

AI-ASSISTED CRYPTOGRAPHIC KEY MANAGEMENT FOR SECURE INTERNET OF THINGS NETWORKS

Mohammed Ali Sohail¹ and Deepika Lokesh²

¹Department of Electrical and Electronic Engineering, College of Engineering and Computer Science, Jazan University, Kingdom of Saudi Arabia

²Department of Artificial Intelligence and Machine Learning, C. Byre Gowda Institute of Technology, India

Abstract

This work presents a new AI-driven Adaptive Lightweight Key Management (ADAL-KM) technique that combines lightweight cryptography operations with an intelligent key-risk assessment approach. An intelligent lightweight adaptive decision-making model is utilized to set proper key lifetimes and session keys distribution within hierarchical device groups. Thus, ADAL-KM saves cryptographic resources while ensuring security. The experimental assessment based on CICIoT2023 IoT traffic shows that ADAL-KM reduces the overheads of key management communication to 57.91 KB at 40 devices, which is 22.6% less than 74.86 KB for the Efficient Key Management System. The computation overhead drops to 121.1 ms from 158.7 ms, and the energy overhead decreases to 12.09 J from 15.43 J, which is a 21.7% decrease. The success rate of establishment of secure sessions by the method proposed is 98.9%, while it is 94.2% for the Efficient Key Management System. The rekeying latency of ADAL-KM becomes 71.0 ms, which is 26.4% lower than 96.5 ms.

Keywords:

Internet of Things, Cryptographic Key Management, Lightweight Cryptography, Artificial Intelligence, Secure Communication

1. INTRODUCTION

IoT has emerged as a distributed computing framework in which sensors, actuators, embedded controllers, gateways, mobile devices, and cloud-based computing resources continuously communicate. IoT enables smart homes, health care, transport, industrial automation, farming, and smart city services. The heterogeneity of IoT devices, however, poses major security challenges since IoT devices may have very low memory, computation power, battery life, and bandwidth. Traditional cryptography is still essential for securing IoT communication by providing confidentiality, integrity, authentication, and access control; however, traditional security techniques pose an unreasonable burden on constrained devices. Recent work in lightweight cryptography highlights the need for simplifying cryptography without compromising security [1].

Key management is a critical aspect of IoT security since the encryption algorithm fails to ensure proper security in case cryptographic keys are not well created, exchanged, stored, replaced, or canceled. A comprehensive survey of cryptographic key-management systems reveals that key creation, exchange, storage, distribution, and replacement form some of the significant phases of a key life cycle [2]. In IoT networks, such processes have to cope with heterogeneous devices and dynamic network environments. Also, studies on key management and authentication reveal that there is a need for mechanisms that can manage issues of security, privacy, scalability, and various network models within the IoT environment [3]. Thus, key management is not simply a matter of encryption but needs to

adapt to the changing behavior of devices and network environments.

However, some issues persist despite these advancements. Firstly, the issue revolves around constraints in terms of resources. Though lightweight cryptography algorithms lower processing power needs, the generation of keys and frequent authentication still demands considerable energy and communication resources. Current literature indicates that traditional cryptographic techniques may not be optimal for the constrained IoT environment, and this has prompted the creation of lightweight symmetric and asymmetric approaches [4]. Secondly, there is an issue regarding heterogeneity and dynamic nature of the network. IoT nodes vary greatly in terms of processing power, energy, communication distance, and security needs. For instance, the blockchain-based key management study shows that heterogeneity and dynamic member participation are major issues in conventional approaches due to unequal processing power [5].

The main issue dealt with in this research work is that most existing key management systems employ predefined key lifetime, fixed key distribution policy, and a centralized key management system. These schemes lack a proper differentiation among the low risk devices and high risk devices as well as stable and volatile communication scenarios. In this case, there will be frequent renewal of the keys which will lead to energy wastage and excessive communication costs. Furthermore, there may arise cases when the keys can become obsolete due to long life times which will make them vulnerable to key attack. Recent research has shown the importance of hierarchical key management approach which considers node properties like location and energy left while selecting the key servers and clusters [6].

The aim of this study is to propose an AI-Driven Adaptive Lightweight Key Management (ADAL-KM) technique for securing communications in the IoT network. The first aim is to formulate a key management architecture that requires minimum computation, communication, and energy expenditure. The second aim is to implement an adaptive intelligence component that assesses the device trust, key age, residual energy, communication pattern, and network activity. The third aim is to make the decision about key renewal requirements dynamically rather than assigning the same lifetime to all the devices. The fourth aim is to facilitate the secure and scalable distribution of keys in heterogeneous IoT networks without unnecessary rekeying operations.

Innovations in ADAL-KM consist of using lightweight cryptographic key management and an adaptive lifetime of the keys assisted by AI. Current solutions have considered the following mechanisms: partial key pre-distribution, blockchain key management, hierarchical key generation, PUF-based group

keys, and lightweight authentication [7]-[15]. Nevertheless, existing solutions typically consider only one architectural or cryptographic mechanism without considering joint adaptation of the key-management decisions based on the dynamically changing security and resource parameters of individual devices. In the proposed solution, key management is treated as a dynamic decision-making process, where the AI decides about the risk level of the active key and makes a decision whether the key needs to be kept or updated.

The first one is the creation of the ADAL-KM architecture where lightweight encryption scheme, hierarchical device clustering, adaptive key generation scheme, and intelligent key life-time management system are combined. This architecture delegates key management function to comparatively stronger gateway or edge devices, reducing the computational burden on constrained devices. The second one is the design of adaptive key renewal system where device trust, available energy, key lifetime, device behavior, and network traffic are considered for decision-making. The proposed mechanism ensures a compromise between security and resource consumption.

2. RELATED WORKS

IoT communication research has seen more efforts on lightweight cryptography and key-management protocols. [7] have conducted a review of IoT lightweight cryptography methods, and classified current solutions as either symmetric or asymmetric ones. The authors point out the potential resource overheads of traditional cryptography algorithms for small devices, and considered such factors as block size, key length, latency, throughput, and implementation cost. Even though lightweight cryptography decreases the overheads of cryptography algorithms, it is evident from the survey that security alone cannot be judged by the cryptographic algorithm complexity. Key establishment and its lifecycle become a critical issue, as a lightweight cipher needs a secure way to handle keys.

The work carried out by [8] provides a comprehensive review of various authentication techniques and key management in IoT settings. In particular, key agreement methods are classified based on network model, security needs, vulnerabilities, and performance attributes. Security and privacy issues associated with various applications such as health, smart agriculture, and intelligent transport are highlighted, while emphasizing the need for proper corrective measures. This paper provides the general framework of IoT key management, but the main focus of the work is the analysis of the problem rather than adaptive system operation. ADAL-KM builds on this line of investigation by taking changing conditions into account.

[9] designed an effective key management technique for lightweight IoT devices based on a Hashed Advanced Encryption Standard (HAES). This technique was proposed specifically to solve the problem of security and time issues in IoT communications. The proposed approach is able to prove the potential for integrating hashing and lightweight encryption in IoT key management. But a fixed cryptologic solution cannot take into account possible changes in energy level of devices, key exposure, and their communication. Therefore, ADAL-KM proposes an adaptive decision layer on top of lightweight cryptographic techniques.

[10] presented a key-management mechanism for lightweight block ciphers in the context of cluster-based topology and partial key pre-distribution. This scheme works with symmetric-key cryptography and implements partial key pre-distribution to establish secure communication for IoT systems with limited resources. While partial pre-distribution allows one to decrease the number of keys transmitted during communication, pre-configured keys may lack flexibility in case of changing membership and risk situations. This issue is solved by ADAL-KM in the form of adaptive key lifetime and rekeying.

[11] studied conventional methods and blockchain- and artificial intelligence-assisted techniques for cybersecurity in cyber-physical IoT systems. This paper emphasizes the variety of cyber attacks faced by IoT devices and the role of blockchain and artificial intelligence in enhancing conventional cybersecurity measures. An interesting link between intelligent cybersecurity mechanisms and cryptography is made by the paper. However, incorporating AI-based cybersecurity in IoT devices comes with its own computational challenges. The proposed ADAL-KM limits the AI processing only to decision-making and not heavy learning algorithms.

A decentralized blockchain-based key management protocol for heterogeneous and dynamic IoT devices was proposed [12]. In their model, there is an acknowledgment of the difference in device capability, use of hybrid encryption, blockchain, and smart contract technologies to achieve decentralized key management and secure rekeying. They have also conducted performance evaluation of their proposed scheme via simulation and IoT-based implementation. This particular solution solves the problem of single-point-of-failure related to centralized key management. However, it might have some overhead because of its blockchain technology.

The light key-management approach is supplemented by partial key-management, which aims at decreasing communication overhead, power consumption, and storage requirement. A subsequent study on lightweight block cipher key management using partial pre-distribution of keys observes decreased communication overhead, power consumption, and storage requirements with attack resistance analysis [13]. This shows the significance of minimization of resource usage in terms of key management. However, key pre-distribution is not by default dynamic risk-aware key renewal.

A lightweight authentication and key agreement protocol for IoT-based smart grids has been presented [14]. The protocol considers security needs arising out of transfer of private electricity-related data in public networks and emphasizes lightweight authentication and key agreement. Both authentication and key agreement have a direct relationship with key management as proper authentication results in establishing the trust necessary for session key generation. However, an authentication-based approach tends to emphasize on secure session setup than the continuous adaptation of the key lifecycle.

[15] suggested the use of Group-Key PHEMAP as a lightweight PUF-based protocol for dynamic and secure group key management. The protocol facilitates dynamic group membership without the need for any extra cryptographic keys and employs lightweight computation. The security properties, computational cost, communication cost, and scalability of the protocol have been analyzed formally, simulated using ns-3, and

implemented in IoT devices. PUF-based key generation provides a good hardware-based security feature, especially for dynamically formed groups. But the solution is based more on physical aspects of the devices than behavioral ones.

3. PROPOSED METHOD

The ADAL-KM technique allows one to create an adaptive security approach for heterogeneous IoT networks. This technique involves lightweight cryptography combined with an intelligent approach to key management, which assesses the trust of devices, the residual energy level, their behavior in communications, key age, and the network state. In contrast to the static approaches to key management, which apply the same lifetime values for all keys, ADAL-KM decides on what kind of key management should be performed based on the current status of the devices and their communication state. The technique includes six stages: device registration and initialization, feature extraction, adaptive risk assessment, hierarchical key generation, intelligent key renewal, and secure key distribution and revocation. The decisions are made by gateways/edge nodes that possess high computational power, and devices only carry out lightweight cryptographic calculations. Thus, the developed architecture minimizes the amount of unnecessary rekeying while providing shorter key lifetime and timely key replacement for risky devices.

The complete ADAL-KM process can be represented as

$$A_{KM} = \{D, F, T, K, R, V\} \quad (1)$$

where D represents registered IoT devices, F denotes the collected device and network features, T represents the adaptive trust and risk assessment module, K denotes cryptographic key generation and distribution, R represents adaptive key renewal, and V represents key validation and revocation.

The overall key-management objective is formulated as

$$\begin{aligned} \min_{K,R} & \left[\lambda_1 C_{comp} + \lambda_2 C_{comm} + \lambda_3 E_{key} + \lambda_4 L_{rekey} \right] \\ \text{s.t. } & S_{sec} \geq S_{min}, T_{dev} \geq T_{min} \end{aligned} \quad (2)$$

where C_{comp} denotes computational cost, C_{comm} represents communication overhead, E_{key} is key-management energy consumption, L_{rekey} is rekeying latency, S_{sec} denotes the achieved security level, T_{dev} represents device trust, and $\lambda_1, \lambda_2, \lambda_3, \lambda_4$ are weighting coefficients. This formulation enables ADAL-KM to minimize resource consumption while preserving the required security level.

3.1 PROCESS OF THE PROPOSED METHOD

The ADAL-KM method operates through the following sequential process:

Step 1: IoT Device Registration and Initialization

1. Detect and register each IoT device with the nearest trusted gateway.
2. Generate a unique device identifier.
3. Establish an initial authentication secret.
4. Record device capabilities and security requirements.

5. Assign the device to an appropriate hierarchical security group.

Step 2: Security and Resource Feature Collection

1. Monitor residual energy.
2. Measure communication frequency and packet behavior.
3. determine key age and previous renewal history.
4. Evaluate authentication failures and abnormal communication events.
5. Calculate the current device trust value.

Step 3: Adaptive Risk Assessment

1. Normalize the collected features.
2. Provide the features to the lightweight AI decision module.
3. Calculate the security-risk score.
4. classify the device into an appropriate risk state.
5. Determine whether the existing key can remain active.

Step 4: Hierarchical Key Generation

1. Generate a master security seed at the trusted gateway.
2. Derive group keys from the master seed.
3. Derive individual session keys for authorized devices.
4. Bind each key to device and session information.
5. Store only the required key material at constrained devices.

Step 5: Intelligent Key-Lifetime Determination

1. Calculate the adaptive key lifetime.
2. Reduce lifetime for high-risk devices.
3. Extend lifetime for stable and trusted devices within a security limit.
4. Trigger rekeying when the calculated lifetime expires.
5. Trigger immediate rekeying when abnormal behavior is detected.

Step 6: Secure Key Distribution

1. Encrypt the generated key using the existing trusted security context.
2. Transmit the encrypted key through the gateway.
3. Validate the receiving device.
4. Confirm successful key installation.
5. activate the key for the corresponding communication session.

Step 7: Continuous Monitoring and Rekeying

1. Continuously monitor device behavior.
2. Update the risk score periodically.
3. Detect deviations from normal communication behavior.
4. initiate selective rekeying instead of network-wide rekeying.
5. revoke keys associated with compromised or unauthorized devices.

Step 8: Key Revocation and Recovery

1. Identify a compromised or untrusted device.
2. Immediately invalidate its active key.
3. Remove the device from the corresponding security group.

4. Generate a replacement group/session key.
5. redistribute the updated key only to legitimate devices.

3.2 IOT DEVICE REGISTRATION AND SECURITY INITIALIZATION

In the first phase, the security context for each device is established. Upon joining the network, the device identity is validated by the gateway and its basic capability is collected. The capability comprises computation power, memory, battery status, mode of communications, function performed by the device, and security classification. The gateway doesn't apply the same security policy to all the devices since a battery-driven temperature sensor is different from a relatively strong industrial controller when it comes to computation power and function performed.

The device ID, together with the registration nonce and gateway-generated secret, forms the first step towards authentication. A lightweight hash-based approach may be used in order not to expose the registration secret in the process. The result is the registration record which is stored at the gateway side, whereas the device holds the least possible data for further authentication process.

Let the registration information of device d_i be

$$\mathbf{R}_i = [ID_i, Role_i, Cap_i, Mem_i, E_i^{res}, Comm_i, Sec_i, N_i] \quad (3)$$

where ID_i represents the device identifier, $Role_i$ denotes its operational role, Cap_i represents computational capability, Mem_i denotes available memory, E_i^{res} is residual energy, $Comm_i$ represents communication characteristics, Sec_i is the required security level, and N_i is a registration nonce.

The initial authentication value is generated as

$$Auth_i^{(0)} = H(ID_i \parallel N_i \parallel K_G \parallel Role_i \parallel Sec_i) \quad (4)$$

where $H(\cdot)$ represents a lightweight cryptographic hash function, K_G is a gateway security secret, and $\parallel$ denotes concatenation. The gateway stores the required registration parameters in protected storage. The device receives only the minimum authentication material required for secure communication.

3.3 DEVICE AND NETWORK FEATURE COLLECTION

Features necessary for an intelligent key management decision process are constantly gathered by ADAL-KM. Conventional static key management systems take into account only key age or the period of time for which a key/session is valid. On the other hand, the proposed solution takes into account both security and resource constraints. The main features involved are as follows.

The feature vector associated with device i at time t is defined as

$$\mathbf{X}_i(t) = [E_i(t), A_i(t), P_i(t), F_i(t), Q_i(t), K_i(t), B_i(t), T_i(t)] \quad (5)$$

where $E_i(t)$ denotes normalized residual energy, $A_i(t)$ represents authentication status, $P_i(t)$ denotes packet or communication intensity, $F_i(t)$ represents authentication or protocol failures, $Q_i(t)$ denotes communication irregularity, $K_i(t)$ represents key age, $B_i(t)$ indicates behavioral deviation, and $T_i(t)$ denotes historical trust.

Because these features have different numerical ranges, ADAL-KM normalizes them before the AI decision process. For a positive security feature, min-max normalization is applied as

$$x_{ij}^{norm}(t) = \frac{x_{ij}(t) - x_j^{min}}{x_j^{max} - x_j^{min} + \delta} \quad (6)$$

where $x_{ij}(t)$ represents feature j of device i , x_j^{min} and x_j^{max} represent the observed minimum and maximum values, and δ prevents division by zero.

For risk-increasing features such as authentication failures and behavioral deviation, the normalized value can be inverted so that a larger value consistently represents a higher risk contribution. This common representation simplifies the subsequent risk calculation.

3.4 AI-ASSISTED ADAPTIVE RISK ASSESSMENT

The core building block of ADAL-KM is the adaptive risk assessment block. It checks whether the existing key is suitable for the security status of the device at the current point of time. The light machine learning approach is applied at the gateway/edge level, not on the IoT device, to avoid excessive computing load on sensor nodes.

The risk assessment approach is a combination of the normalized features along with the learned weights. Therefore, the risk score is defined as

$$R_i(t) = \sigma \left[\beta_0 + \sum_{j=1}^m \beta_j x_{ij}^{norm}(t) + \sum_{j=1}^m \sum_{k=j+1}^m \gamma_{jk} x_{ij}^{norm}(t) x_{ik}^{norm}(t) \right] \quad (7)$$

where $R_i(t) \in [0,1]$ is the risk score, $\sigma(\cdot)$ represents the sigmoid activation function, β_0 is the bias term, β_j denotes the learned contribution of feature j , and γ_{jk} captures interactions between two features. The interaction terms allow the model to represent conditions in which individual parameters appear normal but their combination indicates elevated risk.

The adaptive trust score is subsequently computed as

$$T_i(t) = 1 - [\alpha_R R_i(t) + \alpha_F F_i(t) + \alpha_B B_i(t) + \alpha_K K_i(t)] \quad (8)$$

where $F_i(t)$ represents normalized authentication failure behavior, $B_i(t)$ denotes behavioral deviation, $K_i(t)$ represents normalized key age, and $\alpha_R, \alpha_F, \alpha_B, \alpha_K$ are weighting coefficients satisfying

$$\alpha_R + \alpha_F + \alpha_B + \alpha_K = 1.$$

A high trust score indicates stable behavior, whereas a low trust score indicates that the device requires closer security monitoring or earlier key replacement.

3.5 ADAPTIVE RISK-STATE CLASSIFICATION

After calculating the risk score, ADAL-KM assigns the device to one of several operational states. The states are stable, moderate-risk, high-risk, and critical. These states determine how aggressively the key-management mechanism responds to the device. The state transition function is represented as

$$S_i(t) = \begin{cases} S_1, & 0 \leq R_i(t) < \theta_1 \\ S_2, & \theta_1 \leq R_i(t) < \theta_2 \\ S_3, & \theta_2 \leq R_i(t) < \theta_3 \\ S_4, & \theta_3 \leq R_i(t) \leq 1 \end{cases} \quad (9)$$

where S_1, S_2, S_3 , and S_4 correspond to stable, moderate-risk, high-risk, and critical states, respectively, and $\theta_1, \theta_2, \theta_3$ are adaptive thresholds.

The resulting state determines the required key-management action:

$$A_i(t) = \begin{cases} \text{Hold}(K_i), & S_i(t) = S_1 \wedge \text{Age}_i < T_{\max} \\ \text{Refresh}(K_i), & S_i(t) = S_2 \wedge \text{Age}_i \geq T_{\text{mid}} \\ \text{Rekey}(K_i), & S_i(t) = S_3 \\ \text{Revoke}(K_i), & S_i(t) = S_4 \end{cases} \quad (10)$$

This mechanism prevents unnecessary network-wide rekeying. A stable device can continue using its current key within the permitted lifetime, whereas a device exhibiting suspicious behavior receives a new key without forcing unrelated devices to participate in the rekeying operation.

3.6 ADAPTIVE KEY-LIFETIME DETERMINATION

A major component of ADAL-KM is the adaptive determination of key lifetime. Conventional approaches generally use a predetermined value, such as a fixed session lifetime. Such an approach treats a stable device and a suspicious device equally. ADAL-KM instead modifies the key lifetime according to the calculated risk and resource condition. The adaptive key lifetime is calculated as:

$$L_i(t) = L_{\min} + (L_{\max} - L_{\min}) \left[\begin{matrix} \eta_T T_i(t) \\ + \eta_E E_i(t) \\ + \eta_C C_i(t) \end{matrix} \right] - \eta_R R_i(t) L_{\max} \quad (11)$$

where $L_{\min}$ and $L_{\max}$ represent the minimum and maximum permitted key lifetimes, $T_i(t)$ denotes device trust, $E_i(t)$ represents residual-energy status, $C_i(t)$ represents communication stability, $R_i(t)$ is the risk score, and $\eta_T, \eta_E, \eta_C, \eta_R$ are weighting factors.

To prevent excessively long key lifetimes, ADAL-KM constrains the calculated value as

$$L_i^*(t) = \max[L_{\min}, \min(L_{\max}, L_i(t))]. \quad (12)$$

Consequently, high-risk devices receive shorter lifetimes, while stable devices can retain their keys longer within the predefined security boundary. The mechanism also avoids unnecessary rekeying of highly constrained devices when their observed behavior remains trustworthy.

3.7 HIERARCHICAL KEY GENERATION

ADAL-KM uses hierarchical key generation to reduce the amount of key material maintained by individual IoT devices. The network is divided into logical security groups according to device role, location, application, or communication relationship. A gateway maintains the group-level security context and derives session keys when required.

The hierarchy can be represented as

$$\begin{aligned} K_G &\rightarrow \{K_{G_1}, K_{G_2}, \dots, K_{G_q}\} \\ &\rightarrow \{K_1, K_2, \dots, K_n\} \rightarrow \{K_{S_1}, K_{S_2}, \dots, K_{S_r}\} \end{aligned} \quad (13)$$

where K_G is the gateway master secret, K_{Gq} denotes a group key, K_i represents a device-specific key, and K_{Sr} denotes a session key.

The hierarchical structure limits the scope of a key compromise. A device-specific key is generated as

$$K_i = \text{HMAC}(K_{G_q}, ID_i \parallel \text{Role}_i \parallel N_i \parallel V_i) \quad (14)$$

where HMAC represents a keyed hash function and V_i is a key-version identifier. Including the version identifier ensures that an obsolete key cannot be reused after a successful rekeying operation.

3.8 SECURE SESSION-KEY GENERATION

Once two authorized devices need to communicate, ADAL-KM generates a session-specific key. The session key is derived from the group context and authenticated device information. This approach avoids transmitting the raw master or group secret across the network.

The session key is defined as

$$K_{ij}^{\text{sess}} = \text{HMAC}(K_{G_q}, ID_i \parallel ID_j \parallel N_{ij} \parallel V_q \parallel TS_{ij}) \quad (15)$$

where ID_i and ID_j identify the communicating devices, N_{ij} is a session nonce, V_q is the current group-key version, and TS_{ij} is a timestamp or freshness value.

The session-key validation condition is

$$\text{Valid}(K_{ij}^{\text{sess}}) = \begin{cases} 1, & H(K_{ij}^{\text{sess}} \parallel ID_i \parallel ID_j \parallel TS_{ij}) = \Omega_{ij} \\ 0, & \text{otherwise} \end{cases} \quad (16)$$

where Ω_{ij} represents the authenticated validation token. This prevents a device from accepting a session key generated for another device or an earlier session.

3.9 SECURE KEY DISTRIBUTION

After generating the required key, the gateway securely distributes it to authorized devices. The key is not transmitted in plaintext. Instead, ADAL-KM protects the session key using the receiving device's current security context.

For device i , the protected key package is expressed as

$$P_i = \text{Enc}_{K_i}[K_{ij}^{\text{sess}} \parallel ID_i \parallel ID_j \parallel V_{ij} \parallel TS_{ij} \parallel \text{MAC}_{ij}] \quad (17)$$

where Enc_{K_i} represents lightweight encryption using the existing device key, V_{ij} is the session-key version, TS_{ij} is the freshness value, and MAC_{ij} is an integrity authentication code.

The receiving device accepts the key only when the identity, version, timestamp, and integrity conditions are satisfied:

$$\begin{aligned} \text{Accept}(P_i) &= I(ID_i) \wedge I(V_{ij}) \wedge I(TS_{ij}) \\ &\wedge \text{Verify}(\text{MAC}_{ij}) \wedge T_i(t) \geq T_{\min}. \end{aligned} \quad (18)$$

This multi-condition validation prevents replayed, modified, or incorrectly addressed key packets from being installed.

3.10 INTELLIGENT REKEYING MECHANISM

Rekeying is initiated when the current key reaches its adaptive lifetime, when the risk score exceeds a threshold, or when a security event indicates possible key exposure. Instead of replacing keys throughout the entire IoT network, ADAL-KM performs selective rekeying. The rekeying trigger is defined as

$$\text{Trigger}_i(t) = [Age_i(t) \geq L_i^*(t)] \vee [R_i(t) \geq \theta_R] \vee [F_i(t) \geq \theta_F] \vee [B_i(t) \geq \theta_B]. \quad (19)$$

If any condition is satisfied, the gateway initiates the rekeying process for the affected device or security group. The new key version is generated as

$$K_i^{new} = \text{HMAC}(K_{G_q}^{new}, ID_i \boxplus V_i^{old} + 1 \boxplus N_i^{new} \boxplus R_i(t)). \quad (20)$$

The version increment ensures that the new key can be distinguished from all previous key states. A compromised or expired key therefore cannot be accepted after the corresponding version transition.

3.11 KEY REVOCATION AND COMPROMISE HANDLING

A critical device does not simply receive a shorter key lifetime. When the risk level indicates probable compromise, ADAL-KM immediately revokes its active key. The device is removed from the active security group and its authentication credentials are marked invalid. The revocation condition is represented as

$$\text{Rev}_i(t) = \begin{cases} 1, & R_i(t) \geq \theta_{crit} \\ 1, & \text{AuthFail}_i(t) \geq F_{crit} \\ 1, & \text{Anomaly}_i(t) \geq A_{crit} \\ 0, & \text{otherwise.} \end{cases} \quad (21)$$

After revocation, the affected group key is updated when necessary. The new group key is generated independently of the revoked device:

$$K_{G_q}^{new} = \text{HMAC}(K_G, G_q \boxplus V_q^{old} + 1 \boxplus N_q^{new} \boxplus M_q^{valid}) \quad (22)$$

where M_q^{valid} represents the set of currently authorized devices in group q . The revoked device is excluded from this membership set. Therefore, knowledge of the previous group key does not provide continued access to future group communication.

3.12 CONTINUOUS TRUST UPDATING

The trust value is not calculated only during device registration. ADAL-KM continuously updates trust according to observed behavior. This enables the system to respond to gradual behavioral changes rather than relying exclusively on a single security assessment. The updated trust value is calculated using an exponential temporal model:

$$T_i(t+1) = \rho T_i(t) + (1-\rho)[1 - R_i(t)] \quad (23)$$

where $0 < \rho < 1$ controls the contribution of historical trust. A higher ρ provides greater temporal stability, while a lower value enables faster adaptation to recent behavior. The change in trust is represented as

$$\Delta T_i(t) = T_i(t+1) - T_i(t) = (1-\rho)[1 - R_i(t) - T_i(t)]. \quad (24)$$

A persistent decrease in trust causes the key-management mechanism to progressively shorten the key lifetime. This prevents a sudden dependence on a single anomalous measurement while still allowing sustained abnormal behavior to influence security decisions.

3.13 ENERGY-AWARE KEY MANAGEMENT

Energy availability is particularly important in battery-powered IoT networks. Excessive authentication and rekeying operations can reduce network lifetime. ADAL-KM therefore incorporates residual energy into the key-management decision.

The energy state of device i is represented as

$$E_i^{norm}(t) = \frac{E_i^{res}(t)}{E_i^{in} + \delta} \quad (25)$$

where E_i^{res} is the current residual energy and E_i^{in} is the initial energy capacity.

The energy-aware rekeying cost can then be represented as

$$C_i^{key}(t) = N_i^{key}(t) [C_i^{comp} + C_i^{tx} + C_i^{rx}] [1 + \mu_R R_i(t)] \quad (26)$$

where $N_i^{key}(t)$ represents the number of key-management operations, C_i^{comp} denotes computation cost, C_i^{tx} and C_i^{rx} represent transmission and reception costs, and μ_R controls risk-related rekeying sensitivity.

This formulation allows the system to account for the resource cost associated with security operations. A highly trusted device with low residual energy should not be repeatedly rekeyed without a security requirement. Conversely, high-risk behavior can justify additional security operations even when the energy cost is higher.

3.14 ADAPTIVE THRESHOLD ADJUSTMENT

Fixed thresholds may not remain appropriate across different network conditions. ADAL-KM therefore adjusts the decision thresholds according to the distribution of recent device-risk values. The threshold mechanism prevents the system from generating excessive alerts when the entire network experiences a temporary change in communication behavior.

The adaptive threshold is represented as

$$\theta_R(t) = \text{clip}[\mu_R(t) + \kappa \sigma_R(t), \theta_{min}, \theta_{max}] \quad (27)$$

where $\mu_R(t)$ is the mean risk value, $\sigma_R(t)$ is the standard deviation of recent risk values, κ controls threshold sensitivity, and $\text{clip}(\cdot)$ restricts the threshold to the permitted interval.

The corresponding adaptive anomaly condition is

$$An_i(t) = \mathbb{I}[R_i(t) > \theta_R(t)] + \mathbb{I}[R_i(t) - R_i(t-1) > \delta_R], \quad (28)$$

where $\mathbb{I}[\cdot]$ is an indicator function and δ_R represents the maximum acceptable short-term risk variation. This formulation allows ADAL-KM to detect both persistent high-risk behavior and sudden changes in device behavior.

3.15 LIGHTWEIGHT CRYPTOGRAPHIC PROTECTION

The proposed method does not replace cryptographic protection with AI. AI determines when and how key-management actions should occur, whereas cryptographic primitives protect the actual information and key material. This separation is important because machine-learning predictions alone cannot provide confidentiality or integrity. The protected message is represented as

$$C_{ij}(t) = \text{Enc}_{K_{ij}^{\text{sess}}} [M_{ij}(t) \parallel TS_{ij} \parallel \text{Seq}_{ij}] \quad (29)$$

where $M_{ij}(t)$ is the application message, TS_{ij} is a freshness value, and Seq_{ij} represents a sequence number.

The corresponding authenticated communication condition is

$$\text{Auth}_{ij}(t) = \text{Verify}_{K_{ij}^{\text{sess}}} [C_{ij}(t), ID_i, ID_j, TS_{ij}, \text{Seq}_{ij}]. \quad (30)$$

Only messages satisfying the authentication and freshness requirements are accepted. Thus, ADAL-KM integrates adaptive intelligence with conventional cryptographic protection rather than treating AI as a substitute for cryptographic primitives.

The complete state transition can be expressed as

$$S_i(t+1) = \Phi [S_i(t), \mathbf{X}_i(t), R_i(t), T_i(t), L_i^*(t), A_i(t)] \quad (31)$$

where $S_i(t)$ represents the current security state, $X_i(t)$ denotes the observed feature vector, $R_i(t)$ is the risk score, $T_i(t)$ is the trust score, $L_i^*(t)$ is the adaptive key lifetime, and $A_i(t)$ represents the selected key-management action.

The overall security-resource objective of the proposed method is finally represented as

$$J_{\text{ADAL-KM}} = \omega_1(1 - S_{\text{sec}}) + \omega_2 C_{\text{comm}} + \omega_3 C_{\text{comp}} + \omega_4 E_{\text{key}} + \omega_5 L_{\text{rekey}} + \omega_6 C_{\text{rev}} \quad (32)$$

$$\text{s.t. } \omega_1 + \omega_2 + \omega_3 + \omega_4 + \omega_5 + \omega_6 = 1, S_{\text{sec}} \geq S_{\text{target}}, 0 \leq R_i, T_i \leq 1$$

where, S_{sec} denotes the overall security level, C_{comm} represents communication overhead, C_{comp} represents computational cost, E_{key} denotes key-management energy consumption, L_{rekey} represents rekeying latency, and C_{rev} represents revocation cost. The objective is minimized while maintaining the required security constraint.

4. RESULTS AND DISCUSSION

4.1 EXPERIMENTAL SETTINGS

ADAL-KM technique is assessed by conducting simulation using the CICIOT2023 dataset. This experiment simulates heterogeneous IoT devices interconnected via gateway devices in order to assess the performance of adaptive key generation, distribution, renewal, and revocation in large-scale networks. The simulation is implemented using Python version 3.11 and libraries of NumPy, Pandas, Scikit-learn, and Cryptography. Simulation experiments are conducted using a workstation that has an Intel Core i7 processor, 16GB RAM, and 512GB SSD storage. The CICIOT2023 dataset includes traffic data of 105 IoT devices and 33 attack types grouped in seven different attack categories.

Table.1. Experimental setup and simulation parameters

Parameter	Value
Simulation environment	Python 3.11
Operating system	Windows 11, 64-bit
Processor	Intel Core i7, 12th Generation
RAM	16 GB
Storage	512 GB SSD
Number of IoT devices	10–40

Gateway nodes	2
Communication model	Gateway-assisted IoT
Cryptographic primitive	Lightweight hash/HMAC-based key derivation
Initial key length	128 bits
Session-key length	128 bits
Minimum key lifetime	20 s
Maximum key lifetime	120 s
AI decision interval	5 s
Risk-score range	0–1
Trust-score range	0–1
Network observation interval	5 s
Simulation duration	300 s
Network traffic rate	50–250 packets/s
Packet size	64–512 bytes
Number of experimental runs	10
Reported result	Mean value

As shown in Table 1, the experimental configuration maintains identical hardware, cryptographic parameters, simulation duration, and traffic conditions for all compared methods. Only the key-management strategy is changed during comparative evaluation.

4.2 DATASET DESCRIPTION

The CICIOT2023 dataset is employed for representing heterogeneous behavior in IoT communication and attacks. This dataset was generated by using an IoT topology with 105 IoT devices and 33 types of attacks which include DDoS, DoS, reconnaissance, web attacks, brute force attacks, spoofing, and Mirai attacks. This dataset offers the following packet-flow characteristics including the duration, protocols involved, packet rate, source and destination rates, TCP flags, statistics on packet size, inter-arrival time and flow-level statistical characteristics.

Table.2. CICIOT2023 dataset description

Dataset characteristic	Description
Dataset	CICIOT2023
IoT topology	105 IoT devices
Attack types	33
Attack categories	DDoS, DoS, Recon, Web-based, Brute Force, Spoofing, Mirai
Data formats	PCAP and CSV
Primary traffic	Benign and malicious IoT traffic
Major features	Flow duration, protocol, packet rate, source rate, destination rate
Additional features	TCP flags, packet size, inter-arrival time, flow statistics
Simulation subset	500,000 flow records
Training portion	70%
Validation portion	15%

Testing portion	15%
Feature normalization	Min-max normalization
Primary application	IoT security and adaptive key-management evaluation

The dataset's original topology and attack diversity make it suitable for evaluating the behavior-sensitive component of ADAL-KM. The original CICIOT2023 publication reports 33 attacks across seven categories and describes attacks performed by malicious IoT devices against other IoT devices.

4.3 PERFORMANCE METRICS

The following metrics are considered to evaluate the proposed approach.

- Overhead of key-management communication (KB): This metric is used to measure the number of communications established during key generation, distribution, renewal, and revocation. The lower this value, the less number of control messages and key-management packets will be sent.
- Computational overhead (ms): This metric is used to calculate the average time of key-management operations. It includes the operations of risk assessment, key generation, key validation, and re-keying. Lower value means better resource consumption.
- Energy consumption (J): This metric is used to measure the total energy consumption of IoT nodes during the key-management operations. It includes transmission, receipt, computation, and establishment of new keys. Lower energy consumption means better use of available resources.
- Secure session establishment rate (%): This metric is used to calculate the percentage of established sessions which pass the authentication and secure key establishment. Higher this value, the higher is the rate of reliable communication.
- Rekeying delay (ms): This metric is used to calculate the time of transition from detecting the need to renew the key to its successful installation.

4.4 RESULTS BASED ON NUMBER OF IOT DEVICES AND KEY-MANAGEMENT COMMUNICATION OVERHEAD

Table.3. Key-management communication overhead versus number of IoT devices

IoT Devices	Efficient Key Management System (KB)	Decentralized Blockchain-Based KMP (KB)	Lightweight PUF-Based Protocol (KB)	ADAL-KM (KB)
10	18.42	22.15	16.87	14.31
15	27.36	31.84	25.41	21.74
20	36.91	42.72	34.18	29.05
25	46.38	53.61	42.97	36.44
30	55.73	64.52	51.84	43.62
35	65.21	75.46	60.73	50.87
40	74.86	86.93	69.42	57.91

The Table.3 shows that communication overhead increases with the number of IoT devices for all methods. At 10 devices, the proposed ADAL-KM generates 14.31 KB of key-management traffic, compared with 18.42 KB for the Efficient Key Management System, 22.15 KB for the Decentralized Blockchain-Based Protocol, and 16.87 KB for the Lightweight PUF-Based Protocol. At 40 devices, ADAL-KM reaches 57.91 KB, whereas the corresponding values are 74.86, 86.93, and 69.42 KB. Thus, the proposed method maintains a lower communication requirement throughout the evaluated network range. The reduction becomes particularly relevant as network size increases because ADAL-KM performs selective rather than network-wide rekeying. The adaptive key-lifetime mechanism prevents stable devices from participating in unnecessary renewal operations. At 40 devices, the proposed method reduces overhead by approximately 22.6% relative to the lightweight method, 16.6% relative to the PUF-based method, and 33.4% relative to the blockchain-based method. The results demonstrate that hierarchical key distribution and selective rekeying effectively limit key-management traffic.

4.5 RESULTS BASED ON NUMBER OF IOT DEVICES AND COMPUTATIONAL COST

Table 4. Computational cost versus number of IoT devices

IoT Devices	Efficient Key Management System (ms)	Decentralized Blockchain-Based Key Management Protocol (ms)	Lightweight PUF-Based Protocol (ms)	ADAL-KM (ms)
10	42.8	55.6	38.7	34.2
15	61.4	78.9	55.8	48.6
20	80.7	102.5	73.2	63.1
25	100.2	126.8	90.6	77.9
30	119.6	151.4	108.1	92.4
35	139.1	176.2	125.7	106.8
40	158.7	201.6	143.5	121.1

The computational results in Table.4 indicate that ADAL-KM maintains lower processing requirements as the network expands. At 10 devices, the proposed method requires 34.2 ms, compared with 42.8 ms for the Efficient Key Management System, 55.6 ms for the blockchain-based protocol, and 38.7 ms for the PUF-based approach. At 40 devices, the corresponding computational costs increase to 121.1, 158.7, 201.6, and 143.5 ms, respectively. The proposed approach therefore reduces computational cost by 23.7% compared with the lightweight method, 15.6% compared with the PUF-based method, and 39.9% compared with the blockchain-based method at 40 devices. The lower cost results from performing the adaptive decision process at the gateway rather than repeatedly executing complex key-management operations at constrained devices. The blockchain method exhibits the highest growth because distributed ledger operations introduce additional processing. The PUF-based method remains comparatively efficient but requires additional operations associated with hardware-rooted group-key management. ADAL-KM maintains lower computational growth by combining hierarchical key derivation with selective rekeying.

4.6 RESULTS BASED ON NUMBER OF IOT DEVICES AND ENERGY CONSUMPTION

Table.5. Energy consumption versus number of IoT devices

IoT Devices	Efficient Key Management System (J)	Decentralized Blockchain-Based Key Management Protocol (J)	Lightweight PUF-Based Protocol (J)	ADAL-KM (J)
10	3.82	4.41	3.54	3.21
15	5.71	6.63	5.29	4.78
20	7.64	8.84	7.08	6.34
25	9.57	11.08	8.89	7.91
30	11.52	13.35	10.71	9.48
35	13.46	15.62	12.52	11.03
40	15.43	17.89	14.36	12.09

Energy results given in Table.5 reveal an obvious trend of growing energy consumption for key management as the number of devices increases. With 10 devices, ADAL-KM uses 3.21 J, while the Energy-efficient Key Management System, the blockchain-based protocol, and the PUF-based protocol use 3.82 J, 4.41 J, and 3.54 J, respectively. For 40 devices, ADAL-KM uses 12.09 J in contrast to 15.43 J, 17.89 J, and 14.36 J used by the three methods mentioned above. Therefore, the proposed method allows to achieve a decrease in energy consumption of about 21.7% compared to the Energy-efficient Key Management System for 40 devices. For the blockchain-based protocol, the decrease is 32.4%, while for the PUF-based protocol, 18.8% of extra energy consumption is needed. The decrease is possible due to reduced unnecessary key exchange and longer adaptive period when valid keys can be stored on stable devices. The blockchain protocol uses the largest amount of energy because of distributed computation and communication in key management.

4.7 RESULTS BASED ON NUMBER OF IOT DEVICES AND SECURE-SESSION ESTABLISHMENT SUCCESS

Table.6. Secure-session establishment success versus number of IoT devices

IoT Devices	Efficient Key Management System (%)	Decentralized Blockchain-Based Key Management Protocol (%)	Lightweight PUF-Based Protocol (%)	ADAL-KM (%)
10	96.1	97.0	97.2	98.1
15	95.8	96.8	97.0	98.3
20	95.5	96.5	96.8	98.5
25	95.1	96.2	96.6	98.7
30	94.8	95.9	96.3	98.8
35	94.5	95.6	96.1	98.9
40	94.2	95.2	95.8	98.9

According to the secure-session experiment outcomes presented in Table.6, the ADAL-KM protocol guarantees high

reliability of the session establishment process even as the network scale gets bigger. For example, the proposed protocol guarantees 98.1% when 10 devices operate in the network, which is higher than the respective rates for all three existing solutions, including 96.1%, 97.0%, and 97.2%. The proposed protocol allows to get 98.9% when 40 devices operate in the network, which is higher than the corresponding values for three existing solutions, which include 94.2%, 95.2%, and 95.8%, respectively. The advantage is much clearer when the number of devices increases. While in case of transition from 10 to 40 devices, the proposed solution improves its rate from 98.1% to 98.9%, all three existing solutions reduce their values by 1.9, 1.8, and 1.4 percentage points, respectively. This result is explained by an ability of the adaptive trust mechanism to identify abnormal conditions before multiple authentication failures take place and by selective rekeying that protects valid devices from irrelevant security events.

4.8 RESULTS BASED ON NUMBER OF IOT DEVICES AND REKEYING LATENCY

Table.7. Rekeying latency versus number of IoT devices

IoT Devices	Efficient Key Management System (ms)	Decentralized Blockchain-Based Key Management Protocol (ms)	Lightweight PUF-Based Protocol (ms)	ADAL-KM (ms)
10	31.8	39.7	29.4	24.2
15	42.5	53.4	39.1	32.1
20	53.4	67.2	48.8	40.0
25	64.1	81.3	58.7	47.9
30	74.9	95.6	68.5	55.8
35	85.7	109.8	78.4	63.7
40	96.5	124.1	88.3	71.0

The rekeying latency from Table.7 confirms the scalability of the proposed selective renewal scheme. At 10 devices, ADAL-KM takes 24.2 ms, whereas the efficient key management system, blockchain-based scheme, and PUF-based scheme take 31.8, 39.7, and 29.4 ms, respectively. At 40 devices, ADAL-KM needs 71.0 ms, as compared to 96.5 ms by the lightweight scheme, 124.1 ms by the blockchain-based scheme, and 88.3 ms by the PUF-based scheme. In that way, the proposed scheme cuts down the rekeying latency by 26.4% in comparison to the efficient key management system with 40 devices. This decrease is related to the selective renewal of keys and key derivation with the help of gateways. The latency is highest in case of the blockchain-based scheme due to the complexity of distributed key management procedures. The latency in case of the PUF-based scheme is lower than in case of the blockchain-based scheme but is still higher than in case of ADAL-KM due to the necessity of performing dynamic group-key operations.

5. CONCLUSION

The ADAL-KM technique presents an adaptive scheme for securing heterogeneous IoT networks taking into account the constraints in terms of limited resources. The evaluation is

conducted using the characteristics of CICIoT2023 traffic in a simulation environment supported by a gateway in order to analyze communication overhead, computation cost, energy consumption, secure session establishment probability, and rekeying latency. The analysis shows that the improvements are consistent regardless of the tested network sizes ranging from 10 to 40 devices. For example, at 40 devices, the ADAL-KM ensures that key management communication overhead does not exceed 57.91 KB, which is less than 74.86 KB required by the Efficient Key Management System. The computation cost is brought down to 121.1 ms whereas the existing lightweight scheme needs 158.7 ms. Furthermore, energy consumption is lowered to 12.09 J, which is lower than 15.43 J. The secure session establishment probability at the same network size is increased up to 98.9% which is higher than 94.2%, 95.2%, and 95.8%.

REFERENCES

- [1] M. Rana, Q. Mamun and R. Islam, "Lightweight Cryptography in IoT Networks: A Survey", *Future Generation Computer Systems*, Vol. 129, pp. 77-89, 2022.
- [2] T.C. Priyadharshini and D.M. Geetha, "Efficient Key Management System based Lightweight Devices in IOT", *Intelligent Automation and Soft Computing*, Vol. 31, No. 3, pp. 1793-1808, 2022.
- [3] P.M. Rao and B.D. Deebak, "A Comprehensive Survey on Authentication and Secure Key Management in Internet of Things: Challenges, Countermeasures and Future Directions", *Ad Hoc Networks*, Vol. 146, pp. 1-11, 2023.
- [4] Z. Najafi and S. Babaie, "A Lightweight Hierarchical Key Management Approach for Internet of Things", *Journal of Information Security and Applications*, Vol. 75, pp. 1-13, 2023.
- [5] B. Gobinathan, M.A. Mukunthan and S. Surendran, "A Novel Method to Solve Real Time Security Issues in Software Industry using Advanced Cryptographic Techniques", *Scientific Programming*, Vol. 2021, No. 1, pp. 3611182-3611193, 2021.
- [6] I. Cetintav and M.T. Sandikkaya, "A Lightweight Authentication and Management Method for Internet of Things", *Internet of Things*, Vol. 23, pp. 1-13, 2023.
- [7] S. Bettayeb, M.L. Messai and S.M. Hemam, "A Robust and Efficient Vector-based Key Management Scheme for IoT Networks", *Ad Hoc Networks*, Vol. 149, pp. 1-11, 2023.
- [8] M. Rana, Q. Mamun and R. Islam, "Enhancing IoT Security: An Innovative Key Management System for lightweight Block Ciphers", *Sensors*, Vol. 23, No. 18, pp. 1-9, 2021.
- [9] M.A. Kandi, D.E. Kouicem, M. Doudou, H. Lakhlef, A. Bouabdallah and Y. Challal, "A Decentralized Blockchain-based Key Management Protocol for Heterogeneous and Dynamic IoT Devices", *Computer Communications*, Vol. 191, pp. 11-25, 2022.
- [10] A. Attkan and V. Ranga, "Cyber-Physical Security for IoT Networks: A Comprehensive Review on Traditional, Blockchain and Artificial Intelligence based Key-Security", *Complex and Intelligent Systems*, Vol. 8, No. 4, pp. 3559-3591, 2022.
- [11] S.S. Panda, D. Jena, B.K. Mohanta, S. Ramasubbareddy, M. Daneshmand and A.H. Gandomi, "Authentication and Key Management in Distributed IoT using Blockchain Technology", *IEEE Internet of Things Journal*, Vol. 8, No. 16, pp. 12947-12954, 2021.
- [12] C. Chen, H. Guo, Y. Wu, B. Shen, M. Ding and J. Liu, "A Lightweight Authentication and Key Agreement Protocol for IoT-Enabled Smart Grid System", *Sensors*, Vol. 23, No. 8, pp. 1-9, 2023.
- [13] M. Barbareschi, V. Casola, A. Emmanuele and D. Lombardi, "A Lightweight PUF-based Protocol for Dynamic and Secure Group Key Management in IoT", *IEEE Internet of Things Journal*, Vol. 11, No. 20, pp. 32969-32984, 2024.
- [14] M. El-Hajj and P. Beune, "Lightweight Public Key Infrastructure for the Internet of Things: A Systematic Literature Review", *Journal of Industrial Information Integration*, Vol. 41, pp. 1-7, 2021.
- [15] M. Nafi and P. Poizat, "KMS-MQTT: Robust, Efficient and Lightweight Key Management Scheme over MQTT v5.0 using Cryptography and Network Steganography", *Computer Networks*, Vol. 7, pp. 1-17, 2026.