ABA-LEP: Autonomous Bidirectional Authentication and Lightweight Encryption Protocol for Drones under ARM Architecture

Qian Zhou[a,*], Jiayang Wu[c], Weizhi Meng[b]

[a]*School of Computer Science,Nanjing University of Posts and Telecommunications, 9 Wenyuan Road Nanjing 210023 Jiangsu China*
[b]*School of Ccomputing and Communications, Lancaster University Lancaster LA1 4YR United Kingdom*
[c]*School of Modern Posts, Nanjing University of Posts and Telecommunications, Nanjing 210003 Jiangsu China*

Abstract

Secure communication protocols for drones are crucial in ensuring safety in potentially threatening network environments. However, existing protocols often suffer from weak autonomy, lack of optimization for ARM architecture, and inefficient utilization of lightweight cryptographic algorithms. To address these issues, this paper designs and analyzes an Autonomous Bidirectional Authentication and Lightweight Encryption Protocol (ABA-LEP) for drones under ARM architecture. The protocol optimizes the fixed-point scalar multiplication in SM2 for ARM architecture to accelerate authentication and key agreement efficiency, and employs simple operations like one-time pad limited XOR for lightweight secure communication encryption. Experiments conducted on the ARM Cortex M-4 based CrazyFlie 2.1 drone demonstrate that, in resource-constrained environments, the ABA-LEP achieves a performance improvement of up to 80.18% in fixed-point scalar multiplication with a 256-bit operand, compared to existing techniques. Additionally, the number of transmitted messages per unit time increases by up to 97.02%. The protocol's resilience against multiple types of attacks has also been verified using the formal verification tool ProVerif.

[*] Corresponding author
Email addresses: zhouqian@njupt.edu.cn (Qian Zhou), 1223096926@njupt.edu.cn (Jiayang Wu), weizhi.meng@ieee.org (Weizhi Meng)

Keywords: Security Protocols,ARM Cortex M-4 Architecture,Elliptic Curves Cryptography,Lightweight Cryptography

1. Introduction

Drones, also known as Unmanned Aerial Vehicles (UAVs), are unmanned aircraft controlled by remote systems and other control mechanisms [1]. In recent years, their extensive applications in package delivery, reconnaissance, environmental monitoring, and disaster response have made them a crucial component of Internet of Things (IoT) networks [2]. However, with the continuous expansion of UAV application scenarios, the security threats associated with transmitting sensitive data over networks are also increasing [3]. Attackers can interfere with UAV operations through eavesdropping, tampering, or identity impersonation, leading to information leakage and mission failure [4]. Therefore, ensuring communication security for UAVs in high-threat network environments has become a critical research issue [5].

Currently, UAV communication protocols face a series of challenges. Firstly, the encryption and security authentication technologies and standards used in mainstream UAV communication protocols are not unified, making efficient collaboration in heterogeneous environments difficult and lacking in autonomous control [6]. Secondly, according to RFC 7228 [7], most micro UAVs fall under the category of Periodic Energy-limited nodes, and thus, applying Lightweight Cryptography on UAVs can enhance response speed, reduce energy consumption, and extend endurance [8, 9]. However, the application of lightweight cryptography in existing communication protocols is not widespread, primarily due to the difficulty in balancing low latency with high security requirements [10, 11].

To address the issue of weak autonomous control, recent research has introduced the domestic SM2 standard for secure UAV communication. The SM2 elliptic curve public-key cryptography, designed and developed autonomously by the country, offers high autonomous control over parameters and enables secure and efficient key sharing [12]. However, SM2 involves complex elliptic curve operations that are computationally intensive, especially when executed on resource-constrained UAVs. This challenge becomes even more significant on lightweight processors such as those based on the ARM architecture, which, despite their energy efficiency and widespread use in embedded UAV systems, lack native support for large-integer multiplications required by SM2 operations. While previous work has discussed applying SM2 in embedded systems, little attention has been paid to fine-grained computational optimization of SM2 on ARM platforms, where hardware upgrades are commonly favored over low-level instruction optimization [13, 14].

Furthermore, the One-Time Pad (OTP) can achieve Perfect Secrecy and, due to its use of a limited number of simple XOR operations, qualifies as lightweight cryptography [15], making it highly suitable for constrained environments like UAVs [16, 17]. However, as UAVs are constrained nodes, securely and efficiently sharing random keys of lengths equal to communication protocol messages is challenging [6, 18], which compromises the confidentiality of UAV communications [19, 20].

Therefore, we proposed an Autonomous Bidirectional Authentication and Lightweight Encryption Protocol for Drones under ARM Architecture (ABA-LEP), and the contributions are as follows:

- An Autonomous Bidirectional Authentication Method for Drones under ARM Architecture (ABA) was designed, enabling rapid and autonomous bidirectional identity authentication between ground stations and UAVs, as well as efficient and secure lightweight key agreement. To ensure the autonomy of the key agreement process, this paper utilizes the SM2 key agreement algorithm and addresses its computational bottleneck by optimizing elliptic curve multiplication at the ARM assembly level. This novel optimization improves the practicality of deploying SM2 in lightweight UAVs.

- Developed an One-Time Pad based Lightweight Encryption Protocol (LEP). This protocol derives short keys obtained through negotiation using simple operations such as limited displacements and XOR, enabling encryption of long messages while ensuring perfect secrecy, forward and backward security, and reduced time and energy consumption.

- Conducted experiments on ABA-LEP using the CrazyFlie 2.1 UAV with the ARM Cortex M-4 architecture. Additionally, used the formal verification tool ProVerif to analyze the protocol's security. Experimental results indicate that the proposed communication protocol has low energy and time overhead on resource-constrained lightweight UAVs, while analysis results show that the protocol can withstand various types of attacks such as identity impersonation and eavesdropping, ensuring the security of communication data.

2. Related Work

2.1. SM2 Algorithm

The SM2 algorithm, introduced by China's National Cryptography Administration in 2010, is based on Elliptic Curve Cryptography (ECC) [12]. In 2004, Gura et al. [21] showed that ECC offers better security and higher computational efficiency than RSA at the same key length, making ECC a focus for lightweight encryption research.

Compared to traditional asymmetric encryption schemes, SM2 allows for autonomous control over parameter selection. In recent years, many studies have optimized the SM2 algorithm by improving chip circuits and reducing multiplication operations. In 2014, Gueron et al. [22] proposed an optimized Montgomery modular multiplication based on modulus characteristics, using a 7-bit window width for fixed-point scalar multiplication and a 5-bit window for non-fixed-point scalar multiplication. In 2015, Adalier et al. [23] optimized scalar computation using techniques like fixed-base NAF windows, achieving high-performance, side-channel resistant elliptic curve cryptography on low-cost commercial hardware for the x86-64 platform. In 2019, Mai et al. [24] redesigned the SM2 digital signature scheme by leveraging extended arithmetic instructions and large cache capacity. In 2023, Zhang et al. [13] implemented a fast modular inverse algorithm based on Fermat's Little Theorem and improved SM2 performance on ARM architecture by using point addition instead of modular multiplication.

Only reference [13] has optimized SM2 at the assembly level for the ARM architecture. With the rapid development of IoT, energy consumption has become a critical factor for IoT terminal processors, making the ARM architecture the mainstream choice for UAV processors.

2.2. One-Time Pad Encryption Method

The One-Time Pad (OTP) is a symmetric encryption algorithm that, according to Shannon's theorem, offers unconditional security by XORing plaintext with a random key of the same length [25]. However, its practical use faces challenges in generating, distributing, and keeping keys secret. The key must be as long as the plaintext, truly random, and never reused.

In 2019, Manucom et al. [26] combined hardware-based true random number generators with the Fisher-Yates shuffle algorithm to enhance key randomness, improving confidentiality. However, they did not address secure key distribution, and the time cost of key generation was higher than using pseudo-random number generators. In 2023, Yang et al. [6] proposed an OTP scheme based on three-dimensional parity, which rearranged messages into a matrix and used a parity bit matrix to assist key XOR operations, allowing shorter keys to encrypt longer messages without sacrificing security. Despite this, it did not solve the problem of generating and securely sharing random keys. Also in 2023, Jiang [27] introduced an OTP algorithm using finite field key exchange based on a Mersenne prime field. This allowed parties to update keys after each communication round, achieving perfect secrecy without pre-shared keys. However, the scheme lacked an authentication mechanism, leaving it vulnerable to impersonation attacks.

While OTP offers lightweight encryption with perfect secrecy, practical issues like random key generation, secure distribution, and authentication remain challenges. Despite this, OTP could improve the security of UAV communications by reducing resource consumption, warranting further investigation into lightweight OTP-based encryption schemes for UAVs.

2.3. UAV Secure Communication Protocols

UAV secure communication protocols refer to a set of rules and mechanisms designed to ensure the secure and reliable transmission of data between UAVs and ground stations [28]. These protocols use encryption, authentication, and integrity verification techniques to protect the communication process from unauthorized access, data tampering, and information leakage, thereby ensuring the safe operation of UAVs and the execution of their missions. Given the unique characteristics of UAV scenarios, recent research has largely focused on lightweight and secure identity authentication and key exchange protocols under the classic Dolev-Yao threat model for wireless networks [29].

In 2019, Li et al. [30] proposed a UAV encryption scheme based on an improved SM4 algorithm, enhancing encryption and decryption speeds and increasing tolerance to packet loss. However, it did not address the need for bidirectional authentication between UAVs and ground stations, leaving it vulnerable to identity impersonation attacks. In 2021, Khan et al. [31] introduced a lightweight identity authentication and key agreement method for FANETs, utilizing Hyperelliptic Curve Cryptography (HECC). Their optimization of elliptic curve multiplication improved performance, though the benefits were only significant with shorter key lengths. That same year, Guo et al. [32] proposed a lightweight UAV network authentication scheme based on SM2 and SM9 algorithms, reducing the need for computationally expensive elliptic curve operations like point multiplication by pre-sharing secret information. The use of SM9 identity-based cryptography improved the efficiency of the authentication key agreement, but the scheme required one party (e.g., the ground node) to have substantial computational power.

In 2022, Jian et al. [33] introduced two schemes based on elliptic curves: DroneSec for high-performance platforms and DroneSec-lite for low-performance platforms, which relied on symmetric cryptography. Both schemes offered some degree of forward security but could only negotiate a single long-term symmetric key, limiting real-time key updates. This restriction compromised the perfect security required for one-time pad encryption, thus reducing their overall security.

Recent research on lightweight UAV secure communication protocols has largely focused on incorporating elliptic curves to reduce the computational overhead associated with traditional asymmetric algorithms like RSA. However, only a few schemes, such as those in [30, 32], have introduced domestic cryptographic algorithms to achieve autonomous control over parameter selection. Furthermore, no existing schemes can meet the high frequency of key negotiations required for one-time pad encryption, which is necessary to achieve truly lightweight forward and backward security.

3. UAV Secure Communication Protocol Design

3.1. Threat Model and Attack Assumptions

In the UAV communication network model, two main entities are involved: UAV nodes that are constrained in both computing resources and energy and are vulnerable to capture; ground station nodes that have relatively abundant computing resources, storage, and stable energy.

This study assumes that communication occurs only between ground stations and UAVs. In such a topology, the wireless communication link between the ground station and the UAV is open and vulnerable. The capabilities of attackers are modeled based on the classic Dolev-Yao threat model for wireless networks [29], assuming that curious and malicious attackers possess the following abilities:

- The attacker can capture any signal transmitted between the UAV and the ground station via the open wireless communication link.

- The attacker has storage capabilities and can store captured information as well as their own computed data.

- The attacker can establish new wireless communication links with either the ground station or the UAV.

- When the UAV is still on the ground, it is securely connected to the ground station via a wired link, and the attacker cannot launch any attacks during this time.

Based on these assumptions, there are four types of attacks on UAV. Identity Impersonation Attack: The attacker can capture UAV signals, infiltrate the ground station, or analyze captured signals to obtain sensitive information and forge identity authentication data to impersonate a legitimate UAV or ground station. Eavesdropping Attack: The attacker can capture and store signals, attempting to decipher keys to access the plaintext of current communication data. If the security protocol does not meet the requirement for forward secrecy, the attacker could even decrypt previously stored communication data. This type of attack is difficult for both the ground station and UAV to detect. Replay Attack: The attacker can capture signals using various methods and replay them to the legitimate ground station or UAV nodes, thereby impersonating a legitimate communication participant. Man-in-the-Middle Attack: The attacker can hijack the session between the UAV and the ground station, simultaneously impersonating both sides of the communication. By inserting themselves as an invisible third party in the session, the attacker can alter the communication without either party realizing it.

3.2. Security Goals and Protocol Overview

Based on the threat model and attack assumptions outlined in Section 3.1, the UAV secure communication protocol designed in this study must meet the following security goals:

- The two communicating parties must negotiate a consistent key within a secure time frame. This key can only be used by the legitimate communication parties, and a different key must be used for each communication session.

Table 1: Main Symbols Involved in ABA-LEP

Symbols	Meaning	Symbols	Meaning
g	Ground Station	G	Base Point
u	UAV	$OTP()$	One-Time Pad
id_x	Public ID of x	SKM	Short Key Matrix
Z_{task}	Task Summary	DV	Diffusion Vector
s_x	Private ID of x	M	Message Matrix
R_x	Public Key of x	M'	Diffused M
r_x	Private Key of x	CM	Cipher M

- Even if an attacker cracks the key for a single communication session, the attacker can at most obtain the content of that session, but cannot decrypt any past communication (forward secrecy) or future communication (backward secrecy).

The UAV secure communication protocol includes an autonomous and controllable two-way identity authentication and key agreement method based on the SM2 algorithm (Chinese National Standard). The primary goal is to ensure that the communication parties are a trusted UAV and ground station and to negotiate a one-time short key that can be updated in real-time. Additionally, it includes a lightweight encryption method based on the one-time pad for secure UAV communication. The main purpose is to encrypt longer messages using the short key negotiated at a relatively low cost.

Main symbols involved in ABA-LEP are shown in Table 1.

The overall of ABA-LEP is shown in Figure 1.

3.3. Autonomous Bidirectional Authentication Method for Drones under ARM Architecture (ABA)

Due to the task-driven nature of UAV flights, where each mission has a limited duration, the steps for two-way identity authentication and key agreement can be compressed. To achieve autonomous control of the key agreement process and meet the real-time low-latency requirements of UAV flight operations, this protocol employs the Chinese national SM2 elliptic curve algorithm. Considering the characteristics of ARM architecture, the scalar multiplication steps in the SM2 key exchange are optimized by replacing point doubling operations with point addition operations, achieving

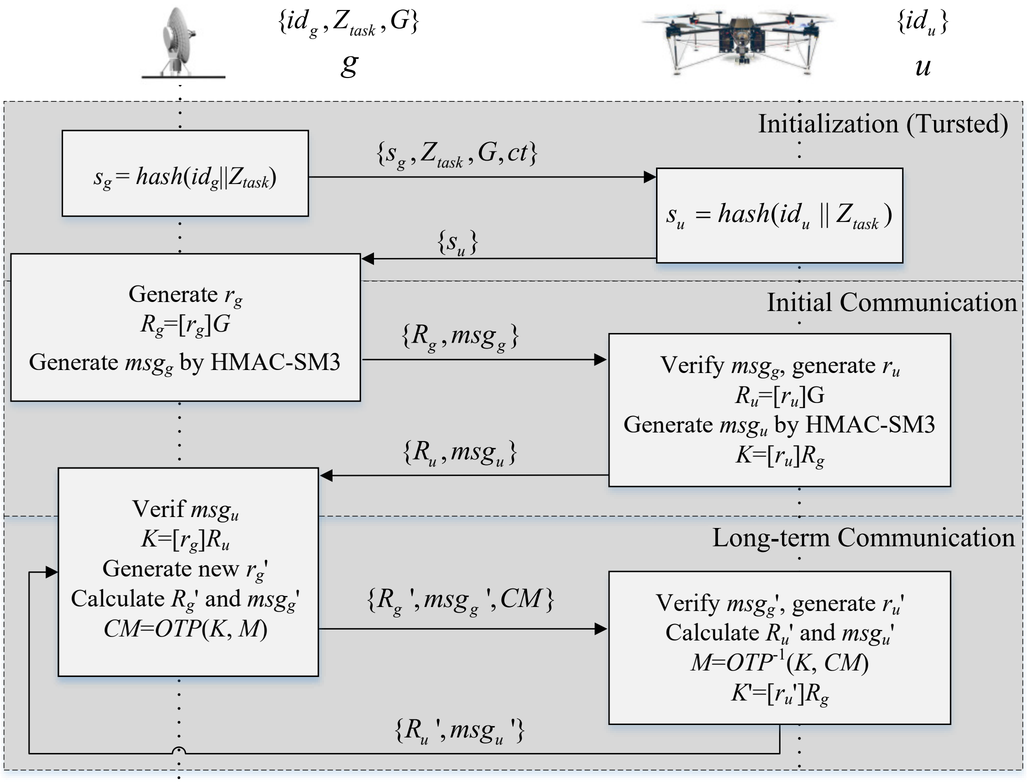

Figure 1: The Overall Protocol Flow

lightweight two-way identity authentication and key agreement with the same level of security. ABA method is divided into two stages: parameter initialization and key agreement.

Stage One: Parameter Initialization. The parameter initialization phase occurs after the task has been defined and just before the UAV is set to take off from the ground station. During this phase, the UAV and the ground station can securely communicate and exchange information via a wired link, making it immune to attacks. Both the ground station and the UAV are assigned unique public identifiers, id_u and id_g, at the time of manufacture. Based on the task summary information Z_{task}, the ground station generates its secret parameter s_g and the UAV's secret parameter s_u:

$$\begin{cases} s_g = hash(id_g||Z_{task}) \\ s_u = hash(id_u||Z_{task}) \end{cases} \tag{1}$$

To ensure the autonomy of the hashing process, the secure hash function $hash()$ used for generating secret identifiers can adopt the SM3 algorithm from the Chinese cryptographic standards. The ground station stores $\{Z_{\text{task}}, s_g, s_u\}$, and securely distributes $\{Z_{\text{task}}, s_g, s_u\}$ to the UAV, which also stores them. Both parties initialize the counter ct to a unified value.

Stage Two: Key Agreement. The ground station and the UAV each use a random number generator approved by the National Cryptography Administration to generate random temporary private keys r_g and r_u, respectively. Let the base point of the elliptic curve be G. The ground station and the UAV can then generate their respective random temporary public keys R_g and R_u as follows:

$$\begin{cases} R_g = [r_g]G \\ R_u = [r_u]G \end{cases} \tag{2}$$

The temporary private key is a variable scalar, while the base point is a known and fixed point, making this a fixed-point scalar multiplication. To optimize scalar multiplication for ARM architecture in the UAV platform, a width-7 window method is employed, using point addition operations to replace point doubling. The scalar multiplication is optimized at the assembly level through the construction of a precomputation table.

Since the point addition efficiency in affine coordinates with Jacobian projection coordinates is far higher than point doubling efficiency, elliptic curve points in the precomputation table are stored in affine coordinates, taking up 148KB of space. The coordinates of the base point G are (G_x, G_y), and in a double loop, the outer loop iterates over i from 0 to 36, and the inner loop iterates over j from 0 to 63. For each combination (i, j), the coordinates of G are multiplied by i, and the result is stored in the precomputation table. The modular operation $\bmod p$ is implemented using an optimized Montgomery multiplication method (CIOS).

The CIOS Montgomery multiplication method performs bitwise multiplication of two large integers, accumulates the results into an intermediate variable, and generates an unreduced product. The modular reduction is handled using a carry mechanism to update the final result, ensuring the output is within the modulus p. Each entry in the precomputation table stores the affine coordinates of the points, specifically the x- and y-coordinates, completing the construction of the precomputation table. The precomputation table generation can be described using Algorithm 1.

Algorithm 1 Precomputation Table Generation

```
Input: Coordinates of $G$: $(G_x, G_y)$
Output: Precomputation Table $Table$
Initialize $Table$ with 148KB of memory
for $i = 0$ to 36 do
    for $j = 0$ to 63 do
        Compute $Table[i][j] = (27 \cdot i) \cdot (j \cdot G) \mod p$
        Store the result in $Table$
    end for
end for
return $Table$
```

The random temporary private key is encoded using an improved Booth encoding method [34], where extension bits are added, and groups of 3 bits are used for encoding, representing integer values between -3 and 3. This reduces the number of operations required. The initial result point (R_x, R_y) is set to the point at infinity. Then, a scalar scan is performed on the temporary private key scalar r, with a window width of 7. In each loop iteration, the value of the current window is extracted from r, and the corresponding value is looked up in the precomputation table. The value from the precomputation table is added to the current result point (R_x, R_y) using point addition. This process repeats until the entire scalar multiplication $r \cdot G$ is completed, yielding the coordinates (R_x, R_y). The optimized scalar multiplication is described in Algorithm 2.

Algorithm 2 Optimized Scalar Multiplication

```
Input: Temporary scalar $r$, Precomputation Table $Table$
Output: Coordinates of $r \cdot G$: $(R_x, R_y)$
Initialize result point $(R_x, R_y)$ to infinity
Set window size to 7
while $r$ is not fully scanned do
    Extract the current window value from $r$
    Lookup corresponding values from $Table$
    Perform point addition: $(R_x, R_y) \leftarrow (R_x + x, R_y + y)$
end while
return $(R_x, R_y)$
```

As a result, the number of point additions required for scalar multiplication is related to the height of the precomputation table, which is 37 operations, with no need for point doubling. In the first communication, the ground station calculates the public key R_g and the verification message msg_g and sends them to the UAV. Upon receiving the public key and verification message, the UAV verifies the data using the stored information and then calculates the UAV's public key R_u and verification message msg_u, which are sent to the ground station. The ground station verifies the information, completing the key exchange. For subsequent communications, after the end of the previous round, the ground station and UAV simultaneously compute their public keys. The ground station sends its public key, verification message, and one-time encrypted ciphertext to the UAV, which verifies the data and immediately transmits its public key back to the ground station, thereby reducing one communication round. After each key agreement, both parties increment the value of the counter *ct* by 1. The verification messages msg_g and msg_u are computed as follows:

$$\begin{cases} msg_g = \mathrm{HMAC} - \mathrm{SM3}(Z_{task}, R_g||s_g||s_u||ct) \\ msg_u = \mathrm{HMAC} - \mathrm{SM3}(Z_{task}, R_u||s_g||s_u||ct) \end{cases} \tag{3}$$

Where HMAC stands for Hash-based Message Authentication Code [35], and HMAC-SM3 indicates that the hash function used in HMAC is implemented using SM3. Since the task digest Z_{task}, the ground station's secret identifier s_g, and the UAV's secret identifier s_u are all securely distributed and stored during the initialization phase, and R_g and R_u are the public keys exchanged during the key agreement process, and *ct* is the key agreement counter for the current round, the HMAC mechanism ensures that the communicating parties are the trusted devices from the initialization phase, that the key agreement messages have not been tampered with, and that the messages are timely and have not been replayed. After the public key exchange and verification, the ground station calculates the shared key K_g, and the UAV calculates the shared key K_u:

$$\begin{cases} K_g = [r_g]R_u \\ K_u = [r_u]R_g \end{cases} \tag{4}$$

Due to the commutative property of elliptic curves, $K = K_g = K_u$, so both the ground station and the UAV derive the same one-time secret key K. This key needs to go through the key derivation operation described in Section 3.4 to meet the requirements of one-time encryption and decryption.

3.4. One-Time Pad based Lightweight Encryption Protocol (LEP)

In LEP, assuming that the ground station and the UAV have shared a short key K of n^2 bits, this short key is arranged into an $n \times n$ matrix, referred to as the Short Key Matrix (SKM):

$$SKM = \begin{bmatrix} k_{1,1} & \cdots & k_{1,n} \\ \vdots & \ddots & \vdots \\ k_{n,1} & \cdots & k_{n,n} \end{bmatrix} \tag{5}$$

Each column i of each row in SKM is viewed as the $n-i$-th bit in binary representation. Each row can thus be converted into a number, and the SKM matrix can be transformed into an n-dimensional vector called the Diffusion Vector (DV):

$$DV = \begin{bmatrix} d_1 \\ \vdots \\ d_n \end{bmatrix} = \begin{bmatrix} 2^{n-1}k_{1,1} + \cdots + 2^0 k_{1,n} \\ \vdots \\ 2^{n-1}k_{n,1} + \cdots + 2^0 k_{n,n} \end{bmatrix} \tag{6}$$

Assuming the message length for communication between the ground station and the UAV is $n \times n^2$ bits, the message is first arranged into an $n \times n^2$ matrix, referred to as the Message Matrix (M):

$$M = \begin{bmatrix} m_{1,1} & \cdots & m_{1,n^2} \\ \vdots & \ddots & \vdots \\ m_{n,1} & \cdots & m_{n,n^2} \end{bmatrix} \tag{7}$$

For the Message Matrix M, each row is cyclically shifted to the left by the corresponding number in the DV, resulting in a transformed message matrix M'. This process performs a diffusion operation on the plaintext:

$$M' = \begin{bmatrix} m_{1,|(0-d_1) \bmod n|} & \cdots & m_{1,|(n^2-1-d_1) \bmod n|} \\ \vdots & \ddots & \vdots \\ m_{n,|(0-d_n) \bmod n|} & \cdots & m_{n,|(n^2-1-d_n) \bmod n|} \end{bmatrix} \tag{8}$$

Each row of M' has n^2 bits. By arranging each row from left to right and top to bottom, it can be organized into n matrices of size $n \times n$, denoted as $D^{(1)}$ through $D^{(n)}$.

Select three perpendicular projection directions for performing Three-dimensional Parity Check on the cubic matrix. For each projection direction, XOR each layer of matrices with the next layer, then XOR the resulting matrix with the subsequent layer, obtaining three projection matrices $T^{(1)}$, $T^{(2)}$, and $T^{(3)}$:

$$\begin{cases} T^{(1)} = D^{(1)} \oplus D^{(2)} \oplus \cdots \oplus D^{(n)} \\ T^{(2)}_{i.j} = \sum\limits_{a=1}^{n} \oplus D^{(i)}_{a,j} \\ T^{(3)}_{i.j} = \sum\limits_{a=1}^{n} \oplus D^{(i)}_{4-j+1,a} \end{cases} \tag{9}$$

Based on the calculated matrices, the encryption party encrypts the plaintext to obtain the ciphertext matrix CM. CM is $n \times n^2$ bits in length, matching M and arranged in an $n \times n^2$ matrix form. Let CM_i denote the i-th row of CM. The encryption party computes:

$$\begin{cases} CM_1 = D^{(1)} \oplus T^{(1)} \oplus SKM \\ CM_2 = D^{(2)} \oplus T^{(2)} \oplus T^{(3)} \\ \vdots \\ CM_{n-1} = D^{(n-1)} \oplus T^{(2)} \oplus T^{(3)} \\ CM_n = D^{(n)} \oplus T^{(1)} \oplus SKM \end{cases} \tag{10}$$

Since the communicating parties have securely shared the short key SKM, the encryption party transmits only the ciphertext CM to the decryption party. The decryption party first restores $T^{(1)}$ by computing:

$$T^{(1)} = CM_1 \oplus CM_2 \oplus \cdots \oplus CM_n \tag{11}$$

After obtaining $T^{(1)}$, the decryption party can restore $D^{(1)}$ and $D^{(n)}$ based on $T^{(1)}$:

$$\begin{cases} D^{(1)} = CM_1 \oplus T^{(1)} \oplus SKM \\ D^{(n)} = CM_n \oplus T^{(1)} \oplus SKM \end{cases} \tag{12}$$

The matrices $D^{(2)}$ to $D^{(n-1)}$ are generated through the reverse process from CM_2 to CM_{n-1}, which can be calculated as follows:

$$\begin{cases} D^{(2)} = CM_2 \oplus T^{(2)} \oplus T^{(3)} \\ \vdots \\ D^{(n-1)} = CM_{n-1} \oplus T^{(2)} \oplus T^{(3)} \end{cases} \tag{13}$$

The decryption party, following the same method as the encryption party, converts $D^{(1)}$ through $D^{(n)}$ back into each row of M', obtaining M', and generates the diffusion vector DV using the short key matrix SKM. Each row of M' is then cyclically shifted to the right by the corresponding number in the DV, restoring the message matrix M:

$$M = \begin{bmatrix} m'_{1,|(0+d_1) \bmod n|} & \cdots & m'_{1,|(n^2-1+d_1) \bmod n|} \\ \vdots & \ddots & \vdots \\ m'_{n,|(0+d_n) \bmod n|} & \cdots & m'_{n,|(n^2-1+d_n) \bmod n|} \end{bmatrix} \tag{14}$$

3.5. Formal Security Proof

We provide a game-based proof to demonstrate that the proposed ABA-LEP protocol achieves mutual authentication and key indistinguishability under the standard elliptic curve and hash-based assumptions.

Game 0 – Real Execution. The adversary interacts with the real protocol between the ground station g and UAV u, observing exchanged messages including R_g, msg_g, R_u, and msg_u. The session key is derived as:

$$K = [r_g]R_u = [r_g][r_u]G = [r_u]R_g \tag{15}$$

Under the hardness of the Elliptic Curve Diffie-Hellman (ECDH) problem, the adversary cannot compute K without knowing either r_g or r_u.

Game 1 – Random Oracle Model. The verification messages msg_g and msg_u are computed using HMAC-SM3 with securely shared secrets s_g, s_u, and the task digest Z_{task}:

$$msg_g = \mathrm{HMAC_SM3}(Z_{task}, R_g | s_g | s_u | ct) \tag{16}$$

Without knowledge of the secrets, the adversary cannot forge valid authentication tags or impersonate a legitimate party.

Game 2 – Key Indistinguishability. Let K' be a uniformly random key of the same length as K. If the adversary can distinguish between encryptions using K and K' under the LEP encryption, then it breaks IND-CPA security:

$$|\Pr[\mathcal{A}(Enc_K(M_0)) = 1] - \Pr[\mathcal{A}(Enc_K(M\_1)) = 1]| \leq \epsilon \tag{17}$$

where ϵ is negligible under the ECDH and random oracle assumptions.

Hence, the protocol ensures mutual authentication and confidentiality against passive and active adversaries in the defined threat model.

4. Experiment and Analysis

4.1. Experimental Design

To validate the security and lightweight characteristics of ABA-LEP, both formal security verification and real UAV experiments are conducted.

For security verification, Section 4.2 employs the ProVerif formal verification tool. For evaluating lightweight aspects, a series of three experiments have been conducted: these include an evaluation of time consumption for elliptic curve operations, an analysis of computational time overhead at various stages, and a study on the number of messages transmitted over a given period.

These experiments explore whether optimizations for the ARM architecture can reduce the time for elliptic curve operations, thus shortening key agreement time. They also investigate if the optimized process can reduce authentication and key derivation times. Ultimately, the goal is to determine if the proposed ABA-LEP can transmit more messages within the same time while maintaining the same level of security.

Real UAV experiments are conducted on the CrazyFlie 2.1 UAV, which uses an ARM Cortex M-4 architecture. The CrazyFlie 2.1 features an STM32F405 MCU, a microcontroller based on the ARM Cortex M-4 core, with a maximum clock frequency of 168 MHz, 1 MB of Flash, and 192 KB of RAM. The ground station node used for initiating requests and measuring time is a virtual machine running Ubuntu Live Server 22.04, with 2 GB of memory and 2 processor cores. The ground station and UAV establish a wireless channel using the open-source USB wireless adapter Crazyradio Dongle.

Since precise timing is only measured at the ground station, the time spent executing operations on the UAV needs to account for communication delays. Therefore, the average overhead for measuring time on the UAV $\overline{t_{func}}$ is calculated as follows:

$$\overline{t_{func}} = \frac{1}{n} \sum_{i=1}^{n} (t_{total}^{(i)} - t_{RTT}) \tag{18}$$

where t_{total} is the total time measured by the ground station, $t_{RTT} = t_{comm1} + t_{comm2}$ represents the communication time, measured through independent experiments sending empty data packets. t_{comm1} is the communication time from the ground station to the UAV for sending computational requests, and t_{comm2} is the communication time for sending results back from

Table 2: Key Variables and Their Meanings in the Verification Process

Definition Statement	Attribute	Meaning
`free k: key [private].`	Private	Shared Key
`free Ztask: bitstring_type [private].`	Private	Task Digest
`free Sg: key [private].`	Private	Ground Station Private Key
`free Su: key [private].`	Private	Drone Private Key
`free c: channel.`	Public	Public Channel
`new Rg: bitstring_type;`	Public	Ground Station Public Key
`new Ru: bitstring_type;`	Public	Drone Public Key

the UAV to the ground station. i denotes the i-th measurement result, n is the number of experiments, with $n = 100$.

4.2. Security Verification

To verify the security of ABA-LEP against spoofing, eavesdropping, replay, and man-in-the-middle attacks, the ProVerif formal verification tool is employed. ProVerif is an automated formal verification tool capable of rigorous analysis of security properties for network protocols.

The key variables and their meanings defined in ProVerif are in Table 2.

Before verifying under the assumptions of no attacker, the presence of identity spoofing attackers, eavesdropping attackers, replay attackers, and man-in-the-middle attackers, the following query statements were executed in each scenario. Both parties were able to securely complete key exchange and identity authentication, as Table 3.

This indicates that ABA-LEP shows strong security against identity spoofing, eavesdropping, replay, and man-in-the-middle attacks.

4.3. Comparison with Existing Schemes against Common Attacks

To complement the formal verification results, we compare the performance of ABA-LEP with existing schemes under common network attacks, in terms of time overhead incurred during defense. The results are summarized in Table 4.

As shown in Table 4, ABA-LEP achieves significantly lower response latency against common attacks. This is attributed to the precomputation-enabled ECC operations and lightweight authentication mechanism tailored

Table 3: Execution of Query Statements and Results

Statement	Result	Meaning
`query attacker: bitstring_type; event(drone_auth) ==> event(gs_auth).`	`Query event(drone_auth) ==> event(gs_auth) is false.`	Both parties can complete mutual authentication, and the authentication must be initiated by the ground station.
`query attacker: bitstring_type; event(gs_auth) ==> event(drone_auth).`	`Query event(gs_auth) ==> event(drone_auth) is true.`	This means that if the ground station is authenticated, the drone must also be authenticated.
`query attacker(k).`	`Query not attacker(k[]) is true.`	The attacker cannot obtain the key `k` under the verification assumptions.

Table 4: Attack Response Time Comparison (in milliseconds)

Attack Type	ABA-LEP	DroneSec	HECC
Identity Spoofing	1.73	4.92	3.01
Replay Attack	1.48	3.97	2.95
Man-in-the-Middle Attack	2.07	5.85	4.12
Eavesdropping Detection	1.66	4.13	3.42

for ARM-based UAVs. For instance, in the case of identity spoofing, ABA-LEP detects and responds in 1.73 ms, which is 64.8% and 42.5% faster than DroneSec and HECC, respectively.

4.4. Comparison of Time Consumption for Elliptic Curve Operations

This experiment primarily explores the time overhead of elliptic curve operations using different methods. First, the time consumption for fixed-point scalar multiplication on elliptic curves is examined as the private key length increases. In elliptic curve-based key agreement processes, the public key is generated by multiplying a randomly generated temporary private key with the base point, which is a frequently involved and costly operation in the key exchange protocol.

Among the compared schemes, ABA-LEP adopts the Fp-256 prime field elliptic curve recommended in the SM2 Elliptic Curve Public Key Cryptography Standard. It optimizes fixed-point scalar multiplication by setting the window width based on the characteristics of the ARM architecture and constructing a precomputed table. The DroneSec scheme proposed by Jian et al. [33] uses the secp256rl prime field elliptic curve, with fixed-point scalar multiplication implemented using the public Crypto library without further optimization. The HECC scheme proposed by Khan et al. [31] uses hyper-elliptic curve divisor multiplication (Elliptic Curve Point Multiplication, ECPM) to optimize multiplication operations on elliptic curves.

The scalar (private key) length is randomly set to 64, 128, 256, and 512 bits, and the time consumption for multiplying it by the fixed point (base point) is measured. The scalar length is set as the x-axis, and the time consumption of different schemes is set as the y-axis. The results are shown in Figure 2.

The experimental results show that time overhead for fixed-point scalar multiplication increases with scalar length across all schemes, but ABA-LEP

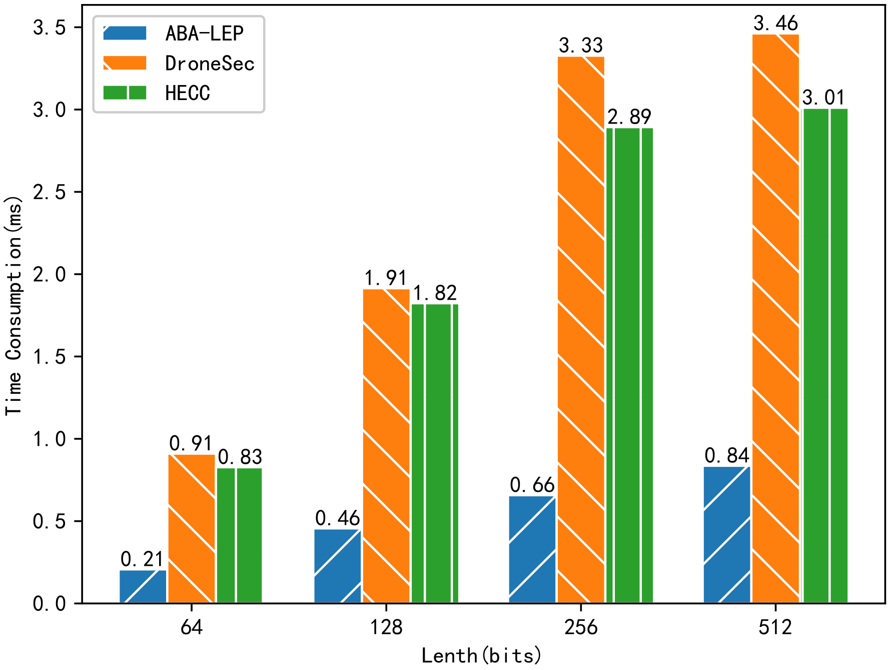

Figure 2: Comparison of Time Consumption for Fixed-Point Scalar Multiplication

has the lowest overhead and smallest increase. This is due to affine coordinate storage in the ABA-LEP precomputed table, reducing memory access, and using only point addition during precomputation. DroneSec, lacking such optimizations, has the highest overhead, serving as the baseline. HECC performs well only with short keys. For 256-bit keys, ABA-LEP improves speed by 80.18% over DroneSec and 77.16% over HECC.

When the key length is set to 256 bits, the comparison of time overhead for elliptic curve modular multiplication, modular addition, point addition, and fixed-point scalar multiplication is shown in Figure 3.

It can be seen that at a key length of 256 bits, by constructing a precomputed table that converts the more time-consuming multiplication into a certain number of additions, the performance of fixed-point scalar multiplication is improved.

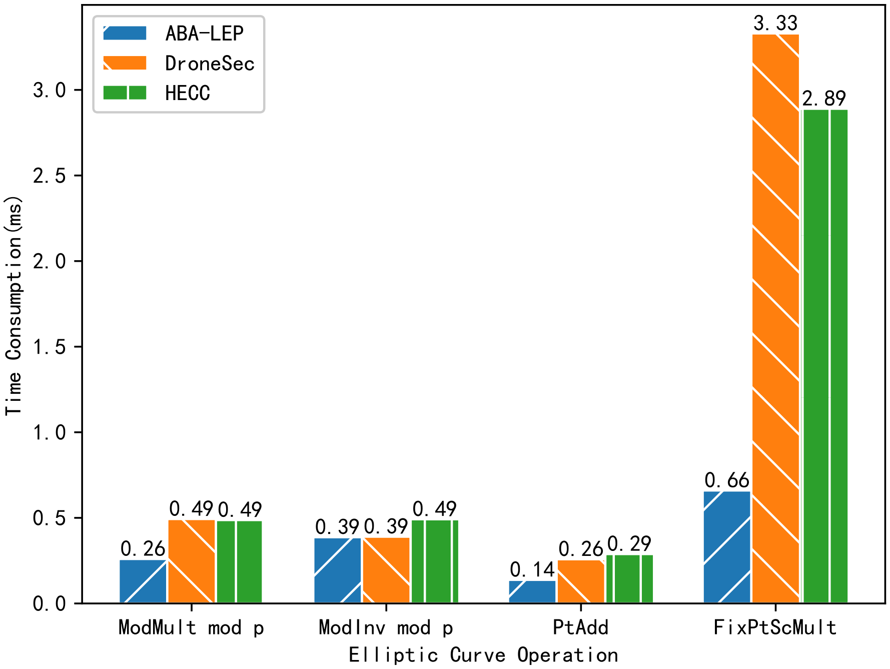

Figure 3: Comparison of Time Consumption for Elliptic Curve Operations

4.5. Comparison of Time Overhead in Different Stages

This experiment mainly explores the comparison of time overhead in different stages of different schemes. ABA-LEP, DroneSec, and HECC schemes all involve system initialization, key agreement, identity verification, key derivation, and message encryption/decryption. The specific experimental results are shown in Figure 4.

The experimental results show that ABA-LEP has the highest time overhead during system initialization due to the construction of a 148KB precomputed table and the need to share secret parameters via a secure channel. DroneSec and HECC have simpler initialization processes with lower overhead. However, as initialization is executed only once per UAV mission, its overall impact is minimal.

In the key agreement stage, ABA-LEP improves speed by 53.00% compared to DroneSec and by 48.87% compared to HECC with 256-bit keys. All schemes perform similarly in identity verification, as they all use HMAC, with ABA-LEP using HMAC-SM3. For key derivation and encryption/de-

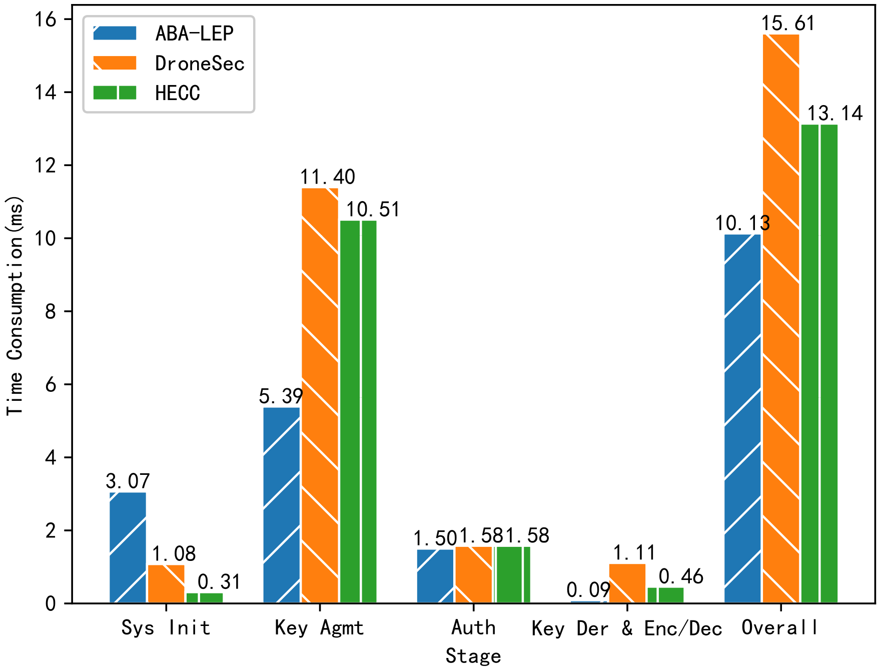

Figure 4: Comparison of Time Overhead in Different Stages

cryption, ABA-LEP, using a one-time pad key derivation method and simple XOR operations, is 91.00% faster than DroneSec and 79.25% faster than HECC, which does not encrypt keys during transmission.

4.6. Comparison of Cumulative Message Transmission Over Time

This experiment explores the change in the number of messages transmitted as time progresses under the premise that the key is updated after each round of communication. The specific experimental results are shown in Figure 5.

The experimental results show that when the communication time is short, the differences in the number of transmitted messages between the three schemes are small. However, as communication time increases, ABA-LEP transmits 97.02% more messages than DroneSec and 28.45% more than HECC. This is mainly due to the lower time overhead of ABA-LEP in the key agreement, key derivation, and message encryption/decryption stages, while the identity verification stage remains relatively constant across schemes.

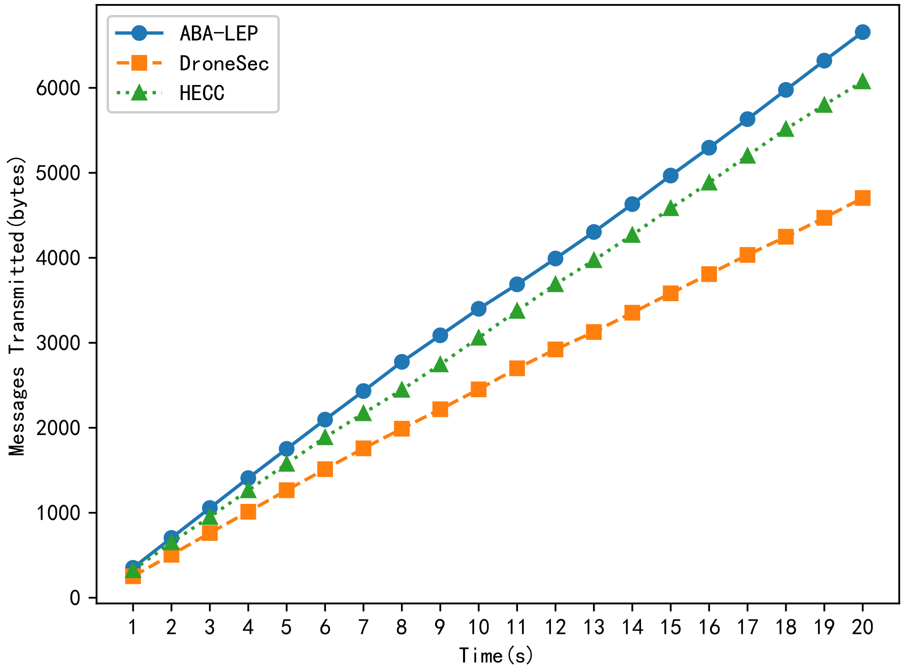

Figure 5: Cumulative Number of Messages Transmitted Over Time

Since the key is updated after each round of communication, each round involves key agreement, identity verification, key derivation, and message encryption/decryption. Therefore, ABA-LEP can cumulatively transmit more message data over time.

5. Conclusion

This study designs and implements an ARM-based UAV bidirectional authentication and lightweight encryption protocol based on the domestic SM2 standard, aiming to address the shortcomings of existing UAV communication protocols in terms of autonomy and limited computational resources. The protocol improves key agreement efficiency by optimizing SM2 fixed-point scalar multiplication at the assembly level for ARM architecture. Additionally, lightweight encryption is achieved using a one-time pad to meet the security needs of UAV communication under limited energy and computational capacity. Experimental results show that the protocol exhibits low computational and time overhead on the CrazyFlie 2.1 UAV and that its effectiveness in resisting various attack types is verified through ProVerif.

Declarations

Conflict of interest The authors declare that they have no confict of interest.

References

[1] Leandro Marcos Da Silva et al., eds. *Anomaly-based intrusion detection system for in-flight and network security in uav swarm.* IEEE, 2023.

[2] Pericle Perazzo et al. "Drone path planning for secure positioning and secure position verification". In: *IEEE Transactions on Mobile Computing* 16.9 (2016), pp. 2478–2493.

[3] Xiaokun Fan et al. "UAV-Enabled Federated Learning in Dynamic Environments: Efficiency and Security Trade-off". In: *IEEE Transactions on Vehicular Technology* (2023).

[4] Muslum Ozgur Ozmen and Attila A Yavuz, eds. *Dronecrypt-an efficient cryptographic framework for small aerial drones.* IEEE, 2018.

[5] Qian Zhou et al. "A location privacy preservation scheme based on consortium block-chain in VANET". In: *Journal of Nanjing University of Posts and Telecommunications (Natural Science)* (2022).

[6] Zewen Yang and Jun Ye, eds. *An Improved Lightweight Cryptography Based on One-Time Pad for IoT Devices.* Springer, 2022.

[7] Carsten Bormann, Mehmet Ersue, and Ari Keranen. *Terminology for constrained-node networks.* Tech. rep. 2070-1721. 2014.

[8] Pietro Tedeschi et al. "Privacy-aware remote identification for unmanned aerial vehicles: current solutions, potential threats, and future directions". In: *IEEE Transactions on Industrial Informatics* 20.2 (2023), pp. 1069–1080.

[9] Xinyao Wang et al. "Secure Communication Against Active UAV Eavesdropper: A Fingerprint-Localization and Channel Tracking Approach". In: *IEEE Transactions on Communications* (2025).

[10] Raffaele Pizzolante et al., eds. *Improving drone security in smart cities via lightweight cryptography.* Springer, 2023.

[11] Yuan Gao et al. “Multi-IRS-Aided Secure Communication in UAV-MEC Networks”. In: *IEEE Transactions on Vehicular Technology* (2025).

[12] Jun Zhao, Xuewen Zeng, and Zhichuan Guo. “Design and implementation of high speed PCIe cipher card supporting GM algorithms”. In: *Journal of Electronics & Information Technology* 41.10 (2019), pp. 2402–2408.

[13] Jipeng Zhang et al. “Research on Efficient Implementation of SM2 for Mobile Devices”. In: *ACTA Electronica Sinica* 51.12 (2023), pp. 3437–3443.

[14] Shay Gueron and Vlad Krasnov. “Fast prime field elliptic-curve cryptography with 256-bit primes”. In: *Journal of Cryptographic Engineering* 5.2 (2015), pp. 141–151.

[15] Teklay Gebremichael, Ulf Jennehag, and Mikael Gidlund, eds. *Lightweight IoT group key establishment scheme from the one time pad.* IEEE, 2019.

[16] Christian Matt and Ueli Maurer, eds. *The one-time pad revisited.* IEEE, 2013.

[17] Xuan Huang et al. “Research on High Reliable Multimode Wireless Communication Technology for UAV”. In: *Frontiers in Computing and Intelligent Systems* 2.3 (2022), pp. 89–93.

[18] Qingyang Zhang, Fumio Machida, and Ermeson Andrade, eds. *Performance bottleneck analysis of drone computation offloading to a shared fog node.* IEEE, 2022.

[19] Qian Zhou et al. “Evss: An efficient verifiable search scheme over encrypted cloud data”. In: *World Wide Web* 26.4 (2023), pp. 1459–1479.

[20] Qian Zhou et al. “A novel semantic-aware search scheme based on BCI-tree index over encrypted cloud data”. In: *World Wide Web* 26.5 (2023), pp. 3055–3079.

[21] Nils Gura et al., eds. *Comparing elliptic curve cryptography and RSA on 8-bit CPUs.* Springer, 2004.

[22] Shay Gueron and Vlad Krasnov. “Fast prime field elliptic-curve cryptography with 256-bit primes”. In: *Journal of Cryptographic Engineering* 5.2 (2015), pp. 141–151.

[23] Mehmet Adalier and Antara Teknik, eds. *Efficient and secure elliptic curve cryptography implementation of curve p-256*. Vol. 66. NIST, 2015.

[24] Long Mai et al., eds. *Accelerating SM2 digital signature algorithm using modern processor features*. Springer, 2019.

[25] Liang Jin et al., eds. *Achieving one-time pad via endogenous secret keys in wireless communication*. IEEE, 2020.

[26] Emraida Marie M Manucom, Bobby D Gerardo, and Ruji P Medina, eds. *Analysis of key randomness in improved one-time pad cryptography*. IEEE, 2019.

[27] Baoan Jiang. "One-time Encryption Algorithm Based on Finite Field Key Exchange". In: *Journal of Information Security Research* 9.5 (2023), p. 457.

[28] Shikang Chen et al. "Literature Review of Secure Communication Protocols for UAV". In: *Communications Technology* 57.3 (2024), p. 213.

[29] Danny Dolev and Andrew Yao. "On the security of public key protocols". In: *IEEE Transactions on information theory* 29.2 (1983), pp. 198–208.

[30] Teng Li et al., eds. *Lightweight secure communication mechanism towards UAV networks*. IEEE, 2019.

[31] Muhammad Asghar Khan et al. "An efficient and secure certificate-based access control and key agreement scheme for flying ad-hoc networks". In: *IEEE Transactions on Vehicular Technology* 70.5 (2021), pp. 4839–4851.

[32] Yue Guo. "Design of Lightweight Authentication and Key Agreement Protocol in UAV Network Based on Chinese Commercial Cryptography". Xi'an, China: Xidian University, 2021.

[33] Qirui Jian, Zemao Chen, and Xiaokang Wu. "Authentication and Key Agreement Protocol for UAV Communication". In: *Computer Science* 49.08 (2022), pp. 306–313.

[34] Andrew D Booth. "A signed binary multiplication technique". In: *The Quarterly Journal of Mechanics and Applied Mathematics* 4.2 (1951), pp. 236–240.

[35] Fengqun Wang et al. “Blockchain-based lightweight message authentication for edge-assisted cross-domain industrial internet of things”. In: *IEEE Transactions on Dependable and Secure Computing* (2023).