

Authentication and Encryption Drone Communication by Using HIGHT Lightweight Algorithm

Hani M. Ismael^{1a}, Ziyad Tariq Mustafa Al-Ta'i^{2b}

^{1,2}Department of computer science, College of Science, Diyala University, Baqubah, Iraq,32001

Author Emails

^amordasshani@gmail.com, ^bziyad1964tariq@gmail.com

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 10 May 2021

Abstract

Extensive use of unmanned aerial vehicles (commonly referred to as a “drone”) has posed security and safety challenges. To mitigate security threats caused by flights of unauthorized drones, we proposed an authentication and security system based on the Chaotic maps and stream cipher lightweight HIGHT algorithm. The proposed system is specifically designed aims to minimize the computational and lower consumption of drone resources and deal with one ground control station (GCS) and one Drone with the number of fly sessions. The proposed system initially generates a flight session key for a drone having a flight plan based on the proposed key generation method using 1d Chebyshev chaotic map and registers the flight session key and its flight plan into a centralized database that can be accessed by. Then drone using HIGHT lightweight to cipher payload data to increase security MAVLink communication protocol and send it to GCS, and finally a GCS checks authentication of the current flight session based on the flight session key and its flight plan as the message authentication code key to authenticate the drone by any flight session while the drone is flying. The experiment results using the NIST test prove the proposed key generation method is able to produce unique and random fly session key, also the results of the error sensitive metrics proved the security of the proposed system using HIGHT block cipher lightweight that is an automatic cryptographic protocol verifier.

Keywords: unmanned aerial vehicle UAV; HIGHT block cipher lightweight algorithm; Chebyshev chaotic maps; MAVLink Protocol.

1. Introduction

Drones have been widely used in recent years not only for the military but also for civil activities, such as environmental control, traffic monitoring, distribution services, and air surveys. In addition, many development programs have adopted drones as mobile collectors for wireless sensor network surveillance applications (WSNs). For example, ground sensors can be used in farms to track soil conditions, and drones can gather information from such sensors regularly and process this information on a network basis[1]. Drones are a kind of flight control tool, and they eventually popularize people's lives without the pilot service. With its market size continually expanding, the attention of academic researchers is on core drone technology. Typically drones need a wireless network to monitor their network throughout the flight. The details gathered during the flight must be returned and supported by the network. The safety output of the network would be critical when the data transmission is confidential [2]. If the viewing line between the drone and its ground controller is interrupted, satellite technology between them can occur. Alas, some deployments do not have encryption functions. This would allow an attacker to take over the control function. Moreover, in 2009 an unencrypted, unmanufactured aerial vehicle video feed was caught with a sky grabber by a militant party[3].

The effects are often different because of various techniques and items of attack. Some attacks are designed to rob information from contact links through security holes while others are aimed at spoofing sensors such as GPS spoofing[4]. Technological progress allows simple handling via mobile phones, instead of using remote controls, to fly mini-drones. Drones are not only used for the purpose of business or personal use. Drones are used by border protection and security teams. Search and rescue teams can collect or drop critical materials in case of natural disasters. Confidence in wireless communications, on the other hand, leaves drones vulnerable to different threats. The attacks, like trade, non-commercial loss, may have dramatic consequences. In this regard, the way hackers carry out attacks and hijack a drone is not correctly understood, To stop or even crash it. For nefarious

reasons, drones can also be affected. It is therefore necessary to detect them and to avoid harm to them[5].

In this paper, we propose authentication and secure drone communication to provide mutual authentication between drones and ground stations. To avoid the computational and traffic overheads caused by certificate exchanges and asymmetric cryptography operations, we introduce the idea about the flight session key between a drone and a ground station created by the key agreement between them before the drone fly. The flight session key is used for efficiently authenticating drones. To provide a unique and secure key agreement for each session fly, we proposed key generation using the chaotic map and Hash Chacha20 lightweight algorithm. To increase the security of MANLink protocol, the proposed system used HIGHT cryptography to cipher payload data without consuming drone enrage resources.

The remaining of the paper is organized as follows: Section 2 explains the related works. Section 3 presents the security of Drone communication. Section 4 illustrates the design proposed model. Sections 5 explain the simulation proposed method, the evaluation metrics, and experimental results and discussion respectively. Finally, Section 6 concludes the proposed model.

2. Related works

Different works were presented such as:

Neil Butcher, et al.[6], established infringements of the US's secrecy and dignity with minimum overhead calculation, for an unmanned aircraft system (UAS). A low-cost encryption mechanism had successfully been implemented for the US. This study was seen as a simple cryptographic break, but it is not possible to break the key under the presumption of a short flight time. This research ensured the system's safety and generated a unique key for each contact path by providing a proper key transmission. For each way of communication, they used the lightweight encryption RC5 algorithm for encryption, decryption, and a unique key. A low-cost encryption method was successfully implemented in the UAS.

Jongho Won et al.[7], suggested an effective certificate-less key encapsulation scheme for sign-encryption (eCLSC TKEM). By reducing the computational overhead on a clever item, eCLSC-TKEM reduced the time taken to create a shared key between a drone and a clever product. The proposed protocol had increased the performance of the drone by using double channels, which enables multiple intelligent subjects to run eCLSC-TKEM at the same time. Through using a parking control testbed, they tested their protocol on usable commercial devices (TelosB and AR. Drone2.0). Experimental data showed that in the computational time (eCLSC-TKEM 9.25 s), (CLSC-TKEM 13.37), (Yang's CL-AKA 32.84s), and (Sun's CL-AKA 15.10s). The proposed protocol is far more effective than other protocols.

Jongho Karel Domin et al., [8], they examined possible specification or application protocol faults using flooding techniques more precisely. The objective was to generate unpredictable machine behavior, injecting invalid or semi-invalid results. They tested with the ArduCopter drone simulator and were able to find a few security vulnerabilities as a result of the test cases mentioned. In the sixth test scenario, the fuzzing script crashed the simulated drone and the payload raised arbitrarily. The floating-point exception aborting and the process aborted can be the result of the fuzzing script's mistake (core dumped). Following that, they discovered a few security bugs. The fuzzing script, in particular, was able to crash the simulated drone in the sixth test case, where the payload was raised at random. They used a key dump of the memory and the gdb debugger when the kernel crashed to figure out what was causing the exceptions. They discovered that errors lead to three distinct functions after conducting an investigation.

Jongho Vishal Dey, et. al. [9], they recommended several approaches and activities that could be used to improve drone communication security. Due to the vulnerability of drones to GPS spoofing, anti-spoofing and anti-jamming receivers were developed. There had been a lot of fruitful work and approaches suggested in the literature for detecting and avoiding civilian GPS anti-spoofing and anti-jamming. These techniques can be divided into two categories: cryptographic (spread spectrum) and non-cryptographic (dual receiver correlation) (antenna array). These techniques, however, were either impossible to adopt or necessitated the purchase of expensive hardware. As a result, they suggested several simplified software-based spoof identification techniques.

Christian Bunse, et. al. [10], they looked at a normal UAV control and control protocol and studied it (i.e., the DSM protocol family). They spoke about typical attack methods, minor assumptions, and the protocol's security vulnerabilities. Since the commercial number of available communication components is low, these findings can easily be ported to other protocols such as (FrSky, S-FHSS, HOTT, and others) by using brute force attack to the message of the radio chip used (CYRF6936) and the DSMX protocol are and they got measurement results of the practical implementation of the attack. When Receive transfer packet a 10848878 μs (≈ 10 s), when Brute force CRC seed 623649 μs (≈ 0.6 s) and the overall 11645488 μs (≈ 11 s). They advised using the longest secret possible (6 bytes at least). This makes the attack of brute force even more complex and means that the rightful owner is better authenticated. Finally, they advocated for the use of cryptographic techniques.

Azza Allouch, et. al.[11], they addressed the MAVLink protocol's security flaws and proposed MAVSec, a security-integrated MAVLink mechanism that relies on encryption algorithms to ensure the security of MAVLink messages sent between GCSs and UAVs. They used Ardupilot to test MAVSec and compared the efficiency of various encryption algorithms (such as ChaCha20, RC4, AES-CBC, and AES-CTR) in terms of memory utilization and CPU consumption. The results show that ChaCha20 outperforms and outperforms other encryption algorithms in terms of performance and efficiency. Integrating ChaCha20 into MAVLink will ensure message anonymity while using less memory and CPU, saving memory and energy for resource-constrained drones.

These studies have mainly focused on obtaining the authority of controlling drones through authentication between drones and ground stations, which are assumed as trusted authorities. Also, these researches have mainly motivated to increase the security of communication channels between drone and ground station using traditional cryptography algorithm that Drone resources consume such as time and energy.

3. Security of Drone Communications

Flying objects pose a potentially high risk of damage and UAV protection and safety threats are crucial to consider [12]. Attackers may use common hacking techniques to quickly control a UAV, preventing it from performing its tasks or, even worse, causing damage. As a result, improvement of UAV protection is urgently necessary. As aggressive security initiatives, drone vendors focus on frequency hopping, spectrum spreading, and key sharing. Legally, however, providers can only run on ISM bands with the 2.4 GHz band dependent on packet-based transfer using networking approaches. Thus protocols such as IPv4 are closely related but do not take safety actions that make them vulnerable to proven attacks [3]. The UAV communication link is shown in Figure 1 [13].

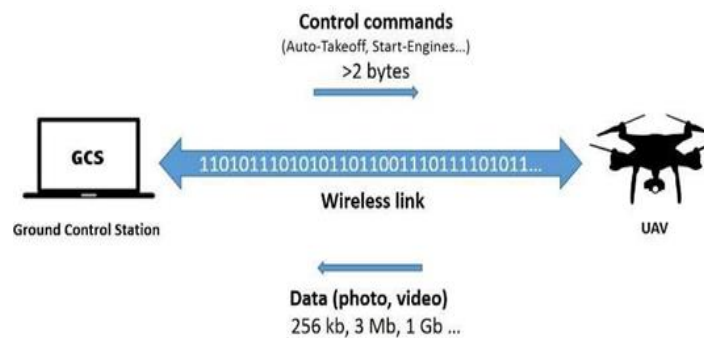


Figure 1. UAV communication link[13].

MAVLink (short for Micro Air Vehicle Link) is a wireless networking protocol that enables entities to communicate[8]. It is used for bidirectional communications between the ground control station and the drone when it is used in drones (GCS). The drone receives orders and controls from the GCS, while the GCS receives telemetry and status information from the drone. Lorenz Meier first launched MAVLink for real-time applications in early 2009 under the Lesser General Public License[9][10]. A single GCS can support up to 255 aircraft using MAVLink. The MAVLink protocol's minimum packet length is 8 bytes (e.g., no payload ACK), and the maximum packet length

is 263 bytes with a complete payload. The MAVLink packet's configuration is seen in Figure 2. To transmit control and telemetry data, a bidirectional connection is needed. The aerial vehicle's telemetry data will be sent to the GCS, while control data will be sent the other direction [11].

A byte-wise MAVLink message is transmitted over the contact channel, followed by an error-correcting checksum. If the checksums do not align, the message is compromised and will be deleted. A packet start sign (STX) is used by MAVLink to sync the start of an encoded message. After receiving the packet start symbol, the packet length (n) is read, and the checksum is checked after n bytes. The decoded packet is processed, an ACK message is sent, and it waits for the next start sign whether the checksum matches. A checksum loss caused by tampered or missing message bytes causes the packet to be discarded, and the receiving system resumes listening for the next start sign packet. As a protection measure for packet loss prevention, MAVLink assigns a sequence number (SEQ) to each packet. If the packet loss rate tends to be high, the pilot will order the UAV to return or at the very least limit its operational range, as seen in Figure 2 [12].

NUMBER	0	1	2	3	4	5	6	7	8	9	10	11
ACRONYMS	STX	LEN	INC FLAGS	CMP FLAGS	SEQ	SYS ID	COMP ID	MSG ID	PAYLOAD	CKA	CKB	SIGNATURE
RANGE	0xFD	1 byte	1 byte	1 byte	1 byte	1 byte	1 byte	3 byte	0-255 bytes	1 byte	1 byte	13 bytes
SHORT DESCRIPTION	Start	Payload length	Incompatibility flags	Compatibility flags	Packet sequence	Sender ID	Component ID	Message type	Actual data	Checksum with seed value A	Checksum with seed value B	Message authentication

Figure

2.MAVLink 2.0 packet structure[12].

4. The Proposed System

The proposed system aims to propose authentication methods to provide mutual authentication between drones and ground control, propose Hash chacha20 lightweight algorithm to cipher the registers the flight session key in a centralized dataset in the ground station to increase the security of proposed authentication methods, and finally increase securing payload data for MAVLink protocol using HIGHT lightweight algorithm. The main idea of the proposed system is to deal with one Ground Control Station (GCS) and one drone with multiple flight sessions. The primitive block diagram for the proposed system is illustrated in Figure 3, where the proposed system includes three main stages (Registration, encryption and decryption, and authentication stages).

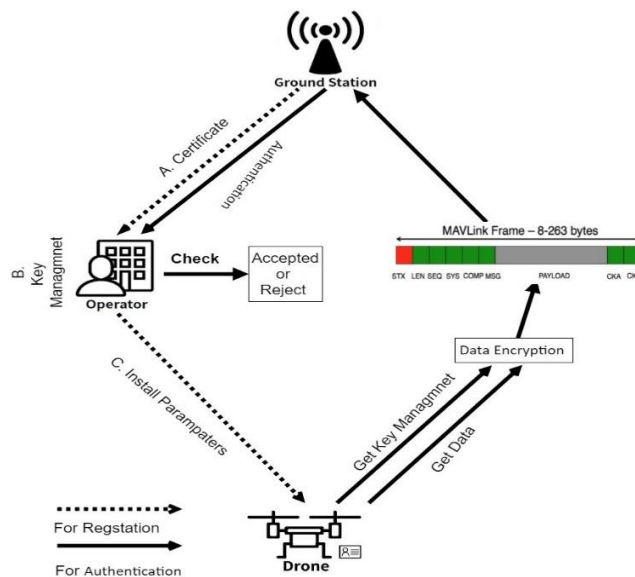


Figure 3. The Primitive Block Diagram for Proposed System.

4.1 Registration Stage

The First stage in the proposed system is the registration drone in the GCS, here the GCS has an operator (computer) that responds to key management. The key management aims to generate a random and unique secret key with a fixed length for each drone session fly based on 1d Chebyshev chaotic function, saved this secret key in the centralization dataset in a secure manner using the

proposed hash chacha20 lightweight algorithm, and finally install parameter (session-id, Hash session key) and initial command and GPS coordinates to the drone. Figure 4. illustrated general block diagrams for key management technique.

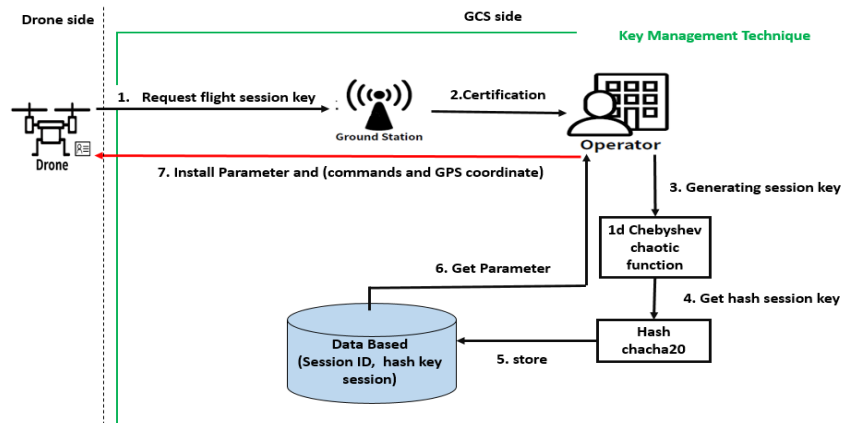


Figure 4. Block diagram of key management technique.

As shown in Figure (4) the key management technique includes the following steps:

1. When a drone attempts to initiate a flight, the drone requests a flight session key from GCS.
2. GCS obtain operator which is a computer that wants to control the security and authentication of the drone. Each flight session of the drone has a unique certificate issued by GCS. The certificate refers to the license GCS gives to the operator to initiate the key generation process.
3. The chaotic maps is series has various useful attributes of application depend on security such as it is a dynamic method in discrete time to result in complex sequence, the noun isn't random however it is deterministic, this characteristic allows us to renew it, has sensitivity is very high of initial condition this leads to any change in primary condition that makes other concatenations, and the chaotic sequence path has entropy behavior in particular space, this causes the recovery of this sequence is impossible in its special space [14]. In this work, a Chebyshev 1D Chaotic Map will be used to generate a unique and random flight session key without complex computation as shown in details as following:

i- For each byte in the session key calculate $Chebyshev_{value}$ using Eq. (1)

$$Chebyshev_{value} = \cos(n \cdot \arccos(x)) \tag{1}$$

Where n is an integer value, $x \in [1, -1]$

ii- Split ($Chebyshev_{value}, '.'$) and take only digits after the dot. For example $Chebyshev_{value}=0.799066410533737$, the results of splitting is "799066410533737".

iii- Convert a value of the "799066410533737" which represents a string to long integer (64 integer number) by using the lookup Table technique, the lookup table has initial parameters which are lookup Table = "0123456789", where this parameter works as an index for each character in string "799066410533737" to return the location of this character in the lookup table.

iv- Convert an integer number (799066410533737) to binary byte or (32-bits) using a Better reduction algorithm, i^{th} as an important role in many cryptographic algorithms. To applied Better for modular reduction of large integers, which is relied on a simple thought avoid the slowness of long division by using involve multiplications, subtractions, and shifts. Algorithm (3) shows Barrett reduction to finds $r=a \text{ mod } p$ given a , and p . The algorithm needs the precomputation of the amount $\mu = \lfloor \frac{b^{2k}}{p} \rfloor$. The pre-computation takes a determined amount of work, which is small in comparison to modular exponentiation cost. Normally, a radix b is selected to be close to the word length of a processor. Suppose $b > 3$ in algorithm (1)[15].

Algorithm 1. Barrett modular reduction

```

Input: two integer number  $a=(a_{2k-1} \dots a_1 x_0)_b, p=(p_{k-1} \dots p_1 p_0)_b$ 
          (with  $p_{k-1} \neq 0$ ), and  $\mu = \lfloor \frac{b^{2k}}{p} \rfloor$ 
Output:  $r = \text{mod } p$ 

Begin
 $q_1 = \lfloor a/b^{k-1} \rfloor$ 
 $q_2 = q_1 \mu$ 
 $q_3 = \lfloor q_2/b^{k-1} \rfloor$ 
 $r_1 = a \text{ mod } b^{k+1}$ 
 $r_2 = q_3 p \text{ mod } b^{k+1}$ 
 $r = r_1 - r_2$ 
if  $r < 0$  then
 $r = r + b^{k+1}$ 
while  $r \geq p$  do
 $r = r - p$ 
return ( $r$ )
End
    
```

4. Generation hashing for session key using proposed hash chacha20 lightweight algorithm as illustrated in the general block diagram in Figure 5.

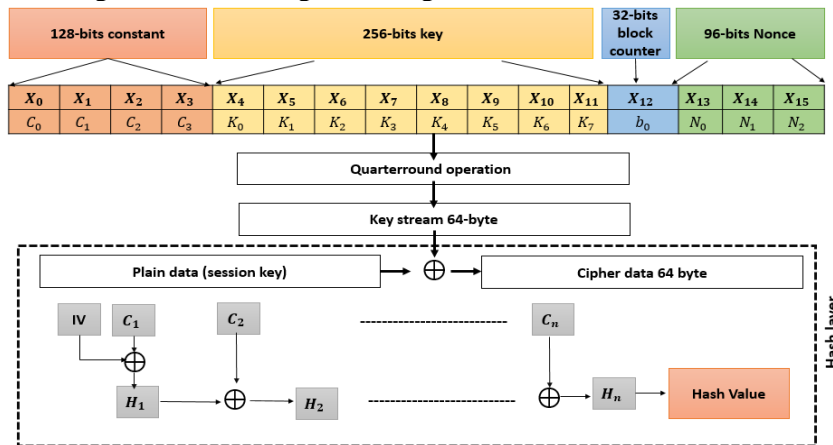


Figure 5. General Block diagram of the proposed lightweight hash chacha20 algorithm.

The proposed Hash Chacha20 lightweight algorithm includes two layers: -

i. Chacha20 lightweight layer: Chacha20 algorithm produces the keystream of 64-Bytes. Input to the keystream matrix is separate from plain text or coded text [16]. This allows the equivalent formation of ciphertext, which improves accuracy. 20 rounds of computations of mathematical utilize X-OR, addition, and rotation utilize as such inputs 4-byte constants, a random 32-byte key, a 12-byte nonce and a 4-byte counter (in original, Bernstein fixed counter and the nonce lengths to be eight values). The 4-Byte constants are $(\sigma_0, \sigma_1, \sigma_2, \sigma_3) = ("0x61707865", "0x3320646e", "0x79622d32", "0x6b206574")$, these strings are successive. The counter typically begins with "0 or 1", increment to every 64Byte plaintext block. ChaCha20 gathers a 256-bit key & a 32-bit nonce (and whose contain a counter). Chacha20 algorithm operates on 32bit words at a time with a key of 256-bits $K = (k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7)$. These blocks are the output of 512-bits for the keystream (Z), and which is X-OR with the stream of plaintext. The encryption state is stored within 16x32bit words value and arranged as a 4x4 matrix [17].

$$\begin{bmatrix}
 X_0 & X_1 & X_2 & X_3 \\
 X_4 & X_5 & X_6 & X_7 \\
 X_8 & X_9 & X_{10} & X_{11} \\
 X_{12} & X_{13} & X_{14} & X_{15}
 \end{bmatrix} \quad (2)$$

ChaCha20 algorithm then locates a four-quarter round function as shown in Algorithm (2) [17].

Algorithm 2. Quarter round
Result: QR (a, b, c, d)
a=a + b ; d = d ⊕ a ; d=(d) << 16;
c=c + d ; b = b ⊕ c ; b=(b) << 12;
a=a + b ; d = d ⊕ a ;d=(d) << 8;
c=c + d ; b = b ⊕ c ;b= b) << 7;

The structure of the primary array defined as shown in Eq.(3) and Eq. (4).

$$\text{for a 256 - bit key, } x = \begin{pmatrix} \sigma_0 & \sigma_1 & \sigma_2 & \sigma_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_0 & k_0 & k_2 & k_3 \\ n_0 & n_1 & c_0 & c_1 \end{pmatrix} \tag{3}$$

$$\text{for a 128 - bit key, } x = \begin{pmatrix} T_0 & T_1 & T_2 & T_3 \\ k_0 & k_1 & k_2 & k_3 \\ k_4 & k_5 & k_6 & k_7 \\ v_0 & v_1 & i_0 & ci_1 \end{pmatrix} \tag{4}$$

The quarter-round functions are exercised to the column (x0, x4, x8, x12), (x5, x9, x13, x1), (x10, x14, x2, x6) & (x15, x3, x7, x11) in odd rounds, and diagonal (x0, x5, x10, x15), (x1, x6, x11, x12), (x2, x7, x8, x13) & (x3, x4, x9, x14) in even rounds. Algorithm 2. characterize the full procedures of Chaha20 algorithm (3)[16]:

Algorithm 3. Chacha20 algorithm
Required: Key K, Block Counter C, and Nonce(N)
Ensure: Keystream (Z)
X ← primary array (K, C, N)
Y ← X;
for i = 1 to 10 do
// Column Round
(x0,x4;x8,x12) ← QuarterRound (x0,x4,x8,x12)
(x5,x9,x13,x1) ← QuarterRound (x5,x9,x13,x1)
(x10,x14,x2,x6) ← QuarterRound (x10,x14,x2,x6)
(x15,x3,x7,x11) ← QuarterRound (x15, x3, x7, x11)
// Diagonal Round
(x0, x5;x10, x15) ← QuarterRound (x0,x5;x10,x15)
(x1, x6, x11, x12) ← QuarterRound (x1,x6,x11,x12)
(x2;x7,x8,x13) ← QuarterRound (x2,x7,x8,x13)
(x3, x4, x9, x14) ← QuarterRound (x3, x4, x9, x14)
end
Z ← X + y
Return Z

ii. Hash layer: the proposed algorithm divides the data encryption into blocks with size 64 bytes, having X-OR with hashing except for the first block X-OR, as it has the initial vector (IV) and produces the first hashing. This operation of accumulator hashing continues until the final hash value is obtained, and this means that the proposed algorithm converts the ChaCha20 from the lightweight encryption algorithm into the lightweight hashing algorithm. The hash function is achieved by converting an input value that is numerical into a compressed numerical value. The hash function input is of arbitrary length, whereas the output length remains always fixed (64 bytes). this step will be stored it in central databased in GCS as pair of parameter for

each session fly (session-id, hash session key). When the operator intends to install the parameters for the drone, he requests these parameters from the database.

The operator provides the drone with two types of commands and coordinates :

1. Before the drone fly, the operator installs the initial command and GPS coordinates for the current session fly.
2. After the drone is flying, the user sends the MAVLink protocol with a new GPS order and coordinates. When GCS wishes to change the direction of the drone or when something has gone wrong during the flight. A heartbeat message from the drone to the GCS is sent to check that the device is ready and alive before sending any new message via payload. The GCS is encrypted with the lightweight algorithm chacha20 to the drone. The payload is encrypted with the hash session key derived during recording and after decoding is calculated to ensure that the message is read by the drone correctly. The checksum is calculated. First, the drone controls and then decrypts the checksum .the payload using chacha20 stream cipher.

4.2 Encryption Stage

The second stage in the proposed system is data encryption/decryption based on HIGHT lightweight encryption algorithm. The objectives of this phase are to use a lightweight to conserve drone resources as time and energy and to improve security in the Micro Aerial Vehicle (MAV) communication protocol for communication between the Ground Control Station (GCS) and Drone.

Let us consider the situation in which the drone flies, bearing in mind that the drone is equipped with parameters (session-id and hash session key), commands, and GPS coordinates. During the flight the drone was able to collect sensitive data (video), so the system proposed here takes several steps :

1. Convert video into N Frames
2. Convert each Frame into a byte
3. For Each byte, Convert (byte to binary) which represents plain data
4. Drone sends the cipher data to GCS via MAVLink protocol by using a lightweight encryption algorithm. Encryption payload data using HIGHT lightweight, so the security of MAVLink protocol is increasing because the payload data in MAVLink packet is a cipher, see section 3. The HIGHT is an asymmetric block chip algorithm that works with the range of bytes and performs simple operations of modular arithmetic, boolean logic, and bitwise with an iterative structured variant of the widespread Feisly Network that can be easily implemented on low-carbon, low-power, and low-computational hardware applications.[18]. In the context of such an application, the cryptography and decryption processes have been established by means of the analysis of transmission characteristics, memory use, and other metrics, to carry out respective comparisons to other known implementations. [19].

4.3 Authentication Stage

The final stage in the proposed system is the authentication stage. In this work, the authentication algorithm was proposed based on a lightweight algorithm to improve the reliability of the proposed system by ensuring a secure transmission channel for data exchange between. When the drone sending cipher data in the MAVLink packet to GCS, the GCS side must check the authentication of this flight session before receiving data to accept data from the drone or reject it. The Proposed Authentication Algorithm follows the procedure as explained below:

1. When GCS receiving MAVLink Packet from the drone, the GCS gives the command to an operator to check the authentication of this session.
2. After receiving the order to check the authentication, the operator begins to extract (session ID and cipher data) from the MAVLink packet.
3. Get session key from the database based on session ID and decryption data using HIGHT lightweight
4. To make the technique of verifying the reliability of the drone more secure, the operator uses the session key of the current session stored in the ground control station database to encrypt the data obtained from the MAVLink packet using the HIGHT lightweight encryption algorithm. If both encrypted data are identical, this means they use the same secret key (Hash session key) and thus

give the authorization to receive and accept the data, but if there is a difference in the encrypted data, this means that the drone used a different key than the one stored inside the ground control station database and therefore this flight session is considered unauthorized and does not receive any data from it.

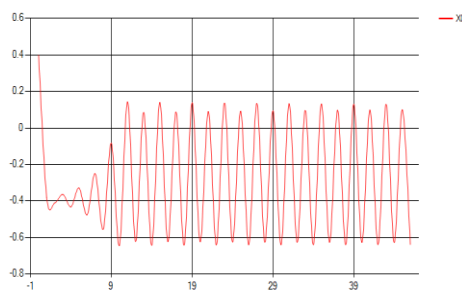
5. Results and Discussion

The results of each stage in the proposed system and performance evaluation based on the NIST test, error sensitivity metrics, correlation coefficient metric, and average security metric are present in this section. The proposed system is executed using C# using a laptop computer. The tests were performed on a Laptop with processor Intel(R) Core(TM)i7-7700HQ @ 2.80GHz (8 CPUs), 64-bit operating system, and Memory 16384 MB RAM. In implementing the proposed scheme, text of various sizes 128 bytes, 256 bytes, 512 bytes, and 1024 bytes will be used. Also, the proposed system takes the test video as shown in Table 1.

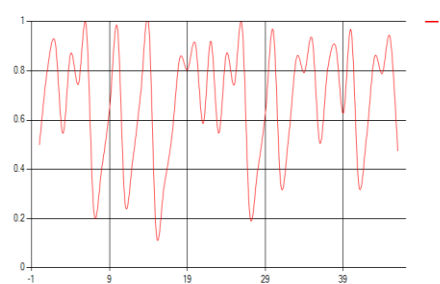
Table 1. Sample of Video.

No.	Video size	Video type	Video length in seconds	Video Dimensions
1	10.2MB	MP4	00:00:33	1280×720
2	15.3MB	MP4	00:00:52	1280×720
3	6.68MB	MP4	00:00:23	1280×720
4	4.05MB	MP4	00:00:39	1280×720

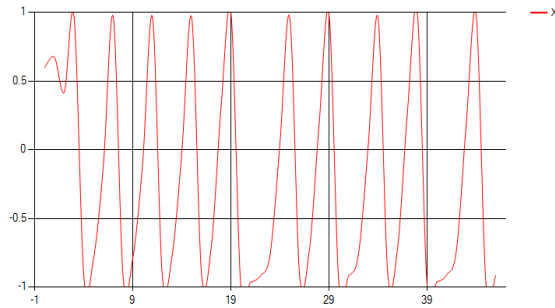
The First stage in the proposed system is the registration drone in the GCS, in this stage will be generated unique and random secret key for each session fly of the drone. Depending on the proposed key generated algorithm and produced Hash for session key using proposed Hash Chacha20 lightweight, see section (4.1).The proposed key generation algorithm using the 1d Chebyshev chaotic function (1). Figure 6 shows the results of the Chebyshev chaotic function using in generating random numbers, based on the values of the initial parameters for the Chebyshev chaotic function. This Figure clarifies the random behavior of Chebyshev chaotic change in the value of initial parameter values (x_0,k) based on three cases of values initial parameters (x_0,k) of the Chebyshev function: case #1 $(x_0=0.4 \text{ and } k=4)$, case #2 $(x_0=0.5 \text{ and } k=5)$, and case #3 $(x_0=0.6 \text{ and } k=6)$. Figure(6) illustrates the best nonlinear behavior of Chebyshev chaotic maps with case# 2 when the initial parameters= $(x_0=0.5 \text{ and } k=5)$.



a) Results of Chebyshev with case #1 $(x_0=0.4 \text{ and } k=4)$



b) Results of Chebyshev with case #2 $(x_0=0.5 \text{ and } k=5)$



c) Results of Chebyshev with), and case #3 ($x_0=0.6$ and $k=6$).

Figure 6. Results of Generating random numbers based on the Chebyshev chaotic map with different values of initial parameters (x_0, k).

Table 2 show results of the proposed key generation algorithm with three cases # of Chebyshev initial parameters (x_0, k) and a number of session =3. The results of the proposed key generation algorithm are secret keys for each drone session fly with size 128 bits. The proposed key generation algorithm produces 16 –byte denotes as key16*8-128-bit for each session and the value of all keys is limited between 0-255.

Table 2. Results of Proposed Key Generation Algorithm





<i>case #1 ($x_0=0.4$ and $k=4$) and the number of sessions =3.</i>																												
Session number	Key 1	Key 2	Key 3	Key 4	Key 5	Key 6	Key 7	Key 8	Key 9	Key 10	Key 11	Key 12	Key 13	Key 14	Key 15	Key 16												
0	10	0	20	6	9	16	6	49	23	3	11	4	23	6	11	1	4	17	25	3	23	4	18	6	24	0	23	6
1	17	1	54	76	20	3	55	63	81	1	1	17	11	1	1	14	4	14	7	25	0	14	0	0	24	7	81	81
2	4	17	1	82	17	1	71	14	8	25	1	21	20	4	23	5	1	49	13	6	49	13	6	49	18	0	19	19
<i>case #2 ($x_0=0.5$ and $k=5$) and the number of sessions =3.</i>																												
0	24	7	19	0	1	1	9	11	1	14	1	71	49	33	24	6	76	30	24	6	76	30	30	18	6	18	6	31
1	16	19	3	31	11	5	22	7	24	90	17	93	10	6	10	3	64	14	5	23	5	89	23	5	89	30	30	30
2	35	19	6	1	21	1	18	0	63	19	1	17	1	16	16	16	16	81	55	41	41	72	55	41	41	72	72	72
<i>case #3 ($x_0=0.6$ and $k=6$) and the number of sessions =3.</i>																												
0	26	22	4	18	6	19	5	17	1	40	17	3	15	0	24	10	6	84	22	6	94	21	0	51	15	15	15	1
1	1	1	14	5	24	0	22	7	3	73	16	0	65	34	23	4	1	14	7	71	10	6	10	10	10	10	10	6
2	20	4	97	18	6	66	76	23	0	10	6	1	19	21	24	0	12	3	15	4	13	7	13	5	10	10	10	6

The proposed system converts 16-byte key to binary with length 128-bit using algorithm 3. and the example of results the key generation algorithm shown in Figure 7.

		4B457D10F
case #2 ($\alpha=0.5$ and $k=5$) and number of sessions =3		
0	÷¼ o G1!öL-°	9A30D0450E7C7F7AD605FB29D1EA4907725DEDCA835C04D3365B16D CF40B146A
1	ÁsPZljg@· ëY- ub/»Ö,)µSL¶ÀææQ	38E3F83EED493B8BF7E3A664BE7C651C2153487877006E9D463A1248E5 B220D4
2	#ÄÖ½?«Q7)H FÚ1¶4¶i);æ@-	7DBE0797F333FBD172A1D98BD38975DD3F142CE600F37A62A388F33B 4AF46057
case #3 ($\alpha=0.6$ and $k=6$) and number of sessions =3		
0	à°Ä«(ÿjTâ^Ö3— □þŠŠ; S□¥ÆØ	84C3AAACE2EC2DE6B5FC422E068381A3FB8519E40CAA772897B51DB2 04C33BEF4
1	·ùpI A"ê"Gjj d;1Ö7nÁ÷Ä0Ö%	42FA5449D75C207D24835C1DE6F4660AC00D286C48899F2613284720F9 3E41AC
2	İa°BLæjü{§%¶j þ8%	CA3FC8677D441920EBC57516DA01569B43033AC86546F7807ABC982E9 2A3CCD3

The second stage in the proposed system is encryption and decryption using the chacha20 lightweight algorithm, see section (4.2). This stage is done on the drone side only on two types of data (video and text). So, if the data is video, firstly convert video in table 1 to images in RGB color space as shown in table 5.

Table 5. Sample of Images.

No	Original image	Image Size	Image Dimensions	No	Original image	Image Size	Image Dimensions
1		67.6 KB	256×256	3		146 KB	256×256
2		58 KB	256×256	4		27.3 kb	256×256

A good cipher algorithm must be sensitive to the secret key and the plain text. If so, the results should be very different when the cipher algorithm makes use of another key with only a small difference for image encryption. Four-color image samples in RGB color space as shown in Table 6 have been used in the experiment of the stream cipher HIGHT algorithm. Table 6 shows the original test image and its corresponding cipher image with both their histograms using HIGHT encryption algorithms. The results in Table 6 illustrate how the HIGHT encryption algorithms can produce cipher images with high randomness, implying that the cipher image cannot be understood and does not seem to affect the original image or the distributed pixels of the original test images and their corresponding cipher images. The results indicate that the histogram images ciphered using HIGHT encryption algorithms appear to the proposed system increase security of MAVLink protocol, making it difficult for unauthorized people to obtain any data from the encoded images.

Table 6. Results of Image Encryption Process Using HIGHT Lightweight Algorithm.

No	Original Image	Cipher Image	Histogram Original Image	Histogram Cipher Image
----	----------------	--------------	--------------------------	------------------------

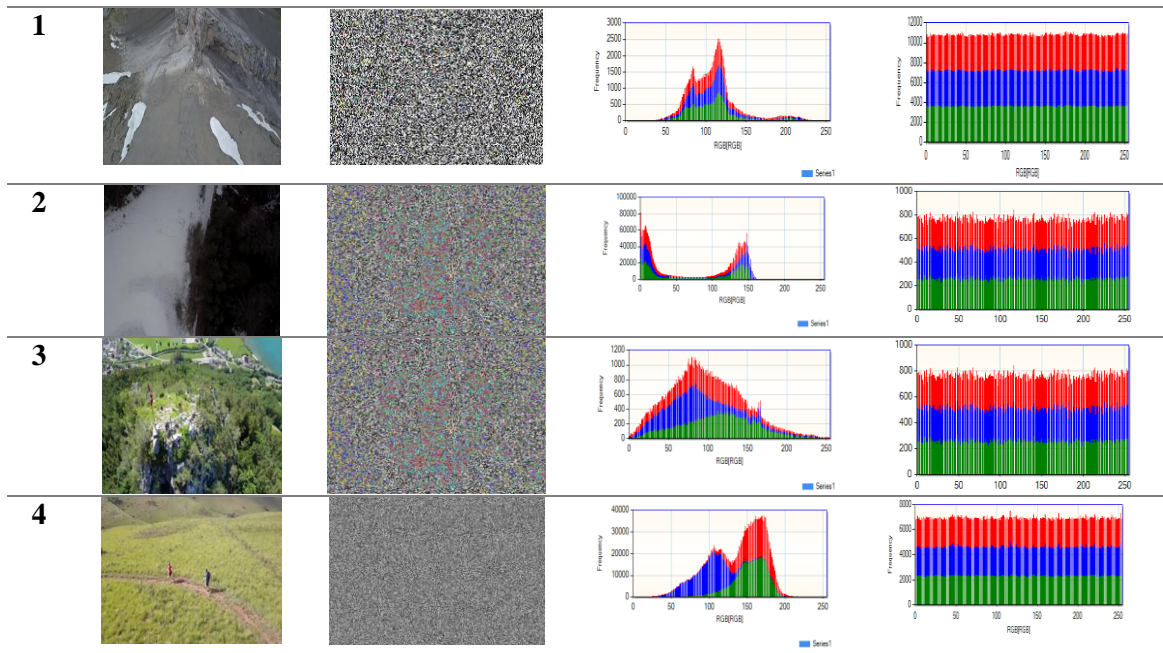


Figure 8 presents the results of error sensitivity-based metrics which are applied between original test images and their corresponding cipher images using HIGHT algorithms. The error sensitivity-based metrics are Mean Square Error (MSE), Peak Signal to Noise (PSNR), Normalized Cross-Correlation, Universal Quality index image (UQI), and Average Difference(AD)[23][23]. Figure 8 proved the proposed system's ability to encrypt images with a higher cipher, where the best results of HIGHT are for all metrics with all test images. The best values of MSE =9064.44; the best values of PSNR =8.56; the best values of NCC =0.53; the best values of UQI = 65.43, finally the best value of AD=89.19.



Figure 8. Results of Encryption HIGHT lightweight Algorithm Based On Error Sensitivity Metrics.

Figure 9 shows the execution time in Mille second of the chacha20 lightweight algorithm cipher test image and it proved the HIGHT cipher image with different size has fast execution time.



Figure 9. Execution Time in MS of Cipher test image Using HIGHT Lightweight Algorithm.

The proposed system using ciphertext which represents commands and GPS coordination that sending from GCS to the drone via MAVLink protocol using HIGHT lightweight algorithm and when receiving these commands and GPS coordination, drone deciphering to follow it using HIGHT lightweight algorithm. Table 7 presents the results of the cipher HIGHT lightweight, this table has shown the HIGHT lightweight algorithm able to produced random ciphertext completely different from plain text and that makes payload in MAVLink protocol more secure against unauthorized persons and attackers.

Table7. Results of cipher HIGHT Lightweight algorithm on text (commands and GPS Coordinates).

No.	Text size	Original text	Ciphertext
1	128	Operating the drone, after which the flight permit is requested. The ground station authorizes the operator to authenticate the	*IDêûŽ JîA´~Õ~²Ãýø. \ñþ·Õ0¶ aœË— IEfVÈø;Rüÿ@šärhM=ÿ æî_Z¹Ú3£xièSÛÚóØ òyûFüzð†¹òGøÙZ/CG± Ô€ÒW×âj,
2	256	Operating the drone, after which the flight permit is requested. The ground station authorizes the operator to authenticate the drone, after which the aircraft receives the necessary orders to fly. The drone begins to fly at an altitude of 200 meters, the	*IDêûŽ JîA´~Õ~²Ãýø. \ñþ·Õ0¶ a œË— IEfVÈø;Rüÿ@šärhM=ÿ æî_Z¹Ú3£xièSÛÚóØ ò yûFü zð†¹òGøÙZ/CG±Ô€Ò W×âj,,dl-BÈ°Ä
3	512	Operating the drone, after which the flight permit is requested. The ground station authorizes the operator to authenticate the drone, after which the aircraft receives the necessary orders to fly. The drone begins to fly at an altitude of 200 meters and the speed is 30 kilometers an hour, the battery capacity is 45 minutes, the imaging distortion is three square kilometers, the amount of signal security 5 Km coordinates and time.HOME(-16.3653,64.0164) 2018.07.06 11:54:11 GPS(-	*IDêûŽ JîA´~Õ~²Ãýø. \ñþ·Õ0¶ a œËIEfVÈø;Rüÿ@šärh M=ÿæî_Z¹Ú3£xiÛÚó Øò yûFüzð†¹òGøÙZ/CG± Ô€ÒW×âj,dl-BÈ°Ä

		16.3654, 64.0166, 18) BAROMETER:	
4	1024	<p>Launching of the picture 0 Denotes the beginning of frames (v2: 0xFD) Duration of payload 1 loading length (n) Flags of inconsistency 2 MAVLink compatibility flags that need to be recognized 2 3 Unless understandable flags may be ignored; 3 Series of packets 4 The submit series of each part is counted. Allows packet failure device identification ID 5 SENDING device identification. ID 5 Allows multiple networks to be distinguished from one network. Item ID 6 Component SENDING identification. Allows differentiation between various elements, such as the IMU and the autopilot, of the same device. ID 7 to 9 of the message Message identification - the id determines the meaning of the payload and how It ought to be decoded correctly. 10 to (n+10) payload The information in the response is dependent on the message.</p>	<p>L” :%o%Aš=,,PÓ)U ĘP– Â,,ó’“`eiôtÜYM0• Tj`c Ýt >P4²¹ýÁG>>JÐ®ÁÁ V°• \Ü ®É</p>

Figure 10 clarifies the results of HIGHT lightweight with a different case size of text (128, 255, 512, and 1024 byte) based on error sensitivity measurements which are correlation coefficient(CC)[24] and average security (AD)[25].Figure 10 shown the chacha20 has the best values of CC=0.2507 with text size 128 byte and average security =6.504 with text overall cases of text size.

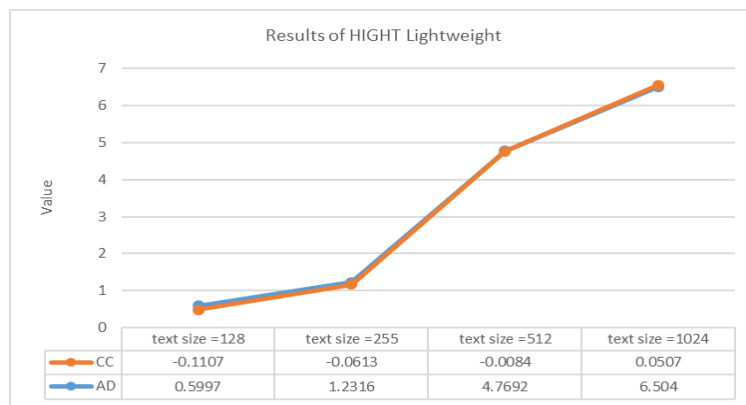


Figure 10. Results of Cipher Text Using HIGHT Algorithm Based on Average Security and CC Metrics.

Figure 11 proved the HIGHT Lightweight algorithm has faster execution time for ciphertext with different sizes which are important features to make the proposed system fast and effective in transmitting cipher commands and GPS from the ground station to Drone and at the same time faster to decryption these command by drone.

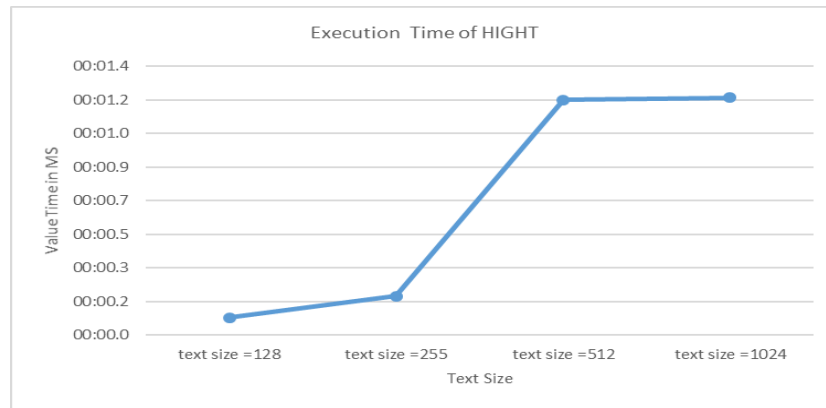


Figure 11. Execution Time in MS of Cipher test Text with different size Using HIGHT Lightweight Algorithm.

Figure 12 show the results of the last stage of the proposed system to check the authentication of drone using HIGHT algorithm.

Set Paramters Step By Step In Preposed System Simulation Proposed system

Select Users

#	Key Ganaration	En Chacha20	Hash Key [Chacha20]	En Hight	Hash K
User[0]	÷%4□□□G16L°	1É ()¥□□ □P	9A30D0450E7C7F7AD605FB29D1EA4907725DEECA835C04D3365B16DCF40B146A	β» &jUa60>%ÉyTR	15282C
User[1]	□Ásb□□□jg@eY	uβ/»Ö,)μ□SL†ÁæQ	38E3F83EED493B88F7E3A664BE7C651C215348787006E9D463A1248E5B220D4	ïμ□□]%-!?"0□□□	35A766
User[2]	#Á□Ó½?□□«□□□Q7)H	FÚ1□□□4¥);æ @□□	7DBE0797F333FBD172A1D98BD38975DD3F142CE600F37A62A388F33B4AF46057	{üð*kÉúµa♣	1BB047

Figure 12. Results of Authentication Algorithm based on HIGHT Lightweight.

Table 8. shows the execution time of the proposed method with the previous methods. The proposed method achieved perfect performance in terms of execution time. This proves that the proposed method can improve the security of drones without drone source consumption.

Table 8. Performances Comparison of the Proposed Method with Previous Based On Execution Time.

No	Reference	Security Methods	Execution Time
1	Jongho Won et al.[7],	eCLSC-TKEM	9.25 s
2	Christian Bunse, et. al. [10]	e Frequency Hopping Spread Spectrum (FHSS)	0.6 s
3	Our proposed method	HIGHT lightweight algorithm	1.4 msec

6. Conclusion

This paper proposes authentication and ciphering of drone communications using chacha20 and HIGHT lightweight encryption algorithm. The proposed system for drone communications is suitable as shown in the results. The comparison of the proposed method with previous works based on execution time shows the proposed method has fast in ciphering execution time (worst time = 1.4 msec).

Reference

[1] Won, J., Seo, S. H., & Bertino, E. (2015, April). A secure communication protocol for drones and smart objects. In Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security (pp. 249-260).

-
- [2] Lv, Z. (2019). The security of the Internet of drones. *Computer Communications*, 148, 208-214.
- [3] Hartmann, K., & Steup, C. (2013, June). The vulnerability of UAVs to cyber attacks-An approach to the risk assessment. In 2013 5th international conference on cyber conflict (CYCON 2013) (pp. 1-23). IEEE.
- [4] Zhi, Y., Fu, Z., Sun, X., & Yu, J. (2020). Security and privacy issues of UAV: a survey. *Mobile Networks and Applications*, 25(1), 95-101.
- [5] Yaacoub, J. P., & Salman, O. (2020). Security analysis of drone systems: Attacks, limitations, and recommendations. *Internet of Things*, 100218.
- [6] Butcher, N., Stewart, A., & Biaz, S. (2013). Securing the mavlink communication protocol for unmanned aircraft systems. Appalachian State University, Auburn University, USA.
- [7] Won, J., Seo, S. H., & Bertino, E. (2015, April). A secure communication protocol for drones and smart objects. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security* (pp. 249-260).
- [8] Domin, K., Symeonidis, I., & Marin, E. (2016). Security analysis of the drone communication protocol: Fuzzing the MAVLink protocol.
- [9] Dey, V., Pudi, V., Chattopadhyay, A., & Elovici, Y. (2018, January). Security vulnerabilities of unmanned aerial vehicles and countermeasures: An experimental study. In 2018 31st International Conference on VLSI Design and 2018 17th International Conference on Embedded Systems (VLSID) (pp. 398-403). IEEE.
- [10] Bunse, C., & Plotz, S. (2018, June). Security analysis of drone communication protocols. In *International Symposium on Engineering Secure Software and Systems* (pp. 96-107). Springer, Cham.
- [11] Allouch, A., Cheikhrouhou, O., Koubâa, A., Khalgui, M., & Abbes, T. (2019, June). MAVSec: Securing the MAVLink protocol for ardupilot/PX4 unmanned aerial systems. In 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC) (pp. 621-628). IEEE.
- [12] Joy, J. P., & Jyothis, T. S. (2016, November). Secure authentication. In 2016 Online International Conference on Green Engineering and Technologies (IC-GET) (pp. 1-3). IEEE.
- [13] Paul, G., & Maitra, S. (2009). On biases of permutation and keystream bytes of RC4 towards the secret key. *Cryptography and Communications*, 1(2), 225-268.
- [14] Al-Tuwajjari, J. M. (2015). Multi-Cipher Technique based on RNA and Chebyshev Map. *Iraqi Journal of Information Technology*, 7(1st).
- [15] Ji, S., & Wan, K. (2017). Adaptive Modular Exponentiation Methods vs Python's Power Function. arXiv preprint arXiv:1707.01898.
- [16] McLaren, P., Buchanan, W. J., Russell, G., & Tan, Z. (2019). Deriving ChaCha20 key streams from targeted memory analysis. *Journal of Information Security and Applications*, 48, 102372.
- [17] Czubak, A., Jasiński, A., & Szymanek, M. (2018, June). A Note on Keys and Keystreams of Chacha20 for Multi-key Channels. In *International Conference on Computer Networks* (pp. 357-372). Springer, Cham.
- [18] Aguilar, J., Sierra, S., & Jacinto, E. (2015, October). Implementation of 'HIGHT' encryption algorithm on the microcontroller. In 2015 CHILEAN Conference on Electrical, Electronics Engineering, Information and Communication Technologies (CHILECON) (pp. 937-942). IEEE.

- [19] Alizadeh, M., Hassan, W. H., Zamani, M., Karamizadeh, S., & Ghazizadeh, E. (2013). Implementation and evaluation of lightweight encryption algorithms suitable for RFID. *Journal of Next Generation Information Technology*, 4(1), 65.
- [20] Niu, X., Wang, Y., & Wu, D. (2014, August). A method to generate a random number for the cryptographic application. In *2014 Tenth International Conference on Intelligent Information Hiding and Multimedia Signal Processing* (pp. 235-238). IEEE.
- [21] Kavun, E. B., & Yalcin, T. (2010, June). A lightweight implementation of keccak hash function for radio-frequency identification applications. In *International Workshop on Radio Frequency Identification: Security and Privacy Issues* (pp. 258-269). Springer, Berlin, Heidelberg.
- [22] Al-Najjar, Y. A., & Soong, D. C. (2012). Comparison of image quality assessment: PSNR, HVS, SSIM, UIQI. *Int. J. Sci. Eng. Res*, 3(8), 1-5.
- [23] Liu, Y., Tang, J., & Xie, T. (2014). Cryptanalyzing an RGB image encryption algorithm based on DNA encoding and chaos map. *Optics & Laser Technology*, 60, 111-115.
- [24] Aoki, K., Ichikawa, T., Kanda, M., Matsui, M., Moriai, S., Nakajima, J., & Tokita, T. (2000, August). Camellia: A 128-bit block cipher suitable for multiple platforms design and analysis. In *International workshop on selected areas in cryptography* (pp. 39-56). Springer, Berlin, Heidelberg.
- [25] Sahib, N. M., Fadel, A. H., & Ahmed, N. S. (2018). Improved RC4 Algorithm Based on Multi-Chaotic Maps. *Research Journal of Applied Sciences, Engineering, and Technology*, 15(1), 1-6.