network structure, which is a design structure that relies on dividing text into two halves, then applying a series of operations (such as XOR, shift, and switch) to one of the halves using a key, then switching The halves and the repetition of the process over several rounds make the algorithm flexible and can be used to encrypt and decrypt using the same structure (easy symmetry). However, it is not resistant to all modern attacks if sufficiently long keys are not used, whereas AES employs a more advanced architecture called the Substitution-Permutation Network, which combines data substitutions through S-boxes with permutation operations. Data locations, which provide a high level of security due to their great ability to disperse data and reduce the connection between the original and encrypted text, making AES resistant to mathematical and analytical attacks. As for RC4, it is a stream cipher, which is a type of encryption that does not operate on fixed blocks of data. Rather, it encrypts data one bit at a time using a series of bits known as the "keystream." A stream cipher is considered fast and is used in applications that require high speed or live streaming, but it is vulnerable to serious attacks if a "keystream" is not generated safely. The number of rounds also varies, with DES and Blowfish using 16 rounds, TDES repeating DES three times (48 rounds), AES using a variable number depending on the length of the key (9, 11, or 13 rounds), and RC4 not relying on traditional rounds.

The power of these systems varies in terms of the attacks they are exposed to and their ability to confront them. DES uses short keys, which exposes it to a brute force attack that tries all possible keys until the correct key is reached and can break the encryption in a reasonable time using powerful devices. TDES faces theoretical attacks, yet is considered safer. With the power of AES, it is vulnerable to side-channel attacks that extract keys by analyzing physical information such as the system's power consumption. Blowfish can be considered safe when used correctly, as no successful attacks have been recorded against it recently. As for RC4, it is vulnerable to attacks on its key-scheduling algorithm, which transforms the original key into subkeys for different encryption rounds. These vulnerabilities can allow an attacker to deduce parts of the original key or text, weakening the overall security of the algorithm.

Overall, AES is the most balanced in terms of security and efficiency, while DES and RC4 have fallen out of favor due to their known vulnerabilities. Blowfish and TDES are relatively secure alternatives, but they are less efficient than AES for certain modern applications.

3.4.1.2 Asymmetric Encryption Algorithms

Asymmetric key encryption systems use a pair of distinct keys: a public key and a private key. These systems provide a higher level of security, but they face several challenges. They are not suitable for large documents due to their slow speed compared to symmetric key systems, which consume more time. They also consume more power and place a higher load on the central processing unit (Alenezi et al., 2020).

The following table presents some types of asymmetric encryption, comparing their key characteristics (Sharma et al., 2021; Saho & Ezin, 2020).

Table 3-7: Comparison Between Asymmetric Encryption Algorithms

Feature/key	**RSA**	**ECC**	**ElGamal**
Block Size/bit	Variable	Variable	Variable
Key Size/bit	1024	160	1024
Algorithm Structure	Integer Factorization	Discrete logarithmic	Elliptical Curve
Rounds	1	1	1
Attacks	Cycle Attack	Side Channel	Side Channel

The table shows a comparison between three asymmetric encryption algorithms: RSA, ECC, and ElGamal, through a set of important properties. First, we note that the block size in these algorithms is variable; that is, it does not depend on a specific length of the data but rather adapts to the length of the key used. In terms of key size, the RSA algorithm requires keys of 1024 bits or more, whereas ECC uses a much shorter 160-bit key, making it well-suited for devices with limited resources. ElGamal also uses a 1024-bit key. In terms of algorithm structure, RSA is based on integer factorization, which means that it is difficult to factor a very large integer, which forms the basis of its power, with very long keys that make the factorization process difficult. In contrast, ElGamal relies on the Discrete Logarithm Problem, which means that it is difficult to find the exponential force in limited arithmetic groups, which increases security because this calculation is not possible to solve quickly at present, at the same time ECC works using Elliptic Curves, which relies on a more complex version of the discrete logarithm problem, but within mathematical sets based on elliptic curves. This provides higher security with greater efficiency using much shorter keys, making them ideal for resource-constrained systems such as phones and IoT devices. All of these algorithms are executed in one round to encrypt or decrypt.

Compared to attack types, RSA is vulnerable to cycle attacks that exploit recurring patterns in encryption or decryption to discover keys. If secure values are not used, an attacker may be able to expose keys or break encryption using analytical attacks based on these cycles. ECC and ElGamal are vulnerable to side-channel attacks, which do not target the algorithm itself but rather exploit physical information such as power consumption or encryption time to extract secret keys. We note that ECC excels in efficiency and security per key size, while RSA remains popular for its simplicity and widespread use. ElGamal is used in special applications that require a different encryption structure but are based on the same mathematical concepts.

3.4.1.3 Hybrid Encryption Algorithms

Hybrid encryption combines the features of symmetric and asymmetric encryption, leveraging the advantages of both systems. It provides a high level of security by ensuring that both the public and private keys are fully protected. The hybrid encryption process begins with generating a symmetric key and encrypting the actual data using it. This key is then encrypted using the receiver's public key through one of the previously mentioned asymmetric encryption algorithms. The packet is sent to the receiver, who decrypts the symmetric key and then uses it to decrypt the sent data. (Muhammed et al., 2024). The following table illustrates the combination of symmetric and asymmetric keys, along with their key properties (Muhammed et al., 2024; Joshi & Prateek, 2024).

Table 3-8: Combination Between Symmetric and Asymmetric Encryption Algorithms

Feature/key	**RSA+AES**	**ECC + AES**	**MKP (RSA+AES Blowfish)**
Block Size / bit	128/192/256	128/192/256	128/256 Blowfish: 64
Key Size/bit	RSA: 2048/4096; AES: 128/256	ECC: 160/521; AES: 128/256	RSA: 2048/4096; AES: 128/256 Blowfish: 448
Algorithm Structure	Uses RSA for key exchange and AES for bulk data encryption	ECC secures the key exchange; AES handles data encryption	Uses RSA for initial key exchange, followed by AES encryption for data transmission
Attacks	Vulnerable to quantum attacks (Shor's Algorithm on RSA); side-channel attacks on AES	Weak ECC curves can be exploited; side-channel attacks on AES.	Relies on RSA's security, which is vulnerable to quantum attacks.

The previous table compares a set of hybrid encryption technologies to key characteristics. When comparing block size, we find that both AES and Blowfish use blocks of different sizes: AES supports 128, 192, or 256 bits, while Blowfish uses a fixed block size. 64 bits. In terms of key size, RSA requires very long keys (2048 or 4096 bits) to ensure security, while ECC provides the same level of security using shorter keys (160 to 521 bits), making it more efficient. Blowfish can use a key up to 448 bits long, which gives it flexibility in some applications.

When comparing algorithms in terms of algorithm structure, we find that these systems use RSA or ECC to secure the key exchange, while AES or Blowfish encrypts the actual data. For example, RSA + AES uses RSA for symmetric key encryption, then AES is used to encrypt data efficiently, and ECC + AES uses ECC instead of RSA, which provides better performance in low-power systems. In MKP technology, RSA is used to exchange keys, and AES and Blowfish are used together to secure data transfer, which enhances security by diversifying encryption algorithms.

As for attacks, all of these systems are vulnerable to different attacks. The RSA system is vulnerable to quantum attacks that rely on quantum computing, which has a great ability to break some traditional encryption algorithms that are considered secure at the present time, such as Shor's algorithm, which is able to factor integers into their prime factors very efficiently using quantum computers. This is very dangerous because it directly threatens algorithms like RSA that rely on the difficulty of this analysis as the basis for their security. As for the AES system, it may in turn be vulnerable to side attacks such as timing attacks, which are a type of side channel attack where the attacker does not attack the algorithm itself directly but rather exploits the difference in the time it takes the algorithm to perform certain operations to deduce confidential information, such as secret keys or parts of encrypted data. While weaknesses of ECC curves may be exploited if not chosen carefully, this can weaken overall security. The MKP hybrid system relies on the power of RSA and the security of both AES and Blowfish and is therefore exposed to the same risks if not implemented properly. Finally, this comparison shows that hybrid systems offer a balance between security and efficiency, but they need good parameter selection and proper implementation to avoid vulnerabilities, especially under the growing threats from quantum computing and side-channel attacks.

3.4.1.4 Comparison Between the Classical Encryption Algorithms

The following table presents a comparison of classical encryption algorithms (Muhammed et al., 2024).

Table 3-9: Comparison Between the Classical Encryption Algorithms

Encryption Method	**Symmetric**	**Asymmetric**	**Hybrid**
Key Type	Single shared key	Pair (public and private keys)	shared and key pairs
Key Distribution	Securely sharing a single key can be challenging	Public keys can be freely distributed, and private keys must be kept secret	Securely share a symmetric key, then use asymmetric encryption for key distribution
Speed	Generally, faster for encryption/decryption	Slower compared to symmetric encryption	Slower than symmetric but faster than pure asymmetric
Security	Vulnerable if the key is compromised	More secure due to the key pair, but still vulnerable if private key is compromised	Combines the strengths of both symmetric and asymmetric encryption for improved security

This table presents the difference between three basic encryption methods symmetric encryption, asymmetric encryption, and hybrid encryption, in terms of key type, distribution, speed, and level of security. In symmetric encryption, a single shared key is used to encrypt and decrypt, making it very fast in terms of performance, but it has trouble distributing the key confidentially between parties, because any compromise of those key compromises the data.

We find that asymmetric encryption uses two keys, public and private, where the public key is published while the private key is used in decryption, which gives higher security but reduces the speed of performance. As for hybrid encryption, it is the best, as it provides a balance between security and efficiency, as it combines asymmetric encryption for key exchange. Such as RSA and symmetric encryption, such as AES, to encrypt data, which provides higher security and greater efficiency.

We conclude that hybrid encryption offers a balanced solution for modern encryption systems in terms of combining double-key security and symmetric encryption speed.

3.4.2 Post- quantum cryptography (considered as extended to classical) :

In 2015, the National Institute of Standards and Technology (NIST) launched a standardization process of post-quantum cryptography (PQC), aiming to create quantum-resistant public key algorithms (Kappler & Schneider, 2022).

Post-quantum cryptography (PQC) is a specialized field focused on designing cryptographic systems that maintain security against the immense computational power of quantum computers. Quantum computers can crack many common cryptographic algorithms, including RSA and ECC. Post-quantum cryptography (PQC) seeks to provide basic cryptographic tools, including encryption systems, digital signatures, and key exchange protocols, designed to withstand attacks from classical and quantum computers. Post-quantum cryptography (PQC) is essential for protecting sensitive data, such as financial transactions, communication channels, and data privacy, ensuring their long-term security with the advent of quantum computers. In addition, quantum cryptography protocols often use quantum key distribution (QKD) schemes, such as BB84, E91, and Kyber, which enable two parties to create a shared secret key with provable security guarantees. These keys can then be used for secure communication via traditional encryption techniques. (Tripathi et al., 2024)

3.4.2.1 Comparison:

The following table compares post-quantum cryptography and classical cryptography (Tripathi et al., 2024).

Table 3-10: Comparison Between Post Quantum Cryptography and Classical Cryptography

Feature	**Post quantum**	**Classical**
Basis	Quantum mechanics	Mathematical computation
Development	Infantile & not tested fully	Deployed and tested
Existing Infrastructure	Sophisticated	Widely used
Digital Signature	Not present	Present
Bit Rate	1Mbit/s average	Depend on Computing power
Cost	Crypto chip 100, 000	Almost zero
Register storage (n-bit) at any moment	One n-bit string	Millions of miles Requirements Devoted h/w & communication lines
Communication Range	10 miles max	Millions of miles Requirements Devoted h/w & communication lines
Requirements	Devoted h/w & communication lines	S/w and portable
Life expectancy	No change based on physics laws	Require changes as computing power increases
Medium	Dependent	Independent

This table compares post-quantum cryptography to classical cryptography across several technical and security aspects. Post-quantum cryptography is based on the principles of quantum mechanics, which is concerned with studying the behavior of very small particles such as electrons and photons at the level of atoms and molecules. Classical cryptography relies on complex mathematical calculations, such as number factorization or discrete logarithms. Post-quantum cryptography is still at an early stage of development and experimentation, while classical cryptography is widely tried and applied. The infrastructure required for quantum cryptography is complex and expensive, requiring encryption chips that can cost up to $100,000. "Cost" refers to the requirements for implementing quantum cryptography and not post-quantum cryptography, while classical cryptography is used at almost minimal costs, using only portable software.

Performance-wise, quantum cryptography offers a relatively low data rate (around 1 Mbit/s), compared to classical cryptography, which relies on available processing power. Quantum cryptography also does not yet explicitly support digital signatures, unlike classical cryptography, which supports them efficiently. As for range, quantum cryptography is limited to short distances (10 miles maximum) due to physical limitations, while classical cryptography can operate over unlimited distances. Also, in quantum cryptography, it is not possible to store more than one n-bit string at a given instant, unlike classical cryptography, which supports storing large amounts of data.

Finally, in terms of lifespan, quantum cryptography is not affected by developments in the laws of physics and, therefore, may be more stable in the long term, while classical cryptography needs periodic updates to adapt to increasing computing capabilities. Overall, quantum cryptography is promising in the future, but it requires specialized and complex infrastructure, while classical cryptography remains the most practical and widespread to date.

3.4.2.2 Kyber (Key Encapsulation Mechanism – KEM)

3.4.2.2.1 Overview

CRYSTALS-Kyber is the only system selected by the National Institute of Standards and Technology (NIST) for standardization to date. Kyber is a network-based KEM system that was selected as the PQC standard in July 2022. Its security relies on the Module Learning With Errors problem in modular lattices (the MLWE problem).

Kyber consists of a secure IND-CPA PKE used for non-discrimination under an adaptive chosen ciphertext attack (IND-CCA2). Shoup proposed the first KEM in 2000. It defines it as a special use case for PKE where the encryption algorithm does not need any input other than the recipient's public key, as it automatically generates a random bit string of a specified length. The encrypted bit string is transmitted to the recipient, who can now decrypt it using his secret key. At this point, only the sender and recipient know this random string of bits, known as the shared secret, which can be used in a symmetric encryption system such as AES (Segatz & Al Hafiz, 2022).

3.4.2.2.2 Kyber Standard Characteristics

In the following table, we review the basic standard characteristics of Kyber algorithms: 512, 768, and 1024 (Demir, Bilgin & Onbaşlı, 2025).

Table 3-11: Kyber Standard Characteristics

Metric	**Kyber 512**	**Kyber 768**	**Kyber 1024**
Key Generation Time (ms)	35	58	89
Encryption Time (ms)	40	63	92
Decryption Time (ms)	52	80	113
Key Size per Byte	Public: 800 Private: 1632 Ciphertext Size: 768	Public: 1184 Private: 2400 Ciphertext Size: 1088	Public: 1568 Private: 3168 Ciphertext Size: 1568
Shared Secret	32	32	32
Maximum load size / Byte	768 B	1088	1568

The table illustrates the differences among the three Kyber algorithms (512, 768, and 1024), which gradually vary in performance and size according to the level of security they provide. Regarding performance, key generation time increases from 35 ms in Kyber-512 to 89 ms in Kyber-1024, representing an increase of approximately 154%. A similar trend is observed for encryption and decryption times, which increase from 40 ms and 52 ms in Kyber-512 to 92 ms and 113 ms in Kyber-1024, respectively. This means that raising the level of security results in a greater time cost in all operations.

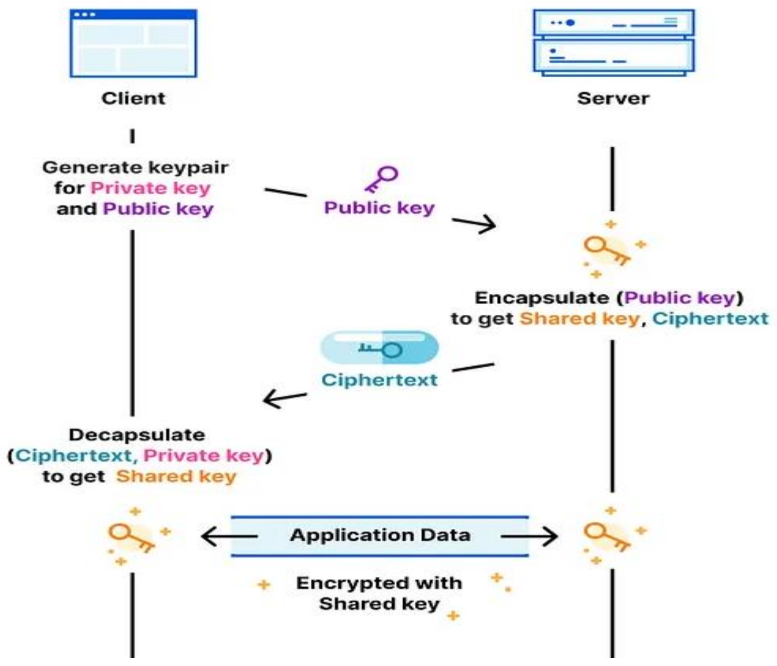
Regarding key sizes, each successive Kyber level shows a clear increase. The public key size rises from 800 B in Kyber-512 to 1568 B in Kyber-1024, while the private key increases from 1632 B to 3168 B, and the ciphertext size increases from 768 B to 1568 B. This increase in volumes affects data transfer and storage in resource-limited systems. At each level, the maximum payload is approximately equal to the corresponding ciphertext size. The maximum payload (maximum load size) is approximately equal to the corresponding ciphertext size at each level. It reflects that Kyber is not intended for transferring a large amount of data directly, but for exchanging session keys.

In the end, we conclude that the Kyber-512 is the fastest in terms of performance and the best in transmission rate, very suitable for systems with limited resources, while the

Kyber-768 offers a strong balance between speed and security, and we find that the Kyber-1024 achieves high security but comes with a larger data volume and time costs.

3.4.2.2.3 Kyber Encapsulation/Decapsulation Mechanism

The mechanism of Kyber encryption is based on key exchange, as shown in the following image (Cambou et al., 2021).



Client
Server
Generate keypair for Private key and Public key
Public key
Encapsulate (Public key) to get Shared key, Ciphertext
Ciphertext
Decapsulate (Ciphertext, Private key) to get Shared key
Application Data
Encrypted with Shared key

Figure 3.5: Key Encapsulation / Decapsulation Mechanism

The image explains the key encapsulation mechanism (Key Encapsulation Mechanism - KEM) used in post-quantum encryption algorithms such as Kyber. The process starts from the client (Client), which generates a key pair: a public key (Public key) and a private key (Private key). The public key is sent to the server (Server), which uses it in the encapsulation process (Encapsulation) to generate a shared key (Shared key) and generate the ciphertext (Ciphertext). The server sends the ciphertext to the client. Next, the client performs the unpacking process (Decapsulation) using the ciphertext and his private key to obtain the same shared key. After these steps, the shared key is used to encrypt the data exchanged between the two parties (Application Data) to ensure confidentiality and security. This mechanism provides strong security against quantum attacks because it relies on exchanging a shared key without directly detecting it, reducing the risk of the key being intercepted during transmission.

3.4.2.2.4 Random Function

The security of Kyber encryption relies on a random function that generates new, unpredictable values each time. Its most critical application is in key generation, where the function produces seeds that expand to create the public and private keys. These keys are unique for each transmission, even when the same parameters are used. This function is also used in the stage of generating random elements within the ciphertext, which makes it impossible for attackers to infer the original text or private keys even if pattern analysis attacks are used, which ensures the confidentiality and quantum power of the keys and protects the system from unauthorized data prediction or retrieval, which ensures the security of encryption and transmission in post-quantum systems. (Zhao et al., 2024)

One of the most famous examples of random generation functions is the esp_fill_random (void *buf, size_t len) function. This function calls a real random number generator that is integrated with ESP32 processors that use buf memory to fill it with a number of bytes depending on the length of the memory. This generator works on thermal noise in the radio units inside the chip as a source of randomness (entropy), where unpredictable electrical oscillations are measured and converted into random bits. These bits are then stored in special registers, and the function reads the values from these registers and distributes them into the required memory. Before writing, the system applies verification and correction algorithms to ensure that bits are free of bias and ensure their quality, making the output suitable for security uses such as key generation, encryption, or unpredictable values in protocols. (Segatz & Al Hafiz, 2022)

3.4.2.3 Comparison

The following table presents the characteristics of Kyber compared to AES and ECC. (Escribano Pablos et al., 2022)

Table 3-12: Comparison Between AES, ECC, and Kyber

Feature/key	**AES**	**ECC**	**Kyber**
Block Size / bit	128	Variable	Use: Key Encapsulation Mechanism (KEM)
Key Size/bit	128,192,256	160	--- Kyber 512: Public key: 800 bytes Secret key: 1632 bytes --- Kyber768: Public key: 1184 bytes Secret key: 2400 bytes --- Kyber1024: Public key: 1568 bytes Secret key: 3168 bytes
Algorithm Structure	Substitution Permutation Network	Discrete logarithmic	Module Learning With Errors (MLWE)
Rounds	9,11,13	1	Use these steps instead of rounds: ✓ Key Generation ✓ Encapsulation ✓ Decapsulation
Attacks	Side Channel	Side Channel	Side-Channel Attacks Randomness Reuse Attacks

The previous table compares three encryption algorithms: AES, ECC, and Kyber. The AES algorithm uses the Substitution-Permutation Network method and works with a different number of rounds depending on the length of the key used (9 rounds for 128-bit, 11 for 192, and 13 for 256). It is considered one of the most important symmetric encryption algorithms. While ECC works using the discrete logarithm method and uses a 160-bit key with one round of calculations, Kyber differs in terms of structure depending on the type adopted, and relies on the Module Learning with Errors (MLWE) algorithm, which is a post-quantum algorithm that does not use traditional rounds, but rather relies on Three steps: key generation, encapsulation, and decapsulation.

In terms of key size, Kyber is more complex than ECC, AES, as the sizes of public and private keys vary according to the public and private security level (Kyber512, Kyber768, Kyber1024) and are measured in bytes, not bits In terms of security, the previous algorithms are exposed to side-channel attacks, while Kyber is also exposed to randomness reuse attacks, which are a type of attack that exploits the reuse of the same random values in encryption operations, which requires precise key management and randomness.

The Comparison table indicates that Kyber is a promising option for a quantum computing-proof future, while remaining within the capability of traditional cryptography, while AES and ECC remain current pillars in classical cybersecurity.

3.5 Conclusion

This chapter describes Internet of Things protocols, security systems, and cyberattacks on computer communications systems. It details communication protocols and encryption mechanisms to ensure data security. The next chapter introduces the experimental systems used to evaluate the characteristics of the proposed systems and describes the design and operational steps of the final system based on MQTT/LoRa.

Chapter 4: System Design

4.1 Introduction

Irrigation systems based on Internet of Things technology are an effective solution to water waste and the scarcity of resources needed for plant irrigation. Therefore, a smart irrigation system is an effective tool that enables farmers to better monitor and manage water resources by providing accurate and timely information about soil conditions and water needs. This system also helps farmers improve irrigation processes and reduce waste, which contributes to increasing agricultural productivity and reducing costs.

We review four models through which data is transmitted, all of which check soil moisture through a moisture sensor and then send this value to an Arduino board using Kyber512 encryption—previously explained in Section 3.4.2.1—which relies on extracting a key and then using the AES system to encrypt through it.

We will explain the fields used in the different models and then review the results through two main criteria—variable size and performance metrics—where the results are reviewed first when integrating Kyber encryption with the MQTT protocol and second when integrating Kyber encryption with the LoRa protocol.

4.2 System Basic Information

4.2.1 Fields Used in Different Forms:

In all forms, we use a set of fields, some or all of which are published in each form, depending on the nature of the form and the degree of credibility in that form. These fields are

1: Public key: pk

2: Secret key: sk

3: Encrypted text: ct

4: Shared secret key: ss

5: Authentication fields: Tag, IV, HMAC

6: Humidity value: moisture

4.2.2 Variable Size

This section provides details about the memory rate used in the project in terms of total size and size of variables, all measured in bytes, and includes:

1: Project Size: The total program size is the size of the static code (firmware) that will be loaded into the Flash memory in the ESP32 module. It includes executable code, constants, text and character strings, and tables or static data.

2: Maximum flash size: The full capacity available to store the program in non-volatile memory (Flash Memory), and all experiments were conducted using the same piece, so the size appears constant in all models and is equal to 1310720 bytes.

3: Size of general variables: Total memory consumed by global variables and static variables. These values are stored in RAM and remain reserved throughout the program's run.

4: Local variable memory: Memory that is used only by variables defined within functions during their execution. These variables are created on the stack and are automatically released after the function exits execution. If the local variables are too large, they cause the program to stop working.

5: Maximum RAM: The total capacity available in RAM to store global variables, local variables, and heap (for dynamic allocation such as malloc/new), and all experiments were performed using the same chunk, so the size appears constant in all models and is equal to 327680 bytes.

4.2.3 Performance Metrics

In this section, we review a set of metrics used to evaluate the model’s efficiency. These metrics include:

1: Key generation time: It is the time it takes the system to generate the public/secret key pair, including calculating random values and mathematical matrices within the encoder. The less time, the faster and more efficient the system is, measured in milliseconds (ms).

2: Encapsulation time: The time required to encrypt data sent through the system, and includes key loading, IV generation, AES/GCM algorithm execution, and signature generation (tag). Measured in milliseconds (ms).

3: Decapsulation Time: The time required to retrieve the original data from the ciphertext, which shows how efficiently the device receives and decodes data in real time, measured in milliseconds (ms).

4: Key size in bytes: The number of bytes occupied by the public or secret key (public/secret key). The size of the public key was adopted in the results because it is what is transmitted through the system. The same type of encryption was used in all systems, so you find the size of the key fixed and equal to 800 bytes. The size of the key affects' memory and transfer speed.

5: Approximate power consumption estimates the power consumed by ESP32 during encryption or transmission execution. It is usually calculated by the equation:

Energy (mJ)= Current (A)×Voltage (V)×Time (s)×1000

The lower value gives higher energy efficiency.

6: Data transfer rate: It is the rate of data sent through the medium (broker) per second. It is calculated from the equation: Rate (Bps) = Data Size(Bytes)/Transmission Time (s) The higher the data transfer rate, the faster and more stable the MQTT connection.

7: Total response time: The period between sending the encrypted message and receiving the response, which includes network time (Wi-Fi latency), encryption and decryption time, and deployment and consumption time in MQTT, is used to evaluate the time of the entire system cycle. It has been adopted as a general evaluation because the message and reception take place in the same piece, but in reality, it cannot be calculated because the system cycle takes place in two different pieces. The times can be calculated separately.

8: Maximum data load size (Payload): The largest data size the system can send in a single message via MQTT, LoRa, or TLS, reflecting cache limits (Buffer Size).

9: Free Remaining Memory (Free Heap Memory): The amount of dynamic memory (RAM) available after all operations have been performed.

10: CPU Usage: The percentage of processor consumption at a given stage is measured by the time consumed by that stage divided by the total time * 100%

CPU Usage = (Stage Time / Cycle Time) * 100%

4.3 MQTT- Based Systems

This section presents the results of experiments conducted to transfer data over the medium using the MQTT protocol combined with Kyber encryption, which encrypts data with the principle of post-quantum encryption.

All of these models have been realistically implemented using a humidity moisture sensor and an ESP32. The actual results of these models have also been reviewed, and the average results are reviewed and presented in subsequent comparisons.

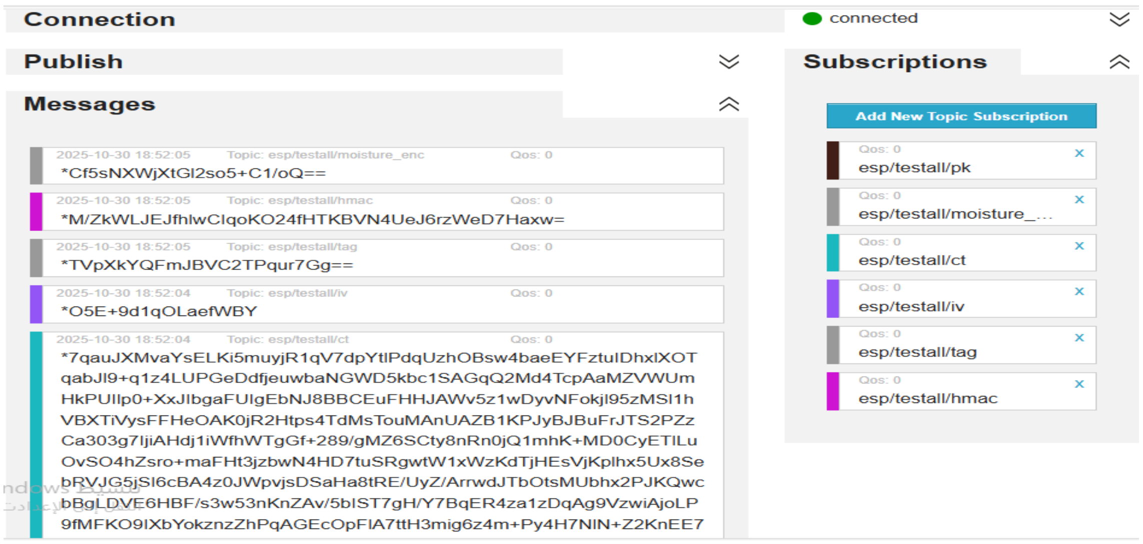
Note: For details of the system parts and results, please see the appendices at the end of the study.

4.3.1 Model 1: General Broker without Authentication

This model represents a data transfer system that does not incorporate any authentication code. This model does not ensure secure and efficient data transfer, as it functions merely as a data obfuscation model: an attacker cannot read the data but can alter the ciphertext by changing even a single bit.

Fields published in this model: PK, CT, Encrypted_ Moisture.

4.3.2 Model 2: General Broker with Authentication



```
2025-10-30 18:52:05    Topic: esp/testall/moisture_enc    Qos: 0
*Cf5sNXWjXtGl2so5+C1/oQ==
2025-10-30 18:52:05    Topic: esp/testall/hmac    Qos: 0
*M/ZkWLJEJfhlwClqoKO24fHTKBVN4UeJ6rzWeD7Haxw=
2025-10-30 18:52:05    Topic: esp/testall/tag    Qos: 0
*TVpXkYQFmJBVC2TPqur7Gg==
2025-10-30 18:52:04    Topic: esp/testall/iv    Qos: 0
*O5E+9d1qOLaefWBY
2025-10-30 18:52:04    Topic: esp/testall/ct    Qos: 0
*7qauJXMvaYsELKi5muyjR1qV7dpYtlPdqUzhOBsw4baeEYFztuIDhxlXOT
qabJl9+q1z4LUPGeDdfjeuwbaNGWD5kbc1SAGqQ2Md4TcpAaMZVWUm
HkPUIlp0+XxJlbgaFUlgEbNJ8BBCEuFHHJAWv5z1wDyvNFokjl95zMSl1h
VBXTiVysFFHeOAK0jR2Htps4TdMsTouMAnUAZB1KPJyBJBuFrJTS2PZz
Ca303g7ljiAHdj1iWfhWTgGf+289/gMZ6SCty8nRn0jQ1mhK+MD0CyETlLu
OvSO4hZsro+maFHt3jzbwN4HD7tuSRgwtW1xWzKdTjHEsVjKplhx5Ux8Se
bRVJG5jSl6cBA4z0JWpvjsDSaHa8tRE/UyZ/ArrwdJTbOtsMUbhx2PJKQwc
bBgLDVE6HBF/s3w53nKnZAv/5blST7gH/Y7BqER4za1zDqAg9VzwiAjoLP
9fMFKO9lXbYokznzZhPqAGEcOpFlA7ttH3mig6z4m+Py4H7NIN+Z2KnEE7
```

Figure 4.1: General Broker with Authentication

This model integrates authentication codes with Kyber-encrypted data into a public communication channel, transforming encryption from protection of “confidentiality only” to comprehensive protection that combines confidentiality (Confidentiality), integrity (Integrity), and reliability (Authenticity). Thus, the system becomes resistant to manipulation, retransmission, and spoofing attacks. The results show that there is a

slight increase in some parameters that do not affect the efficiency of the model but rather increase security.
Fields are published in the model: PK, CT, and authentication fields (Tag, IV, and HMAC). Encrypted_ Moisture.

4.3.3 Model 3: General TLS - MQTT Broker with Authentication

This model also integrates authentication codes with data encrypted with Kyber technology, transforming encryption from protection of “confidentiality only” to comprehensive protection that combines confidentiality, safety (integrity), and reliability (authenticity). This makes the model resistant to tampering, retransmission, and spoofing attacks. However, the difference is that it transmits data over a TLS-secured communication channel, providing comprehensive protection including content confidentiality, channel integrity, and identity verification, making the system more resilient to both classical and quantum attacks.
This integration represents an advanced model of Internet of Things Layer Security (IoT Security Stack), combining post-quantum encryption and secure transmission to ensure data protection from the sensor point to the cloud server. The results show that there is a slight increase in some parameters that do not affect the efficiency of the model but increase security.
Fields published in the model: PK, CT, authentication fields (Tag, IV, HMAC), and Encrypted_ Moisture.

4.3.4 Models 4: General TLS - MQTT Broker via Cloud

This model also integrates authentication codes with data encrypted with Kyber technology, as in the previous two models, transforming encryption from "confidentiality only" protection to comprehensive protection that combines confidentiality, safety, and reliability. This makes the model resistant to manipulation, retransmission, and plagiarism attacks. However, it differs in that it transmits data over the cloud using a TLS-secured communication channel, providing comprehensive protection that includes content confidentiality, channel integrity, and identity verification, making the model more resilient to both traditional and quantitative attacks.
Based on the analysis, it can be said that transferring data encrypted with the Kyber algorithm via Cloud MQTT Broker using TLS achieves the highest level of security

without a significant impact on the overall performance of the model. Despite the slight increase in response time and power consumption, this model is considered the ideal choice among previous methods for smart irrigation applications, as it ensures data confidentiality, integrity, and availability within a secure and scalable operating environment. This cannot be achieved by relying on a public intermediary that lacks full protection for authentication and encryption.

Fields published in this model: PK, CT, authentication fields (Tag, IV, HMAC), and Encrypted_ Moisture.

4.3.5 Proposed MQTT-Based System

Based on the results of the previous comparison, which were added to the appendix, it is better to adopt the MQTT architecture enhanced with the TLS protocol via the cloud (TLS-MQTT via Cloud) in the design of the proposed system, due to the distinct balance it showed between high levels of security and operational performance efficiency compared to other broker configurations. The results showed that using a public broker without authentication achieves the fastest performance in terms of key generation time (4.76 ms) and transfer rate (225,040.37 bytes/s), but the absence of any security layer in this pattern makes it completely unsuitable for applications that deal with sensitive data or environments that require authentication and tight encryption. When authentication was activated or the TLS protocol was applied to the local server, the level of security improved significantly, but this came at the expense of an increase in packaging and unpacking times, which reached about 12.3–15.3 milliseconds, in addition to an increase in power consumption to reach 3.25 millijoules on average, which indicates an additional burden on the processing unit and system resources.

In contrast, the results of the TLS-MQTT system via the cloud showed more balanced performance, as it was able to combine the security advantages provided by the TLS protocol with improved communication and resource efficiency. Unpacking time was only 14.01 ms, while power consumption decreased to 3.24 mJ, maintaining a constant transfer rate of 103,115.63 bytes/s and an acceptable level of free memory usage (150,852 bytes). These values indicate that the use of cloud architecture enables better management of connectivity and load distribution between devices and servers, reducing delays caused by local processing and improving scalability and continuous operation.

Based on the above, the choice of cloud-based TLS-MQTT architecture in the proposed system is scientifically and practically justified, as it represents a compromise that balances communication security with energy and resource efficiency, enhancing system reliability and sustainability in IoT environments that require consistent performance and a high degree of security.
Fields published in the form: pk, ct, iv, tag, hmac, and encrypted moisture.

4.3.5.1 System Architecture

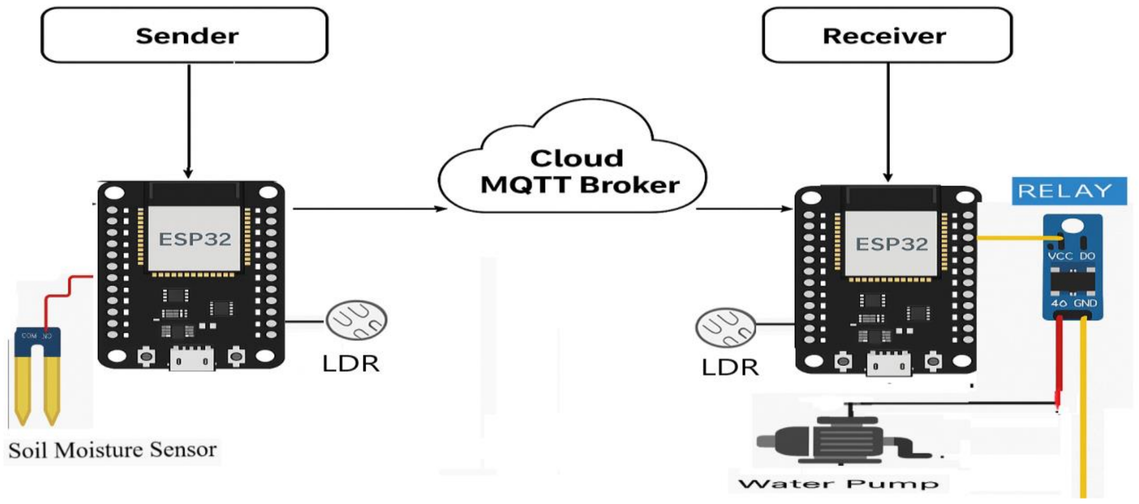


Figure 4.2: MQTT-Based System Architecture

The figure shows the installation of an intelligent irrigation system based on Internet of Things (IoT) technologies using two ESP32 modules connected via a cloud server that adopts the MQTT protocol to exchange data between sender and receiver. At the transmitter, the Soil Moisture Sensor is connected to an ESP32 unit to measure soil moisture level, as well as an LDR to measure light intensity and determine the appropriate irrigation time period. The read data after processing is sent to the Cloud MQTT Broker, which in turn transmits it to the receiver module. At the receiver, the second ESP32 module receives encrypted or processed data from the cloud server and compares it with predefined values to determine soil condition. When irrigation is needed, the unit activates the relay connected to the water pump to turn it on automatically. The presence of light sensors at both ends contributes to improving system efficiency by adjusting pump operation according to lighting conditions and time, achieving an intelligent irrigation system based on self-monitoring and precise cloud control.

4.3.5.2 System Workflow:

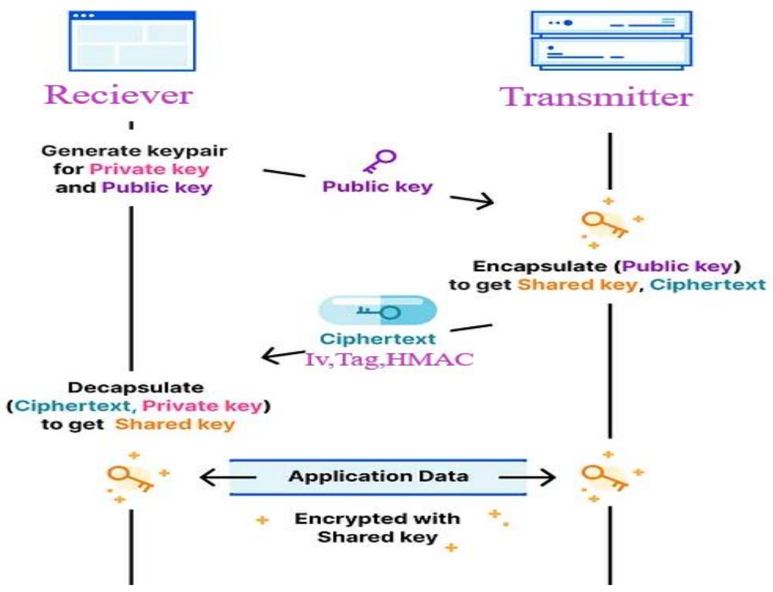

Reciever
Transmitter
Generate keypair for Private key and Public key
Public key
Encapsulate (Public key) to get Shared key, Ciphertext
Ciphertext Iv,Tag,HMAC
Decapsulate (Ciphertext, Private key) to get Shared key
Application Data
Encrypted with Shared key

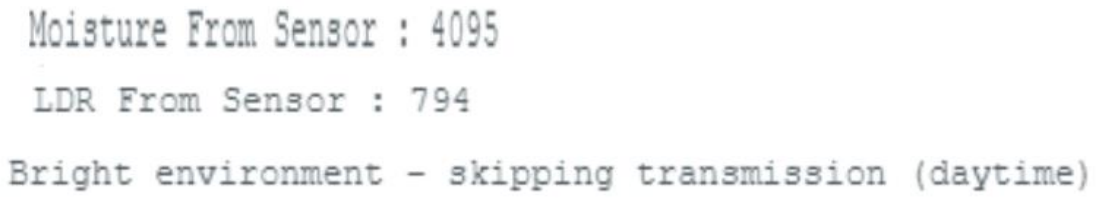
Figure 4.3: MQTT-Based System Workflow

The previous image shows how the proposed system works using the Kyber algorithm to encrypt data and secure communication between sender and receiver. In the first stage, the receiver generates the key pair consisting of the public key (pk) and the private key (sk) and then sends the public key to the sender over a secure communication channel through the cloud using the TLS protocol. After receiving the public key, the sender uses the Kyber algorithm to generate the secret shared key (ss) and generate the ciphertext (ct), then encrypt the moisture value extracted via the sensor using the resulting secret shared key (ss), support the encryption process with a set of authentication fields, and send it to the receiver. Upon receipt of the string, the receiver disassembles it into its original components and uses the private key (sk) with the ciphertext (ct) to unpack and recover the shared key (ss) itself. Both the sender and receiver then use this shared key to secure application data by encrypting and decrypting it using a shared symmetric key, and this key is recreated with each new Internet connection.

This architecture reflects the security of the system based on the principle of post-quantum key exchange based on the Kuiper algorithm, with transmission in a secure cloud communication channel, ensuring data confidentiality and resistance to quantum computing attacks, and enhancing the safety and security of communications in smart Internet of Things systems.

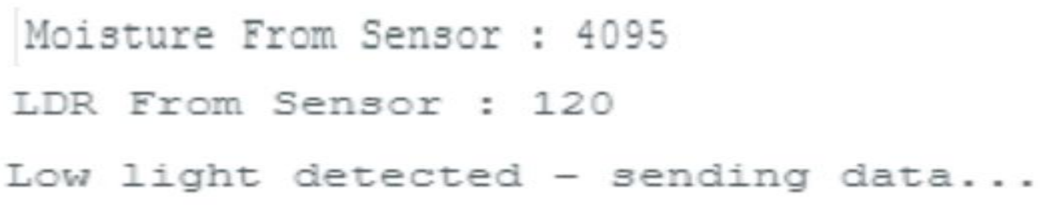
4.3.5.3 System Stages

4.3.5.3.1 Stage1: Collecting Data from Sensors



```
Moisture From Sensor : 4095
LDR From Sensor : 794
Bright environment - skipping transmission (daytime)
```

Figure 4.4: System Decision with LDR Value > 500



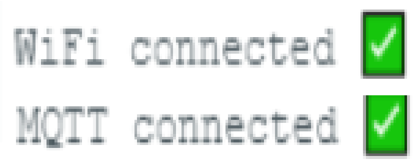
```
Moisture From Sensor : 4095
LDR From Sensor : 120
Low light detected - sending data...
```

Figure 4.5: System Decision with LDR Value < 500

In this step, the humidity value is first checked through a humidity sensor, and then the photoresistor value is checked. If it is less than 500, the system is turned on. If it is greater than 500, the time is considered daytime, and no data is sent, depending on the fact that the pump is not turned on during the day because the sun's rays evaporate the water with heat that can cause burning of plant leaves. Adding variable resistance is considered one of the energy-saving settings in the system by not consuming energy by sending useless information. In the first image, when the value of the optical resistance is greater than 500, it is daytime, and therefore, the system does not send any data, as explained previously, while in the second image, the value of the optical resistance is less than 500, and therefore, operations continue within the system, as will be mentioned later.

4.3.5.3.2 Stage2: Sending Data to Cloud and Distributing It via MQTT

- ✓ Check Network Connection and Broker



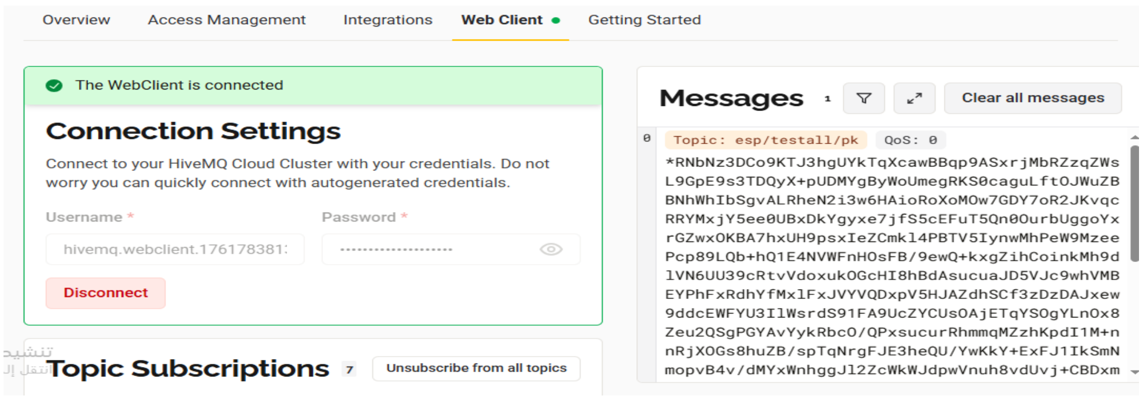
```
WiFi connected ✅
MQTT connected ✅
```

Figure 4.6: Check Connection

Ensure that you are connected to both the network and the broker before sending data. If you are not connected to the network, try every 5 seconds up to 20 times and print "connection again and again failed Wi-Fi" each time, then stop.

When the medium is connected, the device checks the connection before starting the transmission. If it is not connected, it prints "MQTT connection failed" and continues trying up to 10 times, then stops. It is worth noting that the device connects to the medium after the network connection, meaning that it does not achieve this step until after the network connection. Stopping trying to connect after a number of times is considered one of the device's power-saving settings, so that it does not run out of useless connection attempts.

✓ Sending PK



The WebClient is connected

Connection Settings

Connect to your HiveMQ Cloud Cluster with your credentials. Do not worry you can quickly connect with autogenerated credentials.

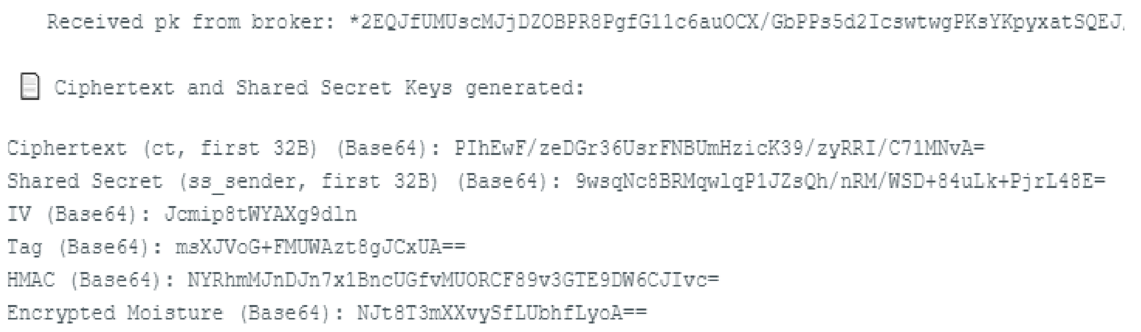
Username * hivemq.webclient.176178381:
Password * ••••••••••••••••••••

```
0  Topic: esp/testall/pk   QoS: 0
*RNbNz3DCo9KTJ3hgUYkTqXcawBBqp9ASxrjMbRZzqZWs
L9GpE9s3TDQyX+pUDMYgByWoUmegRKS0caguLftOJWuZB
BNhWhIbSgvALRheN2i3w6HAioRoXoMOw7GDY7oR2JKvqc
RRYMxjY5ee0UBxDkYgyxe7jfS5cEFuT5Qn0OurbUggoYx
rGZwxOKBA7hxUH9psxIeZCmkl4PBTV5IynwMhPeW9Mzee
Pcp89LQb+hQ1E4NVWFnHOsFB/9ewQ+kxgZihCoinkMh9d
lVN6UU39cRtvVdoxukOGcHI8hBdAsucuaJD5VJc9whVMB
EYPhFxRdhYfMxlFxJVYVQDxpV5HJAZdhSCf3zDzDAJxew
9ddcEWFYU3IlWsrdS91FA9UcZYCUsOAjETqYSOgYLnOx8
Zeu2QSgPGYAvYykRbcO/QPxsucurRhmmqMZzhKpdI1M+n
nRjXOGs8huZB/spTqNrgFJE3heQU/YwKkY+ExFJ1IkSmN
mopvB4v/dMYxWnhggJl2ZcWkWJdpwVnuh8vdUvj+CBDxm
```

Figure 4.7: Sending PK

This system starts when the receiver generates Kyber keys (PK, SK) and publishes PK to the cloud, as shown in the figure below.

✓ Receiving PK



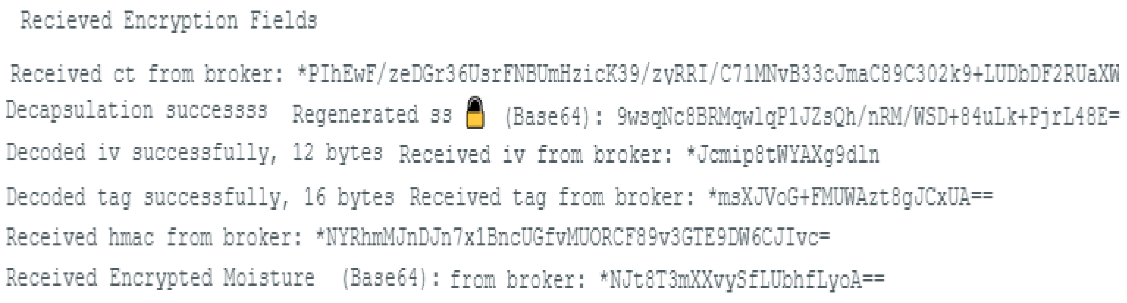
```
   Received pk from broker: *2EQJfUMUscMJjDZOBPR8PgfG11c6auOCX/GbPPs5d2IcswtwgPKsYKpyxatSQEJ

 🗎 Ciphertext and Shared Secret Keys generated:

Ciphertext (ct, first 32B) (Base64): PIhEwF/zeDGr36UsrFNBUmHzicK39/zyRRI/C71MNvA=
Shared Secret (ss_sender, first 32B) (Base64): 9wsqNc8BRMqwlqP1JZsQh/nRM/WSD+84uLk+PjrL48E=
IV (Base64): Jcmip8tWYAXg9dln
Tag (Base64): msXJVoG+FMUWAzt8gJCxUA==
HMAC (Base64): NYRhmMJnDJn7x1BncUGfvMUORCF89v3GTE9DW6CJIvc=
Encrypted Moisture (Base64): NJt8T3mXXvySfLUbhfLyoA==
```

Figure 4.8: Receiving PK

The sender then receives the PK from the cloud and uses it to generate the ciphertext and the shared secret key (ss). It also generates the authentication fields (IV, tag, and

HMAC) and publishes them in the cloud without publishing ss and also uses the ss key to encrypt the moisture value using the AES/GCM system with a 256 key length and publishes it to the cloud. The figure shows that only the first 32 bytes of pk and ct are printed because printing them in full is difficult due to their length.

✓ Receiving CT

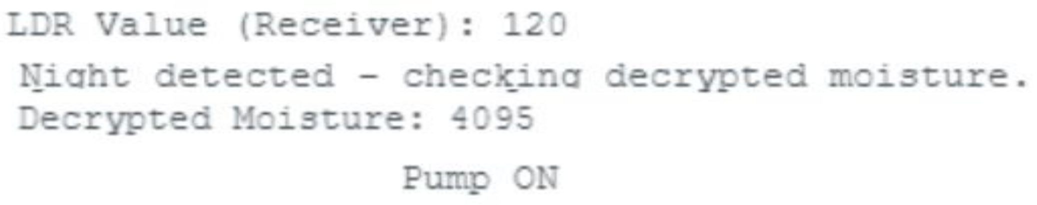


```
 Recieved Encryption Fields

 Received ct from broker: *PIhEwF/zeDGr36UsrFNBUmHzicK39/zvRRI/C71MNvB33cJmaC89C302k9+LUDbDF2RUaXW
Decapsulation successss  Regenerated ss 🔒 (Base64): 9wsqNc8BRMqwlqP1JZsQh/nRM/WSD+84uLk+PjrL48E=
Decoded iv successfully, 12 bytes Received iv from broker: *Jcmip8tWYAXg9dln
Decoded tag successfully, 16 bytes Received tag from broker: *msXJVoG+FMUWAzt8gJCxUA==
Received hmac from broker: *NYRhmMJnDJn7x1BncUGfvMUORCF89v3GTE9DW6CJIvc=
Received Encrypted Moisture  (Base64): from broker: *NJt8T3mXXvySfLUbhfLyoA==
```

Figure 4.9: Receiving CT

After the receiver receives the ciphertext (ct) from the cloud, it retrieves the shared secret key ss through the ciphertext (ct) and the private key (sk) and keeps it to decrypt the data (soil moisture).

4.3.5.3.3 Stage3: Analyzing Data & Making Decision



```
LDR Value (Receiver): 120
 Night detected - checking decrypted moisture.
 Decrypted Moisture: 4095

                    Pump ON
```

Figure 4.10: Decrypting Moisture

After the encoded humidity value arrives and is tested through a set of verifications—which will be explained later—the photoresist value will be tested if the time during the day is neglected, but if the time is at night, the humidity value will be tested after decoding, and the value test if it is greater than 3000 turns on the pump; if it is less, it does not turn it on.

4.4 LoRa - Based Systems

This section presents the results of experiments conducted to transmit data over the air in the form of frequencies using the LoRa protocol with Kyber encryption, which encrypts data according to the principle of post-quantum encryption.

All of these models have been realistically implemented using a humidity moisture sensor and an ESP32. The actual results of these models have also been reviewed, and the average results are reviewed and presented in subsequent comparisons. For details of the results, please see the appendix at the end of the study.

4.4.1 Model 1: (P2P) Transmission without Authentication

This model represents a data transfer model between two LoRa module pieces without linking it to any authentication code. This model does not provide secure and efficient data transfer. It can be considered a data obfuscation model, as an attacker cannot read the data and can even modify the encrypted text by changing a single bit. The text is transferred via a LoRa segment in the form of packets of 240 bytes each.

Fields published in this model: PK, CT. Encrypted_ Moisture.

4.4.2 Model 2: (P2P) Transmission with Authentication

This model transfers data between two LoRa modules after integrating authentication codes with Kyber-encrypted data in a public communication channel, transforming encryption from "confidentiality only" protection to comprehensive protection that combines confidentiality, integrity, and reliability. Thus, the model becomes resistant to manipulation, retransmission, and plagiarism attacks. The results show a slight increase in some parameters, which does not affect the efficiency of the system but rather increases its security.

Fields are published in the model: PK, CT, and authentication fields (Tag, IV, and HMAC). Encrypted_ Moisture.

4.4.3 Model 3: Point-to-Multipoint Transmission

This model is based on connecting several transmitting points with a receiving point, where the pump connected to the ESP unit and the LoRa unit represents the receiving point that receives the signal from several transmitting points distributed in several places on the farm, and these points are composed of an ESP unit connected to the LoRa

Module unit and a humidity sensor. These sensors check the soil moisture in the designated places and send a periodic signal at the appropriate time, based on which the appropriate decision is made to turn on or off the pump.
As in previous models, the ESP unit handles the signal processor, while LoRa acts as the unit for receiving and sending incoming signals.
Fields are published in the model: PK, CT, NodeID, Encrypted_ Moisture.

4.4.4 Proposed LoRa-Based System

Based on the results of the previous comparison - added in the appendix- the adoption of the point-to-multipoint communication mode in the system designed using LoRa technology can be justified, as this mode provides efficiency in managing multiple communications and expands the transmission range compared to the point-to-point (P2P) mode, whether with or without authentication. The data show that key generation, encryption, and decryption time remained approximately constant between the three modes, indicating that the transition to Point-to-Multipoint mode did not impose an additional time burden on processing or encryption phases. Power consumption also remained approximately constant (1.58 mJ) between systems, indicating that this pattern achieves a low and stable level of power consumption despite the increase in the number of connected nodes. In terms of data transfer rate, it decreased slightly in the point-to-multipoint pattern compared to the P2P pattern with authentication, which is a natural decrease resulting from multiple transmission paths and channel distribution over more than one node. However, this rate is adequate for smart agriculture and IoT applications that rely on periodic, distributed transmission of sensor data rather than large-scale data transfer. The response time in the point-to-multipoint pattern also remained within acceptable limits for long-range LoRa applications, which are more concerned with reliability and wide coverage than instantaneous connection speed. Free memory values show that the system in this mode maintains resource efficiency even with multiple nodes, demonstrating its software stability and not increasing the burden on operational memory.

Accordingly, the choice of Point-to-Multipoint mode in the proposed system using LoRa is technically justified, as it enables more efficient management of wide multi-node networks while maintaining stable performance, low energy consumption, and resource efficiency, making it best suited for long-range IoT-based environmental and agricultural monitoring applications.

4.4.4.1 System architecture

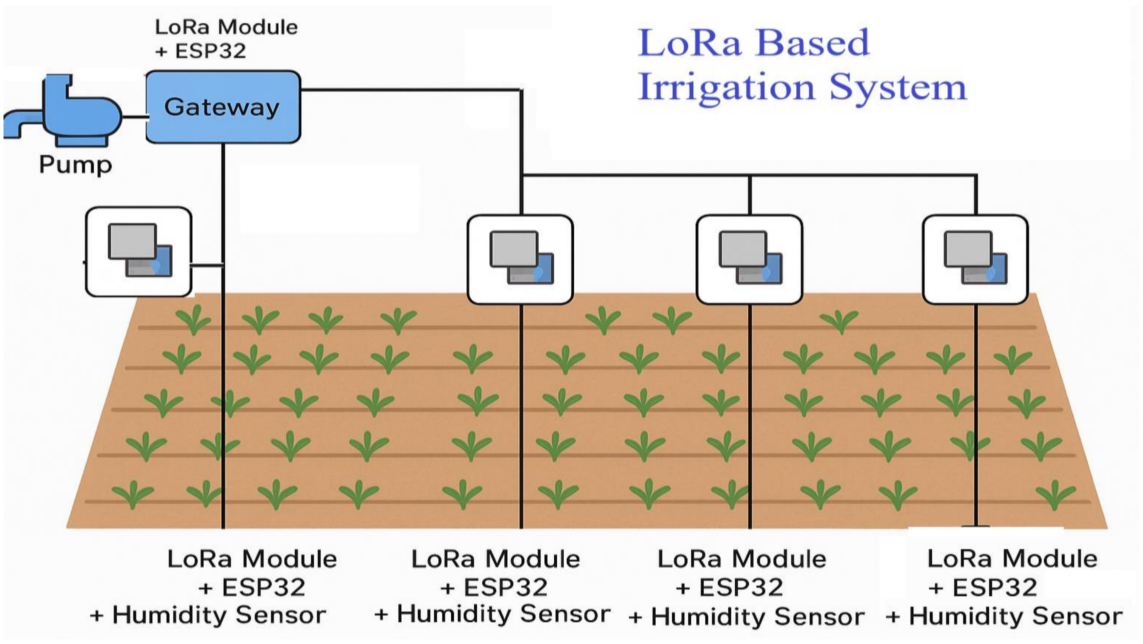


Figure 4.11: LoRa-Based System Architecture

The proposed system applies the P2M approach, consisting of a set of terminal nodes, each consisting of an ESP32 module, a LoRa module, and a humidity sensor. The moisture sensor measures soil moisture and sends the result to the ESP module, which encodes the data and sends it to the LoRa module, which in turn sends it to the (LoRa + ESP) module, which represents a receiving node that receives the data, decodes it, and makes the decision to turn the pump on or off.

This system is similar in style to LoRaWAN networks, except that it is simpler to install and operate. However, this system combines efficiency and security, ease of power and switch management, and scalability in smart irrigation networks using LoRa technology. This usually makes it the preferred choice in these applications, especially on large farms, with many nodes, or places that lack communications infrastructure. This design also demonstrates the integration of sensors, intelligent processing, and real-time control of mechanical devices to achieve more efficient and intelligent irrigation water management.

4.4.4.2 LoRa Configurations

4.4.4.2.1 LoRa Standards

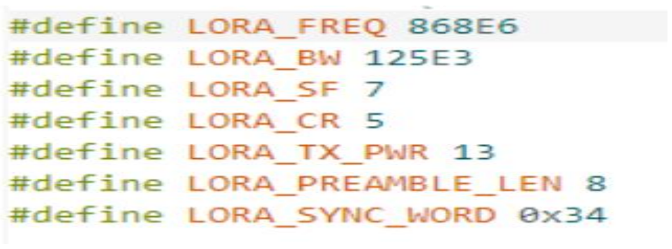

```
#define LORA_FREQ 868E6
#define LORA_BW 125E3
#define LORA_SF 7
#define LORA_CR 5
#define LORA_TX_PWR 13
#define LORA_PREAMBLE_LEN 8
#define LORA_SYNC_WORD 0x34
```

Figure 4.12: LoRa Standards

LoRa settings are a fundamental factor that determines the efficiency of wireless communication in long-range IoT systems. The system in this study is based on:

1: Frequency 868 MHz, an industrial band (ISM Band) intended for low-power communications in Europe and the Middle East, which provides a balance between range and low electromagnetic interference.

2: Bandwidth (Bandwidth = 125 kHz) determines the rate of symbols transmitted per second (symbol rate). Reducing bandwidth increases the sensitivity of the receiver (receiver sensitivity) and thus increases the connection range but reduces the data transfer rate (data rate), and vice versa when it increases.

3: The spreading factor (Spreading Factor = 7) is one of the most important properties of LoRa, as it determines the number of symbols used to represent each bit of data within the modulation technique (Chirp Spread Spectrum). Higher propagation factor values (such as 10 or 12) give greater ability to receive weak signals and improve range, but increase transmission time and consume more power. Choosing SF7 strikes a good balance between speed and range in semi-open environments.

4: The correction rate (coding rate = 4/5) adds error correction bits (error correction bits) within the data frame, enabling the receiver to correct errors caused by noise or interference without the need for retransmission. The higher the correction rate, the greater the reliability, but the lower the net data rate.

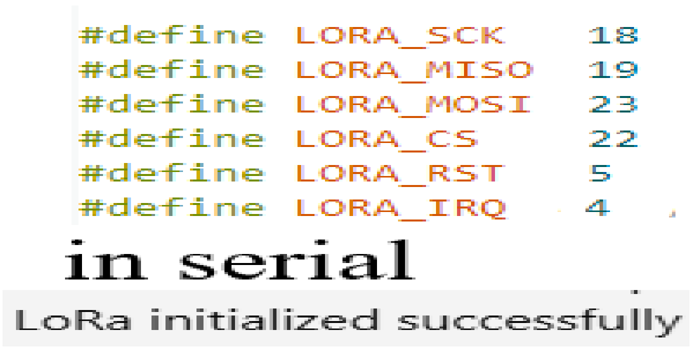
5: Transmission power (Tx Power = 13 dBm) represents the strength of the signal sent from the node to the gateway. Increasing it expands the connection range, but at the expense of battery power consumption. Therefore, choosing 13 dBm is suitable for power systems while maintaining a stable connection range.

6: The boot length (preamble length = 8) is used to facilitate signal detection and receiver synchronization with the start of the data packet. Increasing this length improves the probability of signal detection in interference-filled environments but prolongs transmission time.

7: The synchronization word (Sync Word = 0x34) identifies the network used and separates it from other LoRa networks operating in the same bandwidth, reducing interference and improving packet discrimination accuracy.

These settings provide a balance between range, data rate, power consumption, and reliability, enabling the application to operate at optimal efficiency.

4.4.4.2.2 LoRa Pins



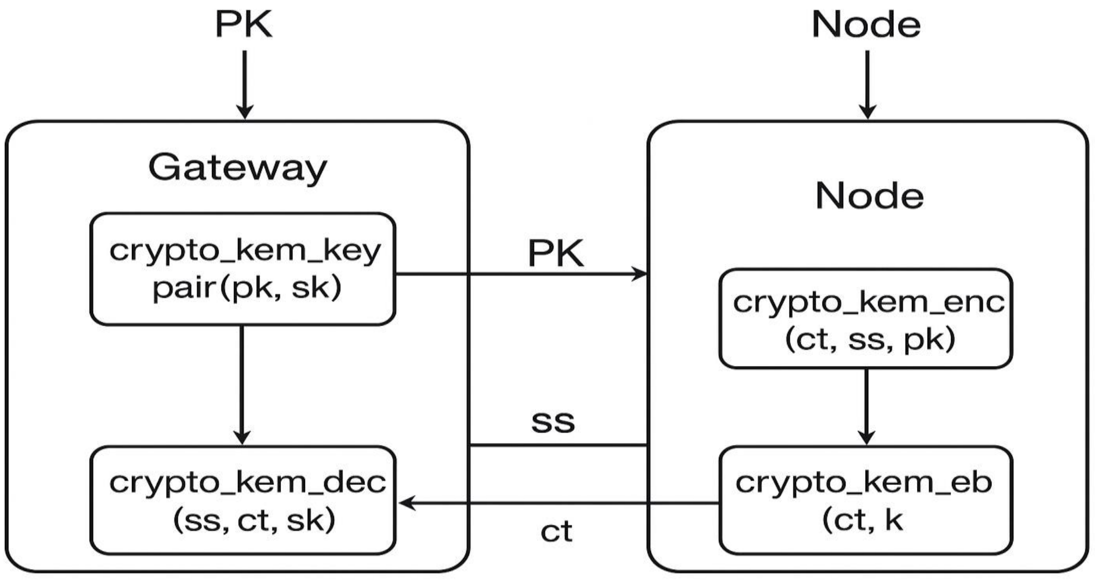
```
#define LORA_SCK    18
#define LORA_MISO   19
#define LORA_MOSI   23
#define LORA_CS     22
#define LORA_RST    5
#define LORA_IRQ    4
```

```
in serial
LoRa initialized successfully
```

Figure 4.13: LoRa Pins

The figure shows the LoRa module's communication pin identification setup with the ESP32 controller, where communication lines are defined according to the SPI protocol through fixed definitions that include clock, transmit, and receive lines, as well as control lines such as the reset (RST) and interrupt line (IRQ). Integrating settings in this way is one of the best practices in designing embedded systems (Embedded Systems), as it contributes to achieving organized functional integration between system components and ensures precise time compatibility in transmission and reception processes. This regulation also helps improve connection stability and reduce error rates resulting from signal conflicts or configuration failures, and allows software reset of the unit in the event of a connection outage, which enhances system reliability and operational continuity. In addition, this approach contributes to reducing power consumption and response time thanks to the use of interrupts to process data as soon as it is received, and facilitates the process of transferring or expanding the project to other applications by modifying definitions only, without the need to rebuild the entire software architecture. Thus, this structural integration represents an essential step towards achieving optimal performance and high reliability in IoT applications that rely on LoRa wireless connectivity, such as smart irrigation systems.

4.4.4.3 System Workflow:



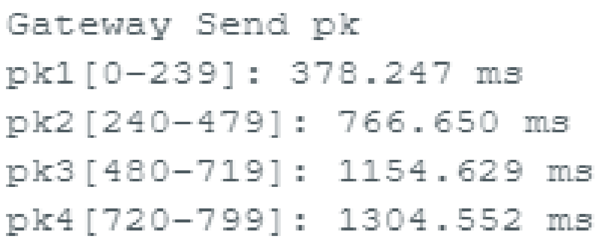
PK
Node
Gateway
crypto_kem_key pair(pk, sk)
PK
Node
crypto_kem_enc (ct, ss, pk)
ss
crypto_kem_dec (ss, ct, sk)
ct
crypto_kem_eb (ct, k

Figure 4.14: LoRa-Based System Workflow

This image shows a diagram showing the mechanism of key exchange and generation of shared secret keys (Shared Secret, SS) between an ESP32 module representing a gateway (Gateway) and a node (Node) using a KEM-based encryption method (Key Encapsulation Mechanism), such as Kyber. The scheme starts at the gateway, where a public & private key pair is generated, then the public key (PK) is sent to all nodes, and each node receives this key and then uses it to encapsulate the secret key, where a ciphertext (Ciphertext, CT) and a temporary secret key (SS) are produced. The node then sends the ciphertext (CT) to the gateway, which in turn uses the private key (sk) to unpack and retrieve the shared secret key. This key is then used to exchange messages between the node and the gateway. This diagram shows a complete cycle of key exchange in a secure manner, ensuring that both the gateway and the node have the same secret key without the need to transfer the key directly over the network, which enhances the security of communication in the network. Fields published in this method: pk, ct, authentication fields (Tag, IV, HMAC), and moisture.

We will then review the detailed steps for data movement in the system:

4.4.4.3.1 Sending Pk

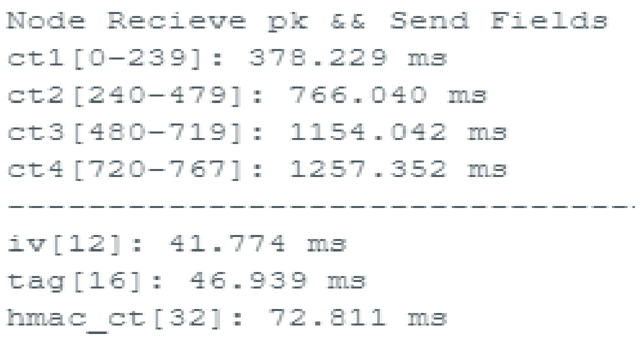


```
Gateway Send pk
pk1[0-239]: 378.247 ms
pk2[240-479]: 766.650 ms
pk3[480-719]: 1154.629 ms
pk4[720-799]: 1304.552 ms
```

Figure 4.15: Sending PK

The gateway then generates the two keys (pk, sk) and sends pk in the form of a 240-bit packet until the entire key is sent. The previous image shows the transmission of the key packets and the special counter that distinguishes this message, along with each packet and the time taken during transmission.

4.4.4.3.2 Receiving PK

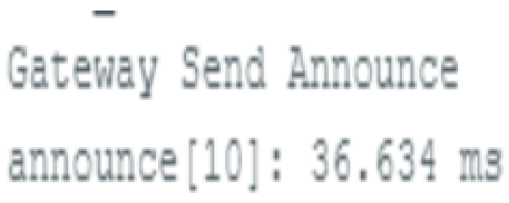


```
Node Recieve pk && Send Fields
ct1[0-239]: 378.229 ms
ct2[240-479]: 766.040 ms
ct3[480-719]: 1154.042 ms
ct4[720-767]: 1257.352 ms
--------------------------------
iv[12]: 41.774 ms
tag[16]: 46.939 ms
hmac_ct[32]: 72.811 ms
```

Figure 4.16: Receiving PK

All nodes receive the pk message sent from the gateway, then each node performs successive operations to generate the shared secret key, after which it sends the following information to the gateway: ct, iv, tag, and hmac. The gateway retrieves the key through the ct sk previously stored with it, and thus the gateway has a private key with this node through which messages are exchanged between them. The image shows the transmission time for each field that was sent.

4.4.4.3.3 Sending Announce

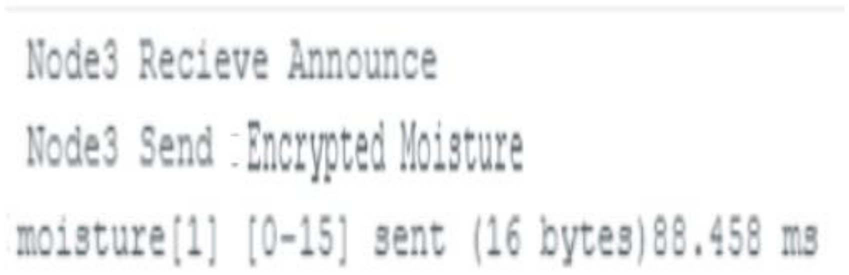


```
Gateway Send Announce
announce[10]: 36.634 ms
```

Figure 4.17: Sending Announce

After the gateway receives the information from the node and retrieves the key, it sends a notification to the node so that the node can send its humidity value encrypted.

4.4.4.3.4 Sending Encrypted Moisture

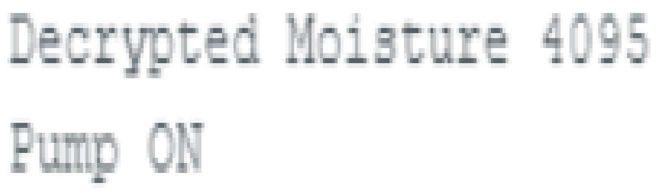


```
Node3 Recieve Announce
Node3 Send :Encrypted Moisture
moisture[1] [0-15] sent (16 bytes)88.458 ms
```

Figure 4.18: Sending Encrypted Moisture

The intended node picks up the message and sends the encoded humidity value based on it, as shown in the picture. Other nodes may pick up the same message, but the unique number of the node is what determines whether the message is received by the gateway or neglected. Waiting for the node to notify the gate puts it into a sleep state until the notification arrives, which is considered one of the most important methods of saving energy consumed in the system.

4.4.4.3.5 Verifying Data and Decision



```
Decrypted Moisture 4095
Pump ON
```

Figure 4.19: *Verifying* & Decision

After the gate receives the humidity value, it performs the necessary checks to ensure the authenticity of the message, then decodes the message when its authenticity is confirmed and makes the decision to stop or start the pump.

4.5 Conclusion

This chapter provides a detailed discussion of the architecture and functional components of the proposed IoT-based smart irrigation experimental systems. The integration of communication and security layers was explained through system diagrams and configuration steps. The next chapter presents the proposed systems and the experimental results obtained from testing these systems under realistic conditions.

Chapter 5: Results and Discussion

5.1 Introduction

This chapter aims to present and analyze the results after applying the proposed systems in a real-world environment, and compare them with previous systems and studies to evaluate their efficiency and performance in an Internet of Things environment. The chapter covers a set of basic metrics used to measure performance, such as key generation, encryption, and decryption time, power consumption, CPU usage, data transfer rate, response time, and memory consumption. It also aims to interpret these results from a scientific perspective that highlights the impact of the system's design and security architecture (based on post-quantum computing algorithms, MQTT protocols, and LoRa technology) in achieving a balance between security and efficiency, thus contributing to supporting the study's main objectives of developing an intelligent, secure, and sustainable system for agricultural applications based on the Internet of Things.

5.2 MQTT- Based Systems

5.2.1 System Variables Size

Table 5-1: MQTT-Based System Variables Size

Variable	Value per Byte
Project Size	1020399 …. (77%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	51452 …… (15%) of RAM
Local Variables Size	276228
Maximum Size	327680

The total project size was 1020399 bytes, equivalent to about 77 percent of the memory allocated for storage (Flash Memory), which has a maximum of 1,310,720 bytes. This ratio is acceptable within Internet of Things (IoT) application standards based on encryption algorithms, especially with the integration of the Kyber post-quantum encryption algorithm, which requires additional space for key processing and extensive calculations. As for dynamic memory (RAM), the total size of general variables was 51,452 bytes, which represents about 15 percent of the total random memory capacity of

327,680 bytes. In contrast, the size of local variables during execution was 276,228 bytes, bringing the total memory consumption during operation to approximately 100 percent of the maximum power, without exceeding the maximum permissible limit.
These values show that the system operates within high memory usage limits but is still within the safe range of operational performance of the ESP32 module, demonstrating that the integration of the Kyber algorithm with encryption functions and security field management (HMAC, IV, Tag, Counter) did not cause a critical drain on resources. Therefore, the system's software architecture can be considered balanced in terms of security, efficiency, and utilization of physical resources, and is scalable in the future through improved memory management or code compression without affecting the overall stability of the system.

5.2.2 System Performance Metrics Results (Data)

Table 5-2: MQTT-Based System Performance Metrics Results (Data per Byte)

Metric	**Trials Values**				**Average**
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Size per Byte	32	32	32	32	32
Data transfer rate (Bps)	61048.3	61082.1	60485.2	61057.6	60918.3
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	99824	99932	99964	99988	99927
Pump (on/off)	Off	Off	On	On	

The previous table shows the results of four experiments (T1–T4) that measure the performance of a system based on soil moisture readings and pump operation control, including encryption and network transmission. It is clear from the values that the humidity rate (moisture) varies significantly across experiments, ranging from 211 to 3774, reflecting the system's response to different environmental conditions. In all experiments, the key size stayed consistent at 32 bytes, demonstrating the stability of the security settings employed. The data transfer rate also showed relative stability, averaging 60,918.3 bytes/s, indicating the efficiency of communication between terminal nodes and the gateway. In addition, the maximum load capacity remained constant at 800 bytes, indicating a consistent data structure during data transmission. Free memory averaged 99,927 bytes across experiments, with slight differences, reflecting efficient resource management.

Finally, the pump operation results (pump on/off) showed that the system activated the pump only when the humidity was high (in the third and fourth experiments), confirming the system's accuracy in making decisions based on the measured data.

5.2.3 Performance Metrics Results (Time)

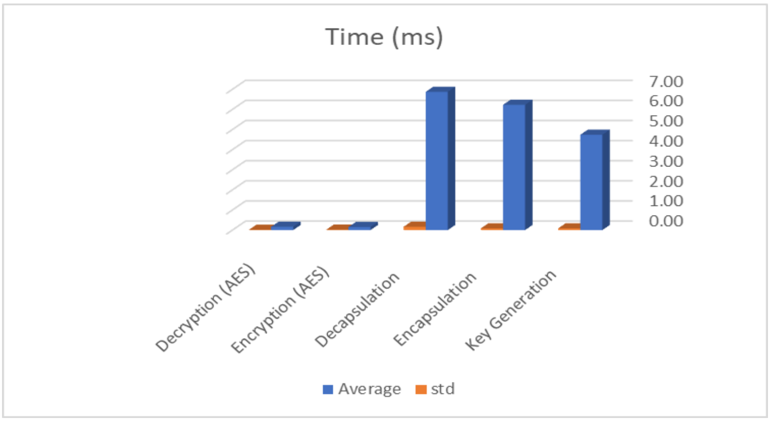


Figure 5.1: Performance Metrics Results (Time)

Table 5-3: MQTT-Based System Performance Metrics Results (Time per Milli Second)

Time / Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation(std)**
Key Generation	4.76	4.614	4.77	4.81	4.74	0.09
Encapsulation	6.27	6.098	6.27	6.27	6.23	0.09
Decapsulation	6.96	6.626	6.95	6.95	6.87	0.16
Encryption (AES)	0.162	0.119	0.16	0.16	0.15	0.02
Decryption (AES)	0.17	0.135	0.17	0.17	0.16	0.02
Response time	104.8	104.7	105.8	104.8	105.02	0.52
Total Cycle Time	1689	1638	1688.8	1688	1675.87	25.00
Sum	18.32	17.59	18.32	18.36	18.15	0.37

The table shows the time performance analysis of encoding and decoding processes within the system across four different humidity value experiments. Key generation time turns out to be an average of 4.74 ms with a low standard deviation (0.09 ms), reflecting the stability and speed of the process. The encapsulation and decapsulation also recorded averages of 6.23 ms and 6.87 ms, respectively, demonstrating the efficiency of implementing the Kyber algorithm within the test environment. Encryption and decryption operations using AES were the fastest, with an average time of 0.15 ms and 0.16 ms, respectively, highlighting the lightness of the algorithm and its suitability for applications with critical time requirements. On the other hand, the system showed an almost constant response time (Response Time) with an average of 105.02 ms, which is an indicator of the speed of decision-making after receiving the data. While the total cycle time (Total Cycle Time) averaged 1675.87 ms, the total cryptographic times averaged only 18.15 ms, a small percentage of the total time, proving that cryptographic mechanisms barely affect the system's overall performance.

5.2.4 Performance Metrics Results (CPU Usage)

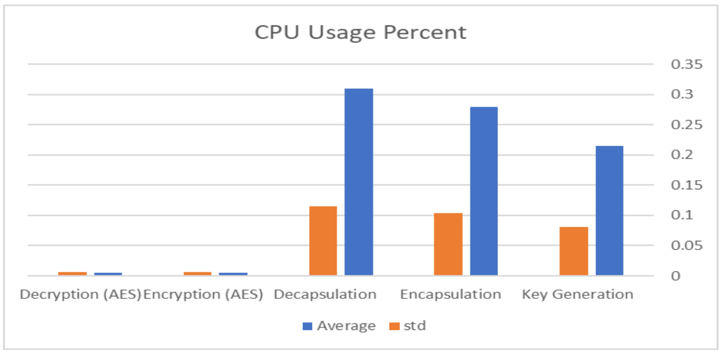


Figure 5.2: Performance Metrics Results (CPU Usage)

Table 5-4: MQTT-Based System Performance Metrics Results (CPU Percentage Usage)

Metric / Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation(std)**
Key Generation	0.14	0.15	0.28	0.29	0.215	0.081
Encapsulation	0.19	0.19	0.37	0.37	0.28	0.1039
Decapsulation	0.21	0.21	0.41	0.41	0.31	0.1155
Encryption (AES)	0	0	0.01	0.01	0.005	0.0058
Decryption (AES)	0	0	0.01	0.01	0.005	0.0058
Sum	0.54	0.55	1.08	1.09	0.82	0.31

The table shows the rates of central processing unit (CPU) usage during the execution of different cryptographic operations across four experiments for varying values of soil moisture. The results indicate that the key generation process consumed an average of 0.215% of the processor's power with a standard deviation of 0.081%, indicating stable performance and efficient algorithm execution. Processor usage rates in the encapsulation and decapsulation processes also averaged 0.28% and 0.31%, respectively, with a moderate standard deviation indicating a slight increase in computational load as the humidity value increases, as a result of the difference in the size of the encrypted data. In contrast, the AES algorithm for both encryption and decryption showed very little processing unit usage, averaging only 0.005%, reflecting its high resource efficiency. Finally, the total processing unit consumption (sum) ratio averaged 0.82%, which is very low and confirms that integrating Kyber and AES algorithms into the system causes little computational burden, making it well-suited for embedded applications and systems with limited resources.

5.2.5 Performance Metrics Results (Energy Consumption)

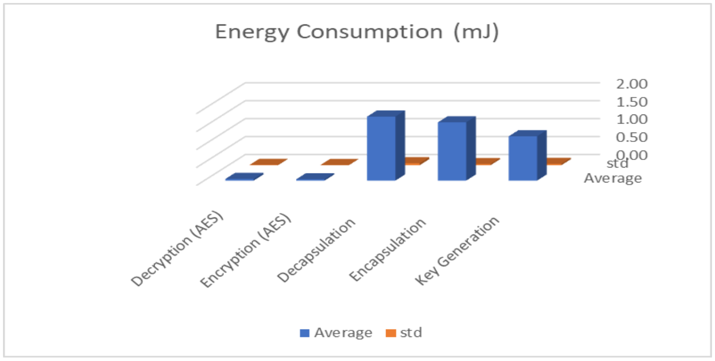


Figure 5.3: Performance Metrics Results (Energy Consumption)

Table 5-5: MQTT-Based System Performance Metrics Results (Energy Consumption per milli Joul)

Metric/ Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation (std)**
Key Generation	1.19	1.22	1.26	1.27	1.24	0.04
Encapsulation	1.58	1.61	1.65	1.65	1.62	0.03
Decapsulation	1.72	1.75	1.83	1.83	1.78	0.06
Encryption (AES)	0.03	0.03	0.04	0.04	0.04	0.01
Decryption (AES)	0.06	0.04	0.04	0.05	0.05	0.01
Sum	4.59	4.64	4.83	4.84	4.73	0.14

The table shows energy consumption measurements for various cryptographic processes within the system across four experiments for varying values of soil moisture. The results show that the key generation process consumed an average of 1.24 milli joul with a standard deviation of 0.04, indicating stable performance and limited power consumption. The encapsulation and decapsulation processes recorded relatively the highest consumption rates at an average of 1.62 milli joul and 1.78 milli joul, respectively, which is due to the extensive calculations associated with the Kyber algorithm during key exchange and encrypted data processing. In contrast, the AES algorithm for both encryption and decryption showed very low power consumption, averaging 0.04 milli joul and 0.05 milli joul, respectively, reflecting its high efficiency in environments with power constraints. Considering the total power consumption (sum), it averaged 4.73 milli joul with a small standard deviation (0.14), indicating that integrating Kyber and AES algorithms into the system did not result in a significant increase in power consumption. These results highlight the efficiency of the proposed system in terms of balancing high security while maintaining energy efficiency suitable for IoT applications and embedded systems.

5.2.6 Comparisons

5.2.6.1 Other models (Variable Size)

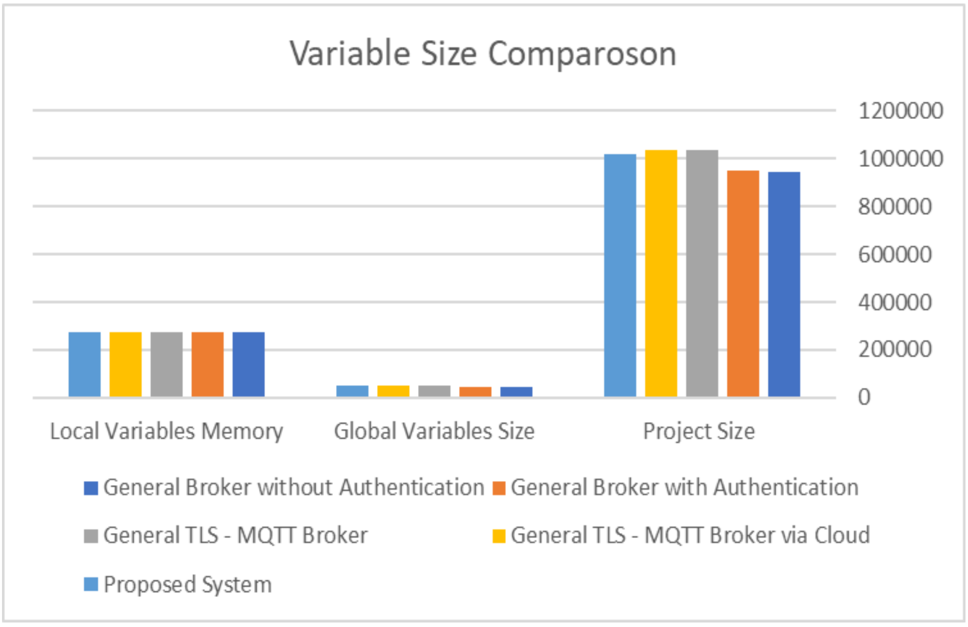


Figure 5.4: Comparison with other models in Variable Size

Table 5-6: Comparison of Variable Size Results in Different MQTT-Based Models

Variable	**Value per Byte**				
	General Broker without Authentication	General Broker with Authentication	General TLS - MQTT Broker	General TLS - MQTT Broker via Cloud	Proposed System
Project Size	945039	948775	1037407	1036539	1020399
Percentage of Storage Memory	72	72	79	79	77
Maximum Flash Size	1310720	1310720	1310720	1310720	1310720
Global Variables Size	46472	46,472	49228	48852	51452
RAM Percentage	15	16	16	16	15
Local Variables Memory	275368	274888	274328	274220	276228
Maximum RAM Size	327680	327680	327680	327680	327680

The table provided a comprehensive comparison between five different systems in terms of memory and software resource consumption during the implementation of MQTT and TLS protocols, including the proposed system. It turns out that the project size (Project Size) in the proposed system was 1,020,399 bytes, which is lower than the cloud-based TLS-MQTT system (1,036,539 bytes) and also lower than the general TLS-MQTT system (1,037,407 bytes), reflecting efficiency in managing the size of the code despite the integration of additional security technologies. The proposed system also recorded a storage memory usage rate (Storage Memory) of 77% of the total capacity (1,310,720 bytes), which is lower than TLS-based systems (79%) and slightly higher than traditional unencrypted intermediaries (72%), indicating a clear balance between security and memory consumption.

As for the size of general variables (Global Variables Size), the proposed system reached 51,452 bytes, which is the highest among comparative systems, reflecting a slight increase in the volume of data related to encryption and key management operations. In contrast, the utilization rate of random-access memory (RAM) maintained a moderate level (15%) similar to other systems, confirming the resource efficiency of dynamic processing. Finally, it appears that the memory size allocated to local variables (Local Variables Memory) in the proposed system was 276,228 bytes, which is within acceptable limits for maximum memory (327,680 bytes).

Overall, these results demonstrate that the proposed system achieves an optimal balance between security and spatial efficiency, providing a high level of security with only a slight increase in memory consumption, making it suitable for applications embedded in IoT environments.

5.2.6.2 Other models (Performance Metrics)

Table 5-7: Comparison of Performance Metrics Results in Different MQTT-Based Models

Metric	**General Broker without Authentication**	**General Broker with Authentication**	**General TLS - MQTT Broker**	**General TLS - MQTT Broker via Cloud**	**Proposed System**
Key Generation Time (ms)	4.762	4.7875	4.875	4.7725	4.74
Encapsulation Time (ms)	6.3125	12.31	12.3775	12.29	6.23
Decapsulation Time (ms)	6.915	15.2965	14.1525	14.01	6.87
Key Size per Byte	32	32	32	32	32
Energy consumption (mJ)	1.66	3.25	3.2675	3.24	4.73
Data transfer rate (Bps)	225040.37	95216.9	108140.1	103115.63	60918.3
Response time (ms)	28.4425	67.195	59.6025	62.0625	123.6325
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	195127	194529	148449	150852	99927

The table shows a comprehensive comparison of the performance of five different systems in implementing the MQTT protocol, including the proposed system based on hybrid encryption mechanisms (Kyber + AES). It is clear from the results that the key generation time in the proposed system is the lowest among all systems (4.74 ms), which reflects high efficiency in implementing the key generation algorithm. It also kept encapsulation and decapsulation time relatively low (6.23 ms and 6.87 ms, respectively), while these values doubled in TLS-based systems, demonstrating the superiority of the proposed system in the processing speed of cryptographic processes.
In terms of key size, it remained constant at 32 bytes across all systems, confirming the consistency of security settings. Regarding energy consumption, it was slightly higher in the proposed system (4.73 mJ) compared to the rest of the systems, which is justified given that more security measures are used than in other systems, which requires additional calculations to ensure a stronger level of security. In terms of data transfer rate, the proposed system showed the lowest value (60918.3 Bps) compared to traditional intermediaries, and this is associated with increased encryption time and additional security measures, but it is still within the acceptable range for small-package IoT applications. The response time in the proposed system was 123.63 ms, which is higher than other systems due to integrated encryption processes, but this time remains suitable for non-instant applications. It appears that the free memory in the proposed system is the lowest (99,927 bytes) as a result of additional consumption resulting from security and encryption processes, but this consumption remains balanced, given the significant increase in security provided by the system.
Overall, the results indicate that the proposed system achieves a high level of security with acceptable time and resource efficiency, making it an ideal choice for smart systems and the Internet of Things that require strong protection without fundamentally compromising performance.

*5.2.6.3 Comparison with Another Paper's Results (*CPU Usage, Response Time*)*

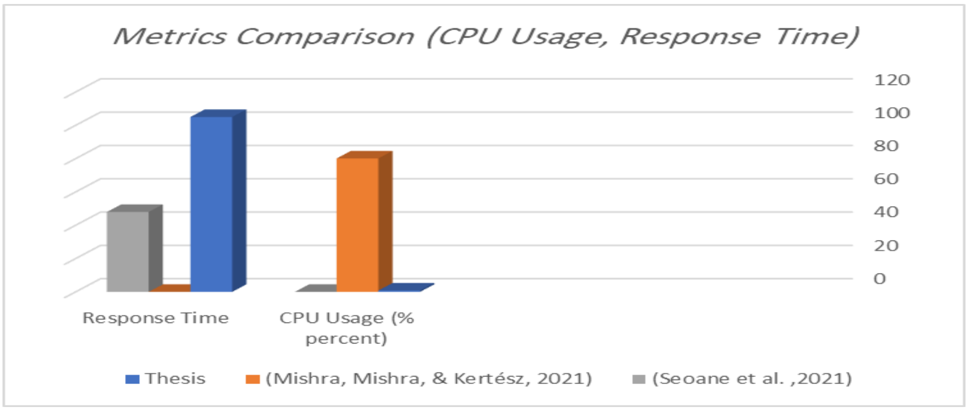


Figure 5.5: Metrics Comparison (CPU Usage, Response Time)

Table 5-8: MQTT- Based System Comparison with Another Papers Results

Metric / Paper	**Thesis**	**(Mishra, Mishra, & Kertész, 2021)**	**(Seoane et al. ,2021)**
CPU Usage (% percent)	0.82	80.09	27-36
Response Time	105	35-80	48

The table presents a comparison of the communication system's performance in this thesis with the results reported by Mishra et al. (2021) and Seoane et al. (2021). Regarding processor usage, the thesis appears to achieve the lowest consumption of 0.82%, compared to the higher values in previous studies, where the percentage reached 80.09% in Mishra et al. And 27–36% in Seoane et al. This suggests that the system in the thesis is more efficient in terms of processor resource consumption, which may reflect a simpler design or reliance on less complex algorithms compared to previous studies. As for response time, the thesis value is 105 ms, which is higher than the values reported in other studies, which ranged between 35 and 80 ms in Mishra et al. And between 48 ms in Seoane et al. This illustrates that reducing processor consumption in the thesis came at the expense of increasing response time, which may be a result of using lighter data processing methods or reducing the resources allocated to each task to ensure processor efficiency. Accordingly, the system presented in this thesis demonstrates high processor efficiency compared to previous studies, although it exhibits a longer response time, which is an important factor to consider when designing IoT systems with limited resources.

5.2.6.4 Comparison with Another Paper's Results (Time)

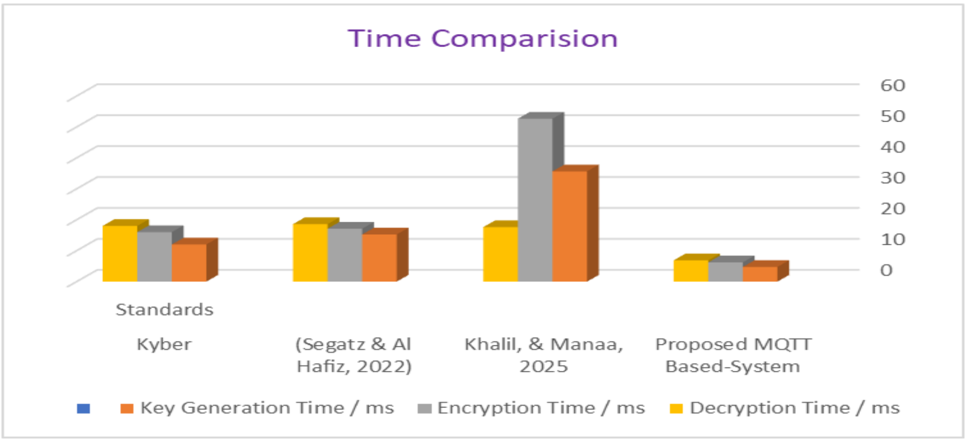


Figure 5.6: Time Comparison (Key Generation, Encryption, Decryption)

Table 5-9: Time Comparison (Key Generation, Encryption, Decryption) with Another Paper Results with Kyber Standards

Metric / Paper	**Proposed MQTT-Based System**	**Khalil & Manaa, 2025**	**(Segatz & Al Hafiz, 2022)**	**Kyber Standards**
Key Generation Time/ms	4.74	35.64	15.24	12 ms
Encryption Time/ms	6.23	52.67	17.1	16 ms
Decryption Time/ms	6.87	17.62	18.57	18 ms

The table presents a quantitative comparison of the proposed system's performance under the MQTT protocol with the results of previous studies by Khalil & Manaa (2025) and Segatz & Al Hafiz (2022), as well as the standard values of the Kyber algorithm used in the benchmarks. The comparison focuses on three main stages in the encryption cycle: key generation, encryption, and decryption. The results show that the proposed system achieved a low-key generation time of 4.74 ms, significantly faster than the 35.64 ms reported in the study by Khalil & Manaa (2025). In the study (Segatz & Al Hafiz, 2022) of 15.24 ms, and it even outperforms the standard value of the Kyber algorithm (12 ms). In the encryption phase, the proposed system recorded 6.23 ms, which is better than other studies (52.67 ms and 17.1 ms) and the Kyber standard (16 ms). The system also demonstrated high efficiency in the decoding phase with a time of only 6.87 ms, compared to 17.62 ms in a study (Khalil & Manaa, 2025), 18.57 ms in a study (Segatz & Al Hafiz, 2022), and the Kyber benchmark of 18 ms Based on these results, it can be concluded that the proposed system achieves significant improvements across all encryption stages compared to previous systems and benchmarks, demonstrating its high efficiency in reducing processing time and improving the performance of secure communications in MQTT-based environments.

5.3 LoRa - Based Systems

5.3.1 System Variables Size

Table 5-10: LoRa-Based System Variables Size

Variable	Value per Byte
Project Size	343647 …. (26%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	26308 …… (8%) of RAM
Local Variables Size	301372
Maximum Size	327680

An academic analysis of the following table shows the basic indicators related to memory consumption in the system used in the project. The values indicate the efficiency of memory resource utilization in terms of storage memory and RAM, which is a critical component in microcontroller-based systems such as ESP32. A project size of 343647 bytes, equivalent to about 26% of storage memory, shows that the program exploits a moderate portion of the total space of 1,310,720 bytes. This reflects an efficient software design, with approximately 72% of memory remaining available for future project expansion or the addition of new software modules, such as additional encryption algorithms or parallel tasks, without exceeding storage capacity. As for the size of global variables, which amounted to 26308 bytes, equivalent to about 8% of random memory (RAM), it appears that the program uses a limited amount of fixed memory to store permanent and shared variables between tasks. This low percentage reflects efficiency in memory organization and avoids excessive use of general variables, which may lead to RAM congestion or task interference. In contrast, it is noted that local variables consume 301372 bytes out of a total of 327,680 bytes of maximum allocated memory, meaning that the temporary memory used during task execution occupies approximately 93% of the capacity. This relatively high value indicates that the computations or encryption algorithms used (such as Kyber or LoRa data processing) rely on large temporary variables, which calls for improved cache management to avoid errors such as stack overflow or poor system performance when multitasking.

Overall, the table reflects a good balance between storage and random memory consumption, with the need to optimize local memory distribution to reduce pressure on RAM without affecting processing performance or the speed of execution of encrypted operations in the system.

5.3.2 System Performance Metrics Results (Data)

Table 5-11: LoRa-Based System Performance Metrics Results (Data per Byte)

Metric	**Trials Values**				**Average**
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Size per Byte	32	32	32	32	32
Data transfer rate (Bps)	23012.89	23012.89	23012.97	23013.97	23013.18
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	261500	261632	261632	261632	261599
Pump (on/off)	Off	Off	On	On	

The table shows the results of four field experiments conducted on the proposed system to measure its performance in intelligent pump control based on soil moisture readings, including encryption mechanisms during data transmission. The values show that the humidity level (moisture) ranged from 211 to 3774, reflecting the diversity of environmental conditions in which the system was tested. The key size remained constant at 32 bytes in all experiments, indicating stable security settings during operation. The data transfer rate also showed clear stability with an average of 23,013.18 bytes/s, indicating the efficiency of the wireless connection and that it is not affected by encryption processes. The maximum permissible load also remained constant at 800 bytes, consistent with the limits of the transport protocol used. As for free, it recorded a high average of 261,599 bytes, which reflects efficiency in consuming software resources and memory while performing multiple tasks. Finally, the pump operation results (pump on/off) show that the system only activated the pump in the third and fourth experiments, i.e., when the humidity values exceeded the specified operating limit, demonstrating the system's accuracy in decision-making based on the measured values. These results thus demonstrate that the proposed system is characterized by stability, efficient resource management, and accurate response to agricultural environmental conditions, while maintaining security and encryption properties without affecting overall performance.

5.3.3 Performance Metrics Results (Time)

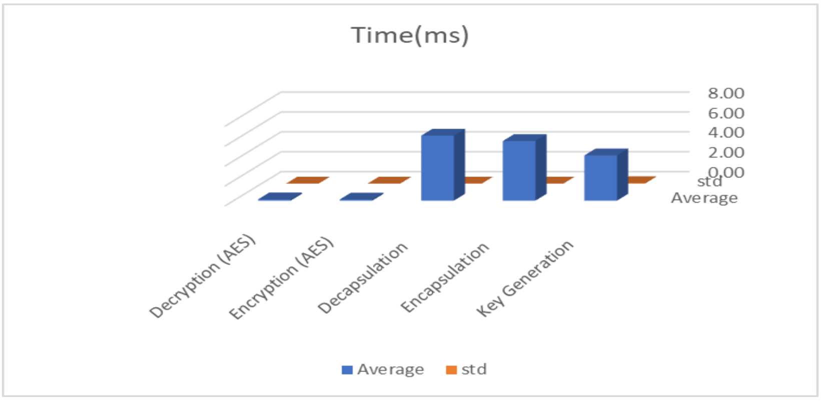


Figure 5.7: System Metrics (Time)

Table 5-12: LoRa -Based System Performance Metrics Results (Time per Milli Second)

Metric / Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation (std)**
Key Generation	4.50	4.64	4.52	4.52	4.54	0.05
Encapsulation	5.98	6.00	5.99	5.99	5.99	0.01
Decapsulation	6.52	6.55	6.53	6.53	6.53	0.01
Encryption (AES)	0.11	0.12	0.11	0.11	0.11	0.00
Decryption (AES)	0.13	0.13	0.13	0.13	0.13	0.00
Response time	4.5	4.635	4.518	4.518	4.54	0.05
Total Cycle Time	3133.01	3133.01	3133.01	3133.01	3133.01	3133.01
Sum	17.24	17.43	17.28	17.28	17.31	0.07

The table presents a detailed analysis of the time performance of the proposed system during the implementation of different cryptographic operations across four experiments for varying values of soil moisture. The results show that Key Generation averaged 4.54

ms with a low standard deviation (0.05 ms), demonstrating process stability and speed of execution within the operating environment. Encapsulation and decapsulation also showed convergent performance with time averages of 5.99 ms and 6.53 ms, respectively, with a standard deviation not exceeding 0.01 ms, reflecting high efficiency in applying the Kyber algorithm within a resource-limited environment. As for the AES algorithm, the Encryption and Decryption times were very low, averaging 0.11 ms and 0.13 ms, respectively, highlighting the lightness and processing speed of symmetric operations. In terms of overall performance, the Response Time system recorded an average of only 4.54 ms, reflecting a high ability to instantly handle transmitted data. Total Cycle Time also reached a constant value of 3133.01 ms across all experiments, indicating the stability of the operational architecture and the stability of the overall performance of the system. Finally, the sum of the cryptographic times showed a low average of 17.31 ms with a simple standard deviation (0.07 ms), representing a very small percentage of the total cycle time. These results reflect that the proposed system has high time efficiency and performance stability, with a clear ability to integrate encryption processes without affecting the speed or responsiveness of the system, making it suitable for IoT applications with critical time requirements.

5.3.4 Performance Metrics Results (CPU Usage)

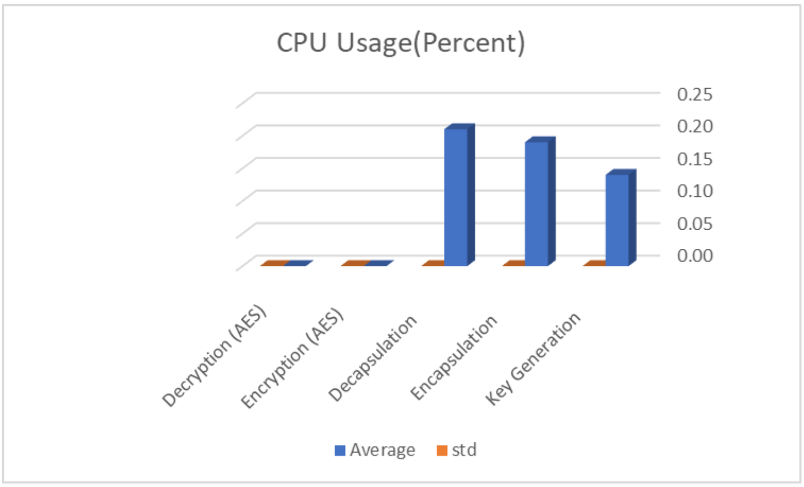


Figure 5.8: System Metrics (CPU Usage Percent)

Table 5-13: LoRa-Based System Performance Metrics Results (CPU Percentage Usage)

Metric / Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation(std)**
Key Generation	0.14	0.14	0.14	0.14	0.14	0.00
Encapsulation	0.19	0.19	0.19	0.19	0.19	0.00
Decapsulation	0.21	0.21	0.21	0.21	0.21	0.00
Encryption (AES)	0	0	0	0	0.00	0.00
Decryption (AES)	0	0	0	0	0.00	0.00
Sum	6.00	0.54	0.54	0.54	0.54	0.00

The table provides an accurate analysis of central processing unit consumption ratios (CPU Usage) while performing various cryptographic operations in the proposed system across four experiments for varying values of soil moisture. The results show that consumption ratios remained approximately constant in all experiments, reflecting stable performance and processing efficiency. The average processor consumption in the Key Generation process was only 0.14%, while the Encapsulation and Decapsulation processes recorded averages of 0.19% and 0.21%, respectively, which indicates that the Kyber algorithm was implemented very efficiently without causing a significant computational burden on the system. As for the encryption and decryption processes using AES, the processing unit consumption was almost zero (0.00%), reflecting the high performance of the algorithm in resource-constrained environments. The total (Sum) also shows a very low overall average (0.54%) with almost no standard deviation, demonstrating consistent processing unit performance across various experimental conditions. These results demonstrate that the proposed system has high computational efficiency and very low processor consumption, making it suitable for embedded and IoT applications that require stable performance while conserving limited resources.

5.3.5 Performance Metrics Results (Energy Consumption)

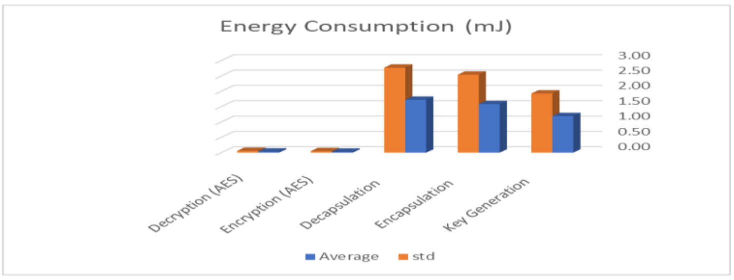


Figure 5.9: System Metrics (Energy Consumption)

Table 5-14: LoRa -Based System Performance Metrics Results (Energy Consumption per milli Joul)

Metric / Moisture	**2243**	**211**	**3765**	**3774**	**Average**	**Standard Deviation(std)**
Key Generation	1.19	1.22	1.26	1.27	1.24	0.04
Encapsulation	1.58	1.61	1.65	1.65	1.62	0.03
Decapsulation	1.72	1.75	1.83	1.83	1.78	0.06
Encryption (AES)	0.03	0.03	0.04	0.04	0.04	0.01
Decryption (AES)	0.06	0.04	0.04	0.05	0.05	0.01
Sum	4.59	4.64	4.83	4.84	4.73	0.14

The table shows energy consumption measurements for a set of cryptographic operations performed in the proposed system under four different soil moisture level experiments, to evaluate the system's energy efficiency during operation. The results indicate that the key generation process consumed an average of 1.24 mJ with a low standard deviation (0.04), reflecting stable performance and consistent power consumption at this stage. Encapsulation and decapsulation showed average consumption of 1.62 mJ and 1.78 mJ, respectively, which are the highest values among operations, and this is due to the computationally intensive nature of the Kyber algorithm used to exchange cryptographic keys within the system. On the other hand, the AES algorithm used for encryption and decryption achieved outstanding energy performance with very low average consumption of only 0.04 mJ and 0.05 mJ, highlighting its high efficiency in compact, limited-power environments. Total energy consumption averaged 4.73 mJ with a slight standard deviation (0.14), which indicates the stability and efficiency of the system in managing energy across all stages of processing. These results reflect that the proposed system has high energy efficiency while maintaining high levels of security, making it suitable for practical applications in Internet of Things (IoT) systems that require a careful balance between energy consumption and security performance.

5.3.6 Comparisons

5.3.6.1 Comparison with other models Variable Size

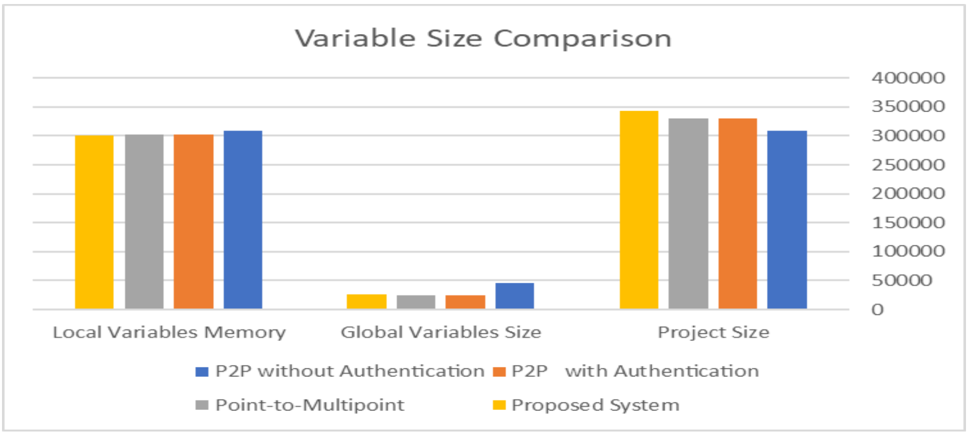


Figure 5.10: Variable Size Comparison

Table 5-15: Comparison of Variable Size Results in Different LoRa-Based Models

Variable	**Value per Byte**			
	P2P without Authentication	P2P with Authentication	Point-to-Multipoint	Proposed System
Project Size	308247	329527	330111	343647
Percentage of Storage Memory	23	25	25	26
Maximum Flash Size	1310720	1310720	1310720	1310720
Global Variables Size	46472	24,928	25036	26308
Percentage of RAM	7	7	7	8
Local Variables Memory	308247	302752	302644	301372
Maximum RAM Size	327680	327680	327680	327680

The table provides a detailed comparison of memory and software resources consumption indicators between four different systems for direct communication (P2P), multi-point communication (point-to-multipoint), and the proposed system that integrates additional security mechanisms. The results show that the project size in the proposed system reached 343,647 bytes, the highest among other systems, reflecting the increase resulting from the integration of encryption and identity verification algorithms within the system architecture. Despite this increase, the percentage of storage memory remained at a moderate level of only 26% of maximum capacity (1,310,720 bytes), demonstrating high efficiency in managing storage resources. Global Variables Size recorded a slight increase in the proposed system (26,308 bytes) compared to the rest of the systems as a result of using additional data structures to support security operations and key exchange. In contrast, random access memory (RAM) consumption remained at a relatively low level (8%), showing that the inclusion of security layers did not impose a significant burden on dynamic resources. The system also kept Local Variables Memory close to other systems (301,372 bytes) out of a maximum of 327,680 bytes, reflecting stable software performance during execution.

Overall, these results demonstrate that the proposed system achieves an optimal balance between security and memory efficiency, adding enhanced security components without negatively impacting overall performance, making it suitable for low-resource wireless communication applications within Internet of Things (IoT) environments

5.3.6.2 Other models (Performance Metrics)

Table 5-16: Comparison of Performance Metrics Results in LoRa-Based Models

Metric	**P2P without Authentication**	**P2P with Authentication**	**Point-to-Multipoint**	**Proposed System**
Key Generation Time (ms)	4.76	4.78	4.87	4.54
Encapsulation Time (ms)	6.31	12.31	12.37	5.99
Decapsulation Time (ms)	6.91	15.29	14.15	6.53
Key Size per Byte	32	32	32	32
Energy consumption (mj)	1.66	3.25	3.26	1.19
Data transfer rate (Bps)	225040.3	95216.9	108140.1	23013.1
Response time (ms)	28.44	67.19	59.60	278.12
Maximum load size (Byte)	800	800	800	800
Free Memory (byte)	195127	194529	148449	261599

The table above presents a comprehensive comparison between different communication systems in terms of key performance metrics associated with encryption processes, power consumption, and data transfer efficiency. The results indicate that the proposed system demonstrated balanced and efficient performance in terms of speed and resource efficiency compared to other systems. The system achieved the lowest key generation time of 4.54 ms, demonstrating its efficiency in performing initial security operations, while encapsulation time decreased to 5.99 ms, a significant improvement over P2P with documentation (12.31 ms) and multipoint communication (12.37 ms). Also, the proposed system recorded the lowest decapsulation time of 6.53 ms, enhancing its ability to reduce the overall processing time of encrypted messages. In terms of energy efficiency, the proposed system emerged as the most economical, with a consumption of only 1.19 mJ, a clear decrease compared to other systems whose values ranged between 1.66 and 3.26 mJ. As for data transfer rate, a decrease in performance was observed in the proposed system (23,013.1 bytes/second) compared to other systems, which is expected as a result of adding multiple security layers that increase the processing volume and total transmission time. Although response time rose to 278.12 ms, this rise is due to additional cryptographic processes that ensure data integrity during transmission, representing an acceptable trade-off between security and speed. As for free memory, it was the highest in the proposed system (261,599 bytes), which reflects efficiency in managing internal resources compared to other systems, especially the multi-point communication system, which showed a significant decrease in free memory (148,449 bytes). Overall, the results show that the proposed system achieves a qualitative improvement in power consumption and memory efficiency while maintaining strong cryptographic performance, making it suitable for Internet of Things (IoT) applications that require a balance between security and limited resources.

5.3.6.3 Comparison with Another Papers' Results (Tran. Rate, Res. Time):

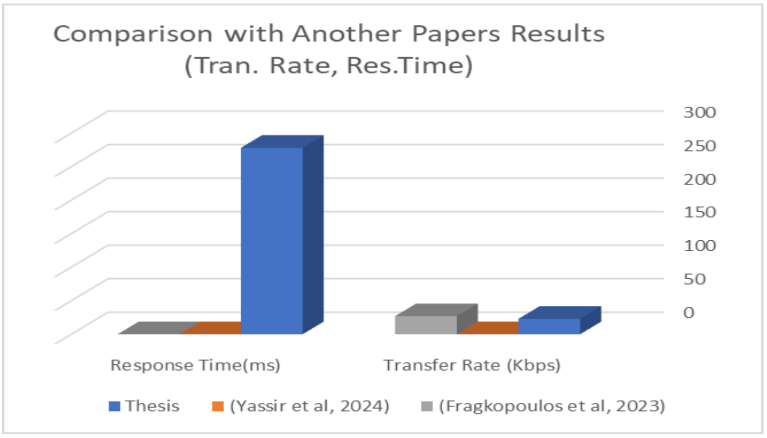


Figure 5.11: Comparison with Another Papers' Results in Lora Systems (Tran. Rate, Res. Time)

Table 5-17: Comparison of LoRa-Based Systems with Other Papers' Results

Metric / Paper	**Thesis**	**(Yassir et al, 2024)**	**(Fragkopoulos et al, 2023)**
Transfer Rate (Kbps)	23	2-6.8	250-27
Response Time(ms)	278	20-30	(1-6)

The table provides a comparison of the performance of LoRa networks across different systems. Analysis indicates that the proposed message system achieves a transmission rate of 23 Kbps and a response time of 278 ms, which is significantly higher in response time compared to the experimental study (Yassir et al. 2024) which recorded a transfer rate between 2–6.8 Kbps and a low response time between 20–30 ms, reflecting a difference in network design or the number of nodes used. While the study by Fragkopoulos et al. (2023) shows a wide transmission rate range of 250–27 Kbps and a very short response time of 1–6 ms, demonstrating significant performance improvements in rural or semi-urban environments with improved LoRaWAN settings. This analysis shows that the proposed system provides an average transfer rate with relatively higher response time, reflecting a balance between stability and data efficiency in practical applications.

5.3.6.4 Comparison with Another Papers' Results Using LoRa (Metrics)

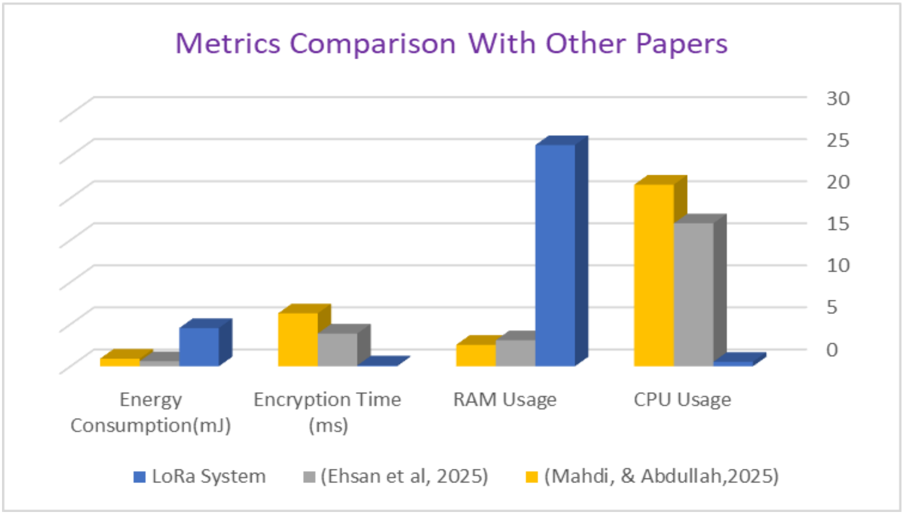


Figure 5.12: Comparison with Another Papers Results in Lora Systems

Table 5-18: Comparison with Another Papers' Results in LoRa-Based Systems

Paper	**Encryption Method**	**CPU Usage Percent %**	**RAM Usage (KB)**	**Encryption Time (ms)**	**Energy Consumption(mJ)**
Proposed System	Kyber512 +AES\GCM	0.54	26.3	0.11	4.55
(Ehsan et al, 2025)	AES/128	Not mentioned	2.8	Less than 100	0.5
(Mahdi, & Abdullah,2025)	Kyber512 +ASCON	21.6	2.56	43	21

The table shows a performance comparison between the proposed system and some previous studies. The table shows that the proposed system combining Kyber512 and AES/GCM features very low processor consumption of 0.54 percent and memory of 26.3 KB with very short encoding time of 0.11 ms and moderate power consumption of 4.55 mJ. In comparison, a system (Ehsan et al.,2025) using AES/128 shows very low power consumption of 0.5 mJ, encryption time of less than 100 ms, and memory of 2.8 KB, but processor consumption is not mentioned, which makes it difficult to evaluate the overall efficiency of the system. While the system (Mahdi & Abdullah,2025) using Kyber512 with ASCON highlights high processor consumption of 21.6 percent, memory of 2.56 KB, long encryption time of 43 ms, and large power consumption of 21 mJ. The analysis shows that the proposed system achieves an optimal balance between encryption speed, resource efficiency, and power consumption, making it more suitable for practical applications in resource-constrained IoT environments compared to other systems.

5.3.6.5 Comparison with Kyber Handshake Time in Lora Systems

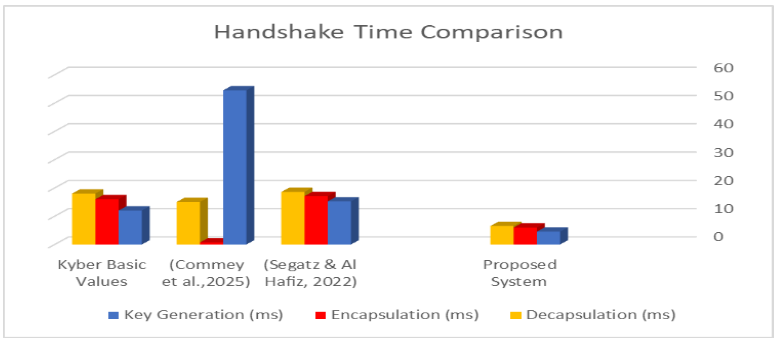


Figure 5.13: Handshake Time Comparison

Table 5-19: Comparison with Kyber Handshake Values in LoRa-Based Systems

Paper	**Encryption Method**	**Key Generation (ms)**	**Encapsulation (ms)**	**Decapsulation (ms)**
Proposed System	Kyber512 +AES\GCM	4.54	5.99	6.53
(Segatz & Al Hafiz, 2022)	AES/128	15.24	17.1	18.57
(Commey et al.,2025)	Kyber	103.7	0.75	8.38
Kyber Basic Values		12	16	18

The table shows a time comparison of encryption performance between the proposed system and some previous studies. The results show that the proposed system combining Kyber512 and AES/GCM has a low-key generation time of up to 4.54 ms, with moderate encapsulation and unpacking times of 5.99 and 6.53 ms, respectively. In comparison, the (Segatz & Al Hafiz, 2022) study shows that using AES/128 alone requires longer key generation (15.24 ms), encapsulation (17.1 ms), and unpacking (18.57 ms), while the (Commy et al, 2025) study shows. Using Kyber alone is a big difference, as generating the key takes a much longer time (103.7 ms) despite the speed of encapsulation (0.75 ms) and unpacking (8.38 ms). Compared with the baseline Kyber values (12–18 ms), it is evident that integrating Kyber with AES/GCM achieves a better balance between key generation speed and the efficiency of wrapping and unwrapping operations, enhancing the overall performance of the proposed system and making it more suitable for practical applications in IoT networks that require both speed and security.

5.4 Final Metrics Analysis

The previous tables highlighted a detailed comparison between the two proposed systems based on the LoRa and MQTT protocols in terms of overall performance, resource efficiency, time, and energy, reflecting fundamental differences in the design and purpose of each system. It turns out that the MQTT system has a much larger project size and high storage ratio compared to the LoRa system, which suggests that the MQTT environment is more complex and relies on a rich network architecture that requires more memory and processing, while the LoRa is characterized by its light architecture and ability to operate in devices with limited capabilities. The MQTT system also showed an increase in the size of general variables versus LoRa, indicating greater reliance on shared data and global functions, while LoRa maintained a better balance in local memory usage, which was reflected in a lower memory consumption ratio versus MQTT. In terms of computational performance, LoRa outperformed MQTT in time in all cryptographic processes (key generation and encryption stages) versus slight but significant differences in constrained systems where every millisecond represents a critical component of the response.

Although LoRa excelled in computational efficiency, the MQTT system showed higher performance in terms of interaction speed compared to LoRa, and the total operational cycle was shorter, reflecting the nature of MQTT based on permanent network connectivity and the speed of packet exchange over the Internet Protocol. It is also noted that the data transfer rate in MQTT exceeds LoRa by more than ten times, making it more suitable for applications that require transferring or broadcasting large amounts of data in real time, while LoRa remains the ideal choice in remote or limited-connection environments. On the other hand, LoRa provided higher efficiency in internal resource management, with free memory being almost twice what is available in MQTT, along with a slight reduction in power consumption and processing unit usage.

These results show LoRa is more energy-efficient for long-lived IoT use in agriculture and distributed environments, while MQTT excels in real-time and high-speed applications for industrial control and monitoring. LoRa highlights operational efficiency in constrained settings; MQTT represents high performance in connected environments. The choice depends on whether efficiency or speed is prioritized.

5.5 Conclusion

This chapter analyzed the experimental results of the implemented systems, evaluating their performance in terms of latency, energy efficiency, and data transmission reliability. The findings confirmed the technical viability of the proposed design. The next chapter focuses on a detailed security analysis to assess system resilience against cyber threats.

Chapter 6: System Security Analysis

6.1 Introduction

In the previous chapter, we reviewed the results of implementing the proposed system, and we also analyzed these results in terms of time, memory, power, and other performance measures. In this chapter, we analyze the security level in the system in terms of the security features provided by each element added to the system. We also review the attacks that this element protects against. Finally, we present a comparison between the system's performance via the MQTT and LoRa protocols.

6.2 MQTT-Based System

6.2.1 MQTT Based-System Security Analysis:

The main objective of the study is to secure connectivity to the Internet of Things. To achieve this objective, several steps must be taken, which we will mention in this section.

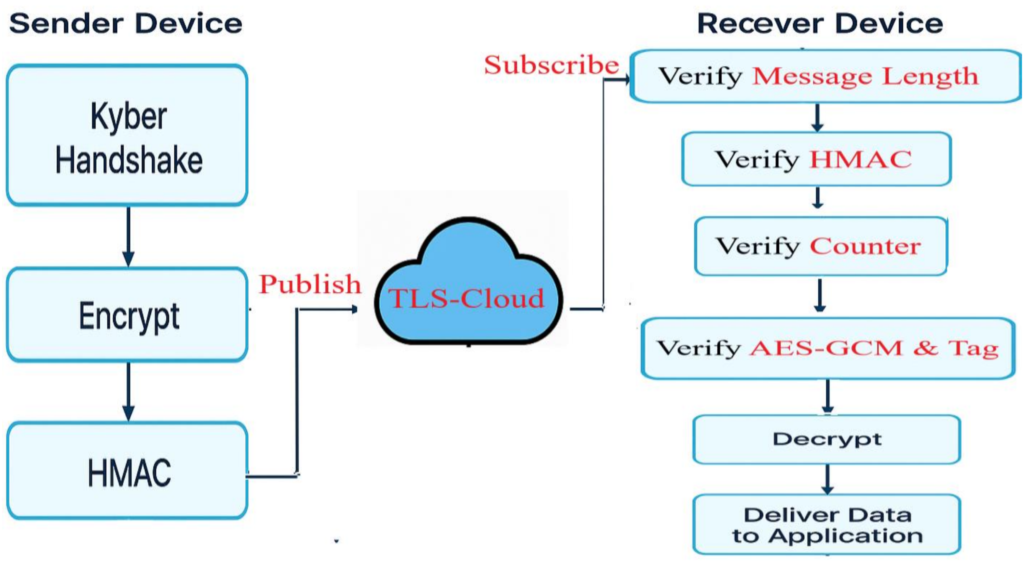
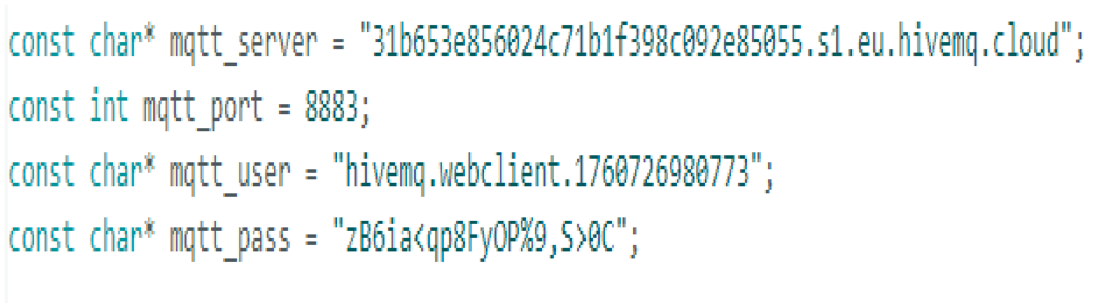


Figure 6.1: Secure MQTT connection with Kyber

The previous image shows one of the modern systems based on the Internet of Things and the cloud. The combination of TLS-Cloud, Kyber, AES-GCM, and HMAC technologies represents a set of integrated layers of protection that enhance the confidentiality, integrity, and authenticity of data during transmission and processing. Below is an explanation of the level of security that each of the previous elements provides to the system.

6.2.1.1 Transmitting / Receiving via TLS-Cloud



```
const char* mqtt_server = "31b653e856024c71b1f398c092e85055.s1.eu.hivemq.cloud";
const int mqtt_port = 8883;
const char* mqtt_user = "hivemq.webclient.1760726980773";
const char* mqtt_pass = "zB6ia<qp8FyOP%9,S>0C";
```

Figure 6.2: Cloud Authentication

When we talk about TLS-Cloud, we mean securing communication between the two parties using the TLS protocol within a cloud computing environment. This usually includes ensuring the confidentiality of the communication (confidentiality) and the absence of any unauthorized change (safety) between the sender and the receiver. Using TLS in the cloud means encrypting and protecting data "on transmission" within the cloud infrastructure (between components or to/from the user), rather than sending it "transparently" or as text over the network. TLS-Cloud is not a new protocol in itself but rather an application of the TLS protocol in the context of a cloud environment, with its own challenges and opportunities.

The use of a TLS layer between the device and the cloud ensures the confidentiality and integrity of transmission and reduces the possibility of messages being intercepted as they cross the network. In the proposed system, the TLS protocol makes it difficult for a network attacker to easily read fields or inject messages, and authentication (using a username and password) reduces the possibility of impersonation at the intermediary level. However, important internal risks include relying on the validity of server certificates and client settings: disabling verification (setting Insecure) or leaking credentials on the device invalidates protection. Hacking a cloud server or leaking session data via the cloud also puts exposed messages at risk, even with the TLS protocol, so in projects with more sensitive data, the device itself needs to be secured to avoid risks related to it.

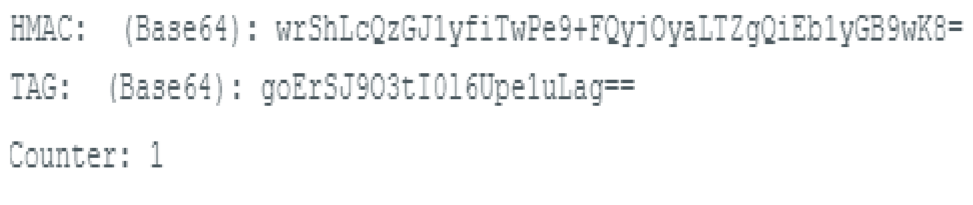
6.2.1.2 The Kyber handshake

The phase creates a strong shared session secret between sender and receiver using a post-quantum algorithm (post-quantum KEM). Within the system, this procedure provides a base layer for generating temporary keys that are later used to encrypt data and authentication derivatives, reducing reliance on fixed keys and reducing the risk of exposing a long-term secret. On the other hand, hacking the device's secret key (sk) represents a serious risk; leaking this key leads to losing the confidentiality of all sessions derived from it. Also, any incorrect implementation of the Kyber algorithm (side leaks, poor memory management, or receiving fake ciphertexts without authentication) may enable an attacker to manipulate the derivation result or exploit the exchange of values (ct) to cause substitutions or duplicates in the derivation of ss. Therefore, to increase security in the network, the system settings require generating a new key every transmission cycle, and the key is also sent with Base64 encoding, which makes it difficult to know directly, and when the key is accessed, it has changed many times.

6.2.1.3 AES-GCM Encryption

The role and limitations of integrated cryptography AES-GCM encryption in the system perform a dual function: keeping sensitive values (such as humidity values) confidential and providing a check mark (tag) for the integrity of the content within the application context. AES-GCM also ensures that any switching in the encrypted text will cause the tag verification to fail, thus rejecting the message before it is delivered. The immediate risks lie in nonce/IV management; reusing IV or generating IV repeatedly undermines GCM safeguards and allows an attacker to infer connections between different messages. In addition, deriving or storing the key insecurely puts all encrypted texts at risk, and errors in handling the results of the mbedtls_gcm_auth_decrypt function may lead to accepting forged texts. Encryption via AES-GCM may cost additional time for operations, as we mentioned in the previous results, but it still provides fast and secure high-speed symmetric encryption suitable for resource-limited devices such as ESP32.

6.2.1.4 HMAC - Verify the Authenticity of the Content



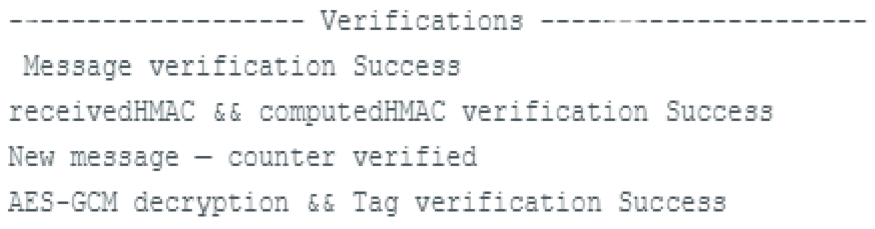
```
IV:  (Base64): b11V+/EtkhE3q2OQ
HMAC:  (Base64): wrShLcQzGJ1yfiTwPe9+FQyjOyaLTZgQiEb1yGB9wK8=
TAG:  (Base64): goErSJ9O3tI016Upe1uLag==
Counter: 1
```

Figure 6.3: HMAC Fields

Calculating HMAC (Hash-based Message Authentication Code) on a set of fields (such as ciphertext || tag || iv || counter) provides additional class verification independent of the transport channel. Within the system, HMAC acts as an immune barrier against attempts to mix (mix-and-match) between fields taken from different messages, because any switch in any field will cause the HMAC value to differ, and verification will not pass. Internal risks include key mismanagement (using the same key to encrypt and authenticate or leaking an HMAC key) that reduces the effectiveness of protection. Also, performing a normal HMAC comparison (not fixed in time) may facilitate time leakage attacks (timing attacks) for a party capable of accurate measurement, and the unstable arrangement of input fields to the HMAC function may lead to legitimate rejection or logical loopholes. Adding and verifying HMAC may represent additional time for the system, but it is not large and is suitable for devices with limited resources without causing any technical problems. In the proposed system, a new HMAC generation was adopted in each message, which is renewed every time iv, counter, or tag is renewed.

6.2.1.5 Verifications

This part includes adding a set of verifications to verify the authenticity of the message, as shown in the following image. It should be noted that verification operations are carried out sequentially in the system, and when one of these operations is not verified, the system stops working.



```
------------------ Verifications --------------------
 Message verification Success
receivedHMAC && computedHMAC verification Success
New message — counter verified
AES-GCM decryption && Tag verification Success
```

Figure 6.4: System Verifications

6.2.1.5.1 Verify Message Length: Check message length

Strict verification of message length when entering the decompilation path protects the system from read states outside the array limit or logical errors when dividing fields. Internally, this scan prevents manipulation attacks that attempt to introduce false sizes to destabilize segmentation logic or cause memory overruns. The associated risks are the possibility of setting the threshold incorrectly, which leads to rejecting correct messages (false positives), or giving false confidence if the verification is superficial and does not guarantee the correctness of the internal fields, which requires precise adjustment of the size of the fields to be used, so it is not enough to check the length alone; it must be followed by a real HMAC/tag. This verification may increase the message size slightly or add additional verification time, but it is not so great as to disable the system, and the added time can be exceeded in order to achieve better security. (Len, Grubbs & Ristenpart ,2021)

6.2.1.5.2 Verify HMAC: Applied Integration and Source Verification

Applying the HMAC verification step in the future ensures that all collected fields have not been changed and that the sender has the correct secret key. In the system architecture, this prevents manipulation of values, hinders the recombination of fields from previous messages, and provides practical evidence of the source of the message within the key frame. Internal risks stem from the possibility of HMAC key theft on the device, time-unsafe comparison, or HMAC calculation in the wrong field order—all of which may impair the effectiveness of this layer. Therefore, when recalculating HMAC, accuracy must be taken into account, and the fields must be arranged in the same order in which they were inserted into the original HMAC. Adding this verification may cause additional time and memory consumption, but this can be compensated for by the security that the HMAC calculation adds to the system. In the proposed system, the field (computed HMAC) represents the additional recalculated field.

6.2.1.5.3 Verify Counter: Prevent Retransmission (Replay Protection)

Counter value increment verification provides a simple and effective mechanism to prevent receiving outdated or retransmitted messages. Within your system, this prevents executing outdated commands (e.g., running a previous pump) and enforces a logical chronology of events. Risks related to the application include loss of synchronization between parties. Also, not storing the Last Counter Value securely or not protecting it from tampering may provide a vulnerability that allows protection to be bypassed, so the counter must be set up, stored, and modified with the correct mechanism that ensures the appropriate value is attached to each message. (Zelle, & Gürgens,2021)

6.2.1.5.4 Verify AES-GCM & Tag

Verification by (tag) before decryption is the last line of defense against any accidental or malicious modification to the encrypted text or its associated fields. Internally, the success of this step represents mathematical proof that the ciphertext has not changed and that the IV/tag is consistent with the key, thus combining confidentiality (encryption) and authentication/integration (tag verification) into a single pattern (AEAD), reducing the need for a separate MAC or additional operations. This verification acts as a powerful way to prevent modification (tampering) or injection (injection) attacks, because any change in encrypted text will cause the tag to fail upon verification. Emerging risks include false acceptance if the tag length is not achieved or if the return value (ret) of the decryption function is ignored; therefore, the field used must be precisely defined and its length set to ensure successful verification.

.

6.2.1.6 Decrypt: Data recovery after verifications

Decryption after going through all the verification stages ensures that the data delivered to the top layer is intact and confidential. Within the system, this arrangement reduces the risk of forged or manipulated data being delivered to the application that controls physical devices. It must be taken into account not to keep clear text (plaintext) in memory after use because this exposes it to subsequent extraction; therefore, the memory must be cleaned immediately after use ends and not decrypted before passing all verifications.

6.2.1.7 Deliver Data to Application: Achieving system goals in the application

Delivering the decoded and verified value to the application enables a control decision (e.g., pump operation) to be made based on reliable data. Its internal advantage is to provide a reliable and simplified decision point for the application, but its risks lie in relying on a single signal without additional policies (delays, multiple trials, or time limits) that may cause erroneous actions when there are spurious cases or sensitive thefts of the scale. The actual action must also be implemented via other security policies (e.g., second verification or protected operating privileges).

Finally, although this integration adds a simple computational load and increases handshake time and power consumption in resource-constrained systems, it achieves a multi-layered defense model (Defense-in-Depth) that enhances the confidentiality, integrity, and reliability of data transmission over the cloud, making it a suitable framework for securing sensitive IoT applications and modern intelligent industrial systems.

6.2.2 MQTT Based-System vs Attacks

In this section, we review the attacks that each element of the proposed system protects against:

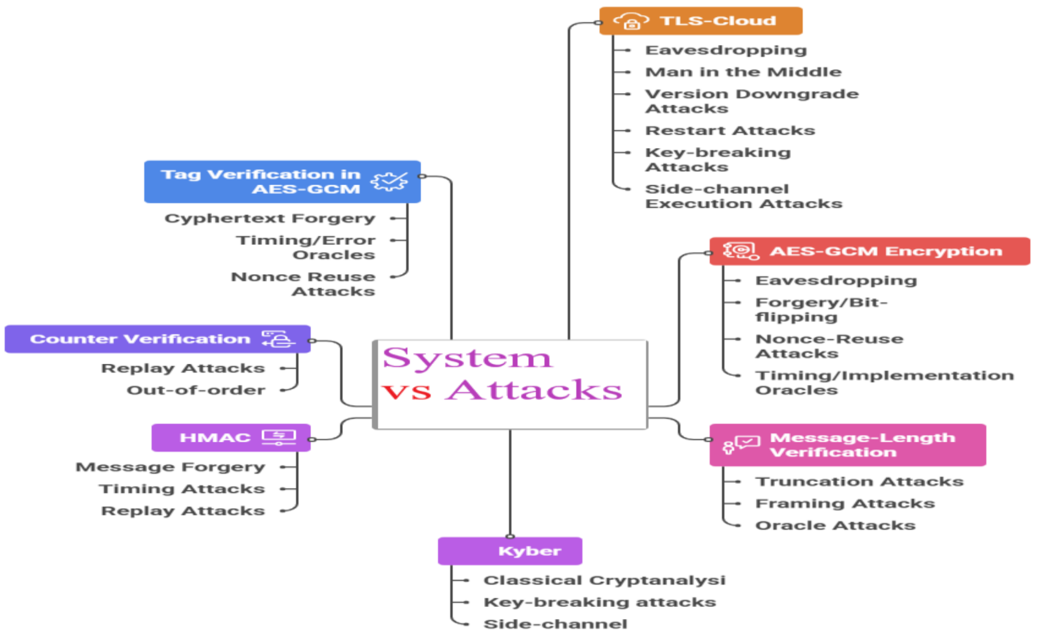


Figure 6.5: MQTT Based-System Vs. Attacks

6.2.2.1 TLS-Cloud:

TLS is the cornerstone of securing cloud communication channels, providing mutual authentication and end-to-end encryption of transmitted data. TLS protects against the following attacks:

1: Eavesdropping: An attacker (sniff) passes network packets to intercept session data, aiming to read unencrypted content. TLS prevents this by exchanging session keys and encrypting the entire payload with the transport layer, so even if an attacker intercepts packets, they cannot interpret the explicit text without the session keys.

2: Man in the Middle (MITM): An attacker interrupts the starting negotiation (handshake) and tries to convince each party that they are the other party to steal keys or change data. TLS 1.3 uses certificate-based authentication, temporary keys (ephemeral keys), and version downgrade prevention techniques, which significantly reduce the chances of MITM success if certificates and trust chains are intact.

3: Version downgrade attacks (downgrade): An attacker forces parties to use a weaker protocol or setup to exploit old vulnerabilities. The TLS 1.3 design eliminates insecure negotiation options and ensures mechanisms that prevent downgrades while being stringent on signature algorithms and version negotiation.

4: Restart attacks: if link properties are not applied (replay/resource exposure): TLS provides lifetime mechanisms for sessions (session recovery), but incorrect implementation can leave vulnerabilities; therefore, cloud applications must define strict session policies.
TLS works because it combines key exchange encryption (authenticated key exchange), party/party certification (certificate-based authentication), and temporary keys that reduce the impact of a long-term key hack. Thus, it prevents content from being detected or forged on the transmission channel as long as the two parties have corrected chains of trust.

6.2.2.2 Kyber Handshake (CRYSTALS-Kyber / ML-KEM)

Kyber is a NIST-approved post-quantum encryption algorithm (ML-KEM), designed to secure key exchange, making it an ideal choice for securing future IoT systems. Kyber provides long-term secrecy (long-term confidentiality) by generating session keys that cannot be broken even with a future quantum computer, and protects Kyber's algorithm from the following attacks:
1: Key-breaking attacks using quantum computing (quantum key recovery): Algorithms such as Shor will break RSA/ECC when a large quantum computer is available; Kyber is built on network problems (lattice-based) that are resistant to currently known quantum acceleration, making key inference from computational exchanges impractical even for a quantum attacker expected in the medium term.
2: Mathematical/Classical Analysis (Classical Cryptanalysis): The resistance of the Kyber algorithm to mathematical analysis attacks/equation attacks depends on the selection of appropriate criteria (parameters); Kyber criteria included in standard assignments (FIPS/NIST) are designed to provide robustness compared to a specific security level (e.g., a level corresponding to AES-128 or higher depending on the version).
3: Side-channel execution attacks (side-channel/fault attacks): Unsafe execution (such as time leaks, power leaks, or value errors) may allow keys to be inferred even if the algorithm itself is robust. Therefore, practical protection requires countermeasures (constant-time operations, masking, and fault detection). NIST documents emphasize monitoring of Kyber implementation.
Kyber succeeds because it ensures that keys are exchanged in an institutionalized manner to resist future quantum attacks as long as the implementation is implemented

with side-channel resistance measures and updated in accordance with international standards and guidelines.

6.2.2.3 AES\GCM Encryption:

AES-GCM provides authentication-synchronous symmetric encryption (AEAD) that protects data from many attacks, such as

1: Eavesdropping: AES-GCM text is encrypted, so intercepting packets does not reveal encrypted data. However, if the nonce/IV is reused with the same key, it may reveal patterns or enable an attacker to infer a relationship between two encrypted texts, leading to a loss of confidentiality.

2: Forgery/Bit-flipping: AES-GCM generates a tag (authentication tag) via GHASH covering the text and accompanying data; any modification will usually result in the tag failing to be verified upon opening, preventing the acceptance of forged text. However, poor verification (such as accepting messages before full verification or leaking various error messages) may open oracle attacks.

3: Nonce-Reuse Attacks: The most common AES-GCM errors; reusing a nonce with a single key may result in partial decryption or create the possibility of forgery. Technical documents and practical reports encourage the use of unique nonces or the transition to error-resistant patterns when necessary.

4: Timing/Implementation Oracles: Response time variation or error messages when verification fails may be used to extract information; therefore, verification must be time constant, and rejecting authentication must give uniform behavior.

The AES-GCM model is very powerful when keys are generated and managed correctly, and verifications are performed correctly.

6.2.2.4 Message-Length Verification:

Used when an attacker attempts to shrink or delete a piece of data to change its meaning or system function, and when comparing the expected length of the message prepared within the system with the length of the received message. Any change in length becomes detectable immediately upon verification, preventing the acceptance of truncated or incomplete messages. It is a simple but necessary measure to ensure data consistency and prevent attacks based on response analysis, including:

1: Truncation Attacks: An attacker deletes the end of a message (e.g., promoting an important part of a command) or shortens it to cause unexpected behavior in the recipient (e.g., deleting a pause field). Comparing the declared length to the actual length and including the length field within MAC/AEAD prevents acceptance of truncated messages.

2: Framing Attacks: An attacker exploits vulnerabilities in the encapsulation protocol to pass fragmented messages or merge multiple messages, leading to misinterpretations; Strict verification of length and frame boundaries interrupts these attempts.

3: Oracle attacks built on different lengths: Differences in server response to messages of different lengths may reveal information that will be exploited later; therefore, your application's handling of different errors and logs should be as uniform as possible.

This verification is successful when the length of the fields is precisely defined and compared correctly.

6.2.2.5 HMAC (Account and Verification):

HMAC is used to ensure the integrity and reliability of data by generating a verification code based on a hash function and a secret key. HMAC is an essential complement to encryption protocols in industrial systems and protects against the following attacks:

1: Message Forgery: If a strong MAC is not used or if an attacker can guess the HMAC key, a valid MAC may be generated for a forged message. HMAC prevents this as long as the key is secret and the hash function is strong (such as SHA-256 or SHA-3).

2: Timing Attacks: Comparing HMAC in a regular way (byte-by-byte) match lengths may be revealed via response time variation; comparison execution must be constant-time (constant-time) to avoid this attack. This protects against generating a new HMAC every time a connection is established, as in the proposed system.

3: Replay paid with missing counters: If HMAC is not combined with a counter or nonces, HMAC may allow accepting old retransmitted messages provided the MAC is still valid; therefore, it is recommended to combine time measurements or serial numbers within the payload covered by HMAC, as was done in the proposed system.

HMAC builds on the properties of collision-resistant hash functions and provides a simple and reliable method for verifying the integrity and provenance of messages, provided the keys and timing are managed, and the comparison is performed correctly, as recommended by recent NIST documentation.

6.2.2.6 Counter Verification

By adding a serial number to the message and by comparing the serial number of each message with the last accepted value, the system can reject any duplicate or delayed packet, preventing malicious control of system components. This mechanism is essential in time-sensitive IoT environments, and attacks on the meter include:

1: Replay Attacks: An attacker records legitimate messages and later rebroadcasts them to cause a duplicate situation or execute unwanted commands. When comparing the received counter with the message with the previously stored counter, duplicate messages are detected and prevented from being executed.

2: Out-of-order delivery explosion: In delayed networks, an attacker may attempt to mix sequences; sliding window policies (sliding window) and timestamp checking help balance accepting delayed packets and preventing reboots.

The counter succeeds when the status is saved using the Tamper-Resistant Storage method, and the counter or timestamp is included within the documentation field (HMAC/AES-GCM) so that its manipulation becomes detectable. As in the proposed system, the counter is saved within the HMAC field to compare with a previously saved value that is modified with each message.

6.2.2.7 Tag Verification in AES-GCM

Tag verification indicates that data has been modified or generated from an unauthenticated source. Failure of this verification indicates a difference in data, which makes it easier for the system to verify and reject it when it differs. Among the attacks it detects are:

1: Cyphertext Forgery: An attacker generates or changes a cyphertext; if the content changes, the GHASH/Tag will not match when opened, and the request is rejected. Ensuring verification before processing prevents the execution of any command based on forged data.

2: Timing/Error Oracles: A different response time when a verification fails may form a channel for inferring information about the key or the internal structure; therefore, the verification process must be time-invariant and exhibit uniform error behavior.

3: Nonce reuse attacks affect Tag: Nonce reuse can affect the falsifiability or exposure of parts of text, so managing nonces is crucial.

The system works when you always check the tag before any load handling. Use a fixed formula to compare tags while accurately defining system elements

6.3 LoRa -Based System

6.3.1 Security Analysis:

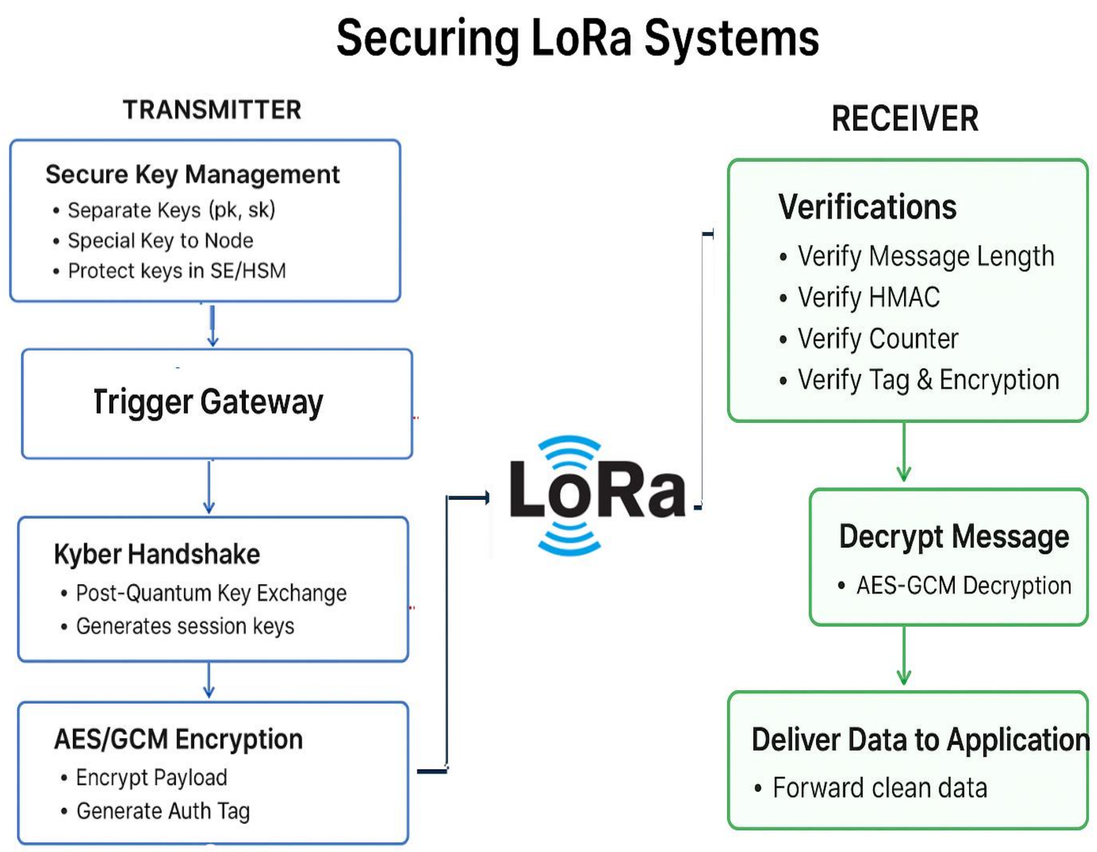


Figure 6.6: Secure LoRa System

Most of the steps mentioned are similar to the insurance steps in the previous section, so we will not re-explain them. We will only explain the additional steps here.

6.3.1.1 Secure Key Management

Secure Key Management is one of the most important pillars of the security of Internet of Things systems, as it ensures that keys are generated, stored, and distributed in a way that maintains their confidentiality and prevents unauthorized access. This mechanism helps protect encryption and authentication processes from attacks such as key extraction and man-in-the-middle, and supports periodic key renewal to enhance security. In resource-limited systems such as ESP32, built-in security protocols and components (HSM/SE) are relied upon to protect private keys and ensure the integrity of communication between nodes.

6.3.1.1.1 Separate Key (pk, sk)

Separating keys between different functions, where the gateway has a public key that sends public messages to all nodes, such as PK when changing keys, and a private key for the gateway with each node, which enhances flexibility and security and makes the system more resistant to complex attacks. This is because it reduces the impact of any single key leak and prevents its reuse in insecure contexts. It provides a high degree of security because it reduces the risk of any partial breach.

6.3.1.1.2 Special Key to Node

Assigning a key to each node prevents impersonation and collisions between nodes and limits the spread of any potential hack. The degree of security is high because it reduces the impact of any leak of a public key and makes each node security independent, and when sending any message, the name of the node sent to verify the key appears.

6.3.1.1.3 Protect Keys In SE/HSM

Keys are stored within secure physical components such as a secure element (Secure Element—SE) or a physical key management module (Hardware Security Module—HSM), where sensitive keys are isolated from the processor's main memory so that they cannot be accessed even if the system is hacked or the device's memory is read. When this feature is enabled, all cryptographic operations, such as generation, encryption, and decryption, are performed within the secure module without the keys exiting the normal execution environment, significantly reducing the possibility of keys being leaked or exposed to side-channel attacks (Side-Channel Attacks) or error injection (Fault Injection). Although this may result in a slight decrease in performance due to communication processes between the processor and the secure unit, it significantly enhances the level of security and reliability. If ESP32 modules are used in applications such as smart irrigation systems based on post-quantum cryptography (Post-Quantum Encryption – Kyber), this option allows Kyber, AES, and HMAC keys to be stored within the secure environment, ensuring that the key material does not leave the protected memory. It is recommended that this feature be activated on permanently operating field devices, while it is preferable to disable it during development to facilitate debugging and testing processes.

6.3.1.2 Trigger Gateway

This feature represents an additional operational security layer within the security architecture of Internet of Things (IoT) systems based on LoRa and ESP32 networks and aims to securely control the timing and response of the communication gateway (Gateway) based on reliable cryptographic verification. This feature prevents the gateway from executing critical commands—such as turning on the pump or sending session keys—unless it receives an encrypted, digitally signed signal or guaranteed verification from a trusted node (Node). The security in this feature is based on integrating cryptographic authentication (Authenticated Encryption) using AES-GCM or HMAC with dedicated keys for each node (Per-Node Keying), making each trigger signal unique and not repeatable or falsifiable. The gateway also only responds when it checks the message length (length verification) and the integrity of the verification tag (tag/HMAC verification) before activating any physical component, such as the pump.

6.3.1.3 Kyber Handshake (explained before)

6.3.1.4 AES/GCM Encryption (explained before)

6.3.1.5 Verification (explained before)

6.3.1.6 Decrypt Message (explained before)

6.3.1.7 Deliver Date to Application (explained before)

6.3.2 System vs Attacks

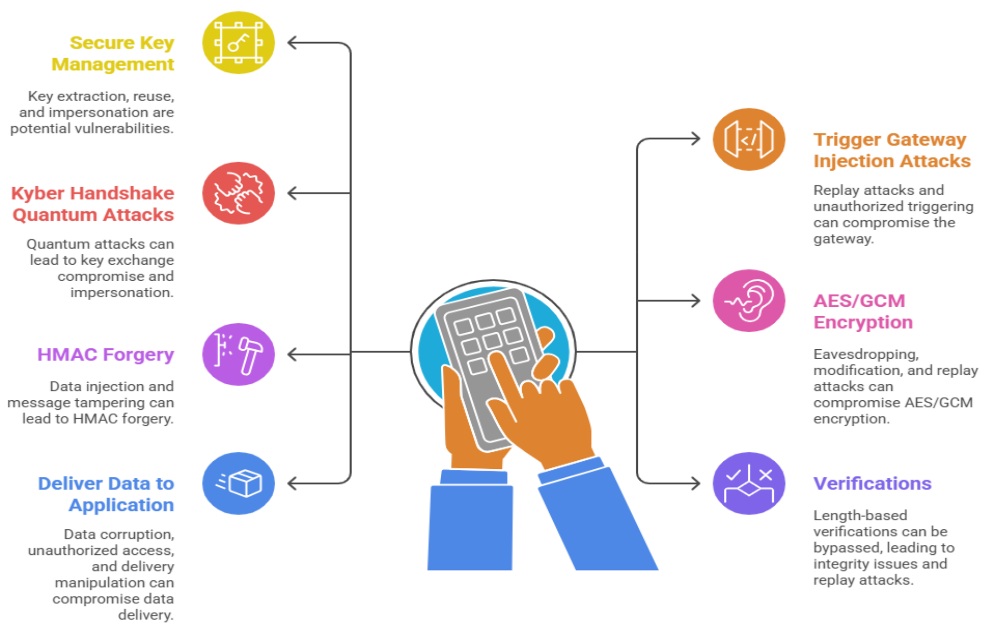

Secure Key Management
Key extraction, reuse, and impersonation are potential vulnerabilities.
Kyber Handshake Quantum Attacks
Quantum attacks can lead to key exchange compromise and impersonation.
HMAC Forgery
Data injection and message tampering can lead to HMAC forgery.
Deliver Data to Application
Data corruption, unauthorized access, and delivery manipulation can compromise data delivery.
Trigger Gateway Injection Attacks
Replay attacks and unauthorized triggering can compromise the gateway.
AES/GCM Encryption
Eavesdropping, modification, and replay attacks can compromise AES/GCM encryption.
Verifications
Length-based verifications can be bypassed, leading to integrity issues and replay attacks.

Figure 6.7: LoRa Based-System vs Attacks

The previous image shows the attacks on LoRa communication systems, a number of which were explained in the previous item, so in this item, we will explain the attacks that were not explained previously.

6.3.2.1 Secure Key Management

6.3.2.1.1 Key Mining Attacks:

It aims to steal private keys or session keys from within devices. They are often performed using energy analysis methods or side channels. Secure Key Management stores keys within physical security elements (Secure Element/HSM) and prevents direct access to them.

6.3.2.1.2 Key Reuse Attacks:

It occurs when the same key is reused to encrypt multiple sessions, making it easier for an attacker to analyze patterns and discover the key. This problem is solved by rotating keys periodically and generating new session keys via Kyber Handshake.

6.3.2.1.3 Impersonation Attacks:

Impersonation occurs when a malicious entity attempts to pretend to be a trusted node or source within a network to trick the gateway or other nodes into accepting fake messages or commands. In the context of IoT networks, impersonation can enable an attacker to control sensitive devices or enter false data that affects automated decisions and control systems.

6.3.2.2 Trigger Gateway

6.3.2.2.1 Unauthorized Command Injection

Unauthorized command injection is a form of active attack in which an attacker seeks to send malicious or modified control instructions to a remote-control system, aiming to change the behavior of the device or network without having legitimate powers. This attack becomes effective when there are no strong mechanisms to verify the source and authenticity of the message. From a system design perspective, mechanisms such as HMAC with unique node keys, AEAD (such as AES-GCM) for locking and authentication, and gateway executive authority restrictions are crucial tools for raising resistance to such attacks.

6.3.2.2.2 Unauthorized Remote Control / Remote Takeover

This type of attack aims to gain physical control over end devices by sending seemingly legitimate commands or exploiting software vulnerabilities in the control gateway and may disrupt services or cause physical damage to systems connected to the physical world. The attack succeeds when the channel lacks strong authentication, which fulfills command execution powers, or when there are vulnerabilities that allow authorization logic to be bypassed.

6.3.2.1 Kyber Handshake (explained before)

6.3.2.2 AES/GCM Encryption (explained before)

6.3.2.3 Verification (explained before)

6.3.2.4 Decrypt Message/ hmac Foregery (explained before)

6.3.2.5 Deliver Data to Application (explained before)

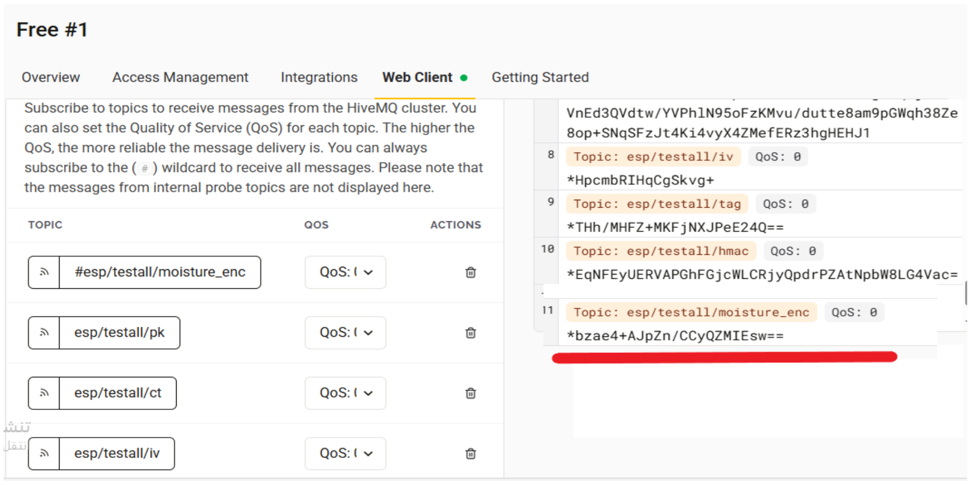
6.4 Security Test Results

In this section, we discuss the results of carrying out several attacks to prove the system's ability to confront cyber-attacks, where the security robustness of proposed irrigation systems against common network attacks is evaluated, especially Replay attacks, and two tests of the Tampering Attack.

6.4.2 Replay Attack

A replay attack is a security threat in which an attacker captures a legitimate message exchanged between two parties and later retransmits it to produce an unauthorized effect or deceive the system.

6.4.2.1 Capture a Message Sent from the Broker



Subscribe to topics to receive messages from the HiveMQ cluster. You can also set the Quality of Service (QoS) for each topic. The higher the QoS, the more reliable the message delivery is. You can always subscribe to the (#) wildcard to receive all messages. Please note that the messages from internal probe topics are not displayed here.

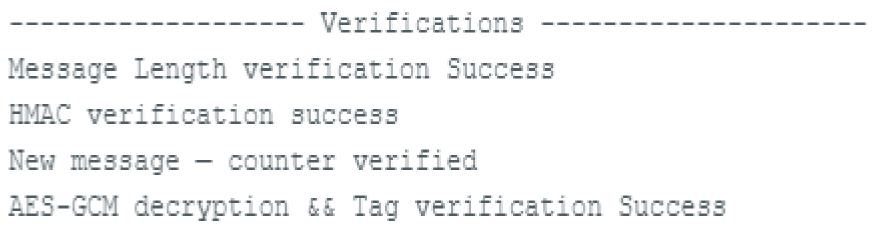
TOPIC	QOS	ACTIONS
#esp/testall/moisture_enc	QoS: (	
esp/testall/pk	QoS: (	
esp/testall/ct	QoS: (	
esp/testall/iv	QoS: (	

```
   VnEd3QVdtw/YVPhlN95oFzKMvu/dutte8am9pGWqh38Ze
   8op+SNqSFzJt4Ki4vyX4ZMefERz3hgHEHJ1
8  Topic: esp/testall/iv  QoS: 0
   *HpcmbRIHqCgSkvg+
9  Topic: esp/testall/tag  QoS: 0
   *THh/MHFZ+MKFjNXJPeE24Q==
10 Topic: esp/testall/hmac  QoS: 0
   *EqNFEyUERVAPGhFGjcWLCRjyQpdrPZAtNpbW8LG4Vac=
11 Topic: esp/testall/moisture_enc  QoS: 0
   *bzae4+AJpZn/CCyQZMIEsw==
```

Figure 6.8: Capture a Message

Finding any message from the broker that has been sent and accepted by the system

6.4.2.2 System Verification for First Sending

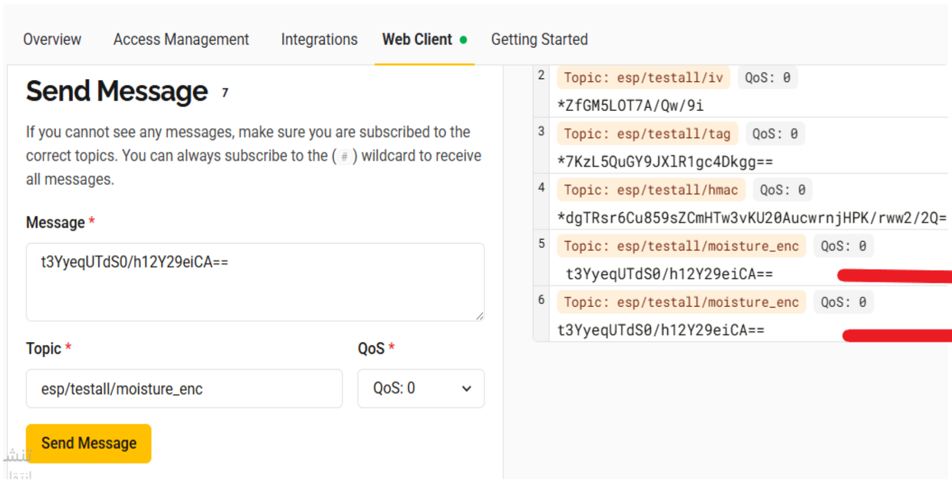


```
------------------- Verifications ---------------------
Message Length verification Success
HMAC verification success
New message – counter verified
AES-GCM decryption && Tag verification Success
```

Figure 6.9: System Verifications for First Sending

The image shows that the message has exceeded all security checks in the system, which indicates that it is a reliable message. Now we will resend it without any change.

6.4.2.3 Resending Message



Send Message

If you cannot see any messages, make sure you are subscribed to the correct topics. You can always subscribe to the (#) wildcard to receive all messages.

Message *: t3YyeqUTdS0/h12Y29eiCA==

Topic *: esp/testall/moisture_enc

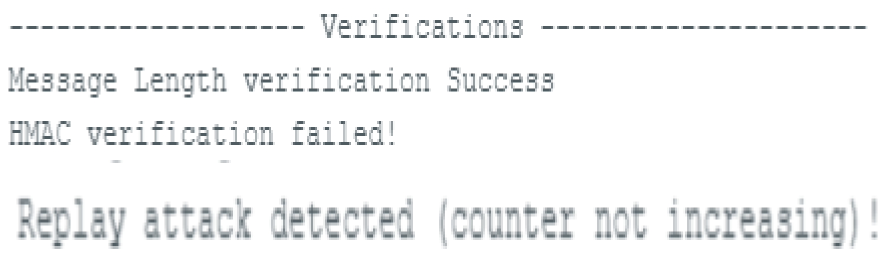
QoS *: QoS: 0

```
2  Topic: esp/testall/iv   QoS: 0
   *ZfGM5LOT7A/Qw/9i
3  Topic: esp/testall/tag   QoS: 0
   *7KzL5QuGY9JXlR1gc4Dkgg==
4  Topic: esp/testall/hmac   QoS: 0
   *dgTRsr6Cu859sZCmHTw3vKU20AucwrnjHPK/rww2/2Q=
5  Topic: esp/testall/moisture_enc   QoS: 0
    t3YyeqUTdS0/h12Y29eiCA==
6  Topic: esp/testall/moisture_enc   QoS: 0
   t3YyeqUTdS0/h12Y29eiCA==
```

Figure 6.10: Resending Message

Now we copy the message and resend it as anew message

6.4.2.4 System Verification after Resending



```
------------------- Verifications ---------------------
Message Length verification Success
HMAC verification failed!

Replay attack detected (counter not increasing)!
```

Figure 6.11: System Verifications for Resent Message

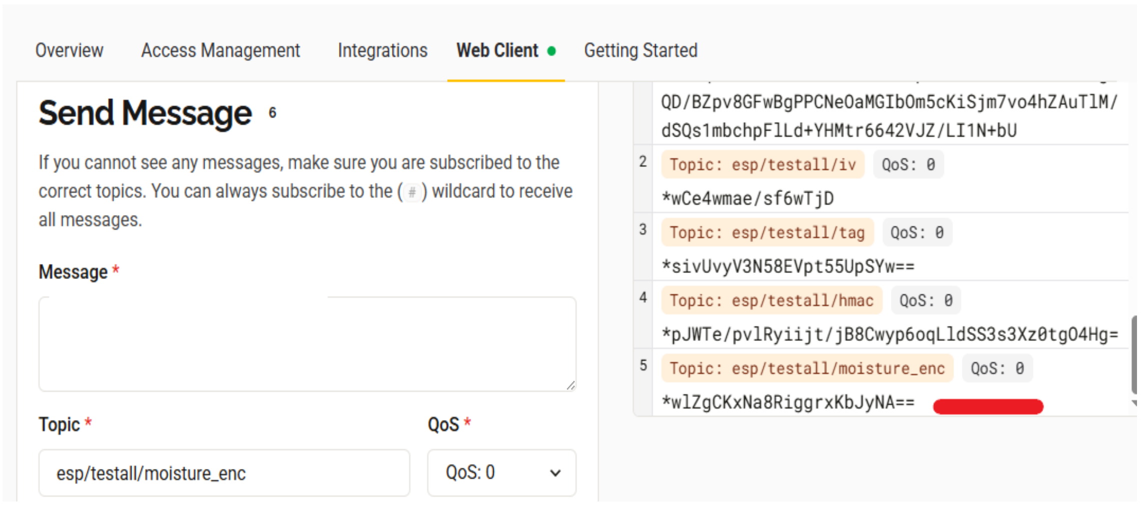
When the signal is picked up, the system checks it and finds that it exceeds the first check, which is the length of the message. But it does not exceed the second check, which is HMAC, as the counter is repeated in the two messages, which makes the system reject HMAC, and the system sends a message so that it shows the frequency of the count and its non-increase, which shows that the message was sent previously.

The above message is an experimental demonstration of the effectiveness of the integration, authentication, and retransmission attack prevention layers in the system architecture. Analytically, it is recommended to document the associated log segments, verify the synchronization of HMAC switches and counter values at the transmitter, and indicate the result in the security analysis section as confirmation of the system's resistance to this type of attack.

6.4.3 Tampering Attack (Test1)

The concept of tampering refers to unauthorized modification of the content of messages or data during transmission and is considered one of the most serious types of attacks targeting information integrity (data integrity). Secure systems rely on integration verification algorithms such as HMAC or digital signatures to detect this type of manipulation. When a difference in the security tag is detected, the message is considered modified (tampered) and refused acceptance, demonstrating the effectiveness of the protection layer in ensuring the reliability of the transmitted data.
As in the steps in the previous experiment, we capture a message, but this time we will change the content of the message by deleting the last two characters of the message to verify the system's ability to detect the error.

6.4.3.1 Capture a Message Sent from the Broker



If you cannot see any messages, make sure you are subscribed to the correct topics. You can always subscribe to the (#) wildcard to receive all messages.

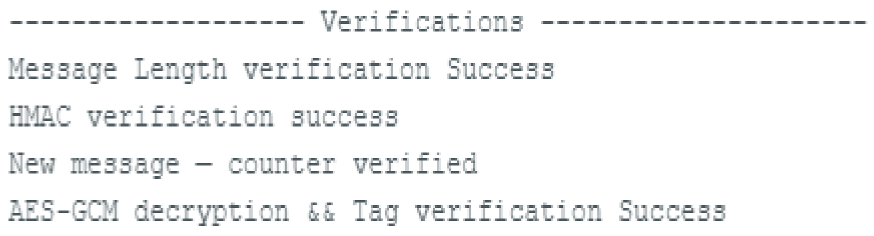
Topic: esp/testall/moisture_enc
QoS: 0

```
  QD/BZpv8GFwBgPPCNeOaMGIbOm5cKiSjm7vo4hZAuTlM/
  dSQs1mbchpFlLd+YHMtr6642VJZ/LI1N+bU
2 Topic: esp/testall/iv  QoS: 0
  *wCe4wmae/sf6wTjD
3 Topic: esp/testall/tag  QoS: 0
  *sivUvyV3N58EVpt55UpSYw==
4 Topic: esp/testall/hmac  QoS: 0
  *pJWTe/pvlRyiijt/jB8Cwyp6oqLldSS3s3Xz0tgO4Hg=
5 Topic: esp/testall/moisture_enc  QoS: 0
  *wlZgCKxNa8RiggrxKbJyNA==
```

Figure 6.12: Capture a Message

Finding any message from the broker that has been sent and accepted by the system.

6.4.3.2 System Verification for First Sending

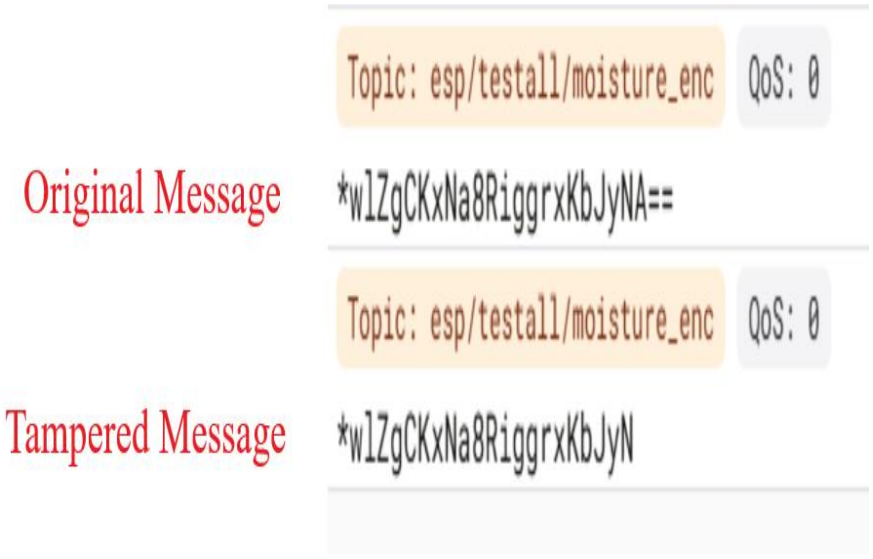


```
------------------- Verifications ---------------------
Message Length verification Success
HMAC verification success
New message - counter verified
AES-GCM decryption && Tag verification Success
```

Figure 6.13: System Verifications for First Sending

The image above shows that the message has exceeded all security checks in the system, which indicates that it is a reliable message. Now we will resend it after **deducting the last 3 codes from it.**

6.4.3.3 Resending Message

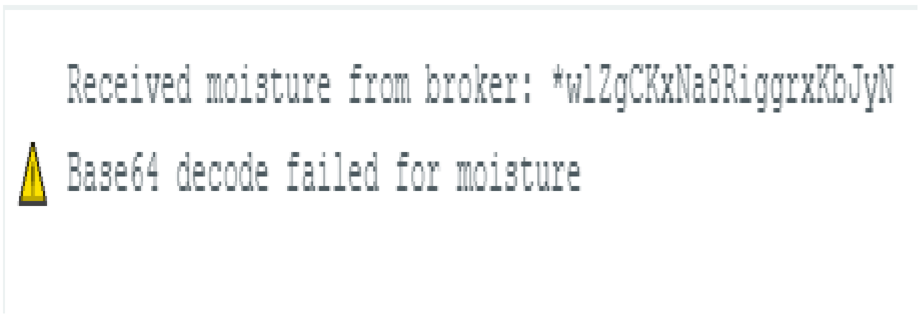


```
Topic: esp/testall/moisture_enc    QoS: 0
*wlZgCKxNa8RiggrxKbJyNA==
Topic: esp/testall/moisture_enc    QoS: 0
*wlZgCKxNa8RiggrxKbJyN
```

Figure 6.14: Resending Message

The previous image represents the message after changing and resending. The image shows the original text and the text after cutting from it and resending.

6.4.3.4 System Verification after Resending



```
Received moisture from broker: *wlZgCKxNa8RiggrxKbJyN
⚠ Base64 decode failed for moisture
```

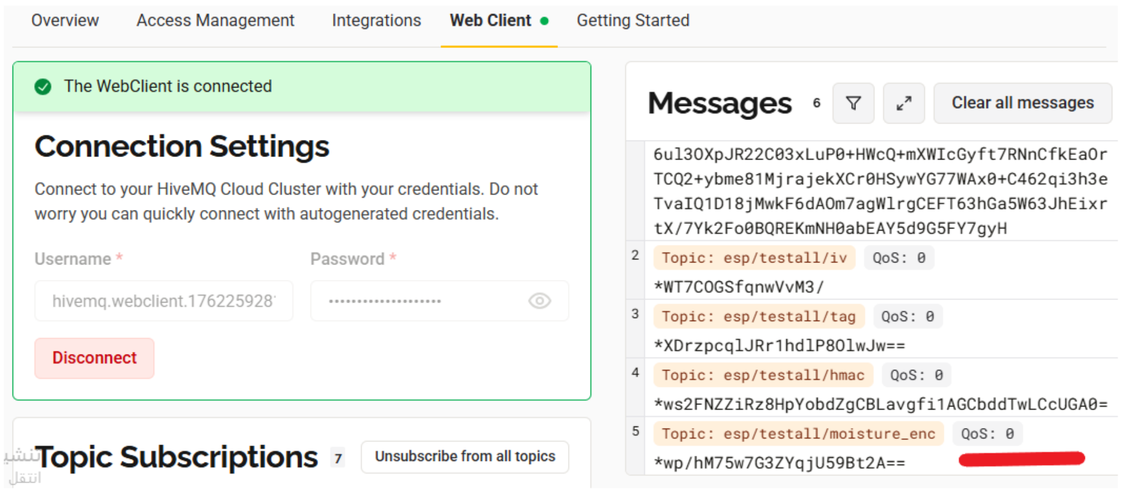
Figure 6.15: System Verifications after Resent Message

After re-sending and receiving the message, the system verifies it and finds that the length of the message does not exceed the initial length test necessary to decode it.

6.4.4 Tampering Attack (Test2)

In another experiment, the same message was retransmitted with a change in its letters.

6.4.4.1 Capture a Message Sent from the Broker

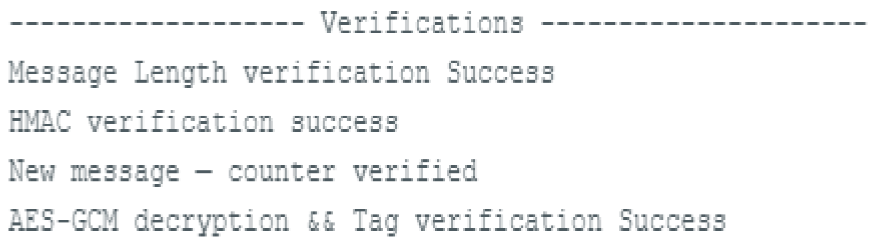

Overview
Access Management
Integrations
Web Client
Getting Started
The WebClient is connected
Connection Settings
Connect to your HiveMQ Cloud Cluster with your credentials. Do not worry you can quickly connect with autogenerated credentials.
Username *
hivemq.webclient.176225928
Password *
Disconnect
Topic Subscriptions 7
Unsubscribe from all topics
Messages 6
Clear all messages

```
6ul3OXpJR22C03xLuP0+HWcQ+mXWIcGyft7RNnCfkEaOr
TCQ2+ybme81MjrajekXCr0HSywYG77WAx0+C462qi3h3e
TvaIQ1D18jMwkF6dAOm7agWlrgCEFT63hGa5W63JhEixr
tX/7Yk2Fo0BQREKmNH0abEAY5d9G5FY7gyH
2  Topic: esp/testall/iv   QoS: 0
*WT7COGSfqnwVvM3/
3  Topic: esp/testall/tag   QoS: 0
*XDrzpcqlJRr1hdlP8OlwJw==
4  Topic: esp/testall/hmac   QoS: 0
*ws2FNZZiRz8HpYobdZgCBLavgfi1AGCbddTwLCcUGA0=
5  Topic: esp/testall/moisture_enc   QoS: 0
*wp/hM75w7G3ZYqjU59Bt2A==
```

Figure 6.16: Capture a Message

Finding any message from the broker that has been sent and accepted by the system.

6.4.4.2 System Verification for First sending

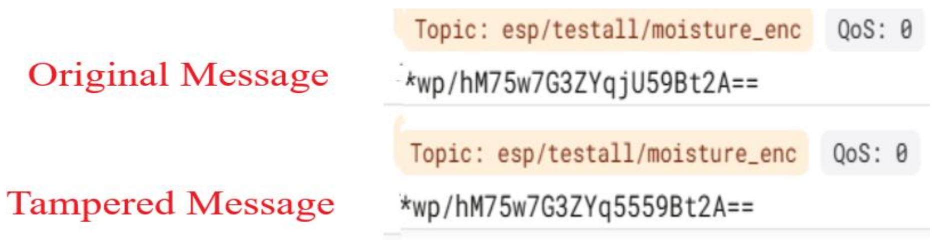


```
------------------- Verifications ---------------------
Message Length verification Success
HMAC verification success
New message - counter verified
AES-GCM decryption && Tag verification Success
```

Figure 6.17: System Verifications for First Sending

The image shows that the message has exceeded all security checks in the system, which indicates that it is a reliable message. Now we will resend it after deducting the last 3 codes from it.

6.4.4.3 Resending Message



Original Message

```
Topic: esp/testall/moisture_enc   QoS: 0
*wp/hM75w7G3ZYqjU59Bt2A==
```

Tampered Message

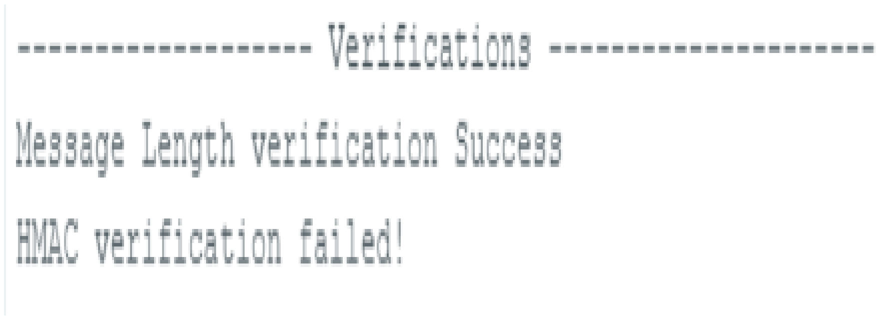
```
Topic: esp/testall/moisture_enc   QoS: 0
*wp/hM75w7G3ZYq5559Bt2A==
```

Figure 6.18: Resending Message

The previous image represents the message after changing and resending. The image shows the original text and the text after cutting from it and resending.

6.4.4.4 System Verification after Resending

The message was recognized, decoded, and passed the message length test, but did not pass the HMAC test because its actual content had changed. When the HMAC match between the sender and the receiver is tested, the test fails.



```
------------------ Verifications --------------------
Message Length verification Success
HMAC verification failed!
```

Figure 6.19: System Verifications for Resent Message

The program considered the message to be incorrect and could not be decoded because it did not exceed the required message length test before the verification series.
Finally, after conducting security experiments in an isolated laboratory environment, the proposed system demonstrated its ability to counter the most prominent types of potential attacks, including data manipulation and retransmission. The results showed that the integrated verification and authentication mechanism, based on the HMAC algorithm and a serial counter, successfully detected any unauthorized modification or duplication of messages, and all incorrect packets were immediately rejected without affecting the overall system performance. These results confirm the effectiveness of the proposed security layer in ensuring the integrity and reliability of data transmitted between nodes and intermediaries, and reflect a successful balance between operational efficiency and the security protection required in sensitive IoT environments. While some other attacks, such as Flooding Attack at Physical, cannot be fully addressed remain under study and work in the future.

6.5 Full Security Analysis

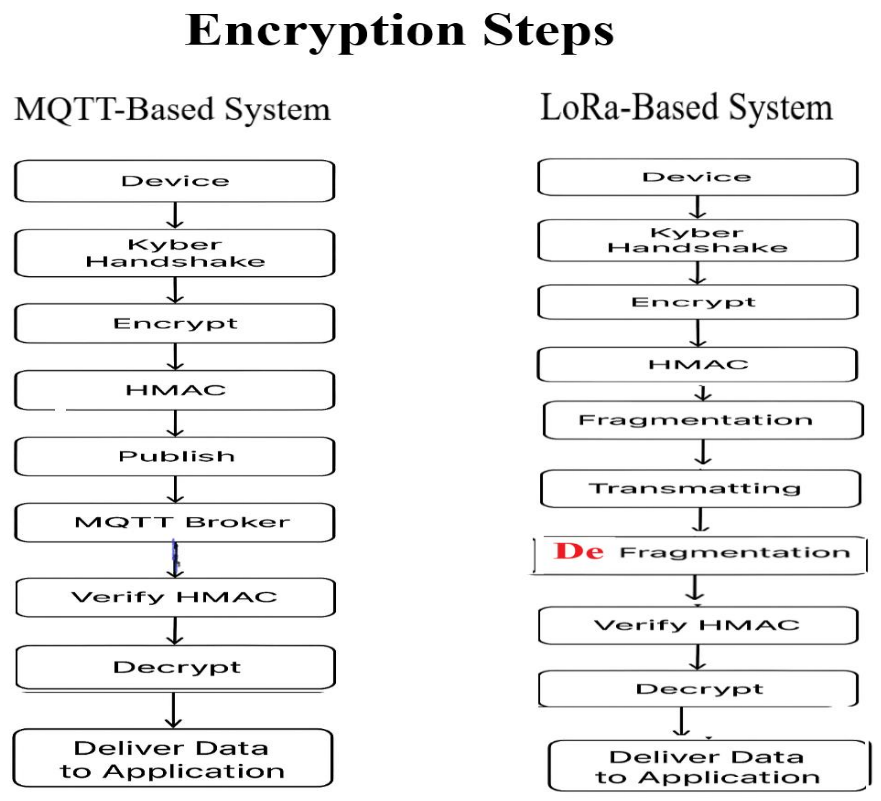


Figure 6.20: MQTT-Based Systems vs. LoRa-Based Systems Encryption Steps

The image shows an in-depth comparison between the data security path in the MQTT-based system and the LoRa-based system when combined with the Kyber encryption algorithm, and clearly shows the impact of the structural structure and working mechanism of each protocol on the level of security, data integrity, and transfer efficiency. In both systems, the process starts from the device that exchanges the quantum keys after encryption (Kyber handshake) to secure a secret communication channel between the two parties. The data is then encrypted (encrypted), and an HMAC code is created to ensure its integrity and credibility. This is where the similarity between the two systems ends, where the structural difference that forms the core of the security effect begins.

In an MQTT system, encrypted data is sent via a central medium (MQTT Broker) that organizes communications between sender and receiver within a publisher/subscriber model (Publish/Subscribe). This design gives the system high efficiency in managing sessions and data flow and also enables additional layers of security, such as identity verification (authentication) and channel encryption using TLS, making communication more reliable within Internet-based networks. However, MQTT's reliance on this central broker means that the overall security of the system is directly related to the integrity of the intermediary server, as this component represents a potential vulnerability (single point of failure). If the broker is compromised or disabled, an attacker can intercept encrypted data or tamper with verification processes (Verify HMAC), making MQTT vulnerable to man-in-the-middle or broker compromise attacks if it is not adequately secured. However, this model remains ideal for applications that require rapid response and extensive data exchange, such as industrial systems or cloud environments.

While the Lora-based system relies on a distributed, low-power transmission mechanism over narrowband wireless channels. After the data is encrypted and a Message Authentication Code (HMAC) is generated, the messages are divided into small parts (hash) due to the small size of the LoRa frame, and then sent over the wireless network to be collected (disassembled) at the receiver. Although these additional steps increase processing and response time compared to the MQTT protocol, they improve the level of physical security of data during transmission, because LoRa uses low frequencies that are difficult to intercept or hack, and often operates in environments isolated from the public Internet, reducing the possibility of direct network attacks. The absence of a central intermediary also reduces the chances of organized hacking or server-level communications being cut off, making the Lora-based system more resistant to attacks directed against infrastructure, despite the limited data transfer rate. Thus, it becomes clear that the different nature of the system imposes a difference in the philosophy of security:

- ✓ In an MQTT-based system, security relies on centralization and network control, where keys and identity verification are managed via a fast but security-sensitive intelligent medium.
- ✓ While security in a LoRa-based system stems from the distributed physical layer and isolation from the Internet, this reduces the chances of hacking at the expense of speed and bandwidth.

Overall, the MQTT protocol provides highly efficient network security with low latency, but requires a robust, centrally protected architecture, while LoRa systems provide robust physical security and higher power efficiency, making them best suited for long-range applications with limited resources. This structural difference between the two models reflects two complementary security philosophies: the first relies on centralized control and deep encryption over the Internet, and the second relies on network isolation and natural resistance to hacking, making their combination – the future step - a strategic step towards achieving an ideal balance between security and flexibility in IoT networks.

The following image shows the location of the proposed system (LoRa / MQTT) in a balanced manner between performance efficiency and security

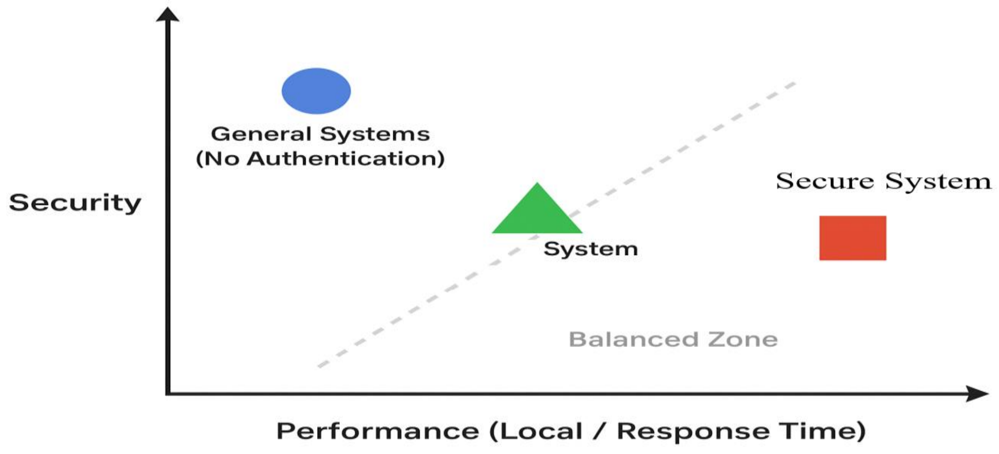


Figure 6.21: System Position

The image expresses a balance chart between performance and security, aiming to clarify the location of the proposed system compared to traditional and unsecured systems on the one hand and ultra-secure systems on the other hand, allowing an assessment of the level of security achieved by the system without sacrificing operational performance. The vertical axis shows the security level, while the horizontal axis represents performance/response time.

It is noted that General Systems is located on the upper left side of the figure, meaning that it provides relatively poor performance despite its security simplicity, as it lacks verification or authentication mechanisms (No Authentication), which makes it vulnerable to hacking or data manipulation despite its quick response locally. In contrast, Secure Systems is located on the lower right side, achieving a very high level of security thanks to the use of advanced encryption and authentication technologies, but at the expense of performance, as response time increases due to complex calculations in the authentication, encryption, and decryption stages. The proposed system, represented by the green triangle in the middle area (Balanced Zone), falls within what is known as the security-performance balance zone, indicating that it has succeeded in achieving a high level of security while maintaining acceptable operational efficiency. This balance indicates that the system uses strong encryption mechanisms (such as Kyber) and employs efficient communication protocols (such as

LoRa or MQTT), without significantly degrading response speed or excessively increasing processing and power consumption.

Therefore, it can be said that the image illustrates the location of the proposed security system from an engineering and analytical perspective, reconciling speed and reliability, and proving that achieving strong security does not necessarily require a complete sacrifice of performance. This visual representation reinforces the basic research idea that modern intelligent systems, especially those based on post-quantum encryption, can maintain a sustainable balance between operational efficiency and a high level of security, making them ideal for time- and security-sensitive IoT applications at the same time.

6.6 Conclusion

This chapter provided a comprehensive security analysis of the proposed systems, explaining the security features that each step added to the system and demonstrating the system's ability to resist threats and attacks in the main IoT environment, such as retransmission attacks and data manipulation. The evaluation confirmed that the multi-layer encryption framework significantly enhances the system's flexibility and achieves the required balance between performance and security. The final chapter concludes the study and suggests future research directions.

Chapter 7: Conclusion and Future Work

This thesis addressed the topic of securing communication standards for smart irrigation systems based on the Internet of Things in Palestine. It analyzed the security challenges associated with smart irrigation systems that rely on low-power wireless communication technologies, with a particular focus on their suitability for the Palestinian environment in terms of limited infrastructure, constrained resources, and specific geographical and climatic conditions.

In this study, a secure smart irrigation system based on LoRa technology was designed and implemented. The proposed system integrates multiple protection mechanisms, including securing the communication layer using appropriate encryption protocols, managing cryptographic keys within secure hardware units such as SE/HSM, and exploiting built-in LoRaWAN protection features such as message headers and frame counters. These mechanisms contribute to mitigating common threats, including replay attacks, data manipulation, and eavesdropping, while maintaining low energy consumption suitable for agricultural IoT environments.

To evaluate flexibility and compare different communication standards, an alternative system based on the MQTT protocol was also proposed and designed. This solution is particularly suitable for scenarios where direct Internet connectivity is available, due to MQTT’s lightweight publish/subscribe model, which enables efficient data transmission with low latency. When deployed over the TLS protocol, MQTT supports strong security services such as encryption, authentication, and data integrity, making it a practical option for integrating smart irrigation systems with cloud servers and data analytics platforms. The results demonstrate that adopting secure and energy-efficient communication solutions significantly improves irrigation efficiency, reduces water waste, and enhances system reliability.

Despite these contributions, the study highlights several challenges that continue to affect smart agricultural systems, including technical limitations related to processing capabilities and energy dependency, connectivity issues caused by unstable network infrastructure, cybersecurity threats such as denial-of-service and man-in-the-middle attacks, protocol compatibility constraints, encryption overhead in resource-limited devices, and economic, environmental, and social challenges such as system cost, maintenance difficulties, and limited user awareness.

Based on these findings, the study recommends applying the proposed security framework in real Palestinian agricultural environments to validate its effectiveness under field conditions. It also suggests adopting hybrid communication architectures that combine long-range technologies such as LoRa with Internet-based protocols such as MQTT to achieve a balance between energy efficiency and communication reliability. Furthermore, integrating post-quantum cryptographic algorithms such as Kyber, enhancing hardware-based security, expanding large-scale testing scenarios, and leveraging cloud platforms for secure data analysis are recommended to improve system robustness.

Finally, this work opens the door to future research directions, including studying the impact of environmental factors on communication performance, analyzing the economic feasibility of secure smart irrigation systems, integrating artificial intelligence for proactive attack detection, and exploring advanced technologies such as 6G and satellite-based communications to support smart agriculture in remote areas of Palestine.

References

1. The Impact of Disasters on Agriculture and Food Security 2023: Executive summary**.** (2023, October 13). FAO. https://openknowledge.fao.org/server/api/core/bitstreams/7802713a-e442-416b-981b-addd4fce0492/content/impact-of-disasters-on-agriculture-and-food-2023/executive-summary.html
2. Kerr Casper, J. (2007). Agriculture: The food we grow and animals we raise.
3. Banerjee, P. S. (2022). Economic development in agro-finance sector.
4. Hegde, P. D. (2021.). Global warming: Effects and remedy.
5. Abuzir, Y. (2017). Predict the main factors that affect the vegetable production in Palestine using WEKA data mining tool. DOI: **10.33977/2106-000-001-005**
6. Abuzir, Y. S., Awad, W. A., & Khdair, M. H. (2021). Web-based market information system for farmers in Palestine. DOI: **10.33977/2106-000-005-002**
7. Ahmadi, F., Abedi, O., & Emadi, S. (2024). Enhancing smart agriculture monitoring via connectivity management scheme and dynamic clustering strategy. DOI**: 10.3390/inventions9010010**
8. Ahmed, R. A., Hemdan, E. E.-D., El-Shafai, W., Ahmed, Z. A., El-Rabaie, E.-S. M., Abd El-Samie, F. E. (2022). Climate-smart agriculture using intelligent techniques, blockchain and Internet of Things: Concepts, challenges, and opportunities., DOI: **10.1002/ett.4607**
9. Alenezi, M. N., Alabdulrazzaq, H., & Mohammad, N. Q. (2020). Symmetric encryption algorithms: Review and evaluation study.
10. Almuhaya, M. A. M., Jabbar, W. A., Sulaiman, N., & Abdulmalek, S. (2022). A survey on LoRaWAN technology, DOI: **10.3390/electronics11010164**
11. Alotaibi, A., Aldawghan, H., & Aljughaiman, A. (2025). A review of the authentication techniques for Internet of Things devices in smart cities: Opportunities, challenges, and future directions. DOI: **10.3390/s25061649**
12. Amjath, M. I. M., & Senthooran, V. (2020). Secure communication using steganography in IoT. DOI**: 10.1109/ICAC51239.2020.9357260**
13. Aslan, Ö., Serkant, S., Ozkan-Okay, M., Yilmaz, A. A., & Akin, E. (2023). A comprehensive review of cyber security vulnerabilities, threats, attacks, and solutions. DOI**: 10.3390/electronics12061333**

14. Bayılmış, C., Ebleme, M. A., Çavuşoğlu, Ü., Küçük, K., & Sevin, A. (2022). A survey on communication protocols and performance evaluations for Internet of Things key. DOI: **10.1016/j.dcan.2022.03.013**
15. Blessy, A., & Kumar, A. (2021). Smart irrigation system techniques using artificial intelligence and IoT. DOI: **10.1109/ICICV50876.2021.9388444**
16. Cambou, B., Gowanlock, M., Yildiz, B., Ghanaimiandoab, D., Lee, K., Nelson, S., Philabaum, C., Stenberg, A., & Wright, J. (2021). Post-quantum cryptographic keys generated with physical unclonable functions. DOI: **10.3390/app11062801**
17. Choudhary, D. S., & Meen, G. (2022). Internet of Things: Protocols, Applications and Security Issues. DOI: **10.1016/j.procs.2022.12.030**
18. Claeys, T., Vučinić, M., Watteyne, T., Rousseau, F., & Tourancheau, B. (2021). Performance of the Transport Layer Security handshake over 6TiSCH. Inria & Univ. Grenoble Alpes, CNRS, Grenoble INP, LIG, DOI: **10.3390/s21062192**
19. Commey, D., Appiah, B., Klogo, G. S., Bagyl-Bac, W., Gadze, J. D., Alsenani, Y., & Crosby, G. V. (2025). Performance analysis and deployment considerations of post-quantum cryptography for consumer. DOI: **10.48550/arXiv.2505.02239**
20. Demir, E. D., Bilgin, B., & Onbaşlı, M. C. (2025). Performance analysis and industry deployment of post-quantum cryptography algorithms. DOI: **10.48550/arXiv.2503.12952**
21. Dubovitski, A. A., Konovalova, M. E., Strelnikova, T. D., Pilipchuk, N. V., & Shvetsova, N. (2021). Assessment of the impact of climate risks on agriculture in the context of global warming. DOI: **10.1088/1755-1315/845/1/012145**
22. Ehsan, M. A., Alayed, W., Rehman, A. U., Hassan, W. ul, & Zeeshan, A. (2025). Post-quantum KEMs for IoT: A study of Kyber and NTRU. DOI: **10.3390/sym17060881**
23. Escribano Pablos, J. I., Marriaga, M. E., & Pérez del Pozo, Á. L. (2022). Design and implementation of a post-quantum group authenticated key exchange protocol with the LiboQS library: A comparative performance analysis from Classic McEliece, Kyber, NTRU, and Saber. DOI: **10.1109/ACCESS.2022.3222389**
24. Faradiba, F. (2024). The effect of global warming on changes in the labor structure of the agricultural sector. DOI: **10.52155/ijpsat. v43.1.5997**

25. Ferrag, M. A., Shu, L., Yang, X., Derhab, A., & Maglaras, L. (2020). Security and privacy for green IoT-based agriculture: Review, blockchain solutions, and challenges. DOI: **10.1109/ACCESS.2020.2973178**
26. Fragkopoulos, M., Panagiotakis, S., Kostakis, M., Markakis, E. K., Astyrakakis, N., & Malamos, A. (2023). Experimental assessment of common crucial factors that affect LoRaWAN performance on suburban and rural area deployments. DOI: **10.3390/s23031316**
27. Froiz-Míguez, I., Lopez-Iturri, P., Fraga-Lamas, P., Celaya-Echarri, M., Blanco-Novoa, Ó., Azpilicueta, L., Falcone, F., & Fernández-Caramés, T. M. (2020). Design, implementation, and empirical validation of an IoT smart irrigation system for fog computing applications based on LoRa and LoRaWAN sensor nodes. DOI: **10.3390/s20236865**
28. Gebremichael, T., Ledwaba, L. P. I., Eldeefrawy, M., Hancke, G. P., Pereira, N., Gidlund, M., & Akerberg, J. (2020). Security and privacy in the industrial Internet of Things: Current standards and future challenges. DOI: **10.1109/ACCESS.2020.3016937**
29. Gulzar, M., Abbas, G., & Waqas, M. (2020). Climate smart agriculture: A survey and taxonomy. DOI: **10.1109/ICETST49965.2020.9080695**
30. Gupta, S. K., Vanjale, S., & Rasal, S. (2020). Securing IoT devices in smart city environments. DOI:**10.1109/ESCI48226.2020.9167630**
31. Guru, A., Mohanta, B. K., Mohapatra, H., Al-Turjman, F., Altrjman, C., & Yadav, A. (2023). A survey on consensus protocols and attacks on blockchain technology. DOI**: 10.1145/3579845**
32. Has, M., Kreković, D., Kušek, M., & Podnar Žarko, I. (2024). Efficient data management in agricultural IoT: Compression, security, and MQTT protocol analysis. DOI: **10.3390/s24113517**
33. Hassan, R., Qamar, F., Hasan, M. K., Mohd Aman, A. H., & Sid Ahmed, A. (2020). Internet of Things and its applications: A comprehensive survey. DOI: **10.3390/sym12101674**
34. Hmissi, F., & Ouni, S. (2022). A survey on application layer protocols for IoT networks. DOI**: 10.48550/arXiv.2405.15901**
35. Ingole, K., & Padole, D. (2024). An Internet of Things (IoT)-based smart irrigation and crop suggestion platform for enhanced precision agriculture. DOI: **10.47974/JIOS1612**

36. Islam, M. M., Al-Momin, M., Tauhid, A. B. M., Hossain, M. K., & Sarker, S. (2020). IoT Based Smart Irrigation Monitoring & Controlling System in Agriculture. DOI: **10.35940/ijrte. E6851.038620**
37. Iqbal, M., Kaynat, A., Safdar, M., Taj, H., Ashraf, S., Akhtar, H., Ishtiaq, M., & Nisha. (2024). Agricultural chemistry and climate change: A review of the impact of global warming on crop yields, food security, and sustainable agriculture. DOI: **10.53555/ecb/2024.13.05.10**
38. Jabal, Z. K., Khayyun, T. S., & Alwan, I. A. (2022). Impact of climate change on crops productivity using MODIS-NDVI time series. DOI: **10.28991/CEJ-2022-08-06-04**
39. Joshi, M., & Prateek, M. (2024). Performance analysis of hybrid encryption algorithm for data security in cloud systems.
40. Jouhari, M., Saeed, N., Alouini, M.-S., & Amhoud, E. M. (2023). A survey on scalable LoRaWAN for massive IoT: Recent advances, potentials, and challenges. IEEE Internet of Things Journal. DOI: **10.1109/COMST.2023.3274934**
41. Kaganurmath, S., Cholli, N., & Anala, M. R. (2025). Post-quantum lightweight key sharing protocol for secure MQTT-based IoT networks (PQLKS-MQTT).DOI: **10.21203/rs.3.rs-6382308/v1**
42. Karpagam, J., Merlin, I. I., Bavithra, P., & Kousalya, J. (2020). Smart irrigation system using IoT. DOI: **10.1109/ICACCS48705.2020.9074201**
43. Käppler, S. A., & Schneider, B. (2022). Post-quantum cryptography: An introductory overview and implementation challenges of quantum-resistant algorithms. DOI: **10.29007/2tpw**
44. Khalil, A.-A., & Manaa, M. E. (2025). Toward quantum-resistant Fog-IoT security: A dual-phase framework with chaotic elliptic-curve Diffie-Hellman and post-quantum encryption. DOI: **10.48084/etasr.11600**
45. Kim, Y., & Seo, S. C. (2025). An optimized instantiation of post-quantum MQTT protocol on 8-bit AVR sensor nodes. DOI: **10.1145/3708821.3733873**
46. Kirdan, E., Rezabek, F., Muhlbauer, N., Carle, G., & Pahl, M.-O. (2023). Real-time performance of OPC UA. DOI: **10.48550/arXiv.2310.17052**
47. Kumar, S., Gaur, M. S., Sharma, P. S., & Munjal, D. (2021). A novel approach of symmetric key cryptography. DOI: **10.1109/ICIEM51511.2021.9445343**

48. Lee, E., Seo, Y. D., Oh, S. R., & Kim, Y. G. (2021). A survey on standards for interoperability and security in the Internet of Things. IEEE Internet of Things. DOI: **10.1109/COMST.2021.3067354**
49. Len, J., Grubbs, P., & Ristenpart, T. (2021). Partitioning Oracle Attacks. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), 1527–1544. USENIX Association.
50. Liaqat, W., Barutçular, C., Farooq, M. U., Ahmad, H., Jan, M. F., Ahmad, Z., Nawaz, H., & Li, M. (2022). Climate change in relation to agriculture: A review. DOI**: 10.5424/sjar/2022202-17742**
51. Mahdi, L. H., & Abdullah, A. A. (2025). A hybrid post-quantum cryptographic framework integrating Kyber-512 and ASCON for secure IoT communication. DOI**: 10.1109/ACCESS.2020.2973315**
52. Mansour, M., Gamal, A., Ahmed, A. I., Said, L. A., Elbaz, A., Herencsar, N., & Soltan, A. (2023). Internet of Things: A comprehensive overview on protocols, architectures, technologies, simulation tools, and future. DOI**: 10.3390/en16083465**
53. Manda, J. K. (2021). IoT security frameworks for telecom operators: Designing robust security frameworks to protect IoT devices and networks in telecom environments.
54. Massaoudi, A., Berguiga, A., & Harchay, A. (2022). Secure Irrigation System for Olive Orchards Using Internet of Things. DOI**: 10.32604/cmc.2022.026972**
55. Mbaraka, J. (2023). Effect of global warming on agricultural productivity. DOI: **10.47604/ija.1971**
56. Memon, K., Umrani, F. A., Baqai, A., & Shah, Z. S. (2024). Comparison of IoT messaging protocols: A novel crop-specific protocol for wheat, banana, and chili
57. Mishra, B., Mishra, B., & Kertész, A. (2021). Stress-testing MQTT brokers: A comparative analysis of performance measurements. DOI: **10.3390/en14185817**
58. Mojarad, M., Ahmadi, H., Behjathaghighi, A., & Karji, M. K. (2021). Design of an IoT-based management and monitoring system for intelligent irrigation. DOI**: 10.1109/ICAC51239.2020.9357260**
59. Moon, S. Y., Jo, B. H., El Azzaoui, A., Singh, S. K., & Park, J. H. (2025). Edge–fog enhanced post-quantum network security: Applications, challenges, and solutions. DOI: **10.32604/cmc.2025.062966**

60. More, P. R. (2025). A comparative study of symmetric key algorithms for network performance
61. Moussa, T. V., Nataraj, C., & Seeralan, A. (2021). IoT based smart irrigation system. DOI**: 10.3390/s22062141**
62. Mrabet, H., Belguith, S., Alhomoud, A., & Jemai, A. (2020). A survey of IoT security based on a layered architecture of sensing and data analysis. DOI**: 10.3390/s20133625**
63. Muhammed, R. K., Faraj, K. H. A., Gul-Mohammed, J. F., Al Attar, T. N. A., Saydah, S. J., & Rashid, D. A. (2024). Automated performance analysis e-services by AES-based hybrid cryptosystems with RSA, ElGamal, and ECC. DOI**: 10.25046/aj090308**
64. Muthuramalingam, R., Velu, R. R., Baskar, H., & Saminathan, M. H. V. (2023). An IoT-based smart irrigation system. DOI**: 10.3390/engproc2024066013**
65. Newaz, A. I., Sikder, A. K., Rahman, M. A., & Uluagac, A. S. (2021). A survey on security and privacy issues in modern healthcare systems: Attacks and defences. DOI: **10.1145/3453176**
66. Nigussie, E., Olwal, T., Musumba, G., Tegegne, T., Lemma, A., & Mekuria, F. (2020). IoT-based Irrigation Management for Smallholder Farmers in Rural Sub-Saharan Africa. DOI**: 10.3390/electronics12122697**
67. Ozdemir, D. (2021). The impact of climate change on agricultural productivity in Asian countries: A heterogeneous panel data approach. DOI**: 10.1109/JIOT.2024.3450123**
68. Palková, Z., Harničárová, M., Valíček, J., & Stehel, V. (2022). Perspective of education in Agriculture 4.0 in selected countries in European Union and Palestine. DOI: **10.1109/EEAE53789.2022.9831232**
69. Prantl, T., & Krupitzer, C. (2021). Performance impact analysis of securing MQTT using TLS. DOI**: 10.1145/3427921.3450253**
70. Prince, N. U., Mamun, M. A. A., Olajide, A. O., Khan, O. U., Akeem, A. B., & Sani, A. I. (2024). IEEE standards and deep learning techniques for securing Internet of Things (IoT) devices against cyber-attacks. IEEE Transactions on Emerging Topics in Computing.

71. Qays, M. O., Ahmad, I., Abu-Siada, A., Hossain, M. L., & Yasmin, F. (2023). Key communication technologies, applications, protocols and future guides for IoT-assisted smart grid systems: A review. DOI: **10.1016/j.egyr.2023.01.085**
72. Qian, M., Qian, C., Xu, G., Tian, P., & Yu, W. (2024). Smart Irrigation Systems from Cyber–Physical Perspective: State of Art and Future Directions. DOI: **10.3390/fi16070234**
73. Qureshi, S. (2024). Denial of Service (DoS) attacks: Innovative cybersecurity solutions for effective attack prevention. DOI: **10.13140/RG.2.2.24587.71200**
74. Qureshi, T., Saeed, M., Ahsan, K., Malik, A. A., Muhammad, E. S., & Touheed, N. (2022). Smart agriculture for sustainable food security using Internet of Things (IoT). DOI: **10.1155/2022/9608394**
75. Quy, V. K., Hau, N. V., Anh, D. V., Quy, N. M., Ban, N. T., Lanza, S., Randazzo, G., & Muzirafuti, A. (2022). IoT-enabled smart agriculture: Architecture, applications, and challenges. DOI**: 10.3390/s25010112**
76. Rahaman, M., Lin, C.-Y., Pappachan, P., Gupta, B. B., & Hsu, C.-H. (2024). Privacy-centric AI and IoT solutions for smart rural farm monitoring and control. DOI: **10.3390/s24134157**
77. Rao, P. M., & Deebak, B. D. (2022). Security and privacy issues in smart cities/industries: Technologies, applications, and challenges. DOI**: 10.1007/s12652-022-03707-1**
78. Rehman, S. U., Singh, P., & Manickam, S. (2020). Towards sustainable IoT ecosystem. DOI**: 10.3390/electronics14030456**
79. Saho, N. J. G., & Ezin, E. C. (2020). Survey on asymmetric cryptographic algorithms in embedded systems. DOI: **10.32628/IJSRSET207292**
80. Saideh, M., Jamont, J.-P., & Vercouter, L. (2024). Opportunistic sensor-based authentication factors in and for the Internet of Things. DOI: **10.3390/s24144621**
81. Saini, J., & Bhatt, R. (2020). Global warming - Causes, impacts and mitigation strategies in agriculture. DOI: **10.9734/cjast/2020/v39i730580**
82. Salem, H. S., Yihdego, Y., & Muhammed, H. H. (2021). The status of freshwater and reused treated wastewater for agricultural irrigation in the Occupied Palestinian Territories. DOI: **10.2166/wh.2020.216**
83. Samawi, V. W. (2021). AN IoT Based Secure Multi-Crop Irrigation System for Smart Farming. DOI: **10.24507/ijicic.17.04.1225**

84. Sasi, T., Lashkari, A. H., Lu, R., Xiong, P., & Iqbal, S. (2024). A comprehensive survey on IoT attacks: Taxonomy, detection mechanisms and challenges. DOI: **10.1016/j.jiixd.2023.12.001**
85. Scott, B. A., Johnstone, M. N., & Szewczyk, P. (2024). A survey of advanced Border Gateway Protocol attack detection techniques. DOI: **10.3390/s24196414**
86. Segatz, F., & Al Hafiz, M. I. (2022). Efficient implementation of CRYSTALS-KYBER key encapsulation mechanism on ESP32. DOI: **10.48550/ arXiv.2503.10207**
87. Seoane, V., García-Rubio, C., Almenares, F., & Campo, C. (2021). Performance evaluation of CoAP and MQTT with security support for IoT environments. DOI: **10.1016/j.comnet.2021.107977**
88. Sharma, D. K., Singh, N. C., Noola, D. A., Doss, A. N., & Sivakumar, J. (2021). A review on various cryptographic techniques and algorithms. DOI: **10.1016/j.matpr.2021.04.583**
89. Singh, S. K., Tiwari, V., Kirti, D., & Vadi, V. R. (2024). Protocols for the Internet of Things. DOI: **10.17148/IJARCCE.2024.134156**
90. Silva, D., Chaves, L. I., & Santos, R. C. (2022). Evaluating MQTT, CoAP, OPC UA. Electronics. DOI: **10.3390/app11114879**
91. Sowmyashree, K. M., & Swaraj, C. M. (2020). IoT based Smart Agriculture Monitoring and Irrigation System. DOI: **10.17577/IJERTCONV8IS14062**
92. Tawalbeh, M., Quwaider, M., & Tawalbeh, L. A. (2021). IoT cloud enabled model for safe and smart agriculture environment. DOI: **10.1109/ ICICS52457. 2021.9464567**
93. Thaher, T., & Ishaq, I. (2020). Cloud-based Internet of Things Approach for Smart Irrigation System: Design and Implementation. DOI: **10.1109/ICPET51420.2020.00015**
94. Tightiz, L., & Yang, H. (2020). A Comprehensive Review on IoT Protocols' Features in Smart Grid Communication. DOI: **10.3390/en13112762**
95. Toh, C. K. (2020). Security for smart cities. DOI: **10.1049/iet-smc.2020.0001**
96. Tripathi, T., Awasthi, A., Singh, S. P., & Chaturvedi, A. (2024). Post quantum cryptography & its comparison with classical cryptography. DOI: **10.48550/arXiv.2403.19299**

97. Ullah, R., Abbas, A. W., Ullah, M., Khan, R. U., Khan, I. U., Aslam, N., & Aljameel, S. S. (2021). An IoT-Based Energy-Efficient Water Management Platform for Smart Irrigation. DOI: **10.1155/2021/5536884**

98. Vairagade, R. S., & Brahmananda, S. H. (2020). Secured Multi-Tier Mutual Authentication Protocol for Secure IoT System. DOI: **10.1109/CSNT48778.2020.9115786**

99. Vangala, A., Das, A. K., Kumar, N., & Alazab, M. (2020). Smart secure sensing for IoT-based agriculture: Blockchain perspective. DOI: **10.1109/JSEN.2020.3012294**

100. Wlazlo, P., Sahu, A., Mao, Z., Huang, H., Goulart, A., Davis, K., & Zonouz, S. (2021). Man-in-the-middle attacks and defence in a power system cyber-physical testbed. DOI**: 10.1049/cps2.12014**

101. Wytrebowicz, J., Cabaj, K., & Krawiec, J. (2021). Messaging protocols for IoT systems—A pragmatic comparison. DOI**: 10.3390/s21206904**

102. Yassir, M., Soepandi, H., Hanani, A., Prakasa, J. E. W., Puspitadewi, G. C., Wibowo, S. H., Adi, P. D. P., & Kitagawa, A. (2024). Performance analysis of LoRaWAN communication utilizing the RFM96 module. DOI**: 10.33096/ ilkom. v 16i3.2326.255-270**

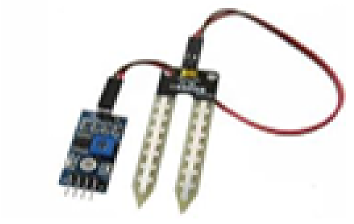
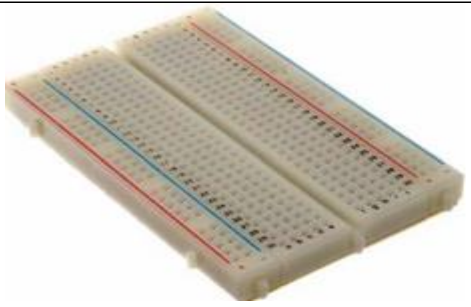
103. Yousefi, A., Jameii, S. M. (2023). Improving the security of Internet of Things using encryption algorithms. DOI: **10.1109/ICIOTA.2017.8073627**

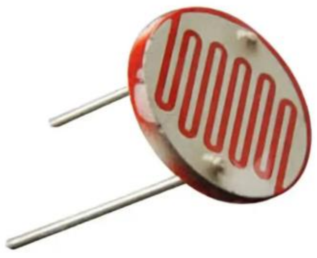
104. Zhang, H., Babar, M., Tariq, M. U., Jan, M. A., Menon, V. G., & Li, X. (2020). Safe City: Toward safe and secured data management design for IoT-enabled smart city planning. DOI**: 10.1109/ACCESS.2020.3014622**

105. Zhao, X., Chen, L.-W., Li, K., Schmidt, H., Polian, I., & Du, N. (2024). Memristive true random number generator for security applications. DOI**: 10.3390/s24155001**

106. Zelle, D., & Gürgens, S. (2021). Bus Count: A Provable Replay Protection Solution for Automotive CAN Networks. DOI**: 10.1155/2021/9951777**

Appendix A: System Parts

	Arduino ESP32: a low-cost, energy-efficient microcontroller (MCU) series from Espressif Systems, known for integrating Wi-Fi and Bluetooth (Classic & BLE) in a single chip, making it perfect for IoT applications like smart home devices, wearables, and sensors. It features dual or single-core Tensilica Xtensa processors, ample memory (SRAM), numerous GPIO pins, and essential peripherals (ADC, I2C, SPI, UART). Programmed via tools like the Arduino IDE or ESP-IDF, it offers a powerful, integrated solution for wireless connectivity in embedded systems.
	Soil Moisture Sensor: contains two tests by methods for which current will go into the dirt, at that point scrutinizes the obstruction of the soil, which will peruse the dampness level. We know the nearness of the water makes the dirt more inclined to lead the power effortlessly, which implies R(resistance) is less in the such kind of soil, while dry soil has poor conductivity of intensity, in this way dry soil upholds more insurance than the wet soil. Sensor is structured on this property of intensity. There should be a point that believers the obstruction into voltage, this is done using circuit which show inside the sensor, which changes over the opposition into voltage
	Breadboard: is a solider less contrivance for ad interim template with test circuit designs Singular bob wires are attached by implanted their "end connectors" into the initial way gave in a breadboar

	Relay: acts like a remote-controlled switch. Since a water pump operates at high voltage/current (AC) and microcontrollers like Arduino or ESP32 operate at low voltage (DC), the relay safely bridges the two, when the microcontroller sends a signal to the relay, which then opens or closes the high-power circuit that controls the pump.
	Water Pump: is responsible for delivering water to the irrigation system. It physically transports water from a source (such as a tank or well) to a field or crops. Automatically switched on or off based on sensor readings (such as soil moisture falling below a certain threshold).,and the signal from the relay
	The LoRa (Long Range Module) is a modern wireless communication technology that enables long-distance data transmission with very low power consumption, making it ideal for Internet of Things (IoT) applications such as intelligent irrigation systems and environmental monitoring. LoRa operates with a modulation technology called Chirp Spread Spectrum (CSS), which enables high noise resistance and connectivity even in environments with weak signals.
 LDR Sensor	The photoresistor (LDR) is a sensitive sensor for light intensity. Its electrical resistance value changes depending on the amount of light falling on it. Its resistance decreases with increasing light and increases in the dark. They are typically used in automatic lighting control systems or to determine day and night periods in intelligent systems, such as operating an irrigation pump only at night

Appendix B: MQTT-Based Systems Models Results

MQTT-Based Systems / Model1

Variable	Value per Byte
Project Size	945039 …. (72%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	52312 …… (15%) of RAM
Local Variables Size	275368
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.76	4.76	4.764	4.764	4.762
Encapsulation Time (ms)	6.32	6.31	6.31	6.31	6.3125
Decapsulation Time (ms)	6.92	6.92	6.91	6.91	6.915
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	1.66	1.66	1.66	1.66	1.66
Data transfer rate bps	224719.101	225940.832	224900.727	224600.807	225040.37
Response time (ms)	28.48	28.35	28.45	28.49	28.44
Maximum load size (Byte)	800	800	800	800	800
Free Memory (Byte)	195120	195128	195120	195140	195127
Pump (on/off)	off	Off	On	On	

MQTT-Based Systems / Model2

Variable	Value per Byte
Project Size	948775 …. (72%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	52792 …… (16%) of RAM
Local Variables Size	274888
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.79	4.78	4.79	4.79	4.7875
Encapsulation Time (ms)	12.31	12.31	12.31	12.31	12.31
Decapsulation Time (ms)	15.796	14.78	15.82	14.79	15.2965
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	3.25	3.25	3.25	3.25	3.25
Data transfer rate (Bps)	94886.5	95528	94885.1	95568	95216.9
Response time (ms)	67.44	66.99	67.45	66.9	67.195
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	194520	194540	194528	194528	194529
Pump (on/off)	Off	Off	On	On	

MQTT-Based Systems / Model3

Variable	Value per Byte
Project Size	1037407 …. (79%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	53352 …… (16%) of RAM
Local Variables Size	275,368
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.82	4.88	4.83	4.97	4.875
Encapsulation Time (ms)	12.35	12.35	12.35	12.46	12.3775
Decapsulation Time (ms)	14.16	14.11	14.09	14.25	14.1525
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	3.26	3.26	3.26	3.29	3.2675
Data transfer rate (Bps)	126115.8	100728.7	103662.1	102053.8	108140.1
Response time (ms)	50.47	63.5	61.73	62.71	59.6025
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	148448	148408	148480	148460	148449
Pump (on/off)	Off	Off	On	On	

MQTT-Based Systems / Model4

Variable	Value per Byte
Project Size	1036539 …. (79%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	53460 …… (16%) of RAM
Local Variables Size	274220
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.79	4.77	4.76	4.77	4.7725
Encapsulation Time (ms)	12.29	12.29	12.29	12.29	12.29
Decapsulation Time (ms)	14.01	14.01	14.01	14.01	14.01
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	3.24	3.24	3.24	3.24	3.24
Data transfer rate (Bps)	103511.2	103319.1	103315.7	102316.5	103115.63
Response time (ms)	61.82	61.94	61.94	62.55	62.0625
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	150864	150884	150852	150808	150852
Pump (on/off)	off	Off	on	On	

Appendix C: LoRa-Based Systems Models Results

LoRa-Based Systems / Model1

Variable	Value per Byte
Project Size	308247 …. (23%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	24904 …… (7%) of RAM
Local Variables Size	302776
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.62	4.62	4.62	4.62	4.62
Encapsulation Time (ms)	6.18	6.18	6.18	6.19	6.1825
Decapsulation Time (ms)	6.5	6.5	6.51	6.5	6.5025
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	1.63	1.63	1.63	1.63	1.63
Data transfer rate (bps)	6789.1	6788.9	6790.4	6792.5	6790.225
Response time (ms)	942.67	942.7	942.5	942.2	942.5175
Maximum load size (Byte)	800	800	800	800	800
Free Memory (Byte)	265448	265448	265448	265448	265448
Pump (on/off)	Off	Off	on	On	

LoRa-Based Systems / Model2

Variable	Value per Byte
Project Size	329527 …. (25%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	24928 …… (7%) of RAM
Local Variables Size	302752
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.62	4.62	4.62	4.62	4.62
Encapsulation Time (ms)	5.98	5.99	5.98	5.98	5.9825
Decapsulation Time (ms)	6.56	6.57	6.57	6.57	6.5675
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	1.58	1.58	1.58	1.58	1.58
Data transfer rate (Bps)	5643.8	56444.5	5644.8	5644.88	18344.495
Response time (ms)	1133.98	1133.83	1133.77	113.77	878.8375
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	265316	265316	265316	265316	265316
Pump (on/off)	Off	Off	on	On	

LoRa-Based Systems / Model3

Variable	Value per Byte
Project Size	330111 …. (25%) of Storage Memory.
Maximum Program Size	1310720
Global Variables Size	25036 …… (7%) of RAM
Local Variables Size	302644
Maximum Size	327680

Metric	Trials Values				Average
	T1	T2	T3	T4	
Moisture	2243	211	3765	3774	
Key Generation Time (ms)	4.62	4.62	4.62	4.62	4.62
Encapsulation Time (ms)	5.99	5.99	5.99	5.99	5.99
Decapsulation Time (ms)	6.57	6.56	6.56	6.56	6.5625
Key Size per Byte	800	800	800	800	800
Energy consumption (mj)	1.58	1.58	1.58	1.58	1.58
Data transfer rate (Bps)	5644.5	5643.74	5644	5643.7	5643.985
Response time (ms)	1133.84	1133.99	1133.94	1133.99	1133.94
Maximum load size (Byte)	800	800	800	800	800
Free Memory (byte)	265056	265056	265056	265056	265056
Pump (on/off)	Off	Off	on	On	

Author’s Biographical Sketch

To insert Biographical Sketch text here, select this text and then either type the text you wish to use or paste text from another document, being sure to keep the text only and not the formatting from the previous document. To keep text only, choose Paste, and then from the drop-down box that will appear, choose the Keep Text Only option on the right, with the icon of a clipboard and the letter A.