This study considers the process that protects a video stream transmitted from the onboard video camera of an unmanned aerial vehicle (UAV) to a ground station in the front and near-front zones. The specificity of the task set is predetermined, on the one hand, by the limited computing resources of onboard equipment, which must encrypt an intensive data stream in real time, and, on the other hand, by the relatively short life span of UAV under combat conditions (especially for FPV kamikaze drones) ranging from 10 minutes to several days.

Most known works in this field consider algorithms focused on application in other settings, with the main efforts aimed at achieving maximal cryptographic security. Unlike the technological advancements highlighted in the available literature, this study has managed to solve the specified problem by taking into account its specific features and utilizing a certain trade-off between the speed and resource intensity of the algorithm, on the one hand, and its security, on the other.

The result was achieved through a detailed comparative analysis and categorization of the closest solutions in terms of characteristics, which is the largest part of the study. Based on its results, the PRESENT algorithm was chosen as the first approximation.

The proposed solution is based on the use of a modification of this algorithm reduced to 16 rounds in counter mode. Analysis of the resulting solution reveals that its cryptographic security requires more than 2 months of computational work when implementing the best attack, i.e., the security of the algorithm is reasonably acceptable.

For practical application of the theoretical results, it is necessary to carefully check the properties of the proposed solution implemented in the equipment under field conditions, as close as possible to combat operations, and, if necessary, to make the necessary adjustments

Keywords: UAV, video data encryption, temporal cryptographic security, lightweight cryptographic algorithms, block ciphers, PRESENT

UDC 004.056.5:629.7.052
DOI: 10.15587/1729-4061.2025.340917

DEVELOPMENT OF AN ALGORITHM WITH TEMPORARY CRYPTOGRAPHIC SECURITY FOR ENCRYPTING VIDEO STREAM FROM AN UNMANNED AERIAL VEHICLE

Liudmyla Kovalchuk
Corresponding author
Doctor of Technical Sciences, Professor*
E-mail: lusi.kovalchuk@gmail.com
Anatolii Davydenko
Doctor of Technical Sciences, Professor*
Tatiana Klymenko
Department of Scientific and Organizational**
Arina Nedashkivska
Department of Applied Mathematics
National Technical University of Ukraine
"Igor Sikorsky Kyiv Polytechnic Institute"
Beresteyskyi ave., 37, Kyiv, Ukraine, 03056
Serhii Hilgurt
Doctor of Technical Sciences, Senior Researcher
Department of Mathematical Modelling and Econometrics**
*Department of Mathematical and Computer Modeling**
**G.E. Pukhov Institute for Modelling in Energy Engineering
General Naumov str., 15, Kyiv, Ukraine, 03164

Received 11.07.2025
Received in revised form 18.09.2025
Accepted date 29.09.2025
Published date 28.10.2025

How to Cite: *Kovalchuk, L., Davydenko, A., Klymenko, T., Nedashkivska, A., Hilgurt, S. (2025). Development of an algorithm with temporary cryptographic security for encrypting video stream from an unmanned aerial vehicle. Eastern-European Journal of Enterprise Technologies, 5 (9 (137)), 41–53. https://doi.org/10.15587/1729-4061.2025.340917*

1. Introduction

Over the past decades, cryptology has evolved in a competitive manner. Cryptanalysts have found new attacks on existing encryption algorithms, and cryptographers have either improved existing algorithms or developed new ones that use different mathematical objects and are based on different mathematical problems. To withstand the rapid development of cryptanalysis and prolong the lifetime of a new crypto algorithm, algorithms have finally had to be designed with a huge margin of security. Typically, for existing algorithms, the time of the best-known attack (with existing computing power) is greater than the lifetime of the universe. This concept of algorithm construction has been used for more than 50 years, and no one expected it to change.

But with the beginning of the full-scale war unleashed by fascist Russia, there was a need for such types of cryptographic algorithms that existed long ago and to which no one was going to return – cryptographic algorithms with temporal security. These are algorithms that, on the one hand, are secure over a short period of time (from 10 minutes to several days) and, on the other hand, require small resources for their implementation.

And first of all, such cryptographic algorithms are needed for use in unmanned aerial vehicles (UAVs). Video data received from UAVs on the front line contains critically important tactical information about the coordinates, location of units, engineering structures, equipment, etc. If the enemy can intercept control over the unmanned aerial vehicle or at least the signal that it transmits to the operator, then it can see the location of UAV pilots and the launch/landing point. After that, the enemy will strike there with artillery in a matter of seconds. Therefore, protecting traffic transmitted from UAVs is primarily about protecting the military who operate them.

The features of UAVs (in particular, disposable FPV kamikaze drones), associated with the speed, weight, dimensions, and cost characteristics of their onboard electronic equipment, put forward a number of specific requirements for the crypto algorithm that can be used to encrypt video data transmitted to the ground station:

– the requirement for the speed of processing the video stream, i.e., for its encryption/decryption: the delay of the "picture" for the UAV operator should not exceed 0.03 seconds [1, 2];

– the total amount of memory used to store a frame of video data for the implementation of the encryption algorithm should be minimal and, if possible, not exceed units – tens of Mbits [3, 4];

– the algorithm should convert the image in such a way that it is impossible to visually restore the original image from the ciphertext;

– the algorithm should not propagate the error, since during signal transmission, distortions are possible, which in the case of significant error propagation will not allow the original image to be restored.

But the strict requirements for the encryption algorithm are partially compensated by weak requirements for its cryptographic security: for different UAVs, depending on the tasks they perform, the required security can range from 15 minutes to several days. Therefore, research into the analysis and development of low-resource lightweight crypto algorithms with temporary cryptographic security, acceptable for protecting video data transmitted from UAVs, is relevant.

2. Literature review and problem statement

An algorithm that can be used to encrypt video data transmitted from an aircraft to a ground station should belong to the class of so-called lightweight algorithms [5]. Therefore, studies that aim at developing encryption algorithms for UAVs primarily consider such algorithms.

Work [6] considers the issue of authentication in UAVs; it is proposed to use the lightweight asymmetric algorithm HIGHT for this purpose. The authors also note that the Feistel scheme can be used for encryption. However, the work does not provide any specific algorithm that, according to its parameters (memory, speed) could be used for encryption of video data. Such an algorithm must have significantly higher speed, so the results of the cited work cannot be directly used for encryption of a video stream.

Paper [7] explores the possibility of using the AES algorithm under an EAX mode (encrypt-then-authenticate-then-translate) for encryption of video and audio data transmitted from UAVs. In this case, experiments showed good results when using this algorithm for an audio signal, even in the presence of failures and errors in data transmission but showed negative results when applied to video. This can be explained by the insufficient speed of the AES algorithm; when designing it, the priority goal was to develop an algorithm with very high cryptographic security, rather than with increased speed.

In [8], a new lightweight Piccolo algorithm with a block length of 64 bits and a key length of 80 or 128 bits is proposed. This is a generalized Feistel scheme that has four 16-bit branches and performs 25 or 31 rounds (depending on the key length). In terms of memory size, this algorithm is almost identical to the well-known and widely used PRESENT algorithm [9]; its throughput reaches up to 21.54 bytes per second. This is not acceptable for use in encrypting video data from UAVs since the delay time should not exceed 0.03 seconds. Even if we limit ourselves to temporal security and reduce the number of rounds, the algorithm cannot be used for image encryption because of its insufficient speed.

Work [10] also considers using the AES algorithm for encrypting data transmitted from UAVs. It is claimed that Ukrainian PD-2 and Shark UAVs apply this algorithm to transmit telemetry data, such as speed, altitude, coordinates, etc. However, the amount of information contained in this data is significantly less than that contained in video streams. In addition, it does not always need to be transmitted in real time and constantly updated, it can be provided periodically or on request. Therefore, the AES algorithm, which is successfully used for transmitting a small amount of data, is not suitable for encrypting video streams in real time due to insufficient speed.

In [11], a new algorithm is proposed for encrypting wildlife images transmitted from UAVs. This algorithm uses the integration of one-dimensional and two-dimensional cellular automata (1D MCA and 2D MCA), combined with bitwise addition and shuffling operations. The work is more focused on analyzing the security of this algorithm, in combination with a key generation algorithm. However, no estimates are given for either the memory size or the speed of the algorithm, so the question of its application for real-time video data transmission remains open. The authors do not analyze the algorithm for compliance with the speed requirements, and do not provide data about the algorithm on the basis of which such an analysis could be performed. Therefore, it is impossible to determine the feasibility of using such algorithms.

Paper [12] proposes a general cryptographic security framework for small drones that, according to the authors, is energy efficient and "has speed advantages over standard cryptographic methods." In particular, this framework includes public key infrastructure (PKI) and elliptic curve algorithms. However, the paper does not provide any specific algorithms that could be used to encrypt video data, let alone estimates of their memory size and bandwidth.

Another lightweight algorithm, HANK-1, is proposed in [13]. It is also a generalized Feistel scheme with four branches and has only 8 rounds; the authors recommend using it under an SHS mode. It requires 64K bytes of memory and its speed is specified as 84.1468 Kbit/s. This makes it a potential candidate for use in resource-constrained environments. But at the same time, questions arise regarding its security. The authors do not obtain analytical estimates of the security against ~~to~~ attacks but only provide certain heuristic considerations and apply statistical tests from the NIST package [14] for statistical analysis of the output sequences. There are also questions regarding the number of rounds – as a rule, for Feistel schemes, the number of rounds is always greater than for SP networks (usually about 30 rounds), so the Feistel scheme with eight rounds raises reasonable doubts regarding security. Therefore, the security of this algorithm, even temporarily, cannot be considered reasonable, and the use of such an algorithm may contain significant risks.

In [15], a new ultra-lightweight block cipher Sriram is proposed, which could be used in a very limited environment, such as RFID (Radio Frequency Identification) tags and sensor networks. This algorithm is also a Feistel algorithm that uses a basic SP network, in which each round consists of a substitution block, a bit permutation, and a

mixing with a key. It has a block length of 64 bits, performs 27 rounds with a key of 96 or 128 bits. In the conclusions to the work, the authors for some reason discuss the hardware implementation of the PRESENT cipher [9] and its 80-bit version, while the work lacks numerical data on the required memory and bandwidth of the algorithm they proposed. Therefore, the question of its suitability for encrypting video streams remains unresolved.

In [16], the issue of developing encryption algorithms for UAVs is also considered, in particular, the CARX block algorithm is proposed. However, the authors of the work note that the ZUC-128 streaming algorithm proposed in [17] is better suited for encrypting images transmitted from UAVs, in which the average gamma generation rate, encryption rate, and decryption rate reach, respectively, 33.74 kbit/s, 23.31 kbit/s, 24.06 kbit/s. Although these indicators are quite high, without experimental results under actual conditions it is impossible to determine whether they provide the required delay time. In addition, the streaming algorithm requires a long time to initialize at the beginning of work, which makes it (without certain modifications) unsuitable for real-time encryption.

A new family of lightweight block algorithms SHIPHER, based on dynamic operations, is proposed in [18]. Ciphers belonging to this family use operations in different algebraic systems, similar to the IDEA cipher [19]. They can have different key sizes, which makes them convenient for different tasks. The authors claim that the memory capacity and speed of these ciphers are significantly better than AES but do not provide corresponding estimates. In addition, any operations in algebraic systems other than XOR significantly reduce the speed of the algorithm. Therefore, the question of the magnitude of the image transmission delay remains open.

In [20], the problems of using UAVs for agricultural work are analyzed and the problems that arise in this case are considered – for example, the possibility of UAV theft by signal spoofing. To prevent this problem, it is proposed to encrypt traffic from UAVs using the Vernam cipher. The authors investigate gamma generation methods for this cipher in the paper but do not address the question of how suitable this encryption method is in terms of memory capacity and speed. This approach seems problematic for at least two reasons: the gamma generation process in the UAV itself may be too slow and resource-intensive, and there is unlikely to be enough memory in the UAV to record a gamma of sufficient length. If short gamma repetition is not used, then such an algorithm categorically cannot be used under an ambush mode.

Work [21] only raises questions about the features of data transmission from UAVs and justifies the need to use encrypted communication in these channels but does not provide any ways to ensure such communication.

Based on the results of our review of available sources of information, it is possible to define known studies on the cryptographic algorithm for encrypting video data transmitted from UAVs to a ground station within the following categories.

Works [6, 8, 11, 20] consider algorithms for encrypting either static images (photographs), or the UAV control channel, telemetry data, or messages for authentication. In other words, these are algorithms for protecting much smaller amounts of information compared to a continuous stream of video data. Such algorithms are not subject to such strict resource and speed restrictions as video stream encryption algorithms, so they are not the subject of our study.

Several papers [10, 16, 18] describe solutions claimed to be intended for high-speed encryption. Cryptographic secure but resource-intensive algorithms are involved here; there are doubts about the possibility of ensuring the required speed. As a rule, such publications do not provide at all, or do not provide enough information necessary for drawing conclusions about the obtained characteristics of the algorithms.

In [13, 15], the emphasis is on lightweight algorithms that could potentially be used for video stream encryption. Regretfully, these papers also mostly lack information that would allow us to draw unambiguous conclusions about compliance with the requirements imposed on UAV video data transmission systems. And in cases where there is a clear opportunity to further simplify the algorithm for the purpose of acceleration, there is no analysis of the cryptographic security that such simplification would lead to.

In works [7, 12, 21], the authors either report negative results when using algorithms for video data encryption, or avoid specific answers regarding the characteristics obtained, limiting themselves to generalized considerations.

Therefore, no work offers a solution acceptable for UAV video data encryption that meets the stringent requirements formulated in our paper's Introduction. Moreover, no literature reviews were found that would contain an exhaustive study and comparison of such algorithms.

The systematization of the reviewed publications and the shortcomings identified in them leads to the conclusion that there is a still unsolved issue. This problem is the lack of currently acceptable solutions for encrypting the intensive stream of video data transmitted from a UAV to a base station that would meet all the requirements imposed on them.

The specified unresolved issue is caused by objective reasons. In particular, this is the rapid evolution of UAV technologies, primarily of the FPV kamikaze drone type, which has radically changed the paradigm of modern combat. As a result, the requirements for speed, weight, dimensions, and cost characteristics of on-board electronic equipment have significantly increased. On the other hand, the much shorter life span of such UAVs opens up a potential path to improving the aforementioned characteristics of the video data protection system. This can be done by reducing its cryptographic security (while developers of encryption algorithms have traditionally tried to maximize cryptographic security). This necessitates the analysis and development of lightweight encryption algorithms, which, by reducing cryptographic security, would make it possible to meet the stringent requirements for the characteristics of on-board equipment in modern UAVs.

3. The aim and objectives of the study

The purpose of our study is to develop an algorithm that could be used to encrypt video streams transmitted from UAVs, in particular under conditions of military operations. Effective encryption of video streams would not only make it possible to hide the information received by the UAV operator from the enemy but also protect the operator from detecting his/her location, which could ultimately save his/her life.

To achieve the goal, the following tasks were set:

– to conduct a thorough analysis of the properties and characteristics of known algorithms that can potentially be used to encrypt video data transmitted from UAVs to a ground station;

– based on identifying the features of existing algorithms, select the most acceptable prototype and modify it in order to eliminate shortcomings and obtain the desired characteristics;

– to analyze the cryptographic security of the modified algorithm;

– to determine the possibilities of practical use of the modified algorithm for encrypting video data from UAVs.

4. The study materials and methods

The object of our study is the process that protects video streams transmitted from UAVs to ground stations in the front and near-front zones.

The main hypothesis of the study assumes the possibility of achieving the required characteristics of the encryption algorithm through a certain trade-off by reducing the cryptographic security, taking into account the short life span of UAVs under modern combat conditions.

The following assumptions were adopted in the work regarding the basic characteristics of the UAV on-board computer:

– RAM capacity is up to 106 bits;

– allowable signal propagation delay time along the entire video information transmission path is up to 0.3 seconds.

To simplify the analysis and development of the required algorithm, a standard simplification was accepted regarding the satisfactory quality of the information transmission path, which does not require the use of noise-resistant coding.

To perform the analytical part of the study, a mathematical description was proposed that makes it possible to represent the video stream in a form convenient for processing – depending on which encryption algorithm will be used.

For each potential candidate encryption algorithm, an evaluation of its characteristics was performed: the amount of memory required for implementation; speed; key space size. Then, an analysis of its properties was carried out, such as resistance to ciphertext distortion during transmission, object recognition when the image is distorted, as well as a preliminary analysis of cryptographic security.

Based on the results of analysis, for each algorithm, its resource intensity, advantages, and disadvantages were determined, in particular, what risks the use of this algorithm carries under certain conditions. Based on this, an algorithm was proposed that could be used under any UAV operating mode, and a number of other algorithms were recommended for various possible practical applications.

To confirm the adequacy of the theoretical results, experimental methods (to verify the correctness of the expected results) and programming methods (to perform image encryption and decryption) were used. The program was developed in the Visual Studio Code environment; the Python programming language and the following Python libraries were used: NymPy, Pillow(PIL), random.

5. Results of the study on the development of an algorithm for encrypting video data from UAVs

5. 1. Analysis of known algorithms potentially suitable for encrypting video data transmitted from UAVs to ground stations

The notations used are given below.

An image from a UAV video camera (one frame of the video stream) can be represented as a matrix

$$A=\left(a_{ij}\right)_{i,j=1}^{u,v},$$

of dimensionality $u \times v$, where each element a_{ij} corresponds to one pixel of the image, which is located in the i-th row ($i = 1, 2, ..., u$) and j-th column ($j = 1, 2, ..., v$) and is represented as a concatenation of three bytes (one for each color, red-green-blue, RGB).

To represent video data as a sequence of images, the following notation was used

$$A^{(t)}=\left(a_{ij}^{(t)}\right)_{i,j=1}^{u,v},\ t\in N, \quad (1)$$

where t is the number (numbering within one session) of the image and increases with increasing session time.

In some cases, for ease of encryption, the entire image was provided either as a single bit string

$$b^{(t)}=\left(b_1^{(t)},...,b_l^{(t)}\right), \quad (2)$$

where $l=24\cdot uv$ or as a byte string

$$B^{(t)}=\left(B_1^{(t)},...,B_L^{(t)}\right), \quad (3)$$

where $L=3\cdot uv$, rows (2) and (3) are formed as a sequential concatenation of rows in matrix (1) (from the first to the u-th).

Below, a number of algorithms that can potentially be used to encrypt image (1) are described; their preliminary analysis is carried out regarding their suitability for the stated purposes.

Algorithm 1. Simplified gamma overlay.

For this algorithm, representation (2) is used.

Let $\Gamma = \{\Gamma^{(1)}, \Gamma^{(2)},..., \Gamma^{(s)}\}$, for some $s \in N$, be a set of random/pseudorandom bit sequences, each of length l

$$\Gamma^{(t)}=\left(\gamma_1^{(t)},\gamma_2^{(t)},...,\gamma_l^{(t)}\right).$$

The value of number s depends on the amount of memory allocated specifically for the encryption process in the UAV. The encryption algorithm is very simple: the sequence of images $b^{(t)}$ is encrypted by bitwise addition with the corresponding gamma, i.e., the sequence of ciphertexts takes the following form

$$c^{(t)}=b^{(t)}\oplus\Gamma^{\left((t-1)\bmod s+1\right)}. \quad (4)$$

Required resource.

To encrypt only one image, $l = 24 \cdot uv$ bits of memory are required.

Disadvantages in application.

Let the values of parameters u and v be relatively small: $u = 380$, $v = 420$. Then, to encrypt one image, $l = 3830400$ bits of gamma are required, which cannot be entered into the limited memory resource of UAV.

If a gamma with a short period is used, then identical parts of the image located at a certain distance will give identical ciphertexts. In addition, if there are large parts of the image of the same color, one can guess by the ciphertext where the image has the same color. In addition, if the UAV is under an ambush mode for a certain time, the image does not change for a long time (i.e., the sequence of images consists of identical matrices). Therefore, the sequence of ciphertexts will also be the same, from which the attacker will understand that the UAV is not moving. Since any information

about the behavior of the UAV is critical, this type of encryption cannot be used.

Algorithm 2. Permutation on a set of colors – 1.

For this algorithm, it is advisable to use the image representation (3)

$$B^{(t)} = \left(B_1^{(t)},...,B_L^{(t)}\right).$$

Each pixel (or rather, its color) is represented by three bytes, or 24 bits. Therefore, a permutation on the set of colors will have a length of 24.

Let S_{24} be the set of all permutations of length 24, and for an arbitrary permutation $\sigma \in S_{24}$, we define a mapping σ: $V_{24} \to V_{24}$ as

$$\sigma\left(a_1,...,a_{24}\right) = \left(a_{\sigma(1)},...,a_{\sigma(24)}\right). \quad (5)$$

To perform encryption, the plaintext corresponding to the image at a certain point in time can be represented as a sequence

$$M = \left(a_{11}, a_{12},...,a_{1v},a_{21}, a_{22}, ...,a_{2v},..., a_{uv}\right), \quad a_{ij} \in V_{24},$$

where a_{ij} corresponds to the filling of the pixel located in row i and column j. Then the key of this encryption algorithm is the permutation $\sigma \in S_{24}$, and the ciphertext will take the following form

$$C = \left(c_{11}, c_{12},...,c_{1v},c_{21}, c_{22}, ...,c_{2v},..., c_{un}\right),$$

where $c_{ij} = \sigma\left(a_{ij}\right)$.

Required resource.

The key is a permutation of length 24, that is, it consists of 24 elements, each of the elements is a number from 1 to 24 and requires 1 byte to record. Then 24 bytes, or 192 bits, are required to store the key.

Disadvantages in application.

The main disadvantage of this algorithm is that it preserves the same colors: that is, if two or more pixels of the plaintext had the same colors, then after encryption these pixels will also have the same colors. Therefore, the image can be recognized quite well.

In addition to this disadvantage, one should also mention a disadvantage that applies to Algorithm 1: if the image does not change, then the encrypted text does not change either.

Algorithm 3. Permutation on a set of colors – 2.

For this algorithm, it is also advisable to use representation (3).

Let S_8 be the set of all permutations of length 8 and $S = \{\sigma_8,..., \sigma_l\}$ be some sufficiently large subset of this set. This set can be either a parameter of the encryption algorithm or its long-term key. It should be noted that $l \le 8! = 40320$. That is, in order to specify the number of permutation from set S, 2 bytes are enough.

The key of the algorithm will be sequence $K = (k_1, k_2,..., k_{uv})$ of length uv, each element of which consists of two bytes and determines the number of the permutation. The encryption is determined by the following algorithm

$$C = \left(c_1, ... , c_{uv}\right),$$

where $c_i = \sigma_{k_i}\left(B_i\right)$.

Required resource.

To store the key, $2uv$ bytes or $16uv$ bits are required, which at $u = 320$, $v = 420$ is more than 2.5 million bits, which may be greater than the limit that the UAV's onboard equipment can provide.

Disadvantages in application.

The main disadvantage of this algorithm is that it requires significantly more memory than is possible in UAV.

Another disadvantage is the immutability of the ciphertext when the image is immutable.

Algorithm 4. Encryption by a stream algorithm.

For this algorithm, we use representation (2) or (3), depending on the data format with which the selected stream algorithm works.

Let $E_k(M)$ be some stream encryption algorithm, where k is the key, M is the plaintext. Naturally, it is advisable to give preference to lightweight algorithms, for example, such as the Enocoro algorithm [22]. Assuming that the UAV has enough memory to store the key and algorithm parameters, the ciphertext can be written as

$$C = E_k\left(M\right),$$

where plaintext M is a sequence of images obtained during the operation of UAV.

It should be noted that in this case, encryption also occurs by applying a gamma, but this gamma does not need to be stored in memory since it is generated according to the encryption algorithm using a short (up to 256 bits) key. If the period of such a gamma is large enough (from 2^{100}), then it will be enough with a large margin to encrypt all images during one session, and this is a significant advantage of this algorithm. Each new image will be encrypted with a different gamma, and identical images, generally speaking, will have different ciphertexts.

Required resource.

Below, as an example, the amount of memory required to implement the Enocoro algorithm is calculated.

The algorithm uses substitution block s_8, which is a bijective mapping of s_8: $V_8 \to V_8$. In the implementation, this mapping can be specified in two ways:

Method 1: with a substitution table (by permuting numbers from 0 to 255, which requires 256 bytes or approximately $2{\cdot}10^3$ bits).

Method 2: as superposition of some 4-bit substitution block and a 2×2 matrix over field F_4, which will need to be calculated (in this case, memory is saved, but speed is lost).

The algorithm also uses matrix $L = \begin{pmatrix} 1 & 1 \\ 1 & d \end{pmatrix}$ over field F_8, where element d can be represented as $d = 0{\times}02 = (00000010)_2 = x$. This is another 4 bytes.

Next, the algorithm uses constants C_0, ..., C_9 given as follows:

$$C_0 = 0{\times}66, \quad C_1 = 0{\times}e9, \quad C_2 = 0{\times}4b, \quad C_3 = 0{\times}d4,$$

$$C_4 = 0{\times}ef, \quad C_5 = 0{\times}8a, \quad C_6 = 0{\times}2c,$$

$$C_7 = 0{\times}3b, \quad C_8 = 0{\times}88, \quad C_9 = 0{\times}4c.$$

This is another 10 bytes. But these constants are needed only at the key initialization stage, and if this stage is not performed (or performed "on the ground"), then they do not need to be stored.

The algorithm also uses calculations in a finite field, that is, division by a polynomial of the 8th power over field F_2. Such an operation is quite slow. If instead of performing the

operation, a table stored in memory is used, then such a table requires $\frac{2^8 \cdot 2^8}{2} \cdot 8 = 2^{18} \approx 2.5 \cdot 10^5$ bits.

In addition, 32 bytes of state and 2 bytes of output must be remembered at each clock cycle, and about 40 bytes will be needed to implement the state update function and the output update function.

As a result, the amount of memory required depends on the implementation and is approximately $2.5 \cdot 2^{11} + 2 \cdot 10^3 + 32 \cdot 8 + 2 \cdot 8 + + 40 \cdot 8$, which is slightly less than 10^6 and can be considered acceptable. An alternative that will make it possible to reduce the amount of memory is to perform operations in a finite field, which are quite slow, and in this case the running time of the algorithm may exceed the acceptable limits of the delay. In any case, before using the algorithm, it is necessary to investigate not only the amount of memory required but also its running time – if it is more than 0.03 seconds, then there will be a significant delay in image processing, and such an algorithm will be unacceptable.

It should be noted that the advantage of this algorithm is the full justification of its security [22].

Disadvantages in application:

1. All stream ciphers have a rather long key initialization procedure, which is performed before the start of encryption. For example, for the Enocoro cipher, it consists of 96 encryption cycles. Such a delay before the start of encryption is unacceptable.

But this drawback can be partially corrected. The initialization phase can be performed before the start of UAV operation, after which the state obtained as a result of the initialization is written to the initial state of the algorithm. After that, the UAV starts its operation, and encryption begins immediately since the initialization has already been performed.

2. The need for a sufficiently large (but not critically large) volume of memory (depending on the specific algorithm).

3. The algorithm will encrypt only a part of the image in one cycle (for example, Enocoro will encrypt only 1 byte). That is, for encryption to occur without delay, the algorithm must perform $L = 3 \cdot uv$ cycles of work in 0.03 seconds, which is quite difficult to achieve.

4. It is necessary to solve a number of issues regarding resynchronization, namely whether it is necessary to resynchronize every time for a new message; how the synchronization packet will change (there are two possible options: either under the counter mode, or the next one is transmitted in the current message); how to detect an error consisting in the loss of part of the message, or a transmission error, and how to correct it in order to restore synchronization.

Algorithm 5. Encryption by a block algorithm under the ECS, OFB, or counter modes.

For this algorithm, it is also advisable to use representation (2) or (3), depending on the data format with which the algorithm works.

Let $E_k(M)$ be the result of applying the block algorithm E in the ECS mode with key k to plaintext M. Using representation (2), it is possible to divide message M into blocks according to the block length of the encryption algorithm (with the addition of padding)

$$M = M_1 M_2 ... M_l.$$

Then in the ECS mode the ciphertext C takes the following form

$$C = C_1 C_2 ... C_l,$$

where $C_i = E_k(M_i)$.

In OFB and counter modes, the encryption will take a more complex form; these transformations are described in detail in [23].

Required resource.

If we consider a "typical" block encryption algorithm, then a sufficient amount of memory is required to store the key, s-blocks, and linear transformation. For example, if 8-bit s-blocks are used, then in order to store one s-block, it is necessary to store 2^8 elements of 8 bits each, i.e., 2^{11} bits. Accordingly, if there are more different s-blocks, then this value must be multiplied by the number of different s-blocks. The amount of memory required for linear transformation depends significantly on the type of this transformation. For example, for the PRESENT algorithm [7], this is a permutation of length 64, and the amount of memory, respectively, is $64 \cdot 8 = 2^9$ bits. To store the key, from 80 to 512 bits are required. In conclusion, the amount of memory required to implement a lightweight block encryption algorithm is reasonably acceptable.

Disadvantages of application:

1. The block cipher has such a feature that changing even one bit in the input message block leads to a change of about half the bits (on average) in the ciphertext. If, for example, the block length is 64 bits (PRESENT), then when using the ESV mode, an error in even one bit when transmitting the ciphertext will lead to color distortion in three pixels during decryption, and under the OFB mode – in six pixels.

It should be noted that this drawback is absent when using the block algorithm under the counter mode since this mode is actually a bit-by-bit gamut overlay.

2. In ECB mode, identical blocks of plaintext are converted into identical blocks of ciphertext, which can potentially provide information about the original image.

In the OFB mode and the counter mode, this drawback is absent.

3. Even for such a lightweight algorithm as PRESENT, the operation time of the full-round algorithm does not meet the requirements for image delay. Therefore, it is most likely necessary to use a truncated version of this algorithm – for example, 16 rounds instead of 32. To do this, it is necessary to assess the security of such a truncated version of the algorithm and make sure that it has sufficient security, at least temporarily – from several hours to several days.

Algorithm 6. Permutations of rows and columns.

For this algorithm, it is advisable to use the representation of the image as matrix $A = (a_{ij})_{i,j=1}^{u,v}$. Let $\sigma_R \in S_u$ and $\sigma_C \in S_v$ be the permutations according to which the rows or columns in the matrix are permuted. This pair of permutations is the key to this algorithm. Then the corresponding ciphertext will take the following form

$$C = (c_{ij})_{i,j=1}^{u,v},$$

where $c_{ij} = a_{\sigma_R(i),\sigma_C(j)}$.

Required resource.

To store the key, one needs (given that a byte element base is used):

– u elements of bit length $\left\lceil \frac{\log u}{8} \right\rceil \cdot 8$;

– v elements of bit length $\left\lceil \frac{\log u}{8} \right\rceil \cdot 8$.

If, for example, $u = 500$ and $v = 600$, then in total one needs $500 \cdot 2 \cdot 8 + 600 \cdot 2 \cdot 8 = 17600$ bits of memory, which is quite an acceptable value.

It should be noted that such an algorithm is tolerant to distortions when transmitting the ciphertext. Fig. 1 shows various variants of distortions that occurred when transmitting the ciphertext. Fig. 1 demonstrates that even with significant distortions, the image is recognizable.

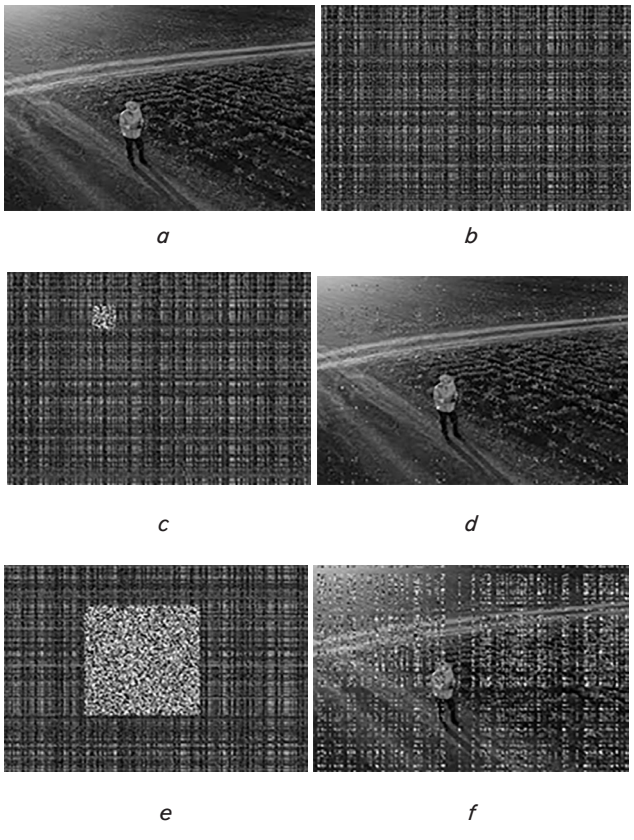

Fig. 1. Decrypted image with distorted ciphertext: *a* — initial image; *b* — encrypted image; *c* — distortion during ciphertext transmission (20 × 20 pixels); *d* — decryption of ciphertext from sub-drawing *c*; *d* — distortion during ciphertext transmission (100 × 100 pixels); *f* — decryption of ciphertext from sub-drawing *e*

Disadvantages of application:

1. With this encryption algorithm, the elements of the matrix that were in the same row will also be in the same row after encryption. Similarly, the elements of the matrix that were in the same column will also be in the same column after encryption. Therefore, wide single-color stripes, vertical or horizontal, can be stored (Fig. 2).

2. If the original image does not change, then the ciphertext will not change either. In addition, if the image has changed insignificantly, then the ciphertext will also have minimal changes. This can provide the enemy with information that the UAV is under an ambush mode or flying for a long time over the same landscape or circling over some point (Fig. 3).

Algorithm 7. Row and column permutations with selective bit inversion.

This algorithm is a complication of Algorithm 6. The key to this algorithm, in addition to two permutations $\sigma_R \in S_u$ and $\sigma_C \in S_v$, is also a bit gamma of length $l = z \cdot u \cdot v$, written as a matrix

$$\Gamma = \left(\gamma_{ij}\right)_{i,j=1}^{u,v},$$

where $\gamma_{ij} \in V_z$, and the value of z is determined by the allowable memory size (which can be roughly considered to be equal to 10^6 bits). After applying the row and column permutations, a selective bit inversion is also applied to each element of matrix $C = \left(c_{ij}\right)_{i,j=1}^{u,v}$ according to the following algorithm:

– vector γ_{ij} is converted into the corresponding integer π_{ij} from 0 to $2^9 - 1$;

– in the filling of the pixel with number i,j, that is, in vector $c_{ij} \in V_{24}$, the bit with serial number π_{ij} is inverted, counting from left to right (some other principle of selecting the bit for inversion is also possible, depending on number π_{ij} – for example, counting from right to left, or not from the first bit, but from the 9th, etc.).

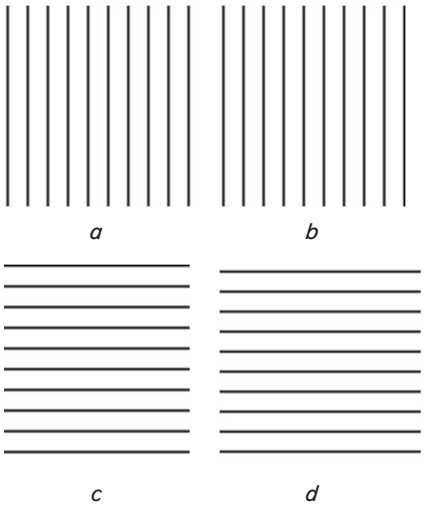

Fig. 2. Ciphertexts obtained from images containing vertical or horizontal lines: *a* — initial image containing vertical lines; *b* — encrypted image *a*; *c* — initial image containing horizontal lines; *d* — decryption of ciphertext from the sub-drawing *c*

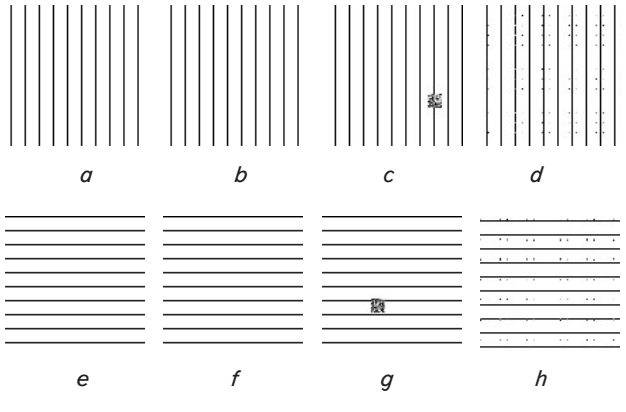

Fig. 3. Ciphertexts obtained with a slight change in the original image: *a* — original image; *b* — encrypted image *a*; *c* — image obtained with slight changes (10 × 10 pixels) of image *a*; *d* — encrypted image *c*; *e* — original image; *f* — encrypted image *e*; *g* — image obtained with slight changes (10 × 10 pixels) of image *e*; *h* — encrypted image *g*

It should be noted that instead of inversion, other transformations can be used – for example, changing the order of the bits.

With this encryption technique, single-color stripes will not be stored.

Required resource.

Similar to Algorithm 6, to store the key, u elements of bit length $\left\lceil \frac{\log u}{8} \right\rceil \cdot 8$ and v elements of bit length $\left\lceil \frac{\log v}{8} \right\rceil \cdot 8$ are required, and $l = z \cdot u \cdot v$ of gamma are also required. If, for example, $u = 500$ and $v = 600$, then, taking into account that the permutations need 17600 bits of memory, we can choose the maximum allowable value $z = 600$, and then, to record the gamma, we need $l = 3 \cdot 500 \cdot 600 = 900000$ bits of memory. Then the integers corresponding to the gamma elements can take values from 0 to 7, that is, the inversion can act on one of the first eight bits (which are numbered from 0 to 7). A more complex method of inversion can be proposed when each of the three bits of vector γ_{ij} controls the inversion in the corresponding pixel filling byte. For example, vector 011 means that in the first byte the first bit is inverted, and in the second and third – the second bit.

It should be noted that this algorithm is also tolerant to distortions when transmitting the ciphertext (Fig. 4).

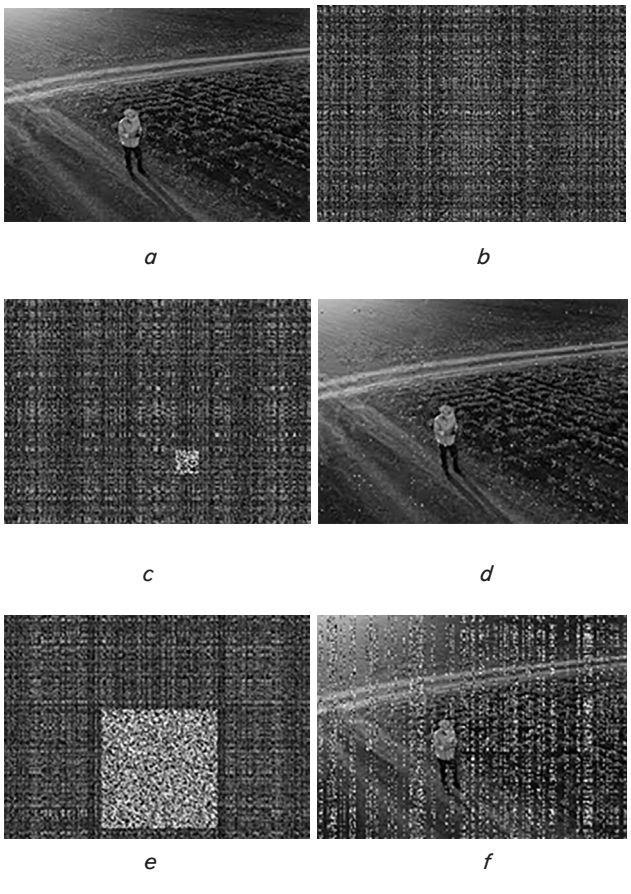

Fig. 4. Decrypted image with distorted ciphertext: *a* — initial image; *b* — encrypted image; *c* — distortion during ciphertext transmission (20 × 20 pixels); *d* — decryption of ciphertext from sub-drawing *c*; *e* — distortion during ciphertext transmission (100 × 100 pixels); *f* — decryption of ciphertext from sub-drawing *e*

Disadvantages in application.

Since the memory capacity does not make it possible to store a gamma of such a length that it would be enough for all consecutive images, but only for one image, the same drawback appears as in Algorithm 6. That is, if the original image does not change, or changes minimally, then the ciphertext will also not change or will change minimally (Fig. 5).

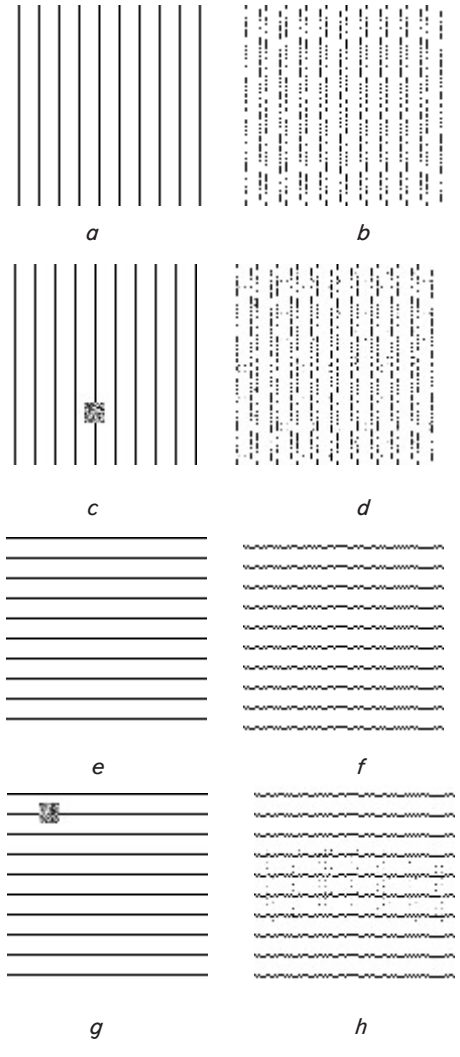

Fig. 5. Ciphertexts obtained with a slight change in the original image: *a* — original image, *b* — encrypted image *a*; *c* — image obtained with slight changes (10 × 10 pixels) of image *a*; *d* — encrypted image *c*; *e* — original image, *f* — encrypted image *e*; *g* — image obtained with slight changes (10 × 10 pixels) of image *e*; *h* — encrypted image *g*

5. 2. Selection of a prototype and its modification to obtain the required characteristics

According to the results of our analysis of the above algorithms, the following conclusions can be drawn regarding their properties:

1. When using such an encryption method in which identical images (frames) correspond to identical ciphertexts, the enemy can obtain sensitive information about the movement of UAV. Therefore, such a technique cannot be considered completely safe, and it should be used with caution, aware of all potential risks.

2. An encryption technique devoid of the above drawback is possible only if a gamma source with a very large period (for example, 2^{100} and more) is used, which can be implemented with a small amount of memory and at an acceptable speed.

3. The possibility of using stream encryption algorithms, even lightweight ones, requires further research using practical implementations. Thus, since it is necessary to achieve acceptable speed, the amount of memory may become critical, and vice versa.

4. The use of block encryption algorithms, even lightweight ones, is possible only for their truncated versions.

5. For block encryption algorithms, it is possible to use only the counter mode since other modes significantly propagate the error when transmitting the ciphertext.

Based on the above considerations, the most promising algorithm as a prototype for developing a specific encryption algorithm that would meet all the stringent requirements is Algorithm 5. This is encryption by a block algorithm in the ECV, OFB, or counter modes.

Regarding the non-propagation of the error, which is one of the specific requirements for a crypto algorithm, the following should be noted.

Of the modes listed above, only the counter mode meets this limitation since it actually implements bitwise gamma overlay. The counter mode also lacks the disadvantage of being able to visually even approximately restore the original image (for example, the horizon separating the terrestrial part from the celestial hemisphere).

As a specific block algorithm, the low-resource lightweight PRESENT algorithm seems to be the most acceptable. But when directly using the full-round version of this algorithm, as shown by calculations, the system does not meet the speed requirements imposed on the video data transmission path. Therefore, in our study, it is proposed to use a truncated version of this algorithm, in which the number of rounds is reduced from 32 to 16 or less. Naturally, the possibility of reducing the number of rounds must have a justification related primarily to ensuring sufficient reliability. The following subchapter analyzes the cryptographic security of the truncated to 16-round version of the PRESENT algorithm.

The version of the PRESENT algorithm modified in the above-described way, operating under a counter mode, in which the number of rounds is exactly 16, will be termed L16-PRESENT (here, L stands for Light).

5. 3. Analyzing the cryptographic security of the L16-PRESENT algorithm to basic attacks

In what follows, we define the indicators of the L16-PRESENT security against basic types of cryptanalytic attacks.

First, it should be noted that the substitution block of the PRESENT algorithm is constructed from such *s*-blocks that satisfy the following requirements [9].

1. For an arbitrary non-zero difference $\Delta_I, \Delta_O \in F_2^4$, the condition is satisfied

$$\#\left\{x \in F_2^4 \mid S(x) \oplus S(x \oplus \Delta_I) = \Delta_O\right\} \le 4.$$

2. For any non-zero difference $\Delta_I, \Delta_O \in F_2^4$, such that $wt(\Delta_I) = wt(\Delta_O) = 1$, the condition is met

$$\left\{x \in F_2^4 \mid S(x) \oplus S(x \oplus \Delta_I) = \Delta_O\right\} = \varnothing.$$

3. For all non-zero $a, b \in F_2^4$, $\left|S_b^W(a)\right| \le 8$, is satisfied, where the Fourier coefficient $S_b^W(a)$ is determined from the following formula

$$S_b^W(a) = \sum_{x \in F_2^4} (-1)^{\langle b, S(x)\rangle \oplus \langle a, x\rangle}.$$

4. For all non-zero $a, b \in F_2^4$, such that $wt(a) = wt(b) = 1$, $S_b^W(a) = 4$. is met.

These four requirements are essential for substantiating the cryptographic security of the PRESENT algorithm and its reduced versions.

Differential cryptanalysis.

A consequence of requirement 2 and the construction of the *Perm* permutation is the following theorem, formulated and proved in [9].

Theorem 1. Each 5-round difference characteristic of the PRESENT algorithm has at least 10 active *s*-blocks.

Using this theorem, as well as requirement 1 for *s*-blocks, and taking into account that there is at least 1 active *s*-block in each round, we can obtain an upper bound on the probability of the differential characteristic for L16-PRESENT as

$$\left(2^{-2}\right)^{10\frac{15}{5}+1} = 2^{-64}. \quad (6)$$

This means that for a successful attack, about 2^{64} pairs of plaintexts (and the same number of encryptions) are required, which is larger than the set of all possible plaintexts, since the block length is 64 bits, and a total of 2^{64} different texts are possible. That is, an attack using differential cryptanalysis is practically impossible.

Linear cryptanalysis.

A consequence of requirements 3 and 4 and the construction of the *Perm* permutation is the following theorem, formulated and proved in [9].

Theorem 2. The maximum deviation ε_{4R} for an arbitrary 4-round linear approximation of the PRESENT algorithm does not exceed $\varepsilon_{4R} \le 2^{-7}$.

Using this theorem, we can obtain an upper bound on the deviation of the 16-round linear approximation for L-PRESENT as

$$\left(2^{-7}\right)^{\frac{16}{4}} = 2^{-28}. \quad (7)$$

This means that a successful attack requires about 2^{56} plaintexts/ciphertexts (and the same number of encryptions). This value is less than the entire set of plain texts, which means that the algorithm can potentially be vulnerable to such an attack.

The time required for its partial execution, i.e., for recovering the key of the last round, is no less than the time required for performing 2^{56} encryptions. In this case, no more than 64 bits of the 80-bit key will be recovered. To recover the remaining 16 bits, it is necessary to perform an enumeration of all options. Therefore, the time during which the algorithm remains secure depends on the computational resources available to the attacker (i.e., how many encryptions s/he can perform in time unit).

It should be noted that this is a lower estimate of the time required for key recovery, while in practice, as a rule, it takes significantly more time.

Attacks on the structure.

Such attacks are mostly applicable to AES-like algorithms that have a clearly defined byte structure. However, in the L16-PRESENT algorithm, the linear transformation uses bit operations, and the *s*-blocks perform operations with tetrads. Therefore, such attacks are not applicable to it.

Algebraic attacks.

In the L16-PRESENT algorithm, a system of 21 quadratic equations with 8 input/output bit variables over field F_2 is required to specify a block. Therefore, to describe the entire

L16-PRESENT algorithm, at least $21 \cdot n$ quadratic equations with $8 \cdot n$ variables are required, where is the number of all s-blocks involved in the algorithm and in the key generation process. In this case

$$n = 16 \cdot 16 + 16 = 272, \quad (8)$$

i.e., a system of quadratic equations with 5712 equations and 2176 variables will be built.

The problem of solving a nonlinear system of equations is NP-hard. Even taking into account that such a system will be sparse, at the moment there are no effective methods to solve it with a solution that could be applied in practice.

Attacks on key schedule.

As a rule, such attacks use certain types of dependences between round keys or their bits. To prevent this, the authors of PRESENT have developed a key schedule algorithm that has the following properties:

– all bits of the key register are nonlinear functions of the 80 bits of the encryption key;

– after the 21st round, each bit of the key register depends on at least 4 bits of the key;

– after the 31st round, 6 bits of the key are polynomials of power 2 of the encryption key bits, 24 bits are polynomials of power 3, all the rest are polynomials with powers 6 to 9.

For the reduced version, the degrees of the corresponding polynomials will be smaller, but still the dependence remains nonlinear, so attacks on the key schedule currently seem to be practically infeasible.

Integral cryptanalysis.

This type of cryptanalysis is the best for the 8-round reduced version of the PRESENT algorithm [24, 25]. To successfully attack this version of the algorithm with a 128-bit key, $2^{24.3}$ plaintexts/ciphertexts are required, the required memory size is 2^{77} bits; and $2^{100.1}$ operations are required. Accordingly, for the 16-round version, all these indicators can only increase, i.e., the 16-round version is secure against integral cryptanalysis.

Thus, studies of the cryptographic security of the L16-PRESENT algorithm have shown that the best attack on this algorithm is a differential attack. A full description of the resources required for this attack can be found in [24, 26]: 2^{64} plaintext/ciphertext pairs; 2^{32} MB of memory; and 2^{64} operations. For a "standard" laptop with a 3.5 GHz processor capable of performing 3.5 billion operations per second, this would take more than $5 \cdot 10^9$ seconds, or more than 167 years.

Even if an advanced computer were used, 1000 times faster than a standard laptop, this attack would take more than 2 months. This is a reasonable time for a situation where only a few days of temporary security is sufficient. Therefore, the L16-PRESENT algorithm (in a counter mode) is a promising candidate for UAV application, but only if hardware is used that will provide acceptable encryption time.

5. 4. Possibilities of practical application of the L16-PRESENT algorithm

The main condition for applying the L16-PRESENT algorithm proposed in our study is its use under a counter mode. This will provide the following advantages when using it:

– fast encryption, performed by gamma overlay;

– a large gamma period, which will provide sufficient resistance to gamma overlap attacks;

– self-synchronization when the transmitted encrypted message is distorted, which will reduce error propagation;

– the impossibility of detecting identical (or very similar) images, which is especially important for operation under an ambush mode.

The use of the L16-PRESENT algorithm under other modes is permissible only in the following cases.

First, these are situations when the transmitted image does not have large fragments with characteristic shapes or clearly defined contours (for example, the horizon separating the earth's surface from the open space above it). Otherwise, even an encrypted video stream will allow the enemy, albeit very approximately, to recognize what the UAV video camera "sees".

Second, the use of the algorithm not under a counter mode is permissible only under conditions of an ultra-low level of radio interference or with the use of additional noise-resistant coding. Under other conditions, the opposite situation will be observed when, due to multiple propagation of errors from distortion of the input information, the video image will be so corrupted that the drone operator will not be able to control UAV normally.

In cases where the limitations on the computing resources of the on-board equipment are particularly critical, it is recommended to use an even more reduced version of the algorithm, for example with a reduced number of rounds to 8 or 12. But there will no longer be a significant margin of temporary security. Therefore, before making such a decision, it is recommended to analyze in detail not only the complexity of the attack but also the enemy's computing resources that it can use.

The temporary cryptographic security provided by the L16-PRESENT algorithm proposed in this study requires an attacker to spend at least 2 months of computing on a powerful computer to brake encrypted data. It should be noted that such security is required only in some cases of UAV use. First of all, under an ambush mode, when the drone lands on the ground and is under standby mode from several hours to several days before being used for an attack. In cases where the ambush mode is guaranteed not to be used, and the lifetime of the FPV drone is reduced to units – tens of minutes, we can recommend the use of other, simpler algorithms. For such situations, Algorithm 6 (row and column permutations), or Algorithm 7 (with additional selective bit inversion), or their modifications, are reasonably acceptable.

For applications that require a slightly longer UAV lifetime – up to several hours (for example, during aerial reconnaissance), image permutation algorithms can be used, or various combinations of varieties of such algorithms and the above-mentioned row and column permutation algorithms.

6. Results of the comparative analysis and development of video data encryption algorithms: discussion

We have determined that the most promising algorithm for encrypting video data from UAVs is the L16-PRESENT algorithm, which is a modification of the PRESENT algorithm reduced to 16 rounds, operating in a counter mode. Algorithm 6 and Algorithm 7 could also be used with certain restrictions, depending on the UAV operating mode. This choice is due to the following arguments:

1. According to Theorem 1, Theorem 2, formulas (6) to (8), as well as the results reported in [24–26], the reduced version of this algorithm has sufficient temporal security. Thus, it takes at least 2 months to break it, which is much

longer than the time of one UAV flight. At the same time, it is recommended to use this algorithm in a counter mode since it is this mode that makes its use universal.

2. According to the results of our work (Algorithm 5, analysis of the required resource), the amount of memory required to implement this algorithm does not exceed 2^{10} bits, which is a completely acceptable value even for a device with very limited resources.

3. According to preliminary experiments, the running time of the algorithm is also acceptable.

4. Fig. 1–5 demonstrate that algorithms based on column and row permutations (even with partial gamma overlap) cannot be applied in the case when the UAV is used under an ambush mode. However, these algorithms can be applied with certain restrictions on the UAV operating mode – for example, if its operating mode does not involve long delays, but rather fast movement for a short time. These algorithms are convenient for performing such tasks in the sense that they were purposefully developed for image encryption.

The features of our research are as follows:

1. Unlike [6, 7] and [10, 11], in this work the research immediately began with the selection of algorithms suitable specifically for encrypting video streams, instead of trying to adapt the algorithms used for encrypting short messages for this purpose.

2. Unlike [8, 13, 15, 18], in this work the requirements for cryptographic security of the encryption algorithm were reduced – only temporal security is required. Due to this, it turned out to be possible to consider a reduced version of the lightweight algorithm, increasing its speed.

It should be noted that the mathematical descriptions used for analytical calculations, like any mathematical abstractions, have certain simplifications and limitations compared to the real world. In particular, the use of initialization vectors was not detailed. The possibility of correcting errors using noise-resistant coding was also not considered, that is, work under conditions of electronic warfare was not considered. These limitations must be taken into account during the practical application of the research results under field conditions or during further research in specific and related areas.

The main drawback of this work is the insufficient amount of empirical data for the final confirmation of the principal hypothesis and the correctness of theoretical results obtained. This is due to the complexity of conducting experimental research under conditions that are as close as possible to combat. Therefore, the next step in further research should be the organization of a large number of such experiments. It should be noted that based on the results of these experiments, some correction of the proposed algorithm is possible – for example, further reducing the number of rounds or using the "wide trail" method, which makes it possible to reduce the number of *s*-blocks of the algorithm. In addition, future work may investigate the possibilities of combining the selected algorithm with noise-resistant coding to improve its functioning under the conditions of using electronic warfare means.

Although the theoretical analysis revealed a clear advantage of the L16-PRESENT algorithm over other candidates, it should be noted that the final decision on choosing a suitable algorithm can be made only after a significant number of practical experiments. This is due to the fact that when implementing the algorithm for the on-board equipment of UAV, certain aspects may emerge that are difficult to predict in theoretical technological advancements.

Therefore, conducting experimental research is a highly relevant task, albeit quite complex, tackling which requires the cooperation of qualified specialists from various fields – mathematicians, programmers, drone pilots, etc.

It should also be noted that in the event of the emergence of a new lightweight block algorithm, which in a certain indicator (resource, speed, cryptographic strength) will be significantly better than PRESENT, it is necessary to investigate the possibility of its application.

Regarding those algorithms that use different types of permutations, their only drawback is the ability to visually reproduce the original image (for example, the horizon line) in general contours. This poses certain risks for aircraft operators, especially when launching and landing UAVs. This drawback could be corrected by introducing an additional transformation into the algorithm, which, on the one hand, would be very lightweight and, on the other hand, could make significant changes to the ciphertext. This issue also requires further research, including experimental part.

The next most promising algorithm for encrypting video data from UAVs after L16-PRESENT, Algorithm 6, and Algorithm 7 is likely the Enocoro algorithm. It also requires further experimental research, the results of which may lead to its surpassing the aforementioned cryptographic algorithms.

7. Conclusions

1. We have conducted a thorough comparative analysis of the properties and characteristics of known algorithms that could potentially be used to encrypt video data transmitted from UAVs to ground stations. In this case, attention was paid primarily to lightweight encryption algorithms and their ability to meet the stringent requirements for the characteristics of modern UAV on-board equipment by reducing cryptographic security. This allowed us to select four algorithms (designated in the work as Algorithm 5, Algorithm 6, and Algorithm 7, as well as the Enocoro algorithm), which, according to their characteristics, turned out to be the most suitable for solving the main task of the study.

2. Based on the results of our analysis, the generalized Algorithm 5 (block algorithm encryption in ECB, OFB, or counter modes) was selected as the basic prototype whose characteristics turned out to be the closest to the requirements for crypto algorithms for encrypting video data from UAVs. Since the known implementations of this algorithm, among which the lightweight PRESENT algorithm was the best, still do not fully satisfy the mentioned requirements, a modification of this algorithm was proposed. The modification involves reducing the number of rounds to 16 and using the counter mode. The resulting L16-PRESENT algorithm turned out to be universal – suitable for use in UAVs under all modes of its operation. Algorithm 6, based on permutations of rows and columns of the video image, as well as similar Algorithm 7 with additional selective bit inversion, were recognized as less promising but acceptable under some modes of UAV operation according to the same criteria.

3. We have analyzed security of the proposed modified algorithm L16-PRESENT against basic cryptanalytic attacks. It has been found to have temporary security sufficient for the use of UAVs with the corresponding video communication system under all known modes of their operation. At

the same time, the algorithm satisfies the restrictions put forward to its resource intensity.

4. The possibilities of practical use of the modified algorithm L16-PRESENT and other algorithms – prototypes for encrypting video data from UAVs – have been considered. That allowed us to determine in which specific situations and modes of combat use of UAVs it is advisable to use one or another algorithm.

Conflicts of interest

The authors declare that they have no conflicts of interest in relation to the current study, including financial, personal, authorship, or any other, that could affect the study, as well as the results reported in this paper.

Funding

The research was conducted with financial grant support from the National Research Foundation of Ukraine within the framework of project No. 2025.06/0109 "Secure jamming-resistant system for transmitting video information from an unmanned aerial vehicle." State registration No. 0125U003164.

Data availability

All data are available, either in numerical or graphical form, in the main text of the manuscript.

Use of artificial intelligence

The authors confirm that they did not use artificial intelligence technologies when creating the current work.

Acknowledgments

The authors express their sincere gratitude to the National Research Foundation of Ukraine for its financial support, which enabled the research within the framework of project No. 2025.06/0109 "Secure jamming-resistant system for transmitting video information from an unmanned aerial vehicle".

References

1. Cox, J., Wong, K. (2019). Predictive feedback augmentation for manual control of an unmanned aerial vehicle with latency. International Journal of Micro Air Vehicles, 11. https://doi.org/10.1177/1756829319869645
2. Kamtam, S. B., Lu, Q., Bouali, F., Haas, O. C. L., Birrell, S. (2024). Network Latency in Teleoperation of Connected and Autonomous Vehicles: A Review of Trends, Challenges, and Mitigation Strategies. Sensors, 24 (12), 3957. https://doi.org/10.3390/s24123957
3. ECP5/ECP5-5G. Family Table. ECP5 and ECP5-5G Device Selection Guide. Lattice Semiconductor. Available at: https://www.latticesemi.com/Products/FPGAandCPLD/ECP5
4. 7 Series FPGAs Data Sheet: Overview. DS180 (v2.6.1) (2020). Product Specification. AMD XILINX. Available at: https://docs.amd.com/v/u/en-US/ds180_7Series_Overview
5. Ashrif, F. F., Sundararajan, E. A., Ahmad, R., Hasan, M. K., Yadegaridehkordi, E. (2024). Survey on the authentication and key agreement of 6LoWPAN: Open issues and future direction. Journal of Network and Computer Applications, 221, 103759. https://doi.org/10.1016/j.jnca.2023.103759
6. Ismael, H. M., Al-Ta'i, Z. T. M. (2021). Authentication and Encryption Drone Communication by Using HIGHT Lightweight Algorithm. Turkish Journal of Computer and Mathematics Education, 12 (11), 5891–5908. Available at: https://www.researchgate.net/publication/392193717_Authentication_and_Encryption_Drone_Communication_by_Using_HIGHT_Lightweight_Algorithm
7. Cecchinato, N., Toma, A., Drioli, C., Oliva, G., Sechi, G., Foresti, G. L. (2022). A Secure Real-time Multimedia Streaming through Robust and Lightweight AES Encryption in UAV Networks for Operational Scenarios in Military Domain. Procedia Computer Science, 205, 50–57. https://doi.org/10.1016/j.procs.2022.09.006
8. Rahiyanath, T. Y. (2015). A Novel Architecture for Lightweight Block Cipher, Piccolo. International Journal of Research in Engineering and Technology, 04 (09), 97–103. Available at: https://ijret.org/volumes/2015v04/i09/IJRET20150409017.pdf
9. Bogdanov, A., Knudsen, L. R., Leander, G., Paar, C., Poschmann, A., Robshaw, M. J. B. et al. (2007). PRESENT: An Ultra-Lightweight Block Cipher. Cryptographic Hardware and Embedded Systems - CHES 2007, 450–466. https://doi.org/10.1007/978-3-540-74735-2_31
10. Safronov, T. (2024). AES-256: V Ukrspecsystems rozkryly detali shyfruvannia danykh BPLA Shark. Militarnyi. Available at: https://militarnyi.com/uk/news/zahyst-danyh-vid-bpla-shark-posylyly-shyfruvannyam-aes-256/
11. Belazi, A., Migallón, H. (2024). Drone-Captured Wildlife Data Encryption: A Hybrid 1D–2D Memory Cellular Automata Scheme with Chaotic Mapping and SHA-256. Mathematics, 12 (22), 3602. https://doi.org/10.3390/math12223602
12. Ozmen, M. O., Yavuz, A. A. (2018). Dronecrypt - An Efficient Cryptographic Framework for Small Aerial Drones. MILCOM 2018 - 2018 IEEE Military Communications Conference (MILCOM). https://doi.org/10.1109/milcom.2018.8599784
13. Eldeeb, H., Shehata, K., Shaker, N., Abdel Hafez, A. (2012). HANK-1, A new Efficient and Secure Block Cipher Algorithm for Limited Resources Devices. The International Conference on Electrical Engineering, 1–12. https://doi.org/10.21608/iceeng.2012.30662
14. Bassham, L. E., Rukhin, A. L., Soto, J., Nechvatal, J. R., Smid, M. E., Barker, E. B. et al. (2010). A statistical test suite for random and pseudorandom number generators for cryptographic applications. National Institute of Standards and Technology. https://doi.org/10.6028/nist.sp.800-22r1a
15. Ramudu, S., Shanthi, G. (2015). Implementation of an Ultra-Lightweight Block Cipher. International Journal & Magazine of Engineering, Technology, Management and Research, 2 (2), 233–242. Available at: http://www.ijmetmr.com/olfebruary2015/SriRamudu-GShanthi-39.pdf

16. Yang, Y., Dong, H., Li, Z., Xiao, S. (2023). LWED: Lightweight white-box encryption communication system for drones over CARX algorithm. Journal of King Saud University - Computer and Information Sciences, 35 (9), 101727. https://doi.org/10.1016/j.jksuci.2023.101727
17. Yatao, Y., Ruoqing, Z., Hui, D., Yingjie, M., Xiaowei, Z. (2023). WBZUC: novel white-box ZUC-128 stream cipher. The Journal of China Universities of Posts and Telecommunications, 78 (11), 96–106. https://doi.org/10.19682/j.cnki.1005-8885.2022.0022
18. Hei, X., Song, B., Ling, C. (2017). SHIPHER: A new family of light-weight block ciphers based on dynamic operators. 2017 IEEE International Conference on Communications (ICC), 1–7. https://doi.org/10.1109/icc.2017.7996731
19. Lai, X., Massey, J. L. (1991). A Proposal for a New Block Encryption Standard. Advances in Cryptology – EUROCRYPT '90, 389–404. https://doi.org/10.1007/3-540-46877-3_35
20. Meleshko, Y., Maidanyk, O., Sobinov, O., Mynailenko, R. (2021). A Method of Encrypting the Traffic of Quadrocopters Through an A nalog Path During Monitoring of Agricultural Ground Objects. National Interagency Scientific and Technical Collection of Works. Design, Production and Exploitation of Agricultural Machines, 51, 216–226. https://doi.org/10.32515/2414-3820.2021.51.216-226
21. Myronchuk, K., Vatslavyk, O. (2017). Zabezpechennia peredachi danykh v BPLA. Available at: https://sci.ldubgd.edu.ua/jspui/bitstream/123456789/4474/1/Дрони-МИрончук-Вацлавик.pdf
22. Hell, M., Johansson, T. (2010). Security Evaluation of Stream Cipher Enocoro-128v2. CRYPTREC Technical Report. Available at: https://lup.lub.lu.se/search/files/5976181/2433492.pdf
23. Block Cipher Modes. Computer Security Resource Center. NIST. Available at: https://csrc.nist.gov/Projects/block-cipher-techniques/bcm
24. Blondeau, C., Gérard, B. (2011). Multiple Differential Cryptanalysis: Theory and Practice. Fast Software Encryption, 35–54. https://doi.org/10.1007/978-3-642-21702-9_3
25. Z'aba, M. R., Raddum, H., Henricksen, M., Dawson, E. (2008). Bit-Pattern Based Integral Attack. Fast Software Encryption, 363–381. https://doi.org/10.1007/978-3-540-71039-4_23
26. Wang, M. (2008). Differential Cryptanalysis of Reduced-Round PRESENT. Progress in Cryptology – AFRICACRYPT 2008, 40–49. https://doi.org/10.1007/978-3-540-68164-9_4