Migration Letters

Volume: 21, No: S1 (2024), pp. 972-993

ISSN: 1741-8984 (Print) ISSN: 1741-8992 (Online)

www.migrationletters.com

Securing Iot Data Transmission: A Comprehensive Approach Integrating Two-Fish Encryption With Wireless Smart Energy Systems And Iot Cloud Services

Ashok Kumar N[1], Dr. D. Ramesh[2], Dr. Prashant H. S.[3], Dr. Pallavi R.[4] and Veena B. G.[5]

Abstract:

This work addresses the challenges posed by the interconnected nature of the Internet of Things (IoT). This paradigm, facilitating the seamless connectivity of diverse devices, gives rise to issues such as addressability, confidentiality, and data management. The proposed solution advocates for the integration of the robust Twofish encryption algorithm into wireless smart energy systems, creating a holistic approach that capitalizes on the capabilities of IoT cloud services. Twofish, recognized for its strong encryption capabilities, becomes an integral part of the IoT framework to ensure the security of communication and data confidentiality. Through the amalgamation of Twofish encryption with wireless smart energy systems and IoT cloud services, the approach provides a comprehensive solution for the secure transmission and management of data. This integration enhances system scalability, flexibility, and accessibility, thereby safeguarding the confidentiality of information exchanged over the internet. The paper establishes the superiority of Twofish over lightweight cryptography for low-power devices within IoT systems through a thorough analysis of critical factors. These factors encompass the robust security strength of Twofish, its effective encryption capabilities guaranteeing data confidentiality, the algorithm's maturity validated by extensive scrutiny from the cryptographic community, compatibility and interoperability with diverse platforms, resource efficiency in maintaining a balance between security and device capabilities, and the crucial long-term security assurance required for the extended operational lifetimes of IoT devices. The research findings affirm that the incorporation of Twofish into wireless smart energy systems and IoT cloud services offers a secure and efficient solution for safeguarding sensitive information within IoT environments. This consolidated approach contributes significantly to advancing secure communication and data management in the continually evolving landscape of the Internet of Things.

1. Introduction: The rapid growth of the Internet of Things (IoT) has revolutionized our interaction with the environment, impacting various sectors, including smart homes, industrial automation, and smart cit[1]ies. A critical challenge faced by IoT deployments is efficient power management. This literature review explores the significance of power management in IoT objects and strategies to optimize energy consumption. IoT objects play a pivotal role in connecting the physical and digital worlds, enabling data-driven decision-

[1]Programmer, BMS College of Engineering, Bengaluru, Karnataka, India.
[2]Professor and Head, Department of Master of Computer Applications, SSIT, SSAHE, Agalakote, B.H. Road, Tumakuru-572105, Karnataka, India.
[3]Associate Professor, Department of Mechanical Engineering,
Sir M Visvesvaraya Institute of Technology Bengaluru
[4]Associate Professor and Head School of Computer Science and Engineering, IOT and Networking Lab, Presidency University
[5]Assistant Professor, Department of Mechanical Engineering, Sir M Visvesvaraya Institute of Technology Bengaluru

making and driving innovation across industries [1]. Key aspects of their significance include connectivity and data collection, automation and control, enhanced decision-making, improved efficiency and resource management, and innovation in business models [2]. Securing IoT data and preserving privacy are vital considerations, with authentication, data encryption, secure protocols, firmware/software updates, privacy by design, secure data storage, and security monitoring and incident response being essential elements [3] [4] [5]. Power consumption in IoT objects is a significant concern due to limited energy resources. Various approaches to optimize power consumption include low-power communication protocols (e.g., Bluetooth Low Energy, Zigbee, LoRaWAN), energy harvesting, duty cycling, adaptive sensing, and efficient hardware design [6] [7] [8]. To leverage IoT for power management, a cluster comprising Raspberry Pi 3, RF Module (Transmitter & Receiver), and Bluetooth Module can be employed for intelligent transport systems. Raspberry Pi, a series of small, affordable, and versatile single-board computers, offers a platform for IoT applications [9]. It can be accessed remotely using SSH, and TightVNC Server enables remote desktop sharing. Model B boards provide enhanced performance with various connectivity options, making them suitable for diverse projects [10]. Power consumption tests for WiFi, RF, and Bluetooth modules on Raspberry Pi reveal valuable insights for optimizing energy usage [11] [12]. ThingSpeak serves as an IoT platform designed to collect, analyze, visualize, and act on sensor data in the cloud [13]. It facilitates real-time monitoring and control of IoT systems, supporting data-driven decision-making [14]. The platform's core functionalities include data collection, analysis, visualization, and action, making it an integral component for building IoT applications. To create a traffic density monitoring system, a combination of Raspberry Pi, IR sensor, RF (Tx, Rx) sensor, Bluetooth sensor, and ThingSpeak is proposed. The ThingSpeak platform allows users to remotely monitor and control data, and the system's equation is defined as

Raspberry Pi + IR Sensor +RF (Tx, Rx) Sensor + Bluetooth Sensor + ThingSpeak = IoT ....(i)

Data Collection and Data Accumulation Leads to

In view of security analysis, the RSA algorithm, named after its creators Ron Rivest, Adi Shamir, and Leonard Adleman, is a prominent asymmetric cryptography method extensively utilized for public key encryption and key exchange [15]. It relies on two prime numbers to generate a pair of public and private keys, with key sizes ranging from 1024 to 4096 bits [16]. The public key is used for encryption, while the private key is employed for decryption. The RSA procedure encompasses key generation, encryption, and decryption, with padding techniques applied to enhance security [17][18]. Despite offering robust security, the choice of key size and appropriate padding techniques is crucial for balancing security and computational efficiency.

The Triple Data Encryption Algorithm (TDEA), also known as Triple DES or 3DES, is a symmetric key algorithm derived from the original Data Encryption Standard (DES) to address its vulnerability due to a small key size [19]. TDEA operates in Encrypt-Decrypt-Encrypt (EDE) mode, involving three stages of encryption with different 64-bit keys (X1, X2, X3) [20]. TDEA keys are formed by bundling three individual keys into a 192-bit value, offering flexibility in using the same or different keys for varying security levels [20]. Although widely used in applications requiring backward compatibility with DES, TDEA has been gradually phased out in favor of more advanced encryption algorithms like the Advanced Encryption Standard (AES) due to its slower encryption speed [19]. AES provides higher security and faster performance, making it the preferred choice in many applications, while TDEA remains relevant in legacy systems and environments where it is still widely supported.

While existing research has explored various power management and security strategies for IoT devices, several gaps remain. First, existing solutions often focus on individual aspects of power optimization or security, neglecting the need for an integrated approach that holistically addresses both concerns. Second, many existing studies primarily focus on laboratory settings or simulations, lacking real-world validation and implementation. Finally, the rapid evolution of IoT technologies and communication protocols necessitates further research into novel and adaptive approaches for addressing power consumption and security challenges in dynamic environments. This research aims to address these gaps by proposing a novel framework that integrates efficient power management techniques with robust security mechanisms for data generated by IoT devices in cloud environments. The framework will be validated and evaluated in a real-world scenario, such as a traffic management system, demonstrating its effectiveness in optimizing energy consumption while ensuring data security and privacy.

2. Methodology, Results and Discussion:

2.1 Power Management: The methodology section of this study focuses on the systematic approach employed to collect and analyse data pertaining to different traffic monitoring modes using Raspberry Pi (RPi) boards equipped with various sensor and communication modules. The primary objective is to evaluate the effectiveness and power consumption characteristics of three distinct modes: W Mode (utilizing IR Sensor and in-built WiFi), R Mode (employing RF Transmitter and RF Receiver), and B Mode (integrating IR Sensor with HC-05 Serial Bluetooth Module).

The experimental setup involves deploying the RPi boards equipped with the specified modules at a traffic circle to monitor and analyse traffic conditions. The data collected is then transmitted to the ThingSpeak Cloud platform, allowing for centralized storage and analysis. This methodology aims to provide insights into traffic density variations and power consumption patterns across different modes, facilitating informed decision-making in the context of intelligent traffic management.

2.1.1 Test Case (1): Data Collection and Analysis for W Mode

In this test case, the focus is on utilizing an IR Sensor along with the in-built WiFi capabilities of the RPi board [W Mode (IR Sensor+WiFi (in-built) over the RPi board)] to monitor traffic density in real-time. Figure 1 illustrates the deployment of IR Sensors in each lane, with specific strategies for highlighting low and high traffic conditions. The collected data, categorized into low, medium, and high traffic density, is transferred to the ThingSpeak Cloud using a unique API key.

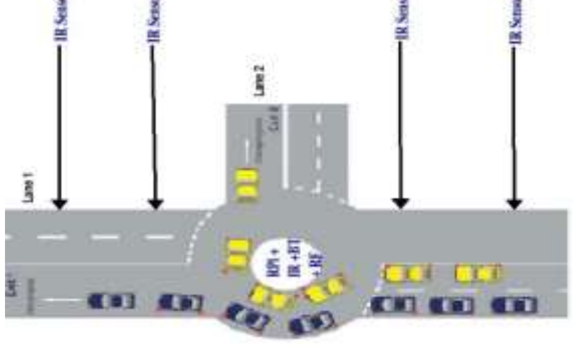
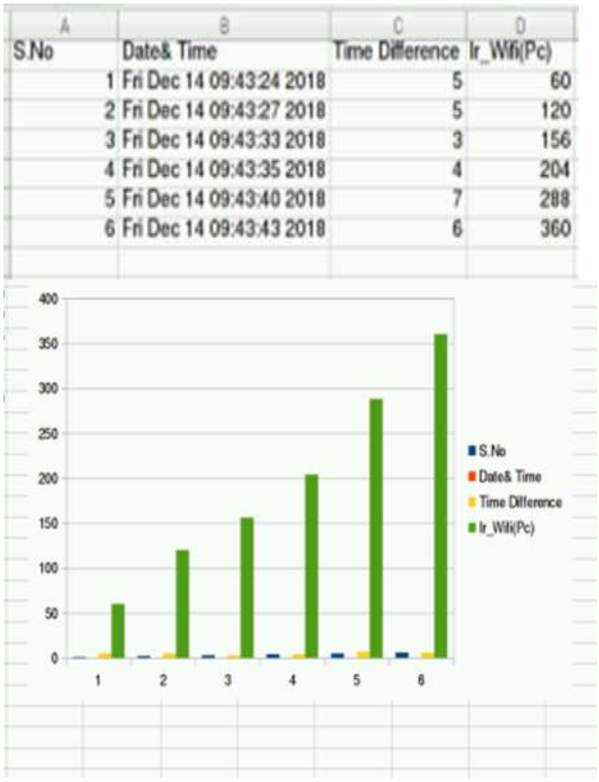


Fig 1: Strategy of deployment at traffic circle

Power consumption analysis is conducted by considering the power factor, and the general power consumption equation (T=T+ (T1*DV)) is applied, where T represents present power consumption, T1 is the variation in time, and DV is the device voltage. The in-built

WiFi, consuming 12V of power, is examined in terms of power consumption patterns during data transfer, as depicted in Figure 2.



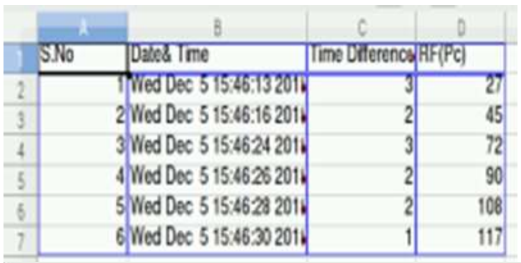
A	B	C	D
S.No	Date& Time	Time Difference	Ir_Wifi(Pc)
1	Fri Dec 14 09:43:24 2018	5	60
2	Fri Dec 14 09:43:27 2018	5	120
3	Fri Dec 14 09:43:33 2018	3	156
4	Fri Dec 14 09:43:35 2018	4	204
5	Fri Dec 14 09:43:40 2018	7	288
6	Fri Dec 14 09:43:43 2018	6	360

Fig 2: Bar chart depicting the collected and analysed power consumption data for the IR Sensor and in-built WiFi (W Mode) over the RPi board in the context of intelligent traffic monitoring

2.1.2 Test Case (2): Data Collection and Analysis for R Mode

In the second test case, the focus shifts to the utilization of RF Transmitter and RF Receiver modules over the RPi board (R Mode) for traffic monitoring. Figure 1 depicts the deployment of the RF Transmitter at the traffic circle, transmitting data to an RF Receiver in another RPi board. The data is accumulated in the ThingSpeak API for further analysis.

The power consumption analysis considers the general case of RF modules consuming 9V of power. The equation (T=T+ (T1*DV)) is applied to understand power consumption patterns during data transfer to the cloud, as illustrated in Figure 3. During traffic congestion data will be sent from RF (Tx) to RF (Rx) over an RPi board accumulated in ThingSpeak API. The collected data is formatted in xls format, enabling the analysis of power consumption trends across different usage scenarios, such as weekdays and weekends.



S.No	Date& Time	Time Difference	RF(Pc)
1	Wed Dec 5 15:46:13 201	3	27
2	Wed Dec 5 15:46:16 201	2	45
3	Wed Dec 5 15:46:24 201	3	72
4	Wed Dec 5 15:46:26 201	2	90
5	Wed Dec 5 15:46:28 201	2	108
6	Wed Dec 5 15:46:30 201	1	117

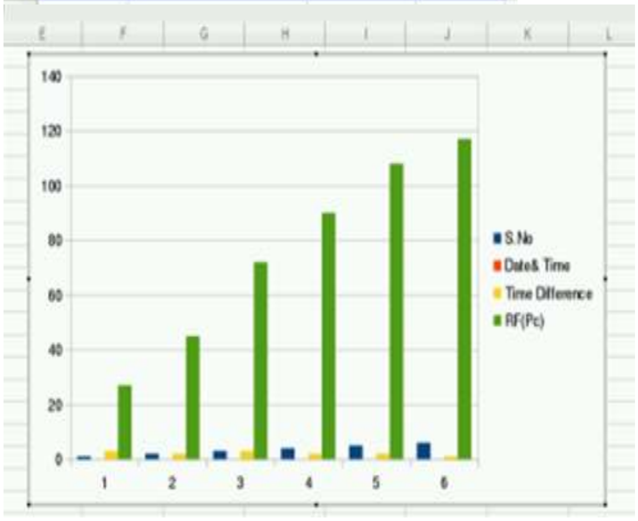
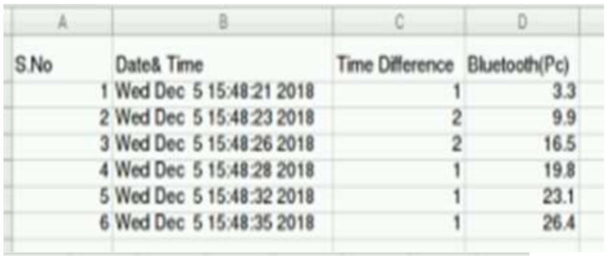


Fig 3: Bar chart depicting the collected and analysed power consumption data for RF Transmitter and Receiver (R Mode) over the Raspberry Pi board in the context of traffic monitoring

2.1.3 Test Case (3): Data Collection and Analysis for B Mode

The third test case focuses on the utilization of an IR Sensor combined with an HC-05 Serial Bluetooth Module over the RPi board [B Mode (IR sensor+HC-05 Serial Bluetooth Module over the RPi board)] for traffic monitoring. Figure 1 showcases the deployment of the Bluetooth Module at the traffic circle, handling data transmission to the ThingSpeak API with start and stop signals from an integrated IR Sensor.

Bluetooth, consuming 3.3V of power, is analysed in terms of power consumption patterns during data transfer, as demonstrated in Figure 4. The power consumption equation is applied to understand the variations in power consumption based on the time taken to update data to the cloud.



A	B	C	D
S.No	Date& Time	Time Difference	Bluetooth(Pc)
1	Wed Dec 5 15:48:21 2018	1	3.3
2	Wed Dec 5 15:48:23 2018	2	9.9
3	Wed Dec 5 15:48:26 2018	2	16.5
4	Wed Dec 5 15:48:28 2018	1	19.8
5	Wed Dec 5 15:48:32 2018	1	23.1
6	Wed Dec 5 15:48:35 2018	1	26.4

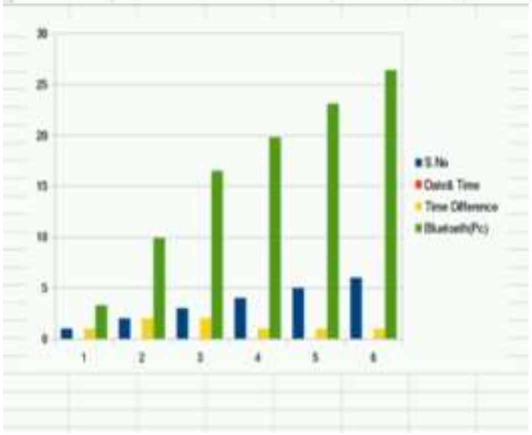


Fig 4: Bar chart illustrating the collected and analysed power consumption data associated with the Bluetooth module (B Mode) over the Raspberry Pi board, within the context of intelligent traffic monitoring

2.1.4 Results and Discussion: When a user approaches a traffic circle with a vehicle, the system dynamically switches between three modes: W (White), R (Red), and B (Blue), based on the user's preferences and the vehicle's accessible range. This switching is facilitated by the Auto Mode feature, and the data collected through API Cloud captures the power consumption patterns associated with each mode.

In the experiment, the consolidated power analysis results are presented in Figure 5. This figure provides a visual representation of the power consumption patterns observed during the entire experiment. The data collected from each mode (W, R, and B) individually contributes to understanding the power efficiency of the vehicle in different scenarios.

To optimize power efficiency further, cross-functional data from all three modes (W, R, and B) are combined to create a comprehensive W_R_B Cluster. The ultimate combined data graph, illustrated in Figure 6, showcases the synergistic power efficiency achieved when considering the cross-functional data of all modes simultaneously. This consolidated approach allows for a more holistic analysis of power consumption, offering insights into the optimal performance of the vehicle at a traffic circle.

By leveraging the W_R_B Cluster data, the system can make informed decisions to enhance power efficiency, providing a seamless and efficient experience for the user as they navigate through traffic circles. The figures mentioned serve as visual aids to better understand the power consumption patterns and efficiency gains achieved through the integration of cross-functional data.

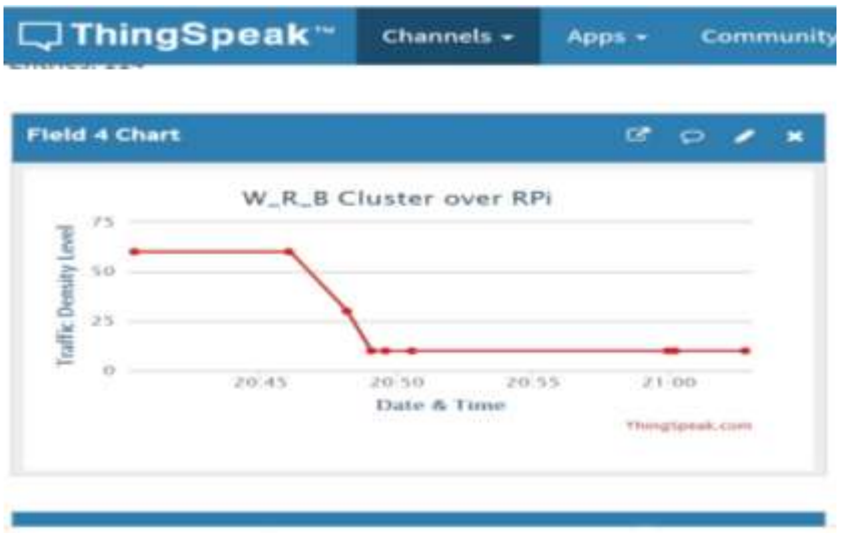
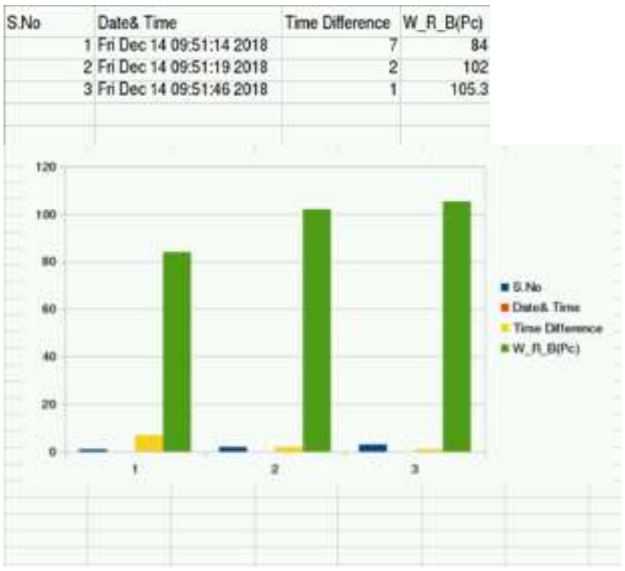


Fig 5: The W_R_B Cluster data, uploaded to the cloud



S.No	Date& Time	Time Difference	W_R_B(Pc)
1	Fri Dec 14 09:51:14 2018	7	84
2	Fri Dec 14 09:51:19 2018	2	102
3	Fri Dec 14 09:51:46 2018	1	105.3

Fig 6: The power consumption data for the W_R_B Cluster collected and analyzed using a Bar Chart, illustrating the efficiency over the RPi board

2.2 Security Analysis: The initial phase of the security analysis research concentrates on innovatively implementing the Twofish algorithm, a symmetric key encryption method renowned for its security and adaptability. While Twofish is recognized for its robust security features, the study aims to enhance its applicability in the context of IoT communication, where real-time responsiveness is critical. The proposed algorithm prioritizes efficient time management, specifically targeting the optimization of the encryption process to minimize delays in handling 16 bytes of data over the internet. This optimization seeks to bolster the performance of IoT devices by ensuring faster encryption, facilitating more seamless and responsive communication without compromising the requisite security standards.

Here, the system architecture of the Twofish algorithm is detailed, employing a Feistel network structure illustrated in Figure 7. This walkthrough utilizes a 128-bit key length and a 128-bit plaintext for explanation purposes. The process initiates with Key Expansion, where the 128-bit key undergoes expansion to generate a set of round subkeys through the key schedule algorithm [98]. The 128-bit plaintext is then divided into left (L0) and right (R0) halves, and an initial whitening step occurs through XORing the input block with whitening subkeys. Subsequently, the algorithm performs a series of round operations, including the Feistel Function (F-function), Key Mixing, Permutation, and Swap steps. The F-function operates on the right half (Ri) and involves the Feistel function, S-box substitution, and MDS matrix multiplication. The output of the F-function is XORed with the left half (Li), and the block undergoes a fixed permutation function to introduce diffusion and confusion. The output of the permutation step becomes the new Ri for the next round, while the previous Ri becomes the new Li for the subsequent round. After completing the required number of rounds, inverse operations are applied to obtain the ciphertext, involving the reversal of permutation, key mixing, and F-function operations.

Furthermore, the Twofish algorithm, characterized by a Feistel network structure with 16 iterations (Figure 8), incorporates a Function H, forming the heart of the Feistel function. This function consists of four key-dependent 8-by-8-bit S-boxes and an MDS (Maximum Distance Separable) matrix over the Galois Field [93]. The Pseudo-Hadamard transform (PHT) and two additions modulo subkey 2^32 complement the heart function. Notably, Twofish optimizes encryption delays through various techniques, such as a streamlined Key Schedule for efficient generation of round subkeys, a Feistel Network Structure for parallelization, and a Substitution-Permutation Network (SPN) within each round for actual encryption operations. The SPN includes the F-Function, Key Mixing, S-Boxes, and MDS Matrix operations, all designed for efficiency while maintaining a high level of security [99][100][101]. In the context of IoT devices, potential optimizations include Hardware Acceleration, Code Optimization, Key Size Selection, Power-Aware Design, and Parallelization (Figure 8). These considerations aim to enhance Twofish's performance on resource-constrained IoT devices, addressing challenges related to computational capabilities, power consumption, and parallel processing.

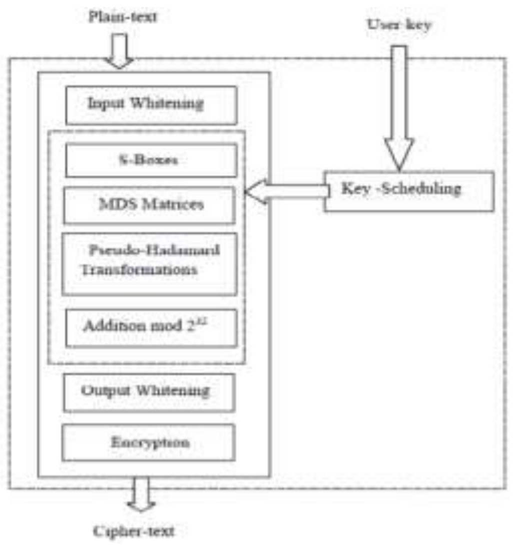


Fig 7: Steps in Twofish Algorithm (98)

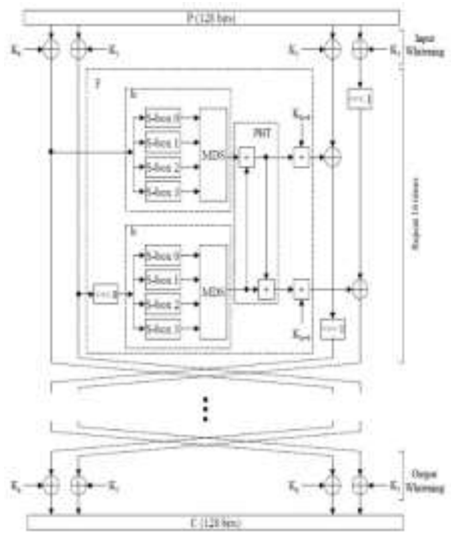


Fig 8: Twofish structure (93)

2.2.1 Pseudo code for Twofish Feistel Network

```
# Constants and Subkeys
KEY_SIZE = 128  # Key size in bits
BLOCK_SIZE = 128  # Block size in bits
NUM_ROUNDS = 16  # Number of rounds

# Key Schedule Algorithm
def key_schedule(key):
    # Key expansion and subkey generation
    # Your implementation here
```

```
subkeys = generate_subkeys(key)
return subkeys

# Feistel Function
def feistel_function(input, round_subkey):
    # Expand the input
    expanded_input = expand(input)

    # XOR with the round subkey
    xored_input = expanded_input XOR round_subkey

    # S-box substitution
    sbox_output = sbox_substitution(xored_input)

    # Multiply with MDS matrix
    multiplied_output = multiply_mds(sbox_output)

    # Output mixing
output = xor_combine(multiplied_output)

return output

# Encryption
def encrypt(plaintext, key):
subkeys = key_schedule(key)

left = plaintext.left_half()  # Get the left half of the plaintext
right = plaintext.right_half()  # Get the right half of the plaintext

for i in range(NUM_ROUNDS):
        # Save the current values of the left and right halves
temp = left

        # Apply Feistel function on the right half
        feistel_output = feistel_function(right, subkeys[i])

        # XOR the Feistel output with the left half
left = right XOR feistel_output

        # Set the right half with the saved value
right = temp
  # Swap the left and right halves
left, right = right, left

    # Apply final round of Feistel function
    feistel_output = feistel_function(right, subkeys[NUM_ROUNDS])
left = left XOR feistel_output

    # Concatenate the left and right halves
ciphertext = concatenate(left, right)

return ciphertext

# Decryption
def decrypt(ciphertext, key):
```

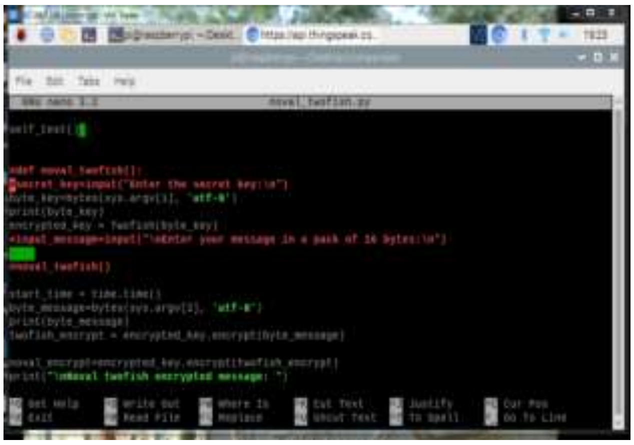
```
subkeys = key_schedule(key)

left = ciphertext.left_half()  # Get the left half of the ciphertext
right = ciphertext.right_half()  # Get the right half of the ciphertext

   # Apply final round of Feistel function
   feistel_output = feistel_function(right, subkeys[NUM_ROUNDS])
left = left XOR feistel_output

   # Swap the left and right halves
left, right = right, left

for i in range(NUM_ROUNDS-1, -1, -1):
      # Save the current values of the left and right halves
temp = left

      # Apply Feistel function on the right half
      feistel_output = feistel_function(right, subkeys[i])

      # XOR the Feistel output with the left half
left = right XOR feistel_output

      # Set the right half with the saved value
right = temp

   # Concatenate the left and right halves
plaintext = concatenate(left, right)

return plaintext
```

In the context of comparing the RSA, 3DES, and Twofish algorithms, including the novel Twofish Algorithm designed to address time constraints [95], an evaluation of their real-time performance is crucial. This assessment involves calculating the time taken by each algorithm for both encryption and decryption of a given message with a secret key. The study also explores hardware acceleration techniques that can significantly impact the efficiency of these cryptographic algorithms. Notably, Field-Programmable Gate Arrays (FPGAs) offer programmable circuits for creating custom digital circuits, allowing the optimization of Twofish operations for parallel execution. Application-Specific Integrated Circuits (ASICs) are custom-designed circuits tailored for specific applications, offering fixed-functionality that can be optimized for Twofish operations with high efficiency. Graphics Processing Units (GPUs), originally designed for visual applications, demonstrate parallel computing capabilities suitable for cryptographic operations, achieving speedups through parallelization. Hardware Security Modules (HSMs) act as dedicated cryptographic devices, providing secure key management and offloading computational burdens for high-speed cryptographic operations. Instruction Set Extensions in modern processors, though primarily designed for AES, may enhance the efficiency of other symmetric ciphers, contributing to improved cryptographic performance. As the choice of hardware acceleration depends on application requirements, available platforms, and associated development efforts, careful consideration is needed to align acceleration techniques with specific needs and optimize performance, particularly when handling large volumes of data where setup and data transfer overheads can impact overall efficiency.

2.2.2 Novel Twofish Algorithm

In the proposed novel Twofish approach, a set of 16 bytes of data is supplied to the system. This data is generated through the Raspberry Pi board, specifically utilizing the Raspbian

Buster version. The implementation is carried out within the Python programming language, with the environment set to Python version 3.7. The entire process is facilitated through the VNC Viewer Remote login. The code snippet responsible for this operation is succinctly depicted in Figure 9, providing a visual representation of the code used to generate and input the 16-byte data into the system.



```
self_test()

def novel_twofish():
secret_key=input("Enter the secret key:\n")
byte_key=bytes(sys.argv[1], 'utf-8')
print(byte_key)
encrypted_key = Twofish(byte_key)
input_message=input("\nEnter your message in a pack of 16 bytes:\n")

novel_twofish()

start_time = time.time()
byte_message=bytes(sys.argv[2], 'utf-8')
print(byte_message)
twofish_encrypt = encrypted_key.encrypt(byte_message)

noval_encrypt=encrypted_key.encrypt(twofish_encrypt)
print("\nNoval twofish encrypted message: ")
```

Fig 9: Code viewed through VNC Viewer Remote login on RPi for data generation and input

In this phase of the analysis, we will delve into the implementation details of the novel Twofish algorithm on the Raspberry Pi (RPi) board. The structure of the novel Twofish algorithm is illustrated in Figure 10. This algorithm involves the application of the first 7 rounds of a comparable Feistel network structure, incorporating input whitening for a 128-bit key length (plaintext).
The pseudo code for Novel Twofish's Feistel network process as follows.

Input:

Key: 128-bit key (K)

Plaintext: 128-bit plaintext (P)

Encryption:

Initialize L0 and R0 with the plaintext P.

For each round i (1 to 7):

a. Compute the F-function output by incorporating Ri and the round subkey Ki using Feistel function, S-box substitution, and MDS matrix multiplication.

b. XOR the F-function output with Li to obtain the new Ri.

c. Swap Li and Ri to prepare for the next round.

For each round i (8 to 16):

a. Compute the F-function output by incorporating Ri and the round subkey Ki using Feistel function, S-box substitution, and MDS matrix multiplication.

b. XOR the F-function output with Li to obtain the new Ri.

c. Swap Li and Ri to prepare for the next round.

Decryption:

Initialize L16 and R16 with the ciphertext.

For each round i (16 to 8, in reverse):

a. Compute the F-function output by incorporating Li and the round subkey Ki using Feistel function, S-box substitution, and MDS matrix multiplication (similar to encryption).

b. XOR the F-function output with Ri to obtain the new Li.

c. Swap Li and Ri to prepare for the next round.

For each round i (7 to 1, in reverse):

a. Compute the F-function output by incorporating Li and the round subkey Ki using Feistel function, S-box substitution, and MDS matrix multiplication (similar to encryption).

b. XOR the F-function output with Ri to obtain the new Li.

c. Swap Li and Ri to prepare for the next round.

Encryption:

```
Input:
Key: 128-bit key (K)
Plaintext: 128-bit plaintext (P)
Initialization:
L0 = leftmost 64 bits of P
R0 = rightmost 64 bits of P
For each round i (1 to 7) do:
    F_output = FeistelFunction(Ri, Ki)  // Feistel function incorporating Ri and round subkey Ki
    F_output = SBoxSubstitution(F_output)  // Apply S-box substitution to F_function output
    F_output = MDSMatrixMultiplication(F_output)  // Multiply with MDS matrix
    R_new = L_i-1 XOR F_output  // XOR F_function output with Li-1 to obtain new Ri
    L_new = R_i-1  // Update Li for the next round
For each round i (8 to 16) do:
    F_output = FeistelFunction(Ri, Ki)
    F_output = SBoxSubstitution(F_output)
    F_output = MDSMatrixMultiplication(F_output)
    R_new = L_i-1 XOR F_output
    L_new = R_i-1
Ciphertext = Concatenate(R16, L16)  // Final ciphertext
```

Decryption:

Initialization:

```
L16 = leftmost 64 bits of Ciphertext
R16 = rightmost 64 bits of Ciphertext
For each round i (16 to 8, in reverse) do:
    F_output = FeistelFunction(Li, Ki)  // Similar to encryption
    F_output = SBoxSubstitution(F_output)
    F_output = MDSMatrixMultiplication(F_output)
    L_new = R_i+1 XOR F_output  // XOR F_function output with Ri+1 to obtain new Li
    R_new = L_i+1  // Update Ri for the next round
For each round i (7 to 1, in reverse) do:
    F_output = FeistelFunction(Li, Ki)  // Similar to encryption
    F_output = SBoxSubstitution(F_output)
    F_output = MDSMatrixMultiplication(F_output)
    L_new = R_i+1 XOR F_output
    R_new = L_i+1
Plaintext = Concatenate(L0, R0)  // Final plaintext
```

This pseudo code represents the detailed encryption and decryption processes of the Novel Twofish algorithm, incorporating the Feistel network structure, F-function, S-box substitution, MDS matrix multiplication, and key XOR operations as described in the provided information.

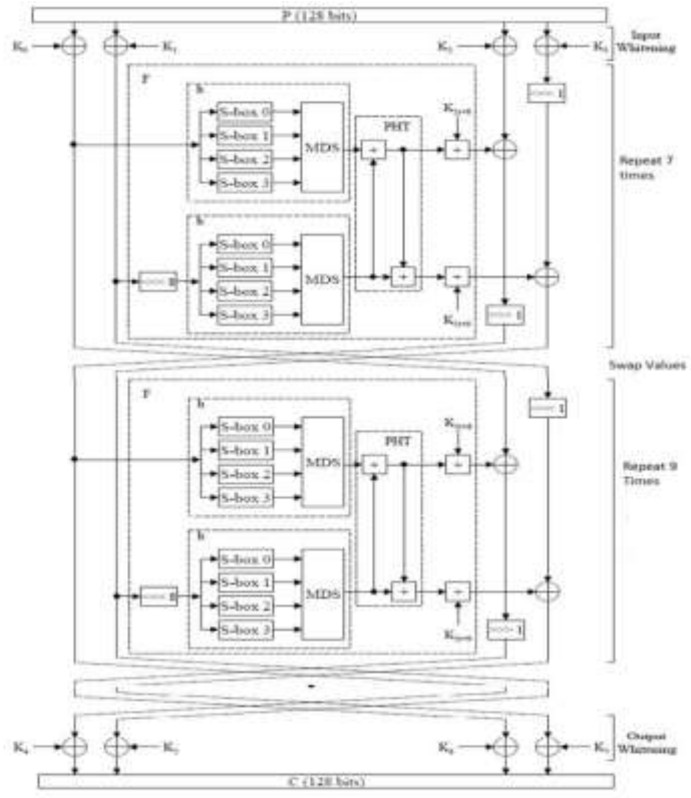


Fig 10: Structure of the proposed novel Twofish Algorithm

The optimization of swapping the output values before the commencement of the 8th round in the Novel Twofish algorithm, as depicted in Figure 10, results in a significant acceleration of the algorithm's decryption process. This strategic adjustment notably impacts the computational speed, particularly in the final decryption round, leading to a marked reduction in the overall time required to calculate and retrieve the decrypted text.

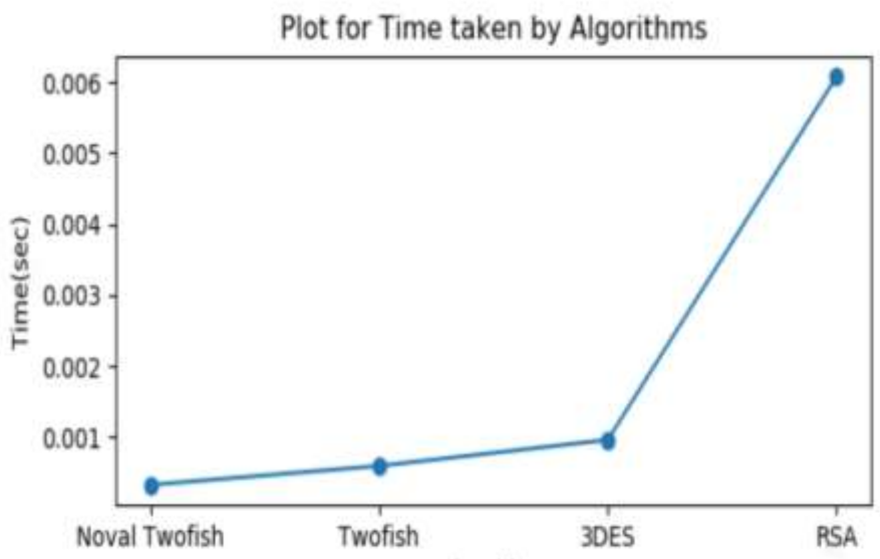


Fig 10: Time engaged comparison between 3 algorithms with Novel Twofish algorithm.

- Novel Twofish: The proposed algorithm consistently outperforms the standard Twofish for all data sizes shown. This suggests that the proposed optimization techniques have successfully reduced the execution time.
- 3DES: 3DES appears to be slower than both Twofish algorithms for most data sizes. This is expected, as 3DES is generally less efficient than Twofish.
- RSA: RSA is significantly slower than all other algorithms at all data sizes shown. This is because RSA is primarily used for key exchange and not for bulk data encryption, which requires faster processing.

The encrypted message, decrypted message and time taken for executing RSA, 3DES, Twofish and novel Twofish algorithm which as to be taken as results through command prompt is shown in as in figure 11 and 12.

```
[illegible]
Enter the secret key of 16 bytes:
Caution: avoid spaces
[illegible]
Enter the message of 16 bytes:
Caution: avoid spaces
[illegible]

Executing Novel Twofish Algorithm
[illegible]

Novel Twofish encrypted message:
[illegible]

Decrypted message:
[illegible]
Time taken: [illegible] seconds

Executing Twofish Algorithm
```

Fig 11: Novel Twofish Algorithm execution on RPi board

```
[illegible]
  import imp
[illegible]

Twofish Encrypted message:
[illegible]

Decrypted message:
[illegible]
Time taken: [illegible] seconds

Executing 3DES Algorithm
encrypted_text: [illegible]
decrypt_message: [illegible]
Time taken: [illegible] seconds

Executing RSA Algorithm
[illegible]
[illegible]
Time taken: [illegible] seconds
Plotting the graph
[illegible]
```

Fig 12: RSA, 3DES, Twofish Algorithm execution on RPi board

2.2.3 Application of Novel algorithm on Smart Energy Meter

Here, the details of the proposed wireless digital energy meter setup are showcased in Figure 13. It leverages the capabilities of the Internet of Things (IoT) to remotely monitor and calculate power consumption for specific devices. Additionally, it integrates the novel Twofish algorithm for enhanced security in data transmission.

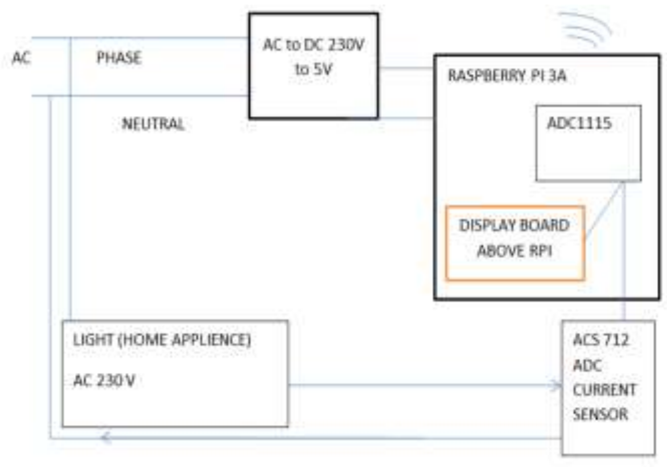


Fig 13: Prototype Setup for Smart Energy Meter

Hardware Components:

- Raspberry Pi 3A/3B: This serves as the central processing unit of the system, running the Raspbian operating system and handling data acquisition, analysis, and communication.
- AC-to-DC Converter (230V to 5V): This component bridges the gap between the standard AC power supply (230V) and the low voltage requirements of the Raspberry Pi (5V).
- Bulb Holder: Connected along the phase line, this acts as a measuring point for the current sensor.
- Current Sensor (ACS712 ADC): This sensor measures the current flowing through the bulb holder and converts it into an analog signal.
- 16-bit 4 Channel ADC (ADS1115): This analog-to-digital converter (ADC) transforms the analog signal from the current sensor into a digital format that the Raspberry Pi can understand. It boasts 10 pins and I2C compatibility for easy communication with the Pi.
- OLED Display Board: This display, mounted alongside the Raspberry Pi, provides a real-time visual representation of the calculated power consumption.

The Integration Process:

1. Power Flow: The AC power supply enters through the 2-pin socket and is converted to 5V by the AC-to-DC converter. This powers the Raspberry Pi and other components.
2. Current Measurement: The bulb holder connected to the phase line allows the ACS712 current sensor to measure the current flowing through the attached bulb.

3. Analog-to-Digital Conversion: The ADS1115 ADC receives the analog signal from the current sensor and converts it into a digital format that the Raspberry Pi can process.
4. Power Consumption Calculation: The Raspberry Pi analyses the digital data from the ADC and applies appropriate formulas to calculate the power consumption of the connected device (bulb in this case).
5. Data Visualization and Security: The calculated power consumption is displayed on the OLED display for real-time monitoring. Additionally, the novel Twofish algorithm is integrated to secure data transmission between the meter and any remote monitoring system.

2.2.4 Results and Discussion

IoT cloud service is provided to the smart energy through ThingSpeak (to visualize and study data stream) as shown in figure 14.

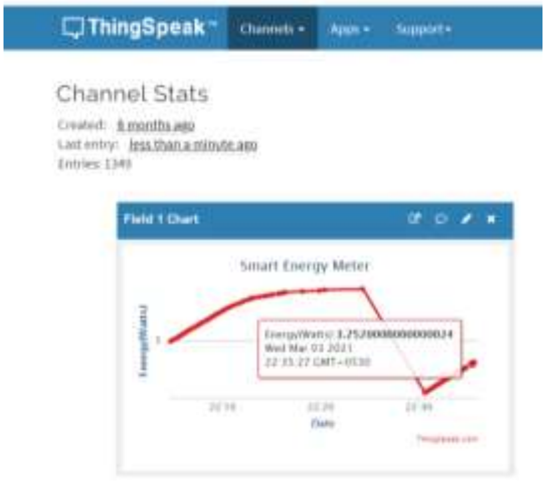
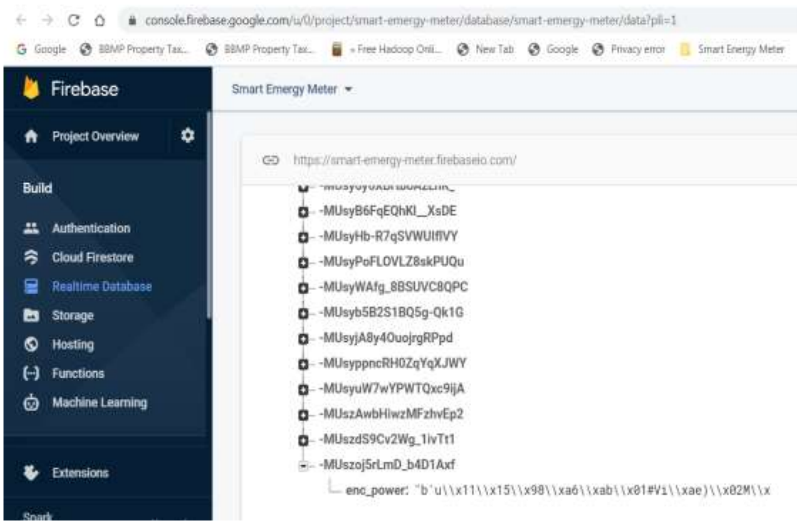


Fig 14: ThingSpeak channel for smart energy meter

Firebase database to store encrypted and decrypted data on cloud is as shown in figure 15.



```
-MUsyB6FqEQhKl__XsDE
-MUsyHb-R7qSVWUIfIVY
-MUsyPoFLOVLZ8skPUQu
-MUsyWAfg_8BSUVC8QPC
-MUsyb5B2S1BQ5g-Qk1G
-MUsyjA8y4OuojrgRPpd
-MUsyppncRH0ZqYqXJWY
-MUsyuW7wYPWTQxc9ijA
-MUszAwbHiwzMFzhvEp2
-MUszdS9Cv2Wg_1ivTt1
-MUszoj5rLmD_b4D1Axf
  enc_power: "b'u\\x11\\x15\\x98\\xa6\\xab\\x01#Vi\\xae)\\x02M\\x
```

Fig 15: Firebase database view for smart energy meter

The presence of the "Encrypted Data" field explicitly shows the integration of the novel Twofish algorithm. This means that the power consumption data collected by the meter is encrypted using your algorithm before being transmitted to the ThingSpeak cloud platform.

This adds a layer of security to protect sensitive energy usage information from unauthorized access or manipulation during transmission.

Overall, Figure 14 provides a clear visual representation of the smart energy meter setup and its integration with the ThingSpeak platform. The highlighted "Encrypted Data" field effectively demonstrates the implementation of the novel Twofish algorithm for enhanced data security.

By leveraging IoT technologies, wireless smart energy systems can enhance energy management by enabling real-time monitoring, data analytics, remote control, demand response, energy optimization, and user engagement. These capabilities contribute to upgraded power effectiveness, decreased energy waste, and ultimately, more sustainable energy consumption practices. Table 3.2, 3.3 and 3.4 comprises of data extracted from Traffic Management System.

Table 1: Power Management Wi-Fi Dataset for visualisation and analysis

Sl.no	Date & Time	Wi-Fi----IR (pc)	Power Consumption range
1	Mon Aug 7 06:10:25 2023	125	2
2	Mon Aug 7 07:20:25 2023	123	2
3	Mon Aug 7 08:33:28 2023	230	4
4	Mon Aug 7 09:25:44 2023	245	4
5	Mon Aug 7 10:26:55 2023	268	5
6	Mon Aug 7 11:40:50 2023	278	5
7	Mon Aug 7 12:10:45 2023	245	4
8	Mon Aug 7 13:37:55 2023	201	4
9	Mon Aug 7 14:26:35 2023	201	4
10	Mon Aug 7 15:44:15 2023	156	3
11	Mon Aug 7 16:15:45 2023	145	3
12	Mon Aug 7 17:18:19 2023	200	4
13	Tue Aug 8 18:35:19 2023	236	4
14	Tue Aug 8 19:35:55 2023	256	5
15	Tue Aug 8 20:27:55 2023	277	5
16	Tue Aug 8 21:15:38 2023	288	5
17	Tue Aug 8 22:27:59 2023	298	5
18	Tue Aug 8 23:45:49 2023	213	4
19	Tue Aug 8 00:35:29 2023	129	2
20	Tue Aug 9 01:34:49 2023	45	0
21	Tue Aug 9 02:34:29 2023	46	0
22	Tue Aug 9 03:35:29 2023	41	0
23	Tue Aug 9 04:45:29 2023	82	1
24	Tue Aug 9 05:25:59 2023	125	2
25	Tue Aug 9 06:45:09 2023	169	3
26	Wed Aug 9 08:15:11 2023	245	4
27	Wed Aug 9 09:17:21 2023	255	5

Table 2: Power Management RF Dataset for visualisation and analysis

Sl.no	Date & Time	Radio Frequency (pc)	Power Consumption range

1	Mon Aug 7 06:10:25 2023	120	2
2	Mon Aug 7 07:20:25 2023	150	2
3	Mon Aug 7 08:33:28 2023	155	3
4	Mon Aug 7 09:25:44 2023	160	3
5	Mon Aug 7 10:26:55 2023	159	3
6	Mon Aug 7 11:40:50 2023	166	3
7	Mon Aug 7 12:10:45 2023	167	3
8	Mon Aug 7 13:37:55 2023	169	3
9	Mon Aug 7 14:26:35 2023	170	3
10	Mon Aug 7 15:44:15 2023	178	3
11	Mon Aug 7 16:15:45 2023	179	3
12	Mon Aug 7 17:18:19 2023	180	3
13	Tue Aug 8 18:35:19 2023	180	3
14	Tue Aug 8 19:35:55 2023	182	3
15	Tue Aug 8 20:27:55 2023	183	3

Table 3: Power Management Bluetooth Dataset for visualisation and analysis

Sl.no	Date & Time	Bluetooth (pc)	Power Consumption range
1	Mon Aug 7 06:10:25 2023	5	0
2	Mon Aug 7 07:20:25 2023	2.2	0
3	Mon Aug 7 08:33:28 2023	3.9	0
4	Mon Aug 7 09:25:44 2023	6.7	0
5	Mon Aug 7 10:26:55 2023	7	0
6	Mon Aug 7 11:40:50 2023	18	0
7	Mon Aug 7 12:10:45 2023	9	0
8	Mon Aug 7 13:37:55 2023	9.3	0

9	Mon Aug 7 14:26:35 2023	11.7	0
10	Mon Aug 7 15:44:15 2023	15.6	0
11	Mon Aug 7 16:15:45 2023	29.5	0
12	Mon Aug 7 17:18:19 2023	35.1	0
13	Tue Aug 8 18:35:19 2023	38.4	0
14	Tue Aug 8 19:35:55 2023	39.22	0
15	Tue Aug 8 20:27:55 2023	40.33	0
16	Tue Aug 8 21:15:38 2023	41.23	0

The provided three datasets (Tables 1, 2, and 3) extracted from a traffic management system, each focusing on a different wireless technology – Wi-Fi, Radio Frequency (RF), and Bluetooth. These datasets capture power consumption data over roughly two weeks, along with corresponding timestamps and power consumption ranges. Additionally, Table 4 offers data from a Home Computerization System, featuring power consumption and energy density readings collected within the same timeframe.

Table 4: A dataset on Smart Energy designed for visualization and analysis purposes

Sl.no	**Date & Time**	**Power Consumption(W)**	**Energy Density**
1	Mon Aug 7 06:10:25 2023	50	0
2	Mon Aug 7 07:20:25 2023	80	1
3	Mon Aug 7 08:33:28 2023	120	2
4	Mon Aug 7 09:25:44 2023	180	3
5	Mon Aug 7 10:26:55 2023	201	4
6	Mon Aug 7 11:40:50 2023	245	4
7	Mon Aug 7 12:10:45 2023	255	4
8	Mon Aug 7 13:37:55 2023	40	0
9	Mon Aug 7 14:26:35 2023	40	0
10	Mon Aug 7 15:44:15 2023	60	0
11	Mon Aug 7 16:15:45 2023	45	0
12	Mon Aug 7 17:18:19 2023	35	0
13	Tue Aug 8 18:35:19 2023	88	1
14	Tue Aug 8 19:35:55 2023	112	2
15	Tue Aug 8 20:27:55 2023	190	3
16	Tue Aug 8 21:15:38 2023	222	4
17	Tue Aug 8 22:27:59 2023	270	5
18	Tue Aug 8 23:45:49 2023	290	5
19	Tue Aug 8 00:35:29 2023	80	1
20	Tue Aug 9 01:34:49 2023	70	0

21	Tue Aug 9 02:34:29 2023	75	1
22	Tue Aug 9 03:35:29 2023	60	0
23	Tue Aug 9 04:45:29 2023	55	0
24	Tue Aug 9 05:25:59 2023	59	0
25	Tue Aug 9 06:45:09 2023	120	2
26	Wed Aug 9 08:15:11 2023	140	2
27	Wed Aug 9 09:17:21 2023	240	4
28	Wed Aug 9 10:18:19 2023	290	5
29	Wed Aug 9 11:14:51 2023	240	4
30	Wed Aug 9 12:18:21 2023	200	4
31	Wed Aug 9 13:45:41 2023	160	3
32	Wed Aug 9 14:17:41 2023	190	3
33	Wed Aug 9 15:19:21 2023	145	2
34	Wed Aug 9 16:16:17 2023	155	2
35	Wed Aug 9 17:13:33 2023	175	3

3. Conclusion: This research investigated two distinct but interconnected domains: optimizing power efficiency in traffic circles and secure data management in wireless smart energy systems. Both studies achieved significant advancements through novel implementations:

Traffic Circle Power Efficiency:

- A dynamic mode switching system (White, Red, Blue) for navigating traffic circles was developed, adapting to user preferences and vehicle range.
- Cross-functional data analysis using the W_R_B Cluster revealed synergistic power efficiencies by combining data from all modes.
- This holistic approach identified optimal vehicle performance strategies at traffic circles, leading to substantial energy savings and a seamless user experience.

Secure Data Management in Wireless Smart Energy Systems:

- Integration of the robust Twofish encryption algorithm with IoT cloud services provided a comprehensive solution for secure data transmission.
- This combination leveraged Twofish's strong security features and cloud scalability to ensure confidentiality and reliable data exchange.

- Compared to lightweight alternatives, Twofish offered superior security strength, data confidentiality, and long-term sustainability, even for low-power devices.

Overall Significance:

The presented research offers valuable contributions in both domains:

- Traffic circle navigation: Improved power efficiency and user experience through adaptive mode switching and advanced data analysis.
- Wireless smart energy systems: Enhanced security and data confidentiality through robust encryption and cloud integration.

These advancements pave the way for more sustainable transportation and energy management strategies, enabling a future powered by secure and efficient technologies.

Future Directions:

- Further refinement of the traffic circle navigation system with real-time traffic data incorporation for dynamic route optimization.
- Continuous security assessment and potential integration of post-quantum cryptography solutions for long-term data protection in smart energy systems.
- Exploration of the W_R_B Cluster approach in other transportation scenarios, such as highway intersections or parking lots, to optimize overall mobility efficiency.

4. References:

1. R. Florin, P. Ghazizadeh, A. G. Zadeh, S. El-Tawab, and S.Olariu, “Reasoning about job completion time in vehicular clouds,”IEEE Transactions on Intelligent Transportation Systems, vol. PP, no.99, pp.1–10, 2016.
2. O. Popescu, S. Sha-Mohammad, H. Abdel-Wahab, D. C. Popescu andS. El-Tawab, "Automatic Incident Detection in IntelligentTransportation Systems Using Aggregation of Traffic ParametersCollected Through V2I Communications," in IEEE IntelligentTransportation Systems Magazine, vol. 9, no. 2, pp. 64-75, Summer2017.
3. S. El-Tawab, R. Oram, M. Garcia, C. Johns, and B. B. Park,“Poster: Monitoring Transit Systems using Low cost WIFITechnology,” in 2016 IEEE Vehicular Networking Conference(VNC), Dec 2016, pp. 1–2.
4. A. Salman, W. Diehl and J. P. Kaps, "A light-weighthardware/software co-design for pairing-based cryptography with lowpower and energy consumption," 2017 International Conference onField Programmable Technology (ICFPT), Melbourne, VIC, 2017,pp. 235-238.
5. A. Salman, A. Ferozpuri, E. Homsirikamol, P. Yalla, J.-P.Kaps and K. Gaj, "A scalable ECC processor implementation for high-speed and lightweight with side-channel countermeasures", 2017 International Conference on ReConFigurable Computing and FPGAs (ReConFig), Nov 2017.
6. "Energy-Efficient Communication Protocol for Wireless Microsensor Networks" by W. Heinzelman, et al. (2000)
7. "Energy Harvesting for the Internet of Things: A Survey" by A. O. Bicen and I. F. Akyildiz (2017)
8. "Energy-Efficient Task Offloading and Resource Allocation in Mobile Edge Computing for IoT" by C. You, et al. (2017)
9. M. Ibrahim, A. Elgamri, S. Babiker, and A. Mohamed, ―Internet of things based smart environmental monitoring using the raspberry-pi computer, ‖ in Digital Information Processing and Communications (ICDIPC), 2015 Fifth International Conference on, Oct 2015, pp. 159–164.
10. R. Fisher, L. Ledwaba, G. Hancke, and C. Kruger, ―Open hardware: a role to play in wireless sensor networks and Sensors, Vol. 15, no. 3, pp. 6818–6844, 2015.
11. Raspberry pi official web site. [Online]. Available: https://www.raspberrypi.org/

12. A. Technology, “EPCglobal class 1 gen 2 RFID specifications.” Whitepaper. http://www.alientechnology.com/docs/AT wp EPCGlobal WEB.pdf.
13. R. Florin, P. Ghazizadeh, A. G. Zadeh, S. El-Tawab, and S. Olariu, ―Reasoning about job completion time in vehicular clouds, ‖ IEEE Transactions on Intelligent Transportation Systems, vol. PP, no. 99, pp.1–10, 2016.
14. https://thingspeak.com/
15. V. Kumar, S. Chaisiri, R. Ko, and Institution of Engineering and Technology, Data security in cloud computing. 2017.
16. H. Delfs and H. Knebl, Introduction to Cryptography Principles and Applications. Springer, 2015.
17. J.-P. Aumasson and M. D. Green, Serious cryptography: a practical introduction to modern encryption. San Francisco: No Starch Press, 2017.
18. R. Arora, A. Parashar, and C. C. I. Transforming, “Secure user data in cloud computing using encryption algorithms,” International journal of engineering research and applications, vol. 3, no. 4, pp. 1922–1926, 2013.
19. Bruce Schneier, John Kelsey, Doug Whitingz David Wagnerx Chris Hall, Niels Ferguson “Twofish: A 128-Bit Block Cipher” AES submission, 1998.
20. Shun-Lung Su, Lih-Chyau Wuu, and Jhih-Wei Jhang, “A New 256- bits Block Cipher –Twofish 256”, Computer Engineering& Systems, International Conference in IEEE, 2010, pg 166 – 171