





Article

On the Cryptanalysis of a Simplified AES Using a Hybrid Binary Grey Wolf Optimization

Rizk M. Rizk-Allah ^{1,2}, Hatem Abdulkader ³, Samah S. Abd Elatif ⁴, Diego Oliva ^{5,*},
Guillermo Sosa-Gómez ⁶ and Václav Snášel ²

- ¹ Department of Basic Engineering Science, Faculty of Engineering, Menoufia University, Shebin El-kom 32511, Menoufia, Egypt; rizk_masoud@yahoo.com or rizk.masoud@vsb.cz
- ² Faculty of Electrical Engineering and Computer Science, VŠB-Technical University of Ostrava, 70800 Ostrava, Czech Republic; vaclav.snasel@vsb.cz
- ³ Department of Information Systems, Faculty of Computers and Information, Menoufia University, Shebin El-kom 32511, Menoufia, Egypt; hatem6803@yahoo.com
- ⁴ Department of Basic Engineering Science, Higher Institute of Engineering and Technology, Tanta 31739, Egypt; eng_samah2011@yahoo.com
- ⁵ Departamento de Innovación Basada en la Información y el Conocimiento, Universidad de Guadalajara, CUCEI, Guadalajara 44430, Mexico
- ⁶ Facultad de Ciencias Económicas y Empresariales, Universidad Panamericana, Álvaro del Portillo 49, Zapopan 45010, Mexico; gsosag@up.edu.mx
- * Correspondence: diego.oliva@cucei.udg.mx

Abstract: Cryptosystem cryptanalysis is regarded as an NP-Hard task in modern cryptography. Due to block ciphers that are part of a modern cipher and have nonlinearity and low autocorrelation in their structure, traditional techniques and brute-force attacks suffer from breaking the key presented in traditional techniques, and brute-force attacks against modern cipher S-AES (simplified-advanced encryption standard) are complex. Thus, developing robust and reliable optimization with high searching capability is essential. Motivated by this, this paper attempts to present a novel binary hybridization algorithm based on the mathematical procedures of the grey wolf optimizer (GWO) and particle swarm optimization (PSO), named BPSOGWO, to deal with the cryptanalysis of (S-AES). The proposed BPSOGWO employs a known plaintext attack that requires only one pair of plaintext-ciphertext pairs instead of other strategies that require more pairs (i.e., it reduces the number of messages needed in an attack, and secret information such as plaintext-ciphertext pairs cannot be obtained easily). The comprehensive and statistical results indicate that the BPSOGWO is more accurate and provides superior results compared to other peers, where it improved the cryptanalysis accurateness of S-AES by 82.5%, 84.79%, and 79.6% compared to PSO, GA, and ACO, respectively. Furthermore, the proposed BPSOGWO retrieves the optimal key with a significant reduction in search space compared to a brute-force attack. Experiments show that combining the suggested fitness function with HPSOGWO resulted in a 109-fold reduction in the search space. In cryptanalysis, this is a significant factor. The results prove that BPSOGWO is a promising and effective alternative to attack the key employed in the S-AES cipher.

Keywords: cryptanalysis; simplified-AES; binary optimization; grey wolf optimizer; particle swarm optimization

MSC: 94A60



Citation: Rizk-Allah, R.M.; Abdulkader, H.; Elatif, S.S.A.; Oliva, D.; Sosa-Gómez, G.; Snášel, V. On the Cryptanalysis of a Simplified AES Using a Hybrid Binary Grey Wolf Optimization. *Mathematics* **2023**, *11*, 3982. <https://doi.org/10.3390/math11183982>

Academic Editor: Cheng-Chi Lee

Received: 3 August 2023

Revised: 14 September 2023

Accepted: 14 September 2023

Published: 19 September 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

With the development of virtual communication, users can safely communicate and share information with other systems worldwide [1]. Cryptography is a key topic for secure data transfer. Cryptology is divided into two branches: cryptography and cryptanalysis [2]. Cryptography is the science of creating ciphertext, which renders communication

meaningless to anyone but the intended receiver. Plaintext is the unmasked text, whereas the ciphertext is the masked version. Only the recipient, who knows the secret key, can read this disguised text. Cryptanalysis, meanwhile, is the science of decrypting ciphertext after obtaining the key. Cryptanalysis is a crucial tool for determining the strengths and weaknesses of cryptosystems. The brute-force attack and the cryptanalytic attack are two types of cryptanalysis techniques. The extensive search approach, commonly known as brute force, searches all potential keys until the cipher is cracked. If the search problem is large, the cryptanalysis task becomes an NP-hard problem; the Advanced Encryption Standard (AES) and Data Encryption Standard (DES) represent two ciphers of NP-hard problems. There are many types of attacks that a cryptanalyst can use to break a cipher based on the attacker's availability of information. The goal is to find the key to recover the ciphertext easily. A brute-force attack represents one of the popular methods of doing so. In this attack, the cryptanalyst attempts each key combination until the right one is found. For long keys, a network of computers can be used to combine their computational power and cumulative power, making a brute-force attack possible at a higher cost.

The Simplified-Advanced Encryption Standard (S-AES) is a not-Feistel cipher that produces a 16-bit ciphertext using a 16-bit plaintext and a 16-bit key. S-AES is utilized in embedded devices with low memory and processing capacity, such as mobile phones and GPS receivers [3]. The encryption consists of one round pre-round, and two rounds are modified. Depending on how much information is accessible to the attacker, a cryptanalyst can employ various techniques to break a cipher. The known-plaintext attack (KPA) is an attack where the attacker possesses plaintext and ciphertext samples. The ciphertext-only attack (COA) is another attack in which the cryptanalyst can only access the ciphertext. Compared to the COA, the KPA is simpler to implement because the attacker owns more information (one pair of plain text and ciphertext) to help them obtain the key.

Recently, cryptanalysis experts have begun considering several methods for cracking cryptographic systems. Metaheuristic optimization methods (MOMs) have attracted the scientific community's attention due to their capacity to handle complex issues that can't be solved using conventional methods [2]. MOMs are divided into three topics: evolutionary-based, physical-based, and swarm-based methods. The first method is a generic population-based metaheuristic inspired by biological evolution processes, such as reproduction, mutation, recombination, and selection. Physical-based algorithms make up the second group. Search agents communicate and move about the search space using these algorithms based on physics rules. The last type is swarm-based algorithms based on social organisms' collective activity. The interaction of the swarms with one another and with their environment inspires collective intelligence. MOMs are becoming more common in several applications due to their advantages, such as being easy to implement, not relying on gradient information, and being able to avoid strikes in local optima due to exploration and exploitation features. Some of the popular MOMs include particle swarm optimization (PSO) [4], binary aquila optimizer (BAO) [5], genetic algorithm (GA) [6], differential evolution (DE) [7], ant colony optimization (ACO) [8], grey wolf optimization (GWO) [9], bat algorithm (BA) [10], and fruit fly algorithm (FA) [11]. However, optimal solutions may not be guaranteed because more exploration and exploitation searches are needed, especially when dealing with complicated optimization tasks. Exploitation is the activity of covering each individual's immediate surroundings. Meanwhile, exploration allows for the research of space that is farther away from the individual's current location. Nature's exploitation and exploration calculations are difficult to comprehend because of a lack of consensus. On the other hand, as one capacity strengthens, the other weakens, and vice versa [12]. To reach a proper balance, one alternative is to develop hybrid techniques, in which two or more algorithms are integrated to increase the performance of each algorithm, and the resulting hybrid technique is referred to as a memetic method [13]. In [14], the exploration–exploitation equilibrium generated by the swap operator during the search in Hamming space was probabilistically evaluated using weight classes of nonlinear, power-attack-resistant S-boxes.

The PSO has been combined with various metaheuristics in several studies, including hybrid PSO with GA (PSOGA) [15,16], PSO with Ant Lion Optimization (PSO-AIO) [17]; PSO was also combined with the Bacterial Foraging Optimization (BFO) algorithm in [18]. These hybrid approaches are meant to complement one another's capabilities to enhance exploitation while reducing the possibility of a local optimum.

Similarly, in the field of hybrid metaheuristics, GWO has sparked significant attention. For example, for test scheduling and continuous optimization, the authors of [19] combined GWO with the sine cosine algorithm (SCA). The authors of [20] hybridized GWO and GA for breast cancer detection. This encouraged us to test the binary hybrid PSOGWO variant, which solves the continuous problems as well as our cryptanalysis problem. The following are the main contributions of the suggested algorithm.

- 1- This work introduces a hybrid technique that combines the GWO and the PSO (HPSOGWO) and converts it to a binary version for simplified AES cryptanalysis.
- 2- This technique improves the exploitation ability in the particle swarm optimization with the ability of exploration in the grey wolf optimizer to produce both variants' strength.
- 3- HPSOGWO is used to describe the cryptanalysis challenge as a combinatorial problem to break the Simplified-AES cryptosystem using KPA.
- 4- The performance of the proposed BPSOGWO is compared to other attacks, where it exhibits faster performance with only one pair of plaintext–ciphertext pairs (i.e., it reduces the number of messages needed in an attack, and secret information, such as plaintext–ciphertext pairs, cannot be obtained easily).
- 5- It can improve the cryptanalysis for the fitness of the S-AES by 82.5% compared to PSO, 84.79% compared to GA, and 79.6% compared to ACO.

The remaining sections of this article are as follows: The scope of work in the field of cryptanalysis is discussed in Section 2. Section 3 introduces the Simplified Advanced Encryption Standard. The basics of the PSO and GWO are introduced in Section 4. Section 5 introduces the proposed HPSOGWO algorithm for attacking S-AES, while Section 6 offers the results and discussion. Finally, in Section 7, the work's conclusion is presented.

2. Related Work

Many researchers have addressed the cryptanalysis of S-AES. For instance, Musa et al. [21] employed linear and differential cryptanalysis to attack S-AES, where their linear analysis method is more efficient than brute force attacks because it only takes 109 plaintext and ciphertext combinations to attack the first round of S-AES. Mansoori et al. [22] performed a linear cryptanalysis on S-AES to break the first round, which took at least 116 plaintext and ciphertext couples, whereas breaking the second round required 548 plaintext and ciphertext pairings, implying that a linear attack on S-AES is possible. Simmons [23] performed algebraic cryptanalysis on S-AES by solving polynomial equations. Vimalathithan and Valarmathi [24,25] are two such metaheuristic-based attacks that have been carried out. An intelligent agent for cryptanalysis of S-AES with only one plaintext was proposed by Rania Saeed and Ashraf Bhery [26]. Moreover, several works have addressed the metaheuristic attacks on stream ciphers, modern symmetric block ciphers, and classical ciphers [27–31]. In [32], authors concentrated on the cryptanalysis of classical ciphers, such as substitution, transposition, and Vigenere ciphers, using a variety of well-liked metaheuristics, including EA, ACO, PSO, and cuckoo search algorithms. Ref. [33] presented the binary cuckoo search technique for SDES ciphertext. In contrast, the dolphin swarm algorithm (DSA), binary firefly (BF), and binary cat swarm optimization (BCSO) have been proposed in [34–36] to deal with the DES cryptosystem. Tabu Search (TS) has been introduced to handle the stream ciphers (RC4, VMPC, and RC4+) in [37,38], while the Grari [39] presented an ACO algorithm-specific Merkle–Hellman cipher. In [40], the effectiveness of eight popular, newly released metaheuristic algorithms for the MHKC were examined. The authors in [41] presented quantum cryptanalysis for binary elliptic curves by reducing the circuit depth, complementing recent research that focused on reducing the circuit width. A novel binary

hybrid PSO-EO algorithm for cryptanalysis of the internal state of the RC4 cipher was presented in [42].

3. Simplified Advanced Encryption Standard (S-AES)

This section introduces the fundamentals of the S-AES algorithm [2,43] with its related information, the S-AES encryption, key expansion, and decryption technologies. S-AES is a block cipher that accepts a 16-bit plaintext and a 16-bit key as input and outputs of a 16-bit ciphertext. A state is created from the 16-bit input plaintext, which is a 4×4 matrix of nibbles (a nibble is a 4-bit block). A state is taken and replaced with another one for the next round; this process occurs in each round. Substitution (Sub. Nibbles), mix columns, shift row, and round key addition represent four S-AES transformations. The S-AES encryption, decryption and key generation, and decryption techniques are described in Figure 1.

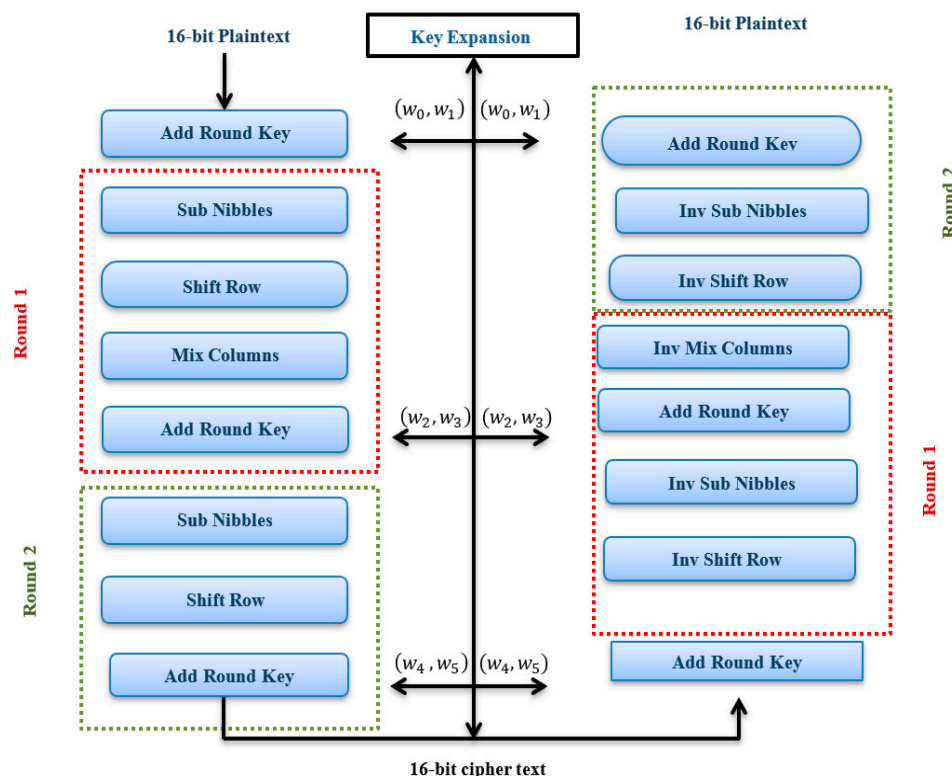


Figure 1. S-AES Encryption and Decrypt.

3.1. Substitution

Substitution (Sub. Nibbles) is done for each nibble as the first step in Round 1. A basic lookup table contains the nibble substitution function, often called an S-box or substitution table. All possible 4-bit values are permuted in the S-box, creating a 44-bit array of nibble values. The state matrix’s component bits are mapped into new nibbles, and a row index is calculated using the nibble’s leftmost two bits. These row and column indices in the S-box imply a unique 4-bit value (the new nibble). The change has a clear, confusing impact, making the relationship between the ciphertext’s statistics and the key as convoluted as possible, so that attempts to crack the key become more difficult. The S-box is used for encryption; for example, the inverse of the S-box used for decryption is s^{-1} (see Equation (1)).

$$S = \begin{bmatrix} 9 & 4 & A & B \\ D & 1 & 8 & 5 \\ 6 & 2 & 0 & 3 \\ C & E & F & 7 \end{bmatrix} \qquad S^{-1} = \begin{bmatrix} A & 5 & 9 & B \\ 1 & 7 & 8 & F \\ 6 & 0 & 2 & 3 \\ C & 4 & D & E \end{bmatrix} \qquad (1)$$

3.2. Shift Row

The state matrix’s first row remains unchanged during this stage. The one-piece circular offset occurs during the second row.

3.3. Mix Columns

The Mix Columns transformation is completed with each column in the matrix in the third stage, converting each state column into a new column. To calculate the transformation, multiply the state column by a constant square matrix. In Galois Field GF (2), polynomial coefficients are regarded as nibbles and moved into the state column and the matrix of constants. To verify that the new matrix remains within the filed GF (2⁴), multiply the bytes with the irreducible polynomial modulus (x⁴ + x + 1). The block at the output (b₀, b₁, b₂, b₃) is defined as follows: Let (a₀, a₁, a₂, a₃) be the input nibbles of the Mix Columns operation (see Equation (2)).

$$\begin{bmatrix} b_0 & b_2 \\ b_1 & b_3 \end{bmatrix} = \begin{bmatrix} 1 & 4 \\ 4 & 1 \end{bmatrix} \begin{bmatrix} a_0 & a_2 \\ a_1 & a_3 \end{bmatrix} \tag{2}$$

3.4. Add Round Key

At the end of each round, the round key is appended. The state matrix and the key round are included by bitwise XOR operation.

3.5. S-AES Key Expansion

In this stage, three-round keys are generated using the primary key, and each one is employed in a different specialized round, making S-AES safer. The keys used in decryption are the same as those used in the encryption algorithm.

The key expansion method creates round keys word per word, with a word being a two-bit array. The procedure generates six words, w₀, w₁, w₂, w₃, w₄, and w₅, where k₀= w₀ w₁, k₁= w₂ w₃, and k₂= w₄ w₅. We generate the two bytes w₀ and w₁ from the original key. The S-AES key expansion algorithm is defined in Algorithm 1.

Algorithm 1: S-AES Key Expansion Algorithm

```

For 2 ≤ i ≤ 5 do
    If i(mod 2) = 0 then
        wi = wi-2 ⊕ REC(i/2) ⊕ SubNib(RotNib(wi-1))
    Else
        wi = wi-1 ⊕ wi-2
    end if
end for
    
```

This method is based on $REC(i) = x^{i+2} \in GF(2^4)$, where $REC(i)$ is described as $REC(i) = x^{i+2} \in GF(2^4)$, so $REC(1) = x^3 = 1000$ and $REC(2) = x^4 = x + 1 = 0011$. If N₀ and N₁ are two nibbles and N₁N₀ is the result of their concatenation, the RotNib and SubNib functions are defined as RotNib (N₀N₁) = N₁N₀ and SubNib (N₀N₁) = Sbox (N₀) Sbox (N₁); this means that the “rotate nibble” is completed first, followed by the “substitute nibble”.

3.6. Decryption

The opposite of encryption is decryption. When ciphertext contains 16 bits and a 16-bit key, the original plaintext also contains 16 bits. As seen in Figure 1, decryption employs single pre-round and two-round modifications, just like encryption. Decryption techniques are opposed to encryption procedures [2].

4. Material and Methods

4.1. Basics of the Particle Swarm Optimization (PSO)

J. Kennedy and R. Eberhat devised the PSO algorithm [4] as an optimization method for describing how individuals (particles) in a population behave. Algorithm 2 shows the pseudo-code for PSO. It mimics the behavior of swarming and flocking in creatures such as locusts, fish schools, and bird flocks. A swarm of particles is given a population of random alternative solutions in the canonical PSO model. They search continuously for novel solutions in the D-dimension issue space. The location of a particle has a direct relationship with its fitness. Each particle j is associated with a velocity (V_j) and a location (X_j) that can be used during the evolution stage (see Equations (3) and (4)).

$$X_j = (x_{j,1}, x_{j,2}, \dots, x_{i,D}) \tag{3}$$

$$V_j = (v_{j,1}, v_{j,2}, \dots, v_{i,D}) \tag{4}$$

The updating or evolution of velocity and the location of the j th particle can be formed for the next time ($t + 1$) as follows in (Equations (5) and (6)).

$$V_j(t + 1) = w \cdot V_j(t) + c_1 \cdot r_1 \cdot (pbest_j(t) - X_j(t)) + c_2 \cdot r_2 \cdot (gbest_j(t) - X_j(t)) \tag{5}$$

$$X_j(t + 1) = X_j(t) + V_j(t + 1) \tag{6}$$

where $j = 1, 2, \dots, N$, w is the inertia weight. $pbest$ defines the personal best associated with each particle, $gbest$ defines the global best location among all particles, c_1 defines the self-recognition component's coefficient, c_2 notes the social component factor, and r_1, r_2 are random numbers generated between 0.0 and 1.0.

Algorithm 2: PSO procedures

```

Define the size of the swarm  $N$ .
Define  $t_{max}$ , which is the greatest number of generations possible.
Create a population of  $N$  particles as a starting point.
Set particle positions and velocities at random.
Determine the fitness of each particle.
Find the most suitable particle, the best
     $t = 0$ 
    While ( $t < t_{max}$ )
        For  $i = 1$  to  $N$ 
            Calculate the  $i$ th particle's new position.
            Find  $pbest$ 
        End For
        Find  $gbest$ 
         $t = t + 1$ 
    End while
Return  $gbest$ 

```

4.2. Binary PSO (BPSO)

Kennedy and Eberhart [44] proposed the BPSO to create binary solutions for searching space's discrete or binary domain. In this context, the BPSO uses the original PSO's velocity and formulates the binary string of solution as follows in Equation (7).

$$X_j(t + 1) = \begin{cases} 1, & \text{if } r_3 \leq sig(V_j(t + 1)) \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

where r_3 is random numbers from $[0, 1]$, sig denotes the sigmoidal function that is determined by Equation (8).

$$sig(V_j(t + 1)) = \frac{1}{1 + e^{-V_j(t+1)}} \tag{8}$$

4.3. Basics of Grey Wolf Optimization (GWO)

GWO was introduced by Mirjalili et al. [8] as a metaheuristic algorithm influenced by the natural leadership of grey wolves' structure and hunting strategy. Grey wolves are apex predators that live in groups of five to twelve individuals. They lived in a well-knit group. To replicate the leadership hierarchy, there are four members in the GWO's population: alpha, beta, delta, and omega. Alphas are the bosses, both male and female. The alpha (α) decides when to sleep, wake up, hunt, etc. The grey wolf hierarchy's beta (β) is the second-highest rank. Subordinate wolves assist the alpha wolf with collective decision-making and other tasks and are called betas (β). The wolves' second-level members (beta) should obey the wolves' first-level members (alpha) and command the pack's lower-level members. The alpha receives information from the second-level wolf (beta), which then enforces the wolf leader's (alpha) directives across the group. Omega (γ) is the grey wolves' third rank. Omega wolves are used as victims. Third-rank grey wolves must always obey the commands of all other dominating wolves. They are the last wolves to eat. Although the omega wolves in the pack may not appear to have a strong character, it is apparent that, without the omega, the entire pack would face internal friction and hardships because the omega receives the other wolves' wrath and violence. This contributes to the fulfillment of the entire pack and the preservation of the dominant structure. Subordinate delta (δ) wolves have to surrender to alpha (α) and beta (β), but they dominate the omega (γ).

The following is a mathematical model of a grey wolf's hunting mechanism:

- i. Searching (looking for the prey).
- ii. Pursuing (following, chasing, and nearing the prey).
- iii. Encircling and pestering the prey until it comes to a halt.
- iv. Prey attack.

To determine each agent's encircling behavior in the group, the following mathematical formulae are used as shown in Equations (9) and (10).

$$D = |C \cdot X_p - X(t)| \tag{9}$$

$$X(t + 1) = |X_p(t) - A \cdot D| \tag{10}$$

The current iteration is denoted by t , X_p is the prey's position, X denotes the grey wolf's position, and A, C mean coefficient vectors. The following formulas (see Equation (11)) are used to calculate the vectors A and C :

$$A = 2a \cdot r_1 - a, \quad C = 2 \cdot r_2 \tag{11}$$

where r_1 and r_2 are random vectors with values between 0 and 1, and a 's components are linearly reduced from 2 to 0 overtime. The value of a is presented as follows (see Equation (12)):

$$a = 2 - t \cdot \frac{2}{t_{max}} \tag{12}$$

where t_{max} defines the maximum number of iterations.

Prey hunting is typically led by α and β , with δ joining occasionally. The most effective proposal solutions, α , β , and δ , have more information about prospective prey locations. The positions of (γ) agents are adjusted according to the positions of α , β , and δ agents. In this regard, the Equations (13) and (14) are provided.

$$\begin{aligned} D_\alpha &= |C_1 \cdot X_\alpha - X|, \\ D_\beta &= |C_2 \cdot X_\beta - X|, \\ D_\delta &= |C_3 \cdot X_\delta - X|, \end{aligned} \tag{13}$$

$$\begin{aligned} X_1 &= X_\alpha - A_1 \cdot D_\alpha, \\ X_2 &= X_\beta - A_2 \cdot D_\beta, \\ X_3 &= X_\delta - A_3 \cdot D_\delta, \end{aligned} \tag{14}$$

The position of the grey wolves is updated as shown in Equation (15):

$$X(t + 1) = \frac{X_1(t) + X_2(t) + X_3(t)}{3} \tag{15}$$

The variable A is a random number between $[-2a, 2a]$. When the random variable $|A|$ is smaller than one, wolves are compelled to attack their victim. The ability to hunt for prey is known as exploration, whereas exploitation is the ability to attack the prey. To divert the hunter’s attention away from the prey, arbitrary values of A are used. When $|A|$ exceeds 1, wolves are compelled to leave this victim and seek a better one. The main steps of GWO are presented in Algorithm 3.

Algorithm 3: The main steps of GWO

Creation the grey wolves’ population $X_i(i = 1, 2, 3, \dots, n)$.
 Initialize the variables a, A , and C .
 Computing the search agent fitness values and agent ranking $(x_\alpha, x_\beta, x_\delta)$
 $t = 0$
While $(t < t_{max})$
 For $i = 1$ to N
 We are updating the current search agent’s position by Equation (15).
 End For
 Updating of a, A , and C .
 Calculation of search agent fitness values and rating of the agents.
 Updating the position of $(x_\alpha, x_\beta$ and $x_\delta)$
 $t = t + 1$
End while
End

4.4. Binary GWO (BGWO)

The BGWO was developed with the binary format for feature selection problems in [39], where the grey wolf position vector is updated according to Equation (16) to firm the conversion from the continuous domain to the binary domain.

$$X(t + 1) = \begin{cases} 1, & \text{if } rand \leq sigmoid\left(\frac{x_1+x_2+x_3}{3}\right) \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

where $rand$ is randomly selected from a uniform distribution, $\in [0, 1]$, $X(t + 1)$ is the updated binary location at iteration t , and $sigmoid(\cdot)$ is defined as the transfer function that is defined as shown in Equation (17).

$$sigmoid(a) = \frac{1}{1 + e^{-10(x-0.5)}} \tag{17}$$

5. The Proposed Hybrid PSO-GWO (PSOGWO)

The primary motivation behind the suggested PSOGWO is thoroughly explained in this section, along with a step-by-step introduction.

5.1. The Motivation of the Proposed PSOGWO

PSO has several flaws, including stagnation and the particles’ inclination to become inactive after a given number of iterations. As a result, local and worldwide search capabilities are lacking. To overcome some of the limitations of the ordinary versions of GWO and PSO, this hybridization of the two optimizers boosts GWO’s population diversity while also increasing PSO’s capacity to escape the local minima. This hybridization technique improves the exploration phase of the PSO algorithm with the ability of exploration in the grey wolf optimizer, allowing it to produce both variants’ strengths and evaluate many

possible solutions. In this regard, rather than utilizing traditional procedures of the PSO, the velocity aspect is updated using the hierarchical structure of the grey wolf aspect.

Also, the grey wolf’s exploitation and exploration were governed by the inertia constant (w). The first three agents’ locations in the search space have been updated by Equations (18) and (14) as follows:

$$\begin{aligned} D_\alpha &= |C_1 \cdot X_\alpha - w \times X|, \\ D_\beta &= |C_2 \cdot X_\beta - w \times X|, \\ D_\delta &= |C_3 \cdot X_\delta - w \times X|, \end{aligned} \tag{18}$$

Then, the velocity of the PSO is integrated with the solutions obtained by the GWO variant to perform the following updating procedures (see Equation (19)).

$$V_j(t + 1) = w \cdot \left(V_j(t) + c_1 \cdot r_1 \cdot (X_1 - X_j(t)) + c_2 \cdot r_2 \cdot (X_2 - X_j(t)) + c_3 \cdot r_3 \cdot (X_3 - X_j(t)) \right) \tag{19}$$

$$X_j(t + 1) = X_j(t) + V_j(t + 1) \tag{20}$$

This algorithm provides a high-level coevolutionary hybrid by combining merits of the PSO and GWO as they work together.

5.2. The Proposed PSOGWO Based on the Binary Aspect (BPSOGWO) for Attacking S-AES

The proposed BPSOGWO is discussed to find the encryption key in the cryptanalysis field. The BPSOGWO is designed to deal with the binary aspect of cryptanalysis tasks. In this regard, after computing Equations (19) and (20), the binary value is obtained using Equation (21).

$$x_d(t + 1) = \begin{cases} 1, & \text{if } rand \leq sigmoid(X) \\ 0, & \text{otherwise} \end{cases} \tag{21}$$

where $sigmoid(X)$ denotes the sigmoid function (i.e., Equation (22)) of the position vector obtained in Equation (20), and $x_d(t + 1)$ represents the resultant binary location at the iteration $(t + 1)$.

$$sigmoid(X) = \frac{1}{1 + e^{-(X/2)}} \tag{22}$$

5.3. Fitness Function of the S-AES

During cryptanalysis, a suggested key created by metaheuristic algorithms must be evaluated for acceptability (fitness or cost). The known plaintext attack (KPA), which requires many plaintext/ciphertext pairs, is adopted in this study. As a result, the fitness should always be built to compare a created plaintext obtained from the trial key and the original plaintext. Using Equation (23), compute the produced plaintext P_G using the known ciphertext C and the trail key obtained by optimization algorithm. The key’s fitness function $F(K_c)$ is specified in Equation (24).

$$P_G = S_AES_{decrypt}(C, K_c) \tag{23}$$

$$F(K_c) = \frac{\#(P_G \oplus P)}{16} \tag{24}$$

where $\#$ stands for the quantity of zeros in $(P_G \oplus P)$, \oplus specifies the *Xor* operation, and P is the original plaintext. F lies in a range of $[0, 1]$. F is equivalent to 1 if all bits in P_G are the same as those in P . The produced key matches the original one in this situation; thus, we want to optimize the fitness function.

Each agent represents a 16-bit binary key in this algorithm. Initialize an agent of N wolves $\in [0.1]$, and then, Equation (24) is applied to calculate the agent’s fitness. Then, solutions are evolved using Equations (14) and (19)–(22). For the updated agent’s position,

the fitness is calculated. The procedure is repeated until the desired number of iterations has been reached. Figure 2 presents the flowchart of the proposed algorithm.

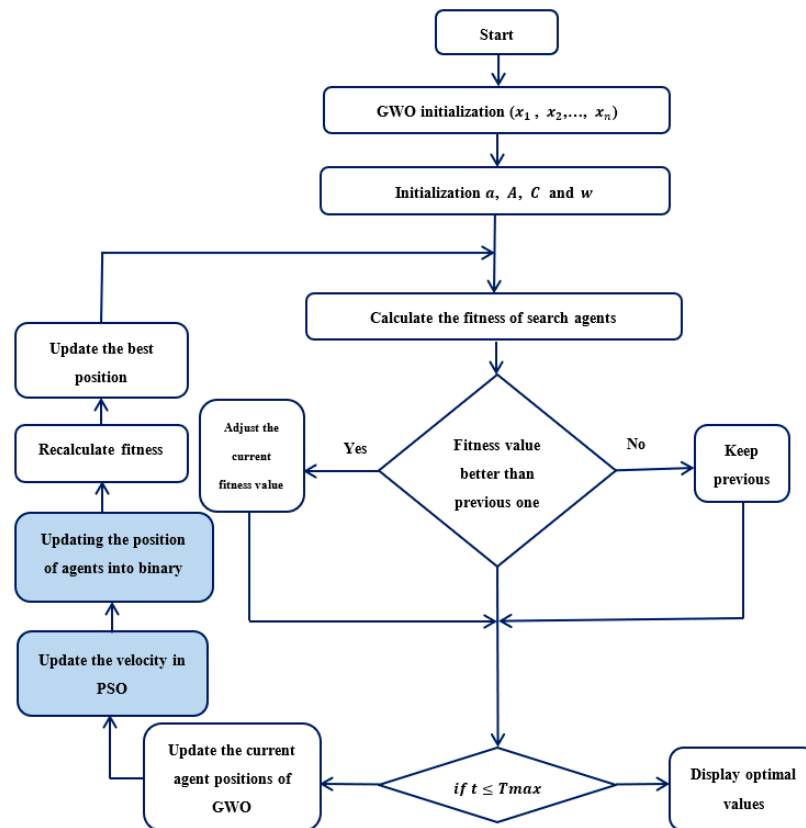


Figure 2. BPSOGWO optimization flow chart.

6. Experimental Setup and Results

This section investigates the assessment of the proposed BPSOGWO on the S-AES, where its performance is compared with the BGWO, BPSO, and binary whale optimization algorithm (BWOA).

6.1. Parameters Configuration

To ensure a fair comparison between the proposed BHOSOFWO algorithm and the compared ones (i.e., BGWO, BPSO, and BWOA), each algorithm is run through 5 independent times with the same number of iterations. As in the original works, the parameter configuration for the implemented optimizers is provided and listed in Table 1. It is important to note that after a certain number of trials, the common parameters, such as population size (Np), of the provided optimizers are set to 30. The BPSO, BGWO, BWOA, and BPSOGWO algorithms were implemented using a MATLAB environment (R2013a) on a laptop with an Intel® core™ i5-5200U CPU, 2.7 GHz, and 6 GB RAM.

The investigation is performed using different sizes of population (10, 20, 30, 40, and 50) and 100 iterations. Figure 3 shows the outcomes of the experiments with two noteworthy observations: (a) For all algorithms BPSOGWO, BGWO, BPSO, and BWOA, with increasing population size, the average fitness of optimal solutions rises, and the proposed BPSOGWO is superior. Also, it was noticed that the proposed BPSOGWO achieves the optimal results with the population size of 30 agents, and (b) the optimum solutions obtained with BPSOGWO in different population sizes are significantly higher than those obtained from other algorithms. Accordingly, the population size is adopted as 30 agents and 100 iterations. By applying the proposed algorithms, it can be noted that the BPSOGWO provides faster convergence than the others, saving the computation efforts as it reaches 30 search agents only with 49 iterations. Furthermore, the statistical results during

the different runs for the proposed BPSOGWO and the compared ones are presented in Table 2. Based on reported results, the proposed BPSOGWO provides competitive results in terms of best value, while the BPSOGWO achieves superior results in terms of mean values. Moreover, the BPSOGWO saves more computational time than the other optimizers.

Table 1. The configurations of parameters for the presented algorithms.

Parameter	Definition	Value
$c1$	Coefficient of cognitive acceleration	0.5
$c2$	Coefficient of social acceleration	0.5
$c3$	Coefficient vector	0.5
W	Inertia weight	$0.5 + \text{rand}()/2$
N	Population size	30
D	No. of variables	16
t_{max}	Maximum iterations	100
R	The number of runs	5

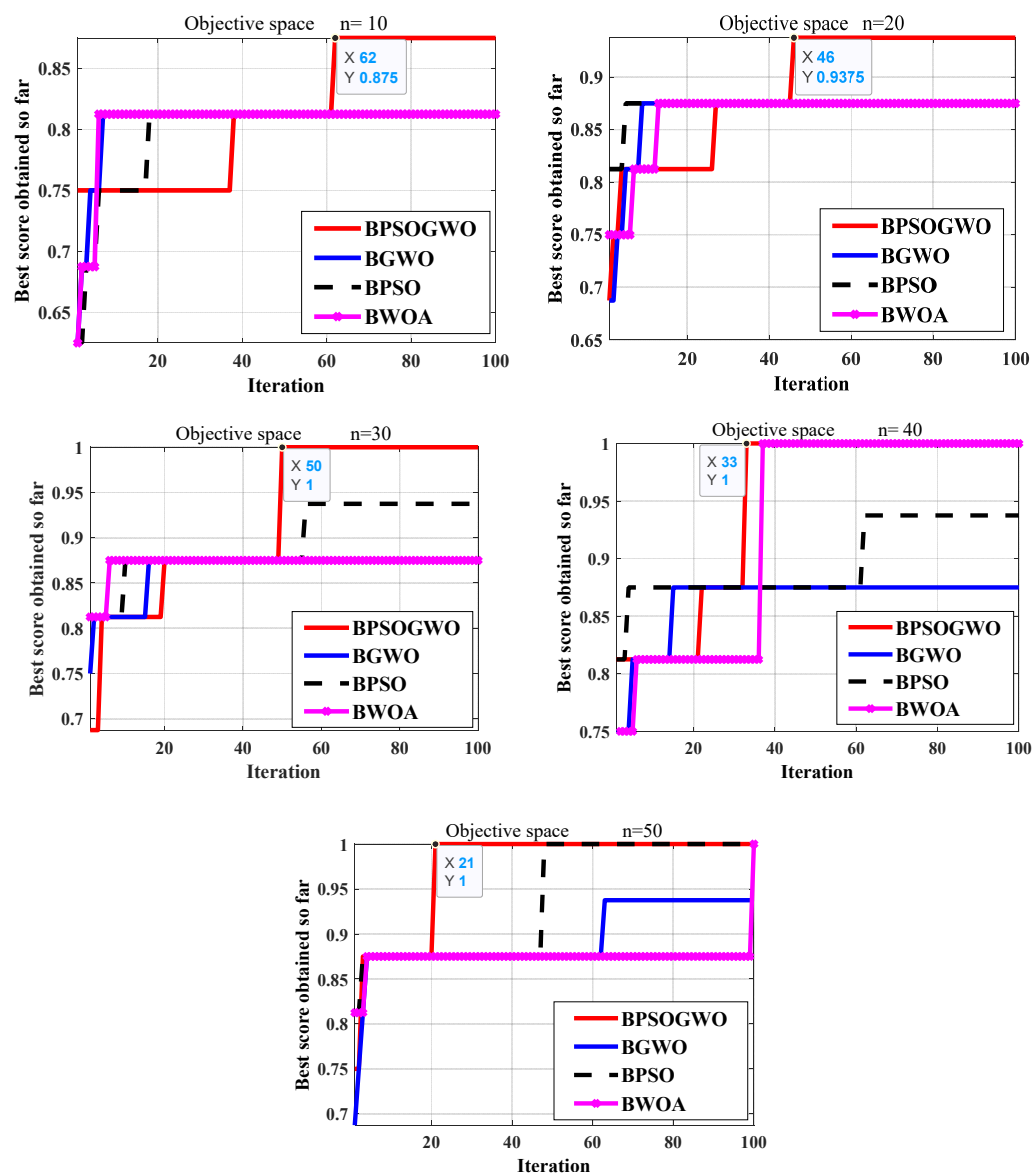


Figure 3. Convergence curve of BPSOGWO, BPSO, BGWO, and BWOA for cryptanalysis of S-AES for different population sizes.

Table 2. The statistical results for cryptanalysis of S-AES.

Statistics	BPSO	BWOA	BGWO	BPSOGWO
Best	1	1	0.9375	1
Mean	0.9187	0.93125	0.89375	0.9375
Median	0.9375	0.9375	0.875	0.9375
Sdt	0.0422	0.035478	0.03019	0.041667
Time	10.397315	11.866	13.40509	9.769446

6.2. Index Storage Space

Storage space is a crucial aspect to be considered in secure retrieval. Data storage in all the presented algorithms has the same number of bytes because we are looking for the encryption key, which consists of 16 bits. Therefore, other variables were used for comparison, such as time, fitness value, and the superiority of the presented algorithm over other algorithms, as shown in Table 2.

6.3. Impact of Population Size on the S-AES Characteristics

As mentioned in this work, the final accuracy of the obtained key and the fitness function strongly depend on the population size (N), which controls the effort of the algorithm and the search size. To exhibit the better characteristics of the S-AES such as fitness, key found, no. of keys, and browsed no. of bits correct, the effective choice of these characteristics is investigated. As shown in Table 3, by increasing the population size, the fitness value increases, and the space search and number of bits needed to obtain a correct key also increased. Therefore, it can be noted that the proposed BPSOGWO reaches the best solution with 16 bits and 3000 evaluations, where some of the compared solutions can reach the correct key but require 4000 and 5000 evolutions.

Table 3. Experimental results of different population size values.

		N = 10	N = 20	N = 30	N = 40	N = 50
BPSOGWO	Fitness	0.8750	0.9375	1	1	1
	Key found	DBAC	95BC	A73B	A73B	A73B
	No of keys Browsed	1000	2000	3000	4000	5000
	No of bits correct	6	9	16	16	16
BGWO	Fitness	0.8125	0.8750	0.8750	0.8750	0.9375
	Key found	5CFE	FBCF	7DF9	EDAA	E7FD
	No of keys Browsed	1000	2000	3000	4000	5000
	No of bits correct	6	7	8	10	11
BPSO	Fitness	0.8125	0.8750	0.9375	0.9375	1
	Key found	DC88	DBAC	BBBE	B1F2	A73B
	No of keys Browsed	1000	2000	3000	4000	5000
	No of bits correct	5	6	10	9	16
BWOA	Fitness	0.8125	0.8750	0.8750	1	1
	Key found	97B5	E7FD	BBBE	A73B	A73B
	No of keys Browsed	1000	2000	3000	4000	5000
	No of bits correct	10	11	10	16	16

6.4. The Fitness Function's Suitability

Designing a fitness function is an important stage in evolutionary algorithms. For different population sizes, calculate the correlation coefficient ($corr$) between it and the number of bits that are corrected in the generated key and the cost function using Equation

(25). The link between the number of items that can be accurately recovered, and the cost function is evaluated using this coefficient. If a perfectly straight linear connection exists, $corr = 1$ is obtained, which means that a good mechanism for assessing the generated key can be obtained, where the proposed algorithm is almost certain to reach the best key. If $corr = -1$, on the other hand, it is in a perfect decreasing linear connection. As seen in Figure 4, the correlation coefficient grows with the number of population sizes (N).

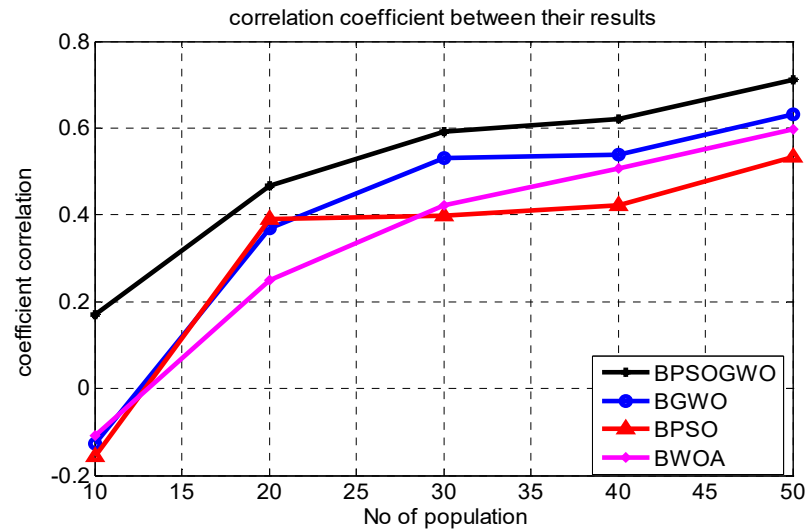


Figure 4. Correlation coefficient of BPSOGWO, BPSO, BGWO, and BWOA for cryptanalysis of S-AES for different population sizes.

It can be seen from the graph that BPSOGWO performs better than other algorithms and has the best correlation between our results. We utilized 50 keys as a sample to construct this coefficient, calculating the fitness of each and the number of correct bits where x and y are n elements arrays.

$$corr(x, y) = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{\sqrt{n \sum x_i^2 - (\sum x_i)^2} \sqrt{n \sum y_i^2 - (\sum y_i)^2}} \tag{25}$$

6.5. Comparison of BPSOGWO with Other Methods

To verify the efficiency of BPSOGWO, it is compared with different optimization methods in this section, where the number of plaintext–ciphertext pairs in previous assaults is listed in Table 4. The proposed suggested method allows determining the key using only one known plaintext-ciphertext combination. The strengths of this algorithm in the cryptanalysis of S-AES are:

- It needs fewer ciphertext and plaintext pairs than other algorithms, as shown in Table 4.
- In comparison to the brute force attack, the space factor can be reduced.
- In comparison to using either PSO or GWO [45], this algorithm it reaches the correct key more efficiently and a smaller number of iterations.
- The BPSOGWO improved the cryptanalysis problem for S-AES by a percentage of 82.5% compared with PSO [23], 84.79% compared with GA [24], and 79.6% compared with ACO [46].

Table 4. Required number of plaintext-ciphertext for attacking.

Strategy	Attacked Rounds	Required Number of Plaintext-Ciphertext Pairs
Linear cryptanalysis Musa [21]	Round 1	109
Linear cryptanalysis Davood [22]	Round 1 Round 1 and Round 2	116 548
Linear cryptanalysis Bizaki [47]	Round 1 and Round 2	96
Using GA Vimalathithan [24]	Round 1 and Round 2	3
Using ACO Gari, Azouaoui, Zine-Dine [46]	Round 1 and Round 2	2
The proposed BPSOGWO	Round 1 and Round 2	1

7. Conclusions

The work attempted to compare the efficiency of Simplified-AES cryptanalysis utilizing KPA. The suggested approach effectively breaks the key, requiring only one known plaintext–ciphertext pair. In contrast, GA and ACO require three and two plaintext–ciphertext pairs to identify the genuine key (i.e., it reduces the number of messages needed in an attack; secret information such as plaintext–ciphertext pairs cannot be obtained easily). Compared to a brute-force approach, the proposed method helps narrow down the key's search space by a factor of 109. In cryptanalysis, this is a significant factor. The challenge in any cryptanalysis system is to quickly obtain the real key to the encryption system and not an approximate key. Compared to traditional methods, such as the brute-force attack, the proposed algorithm exceeds the time tremendously, as the brute-force attack requires 65,536 attempts to reach the real key while the proposed algorithm only needs 5000 attempts to obtain the real key. Compared to modern MOMS methods, such as GWO, PSO, and WOA, the algorithm is superior in terms of time and reaching an exact solution, not an approximate solution. Therefore, the algorithm solved the challenges in the cryptographic solution problem. Finally, because the fitness function utilized in this experiment is unaffected by the cipher under attack, this technique can be easily applied to other recent block ciphers or stream ciphers. Future research can be directed toward employing different swarm intelligence methods to attack AES with large-scale bits. They also present a new mutation mechanism to explore the search space effectively by hybridization with other techniques such as red fox, polar bear, and chimp algorithms.

Author Contributions: Conceptualization, D.O.; Methodology, R.M.R.-A., H.A. and D.O.; Software, R.M.R.-A., H.A., S.S.A.E. and V.S.; Validation, S.S.A.E.; Formal analysis, D.O. and G.S.-G.; Investigation, R.M.R.-A. and D.O.; Resources, G.S.-G.; Writing—original draft, G.S.-G. and V.S. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been supported by an internal grant project of VSB-Technical University of Ostrava (SGS project, grant number SP 2023/076).

Data Availability Statement: All data generated or analyzed during this study are included in this published article.

Conflicts of Interest: The authors declare no conflict of interest.

References

- Chinnasamy, P.; Albakri, A.; Khan, M.; Raja, A.A.; Kiran, A.; Babu, J.C. Smart Contract-Enabled Secure Sharing of Health Data for a Mobile Cloud-Based E-Health System. *Appl. Sci.* **2023**, *13*, 3970. [[CrossRef](#)]
- Stinson, D.R. *Cryptography: Theory and Practice*; Chapman and Hall/CRC: Boca Raton, FL, USA, 2005.

3. Manangi, S.J.; Chaurasia, P.; Singh, M.P. Simplified AES for Low Memory Embedded Processors. *Glob. J. Comp. Comp. Sci. Technol.* **2010**, *10*, 7–11.
4. Jain, M.; Saihjpal, V.; Singh, N.; Singh, S.B. An overview of variants and advancements of PSO algorithm. *Appl. Sci.* **2022**, *12*, 8392. [[CrossRef](#)]
5. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S.; Abualigah, L. Binary aquila optimizer for selecting effective features from medical data: A COVID-19 case study. *Mathematics* **2022**, *10*, 1929. [[CrossRef](#)]
6. Katoch, S.; Chauhan, S.S.; Kumar, V. A review on the genetic algorithm: Past, present, and future. *Multimed. Tools Appl.* **2021**, *80*, 8091–8126. [[CrossRef](#)]
7. Nadimi-Shahraki, M.H.; Taghian, S.; Mirjalili, S.; Faris, H. MTDE: An effective multi-trial vector-based differential evolution algorithm and its applications for engineering design problems. *Appl. Soft Comput.* **2020**, *97*, 106761. [[CrossRef](#)]
8. Mirjalili, S.; Dong, J.S.; Lewis, A. Ant colony optimizer: Theory, literature review, and application in AUV path planning. *Nature-Inspired Optim. Theor. Lit. Rev. Appl.* **2020**, *811*, 7–21.
9. Nadimi-Shahraki, M.H.; Moeini, E.; Taghian, S.; Mirjalili, S. Discrete Improved Grey Wolf Optimizer for Community Detection. *J. Bionic Eng.* **2023**, *20*, 2331–2358. [[CrossRef](#)]
10. Rizk-Allah, R.M.; Hassanien, A.E. New binary bat algorithm for solving 0–1 knap-sack problem. *Complex Intell. Syst.* **2018**, *4* 31–53. [[CrossRef](#)]
11. Rizk-Allah, R.M.; El-Sehiemy, R.A.; Deb, S.; Wang, G.G. A novel fruit fly framework for multi-objective shape design of tubular linear synchronous motor. *J. Supercomput.* **2017**, *73*, 1235–1256. [[CrossRef](#)]
12. Yang, X.S. Swarm intelligence based algorithms: A critical analysis. *Evol. Intell.* **2014**, *7*, 17–28. [[CrossRef](#)]
13. Mafarja, M.M.; Mirjalili, S. Hybrid Whale Optimization Algorithm with simulated annealing for feature selection. *Neurocomputing* **2017**, *260*, 302–312. [[CrossRef](#)]
14. Legón-Pérez, C.M.; Menéndez-Verdecía, J.A.; Martínez-Díaz, I.; Sosa-Gómez, G.; Rojas, O.; Veloz-Remache, G.d.R. Probabilistic Evaluation of the Exploration–Exploitation Balance during the Search, Using the Swap Operator, for Nonlinear Bijective S-Boxes, Resistant to Power Attacks. *Information* **2021**, *12*, 509. [[CrossRef](#)]
15. Shao, K.; Song, Y.; Wang, B. PGA: A New Hybrid PSO and GA Method for Task Scheduling with Deadline Constraints in Distributed Computing. *Mathematics* **2023**, *11*, 1548. [[CrossRef](#)]
16. El Menbawy, N.; Ali, H.A.; Saraya, M.S.; Ali-Eldin, A.M.T.; Abdelsalam, M.M. Energy-efficient computation offloading using hybrid GA with PSO in the Internet of robotic things environment. *J. Supercomput.* **2023**, *79*, 1–40. [[CrossRef](#)]
17. Vinothkumar, T.; Deepa, S.N.; Raj, F.V.A. Adaptive probabilistic neural network based on hybrid PSO–ALO for predicting wind speed in different regions. *Neural Comput. Appl.* **2023**, *35*, 19997–20011. [[CrossRef](#)]
18. Liu, X.; Wu, C.; Chen, P.; Wang, Y. Hybrid Algorithm Based on Phasor Particle Swarm Optimization and Bacterial Foraging Optimization. In Proceedings of the International Conference on Swarm Intelligence, Shenzhen, China, 14–18 July 2023; pp. 136–147.
19. Duan, Y.; Yu, X. A collaboration-based hybrid GWO-SCA optimizer for engineering optimization problems. *Expert Syst. Appl.* **2023**, *213*, 119017. [[CrossRef](#)]
20. Pramanik, R.; Pramanik, P.; Sarkar, R. Breast cancer detection in thermograms using a hybrid of GA and GWO based deep feature selection method. *Expert Syst. Appl.* **2023**, *219*, 119643. [[CrossRef](#)]
21. Musa, M.A.; Schaefer, E.F.; Wedig, S. A simplified aes algorithm and its linear and differential cryptanalyses. *Cryptologia* **2003**, *27*, 148–177. [[CrossRef](#)]
22. Mansoori, S.D.; Bizaki, H.K. On the vulnerability of simplified AES algorithm against linear cryptanalysis. *Int. J. Comp. Sci. Netw. Secur.* **2007**, *7*, 257–263.
23. Simmons, S. Algebraic cryptanalysis of simplified AES. *Cryptologia* **2009**, *33*, 305–314. [[CrossRef](#)]
24. Vimalathithan, R. Cryptanalysis of Simplified-AES Encrypted Communication. *Int. J. Comput. Sci. Inf. Secur.* **2015**, *13*, 142–150.
25. Vimalathithan, R.; Valarmathi, M.L. Cryptanalysis of Simplified-AES using Particle Swarm Optimisation. *Def. Sci. J.* **2012**, *62*, 117–121. [[CrossRef](#)]
26. Saeed, R.; Bhery, A. Cryptanalysis of Simplified-AES Using Intelligent Agent. In Proceedings of the Hybrid Artificial Intelligent Systems: 10th International Conference, HAIS 2015, Bilbao, Spain, 22–24 June 2015.
27. Ali, I.K. Cryptanalysis of simple substitution ciphers using bees algorithm. *J. Baghdad Coll. Econ. Sci. Univ* **2013**, *36*, 373–382.
28. Mekhaznia, T.; Menai, M.E.B. Cryptanalysis of classical ciphers with ant algorithms. *Int. J. Metaheuristics* **2014**, *3*, 175–198. [[CrossRef](#)]
29. Bhateja, A.K.; Bhateja, A.; Chaudhury, S.; Saxena, P.K. Cryptanalysis of vigenere cipher using cuckoo search. *Appl. Soft Comput.* **2015**, *26*, 315–324. [[CrossRef](#)]
30. Jain, A.; Chaudhari, N.S. A new heuristic based on the cuckoo search for cryptanalysis of substitution ciphers. In Proceedings of the International Conference on Neural Information Processing, Istanbul, Turkey, 9–12 November 2015; pp. 206–215.
31. Jain, A.; Chaudhari, N.S. A novel cuckoo search strategy for automated cryptanalysis: A case study on the reduced complex knapsack cryptosystem. *Int. J. Syst. Assur. Eng. Manag.* **2018**, *9*, 942–961. [[CrossRef](#)]
32. Sabonchi, A.K.S.; Akay, B. Cryptanalysis of polyalphabetic cipher using differential evolution algorithm. *Teh. Vjesn.* **2020**, *27*, 1101–1107.
33. Kamal, R.; Bag, M.; Kule, M. On the cryptanalysis of S-DES using binary cuckoo search algorithm. In *Computational Intelligence in Pattern Recognition*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 23–32.

34. Amic, S.; Soyjaudah, K.M.S.; Mohabeer, H.; Ramsawock, G. Cryptanalysis of DES-16 using binary firefly algorithm. In Proceedings of the 2016 IEEE International Conference on Emerging Technologies and Innovative Business Practices for the Transformation of Societies (EmergiTech), Balaclava, Mauritius, 3–6 August 2016; pp. 94–99.
35. Amic, S.; Soyjaudah, K.M.S.; Ramsawock, G. Dolphin swarm algorithm for cryptanalysis. In *Information Systems Design and Intelligent Applications*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 149–163.
36. Amic, S.; Soyjaudah, K.M.S.; Ramsawock, G. Binary cat swarm optimization for cryptanalysis. In Proceedings of the 2017 IEEE International Conference on Advanced Networks and Tel-communications Systems (ANTS), Bhubaneswar, India, 17–20 December 2017; pp. 1–6.
37. Polak, I.; Boryczka, M. Tabu search against permutation based stream ciphers. *Int. J. Electron. Telecommun.* **2018**, *64*, 137–145. [[CrossRef](#)]
38. Polak, I.; Boryczka, M. Tabu Search in revealing the internal state of RC4+ cipher. *Appl. Soft Comput.* **2019**, *77*, 509–519. [[CrossRef](#)]
39. Grari, H.; Lamzabi, S.; Azouaoui, A.; Zine-Dine, K. Cryptanalysis of Merkle-Hellman cipher using ant colony optimization. *IAES Int. J. Artif. Intell.* **2021**, *10*, 490. [[CrossRef](#)]
40. Abdel-Basset, M.; Mohamed, R.; Elkomy, O.M. Knapsack Cipher-based metaheuristic optimization algorithms for cryptanalysis in blockchain-enabled internet of things systems. *Ad. Hoc. Netw.* **2022**, *128*, 102798. [[CrossRef](#)]
41. Putranto, D.S.C.; Wardhani, R.W.; Larasati, H.T.; Ji, J.; Kim, H. Depth-optimization of Quantum Cryptanalysis on Binary Elliptic Curves. *IEEE Access* **2023**, *11*, 45083–45097. [[CrossRef](#)]
42. Rizk-Allah, R.M.; Abdulkader, H.; Elatif, S.S.; Elkilani, W.S.; Al Maghayreh, E.; Dhahri, H.; Mahmood, A. A Novel Binary Hybrid PSO-EO Algorithm for Cryptanalysis of Internal State of RC4 Cipher. *Sensors* **2022**, *22*, 3844. [[CrossRef](#)] [[PubMed](#)]
43. Jawed, M.S.; Sajid, M. Cryptanalysis of Lightweight Block Ciphers using Metaheuristic Algorithms in Cloud of Things (CoT). In Proceedings of the 2022 International Conference on Data Analytics for Business and Industry (ICDABI), Sakhir, Bahrain, 25–26 October 2022; pp. 165–169.
44. Kennedy, J.; Eberhart, R.C. Discrete binary version of the particle swarm algorithm. *Proc. IEEE Int. Conf. Syst. Man Cybern.* **1997**, *5*, 4104–4108. [[CrossRef](#)]
45. Emary, E.; Zawbaa, H.M.; Hassaniien, A.E. Binary grey wolf optimization approaches for feature selection. *Neurocomputing* **2016**, *172*, 371–381. [[CrossRef](#)]
46. Grari, H.; Azouaoui, A.; Zine-Dine, K. A cryptanalytic attack of simplified-AES using ant colony optimization. *Int. J. Electr. Comput. Eng.* **2019**, *9*, 4287–4295. [[CrossRef](#)]
47. Bizaki, H.K.; Falahati, A. Second Round Mini-AES MC. In Proceedings of the 2006 2nd International Conference on Information & Communication Technologies, Damascus, Syria, 24–28 April 2006; pp. 958–962.

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.