drones

MDPI

Article

Hybrid Encryption for Securing and Tracking Goods Delivery by Multipurpose Unmanned Aerial Vehicles in Rural Areas Using Cipher Block Chaining and Physical Layer Security

Elias Yaacoub [1], Khalid Abualsaud [1,*] and Mohamed Mahmoud [2]

[1] Department of Computer Science and Engineering, Qatar University, Doha 2713, Qatar; elias.yaacoub@gmail.com
[2] Department of Electrical and Computer Engineering, Tennessee Technological University, Cookeville, TN 38505, USA; mmahmoud@tntech.edu
* Correspondence: k.abualsaud@qu.edu.qa

Abstract: This paper investigated the use of unmanned aerial vehicles (UAVs) for the delivery of critical goods to remote areas in the absence of network connectivity. Under such conditions, it is important to track the delivery process and record the transactions in a delay-tolerant fashion so that this information can be recovered after the UAV's return to base. We propose a novel framework that combines the strengths of cipher block chaining, physical layer security, and symmetric and asymmetric encryption techniques in order to safely encrypt the transaction logs of remote delivery operations. The proposed approach is shown to provide high security levels, making the keys undetectable, in addition to being robust to attacks. Thus, it is very useful in drone systems used for logistics and autonomous goods delivery to multiple destinations. This is particularly important in health applications, e.g., for vaccine transmissions, or in relief and rescue operations.

Keywords: unmanned aerial vehicles (UAVs); cipher block chaining (CBC); delay tolerant networking (DTN); cryptography; physical layer security (PLS); rural connectivity

Citation: Yaacoub, E.; Abualsaud, K.; Mahmoud, M. Hybrid Encryption for Securing and Tracking Goods Delivery by Multipurpose Unmanned Aerial Vehicles in Rural Areas Using Cipher Block Chaining and Physical Layer Security. *Drones* **2024**, *8*, 111. https://doi.org/10.3390/drones8030111

Academic Editors: Tom Cherrett, Paul Royall and Andy Oakey

Received: 1 February 2024
Revised: 11 March 2024
Accepted: 15 March 2024
Published: 21 March 2024

1. Introduction

Three major use cases are generally adopted in fifth generation (5G) cellular networks: enhanced mobile broadband (eMBB), massive machine type communications (mMTC), and ultra-reliable low-latency communications (URLLC) [1]. In addition to these use cases, there is a global trend in the sixth generation (6G) to add a humanitarian aspect that consists of providing basic internet access for all, thus reducing the digital divide [1–4]. The main targets for basic internet access are remote rural and underprivileged areas. Other areas, e.g., urban areas in general, have adequate communications infrastructure. With this infrastructure, they can enjoy boosted versions of the 5G use cases when 6G networks are deployed, like massive URLLC and mobile broadband reliable low latency communication (MBRLLC) [5]. Furthermore, 6G includes an "intelligence" layer that will be ubiquitous due to the deep synergy expected between artificial intelligence (AI) and 6G [6]. But this luxury will not soon be available in rural areas. In fact, various techniques have been investigated in the literature to provide connectivity to remote rural areas, especially for the backhaul. They were surveyed in [7] and include high altitude platforms and satellite links, for example. The reason is that traditional terrestrial solutions based on fiber optics or microwave towers might be too expensive to deploy, especially when road and transportation infrastructure is absent or inadequate in certain remote rural areas.

Although the use of UAVs for fronthaul access has been considered in the literature [8,9], it was assumed that the areas had adequate power and transport infrastructure. Moreover, it was assumed that a sufficient number of drones was available to provide continuous hovering over the area while other drones were recharging. In many rural

areas, power availability is absent or intermittent, and inhabitants have to rely on local sources [10,11]. Furthermore, distances between different areas/villages might be too large to allow continuous UAV hovering to be affordable. Therefore, local networks were adopted in certain villages, where a base station (BS) tower can provide local connectivity to the inhabitants without backhaul connectivity to the core network [12,13]. Backhaul can be provided later when suitable solutions become available. Such solutions could include satellites and UAVs (easier to provide a backhaul link than continuous hovering over the village), but the simplest and most cost-effective approach is delay-tolerant networking (DTN). In fact, DTN has been investigated extensively in the literature, e.g., [14,15], and relies mostly on moving vehicles and buses. More recent investigations have considered UAVs for DTN [16,17], which are more suitable when the transport infrastructure does not allow frequent vehicle trips.

The works in [7–17] have focused on using UAVs for connectivity. However, the use of UAVs for goods delivery has been gaining momentum in recent years [18–21]. UAV-based delivery was mostly studied as a substitute of, or a complement to, vehicle-based delivery (using trucks and cars). In addition, the main motivation for using UAV-based delivery in these works was cost-effectiveness, in addition to meeting the increasingly large demand for online orders. In rural areas, additional challenges need to be overcome. In fact, suitable connectivity might not be available to perform online ordering, and the distances involved are generally far greater than last-mile delivery in cities.

Finally, we note that although the use of UAVs for joint connectivity and goods delivery has received little attention in the literature, compared to the separate scenarios of UAVs for goods delivery only and UAVs for providing connectivity only, there have been some attempts at studying the joint scenario, e.g., [17]. However, it was assumed that the drone has only one parcel to deliver and collect/forward data along the way to certain destinations as an added value service. In addition, it was assumed that the drone remains connected to the source and destination throughout the data delivery process, which is hard to maintain in isolated remote areas.

In this work, we consider the use of UAVs jointly for providing DTN connectivity and goods delivery to remote rural areas. In the case of remote delivery, there is a need to track each item carried and/or delivered and to log the time and location of delivery. In addition, suitable measures need to be taken to secure the carried data (for confidentiality and privacy purposes), the carried goods (protection from loss or theft), and the UAV itself (protection from attack, damage, or theft). Therefore, the main novelty of this paper is related to proposing techniques for recording and storing the transaction information securely in the absence of internet connectivity. Thus, the UAV is assumed to deliver goods, e.g., medicine, humanitarian support, etc., to remote areas, while carrying data in a DTN fashion. We propose efficient techniques for encrypting the transaction logs. After the UAV's return, this encrypted information can be recovered and processed at the Control Center. The novel contributions of this paper can be summarized as follows:

- Proposing a novel framework for encrypting the transaction logs when using UAVs for autonomous delivery with multiple stops at various destinations;
- Describing how cipher block chaining, physical layer security, and symmetric and asymmetric encryption techniques work in tandem with the proposed approach to track goods delivery and secure the transaction logs;
- Showing that the proposed framework is feasible and securely encrypts the data in the absence of connectivity until the drone returns safely to base;
- Analyzing the proposed approach and showing its robustness to attacks;
- Discussing the suitability of the proposed approach in the face of malicious attempts to tamper with the UAV or its payload; and
- Demonstrating that the proposed approach can work with existing encryption techniques to achieve the desired objectives.

The rest of this paper is organized as follows. Section 2 presents the system model considered in this paper. The proposed approach is presented in Section 3. The results

related to the probability of detecting the encryption keys are presented in Section 4, and the security features are analyzed and discussed in Section 5. Finally, Section 6 concludes the paper.

2. System Model

The system model is shown in Figure 1. We investigated the use of UAVs for providing remote delivery to areas without internet connectivity (thus also providing DTN connectivity in the process). From the perspective of goods delivery, two main scenarios can occur. In the first scenario, illustrated in Figure 1a, each remote rural area/village has a single offloading zone where the goods can be delivered and where a telecommunication base station (BS) or access point (AP) can be made available for data exchange (offloading data from the UAV to the AP/BS and carrying the stored data destined to the UAV from the AP/BS). In this scenario, it is assumed that the village has a local communications network, but that network is not connected to the backhaul. Users in the village communicate locally with the central BS, where the data are stored for the UAV to collect and carry to the core network in a DTN fashion. For example, readings from Internet of Things (IoT) sensors can be transmitted regularly to the BS, where possibly some edge processing/intelligence can take place. When a UAV visits the area, it collects the stored data at the BS and transmits to the BS any information coming from the core network. The BS would then take care of distributing it locally to the users in the remote village. An example of this scenario is shown in Figure 1a.

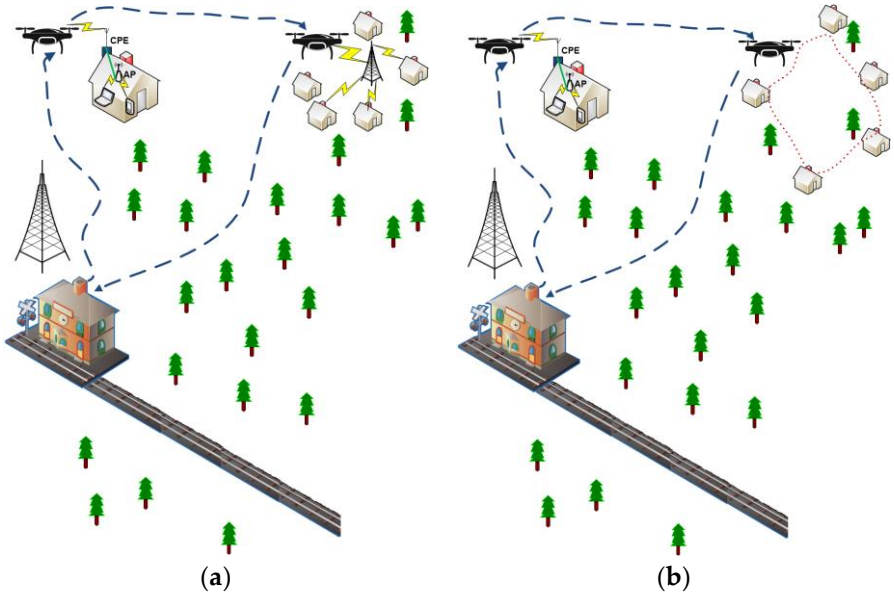

Figure 1. DTN connectivity provided by UAVs benefiting from railroad stations as recharge points: (**a**) village with central BS; (**b**) village without central BS.

The example in Figure 1 consists of having the UAVs provide DTN connectivity to remote areas while using the nearest railway stations as points for recharging. This solves some practical constraints since the train stations are connected to the power grid and could provide a safe place for the UAVs to be stored and recharged. This can also allow the use of trains to transport the UAVs between stations, e.g., to cover different sets of remote zones on the same day. When the start and end stations are the same, this was modeled in [22] as a Traveling Salesman Problem (TSP), where the UAV has to visit each remote area once during its trip while minimizing the trajectory time, which also minimizes the energy consumed during flight.

In the second scenario, illustrated in Figure 1b, at least some of the remote rural areas/villages do not have a single offloading zone, e.g., due to having fewer houses that

are sparsely scattered and relatively far from each other. In that case, the UAV must perform doorstep delivery of the items to their final destinations. Hence, the UAV must still visit each village, but with the absence of a central offload site, the UAV will have to go over each house.

2.1. RFID Tracking of Goods Delivery

When performing multihop delivery as in the scenario of Figure 1, in the absence of real-time internet connectivity, measures need to be put in place to make sure that the goods are delivered to the correct destination and collected by the right people since the UAV stops at multiple locations. This can be accomplished by tagging the goods with RFID tags and having the UAV carry an RFID reader. When the UAV reaches one of the destinations, the local person (to whom the delivery is addressed) can open the box and then collect the relevant goods (also tracked through RFID), as illustrated in Figure 2. In case they make a mistake by collecting items destined for other locations, the UAV's RFID reader detects that the wrong object was removed from the UAV's carried package. In this case, a message can be displayed to the local personnel (e.g., communicated to a suitable mobile app by the UAV, or simply an alarm message/signal issued by the UAV) to warn them of the error. In case the item is still offloaded at the wrong location, this can be reported in the logs (ID of the item, actual intended destination, location where the incident occurred, time, etc.). A picture could also be taken (by the UAV) of the user to confirm the correct delivery and include it in the logs of any incident. Preliminary investigations for using RFID for ensuring correct data delivery were presented in [23], where a testbed/prototype showcasing the implementation of this approach was designed by a student team supervised by the first author. The prototype of [23] included an RFID reader and tags, a GPS for tracking the locations, a database for storing the logs, and a mobile application. However, it proved the concept using a robot instead of a UAV, without involving encryption. The same approach can be easily extended to UAV delivery, while the present work proposes efficient encryption techniques that can be used in this scenario.

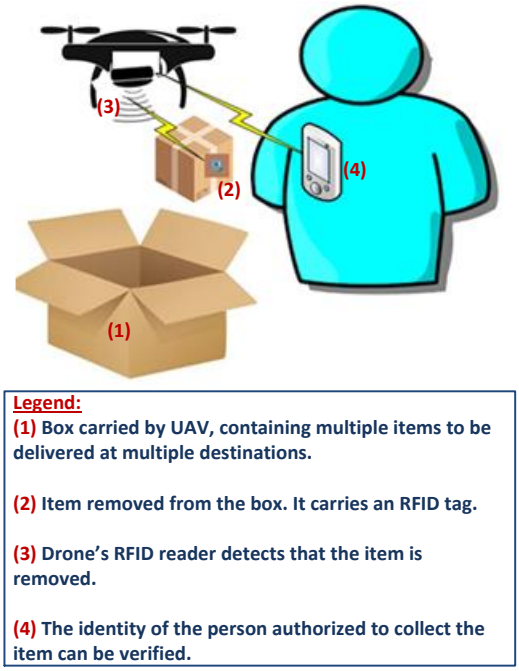

Figure 2. Tracking goods delivery.

2.2. Hybrid Encryption Approach with CBC

In Section 2.1, we described a framework for tracking the correct delivery of goods using RFID. Thus, suitable logs of the transactions made are recorded. However, these

logs need to be secured, as malicious users who might access the drone during one of its stops could tamper with this information. Even if this information is encrypted, the key used would have to be stored somewhere in the UAV's memory, which also puts it at risk. Therefore, the main contribution of this paper is to address this problem by using an enhanced cipher block chaining (CBC) approach to secure the transaction logs, as illustrated in Figure 3.

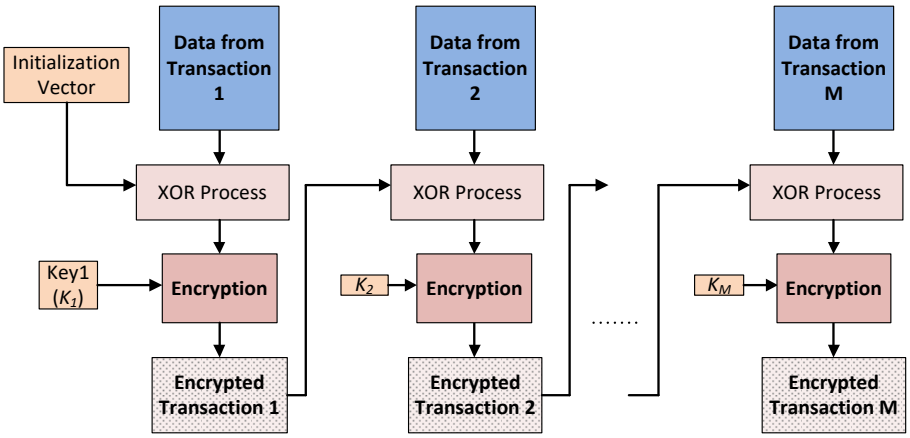

Figure 3. CBC process for encrypting the transaction data.

By encrypting each transaction data block and feeding it as input to the next block, corresponding to the next transaction (related to the next delivery operation), a malicious user accessing the UAV's data (e.g., at one of its stops) would only see the encrypted CBC data. Moreover, if the first block was generated at the starting station or Control Center (e.g., a train station as seen in Figure 1), using a timestamp and/or nonce to encode the start of the journey, it will be very hard for a malicious user capturing the drone to decrypt the whole chain. Added security can also be achieved if encryption is added to that first block using a key and/or initialization vector stored at the station, where it is secure from external malicious access. When the UAV returns to the station, this information can be properly decrypted and stored, and the logs of the transactions are duly recorded on a central server. A new chain can be started when the UAV departs for its next trip, which keeps the size of the chains manageable and makes this approach scalable. When satellite communications are available, even if intermittently during the UAV's trip, the encrypted chain could be transmitted periodically over the UAV–satellite link. This would save the records in case the whole UAV was compromised or shot down before its return.

3. Proposed Approach

The proposed approach for secure UAV delivery in delay-tolerant networks consists of using an enhanced CBC approach to encrypt the data of consecutive transactions, as illustrated in Figure 3. The first encrypted block uses an initialization vector (IV). In the proposed approach, this block corresponds to logging the start of the UAV mission and it indicates the date, time, mission number, etc. when leaving the Control Center. Thus, the IV needs only to be stored at the Control Center, and not at the UAV. Afterwards, the UAV follows its planned trajectory. At each delivery location, it captures the relevant information, such as the item delivered, location, person/entity to which it was delivered to, and indication of correct delivery or if a problem occurred, such as theft or pickup by the wrong person. Then, it encrypts this transaction data using symmetric encryption (the blocks labeled "Encryption" in Figure 3) and CBC.

With CBC, the data of the current transaction are XOR-ed with the encrypted data of the previous transaction, and then the result is encrypted using a symmetric key. In traditional CBC, the key used for symmetric encryption is the same for each CBC transaction. Having this key stored at the UAV poses some security risks, e.g., in case the UAV is captured

by malicious end users, who might then be able to locate the key in the memory and trace/manipulate the transactions performed. Therefore, in the proposed approach, we provide a remedy to this problem by adopting the following steps:

- Use a different key for encrypting each transaction data, i.e., generate a key Ki for transaction number i;
- Generate this key on the spot, using physical layer security (PLS) techniques;
- After symmetric key encryption using CBC, use asymmetric encryption to encrypt the key only, with the public key of the Control Center Kpublic; and
- Store the symmetric key K_i encrypted with K_{public} as $[K_i]_{K_{public}}$, but never the unencrypted key K_i as it is.

After the end of the UAV's flight mission, in the Control Center, the encrypted data can be recovered along with the encrypted keys. The keys can be decrypted using the Control Center's private key and then used to decrypt the CBC data of the various transactions. The process is illustrated in Figure 4.

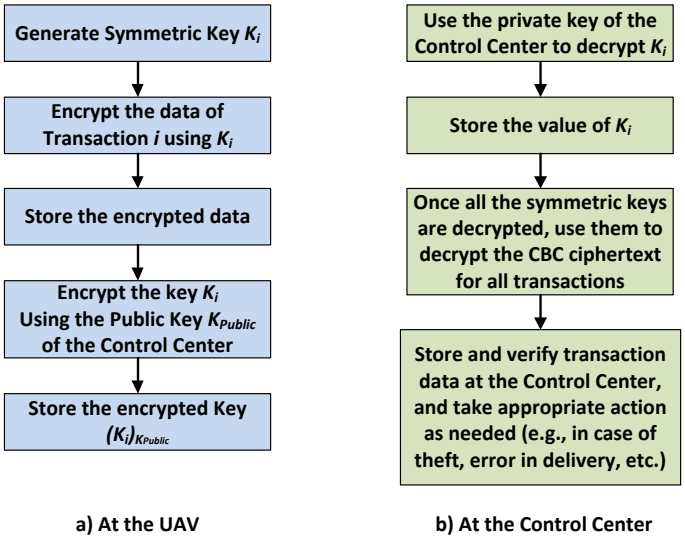

Figure 4. (**a**) Encryption at the UAV; (**b**) Decryption at the Control Center.

4. Results

Considering the proposed approach shown in Figure 3, and assuming that N_b is the number of bits per key K_i, then the number of available keys is $2^{N}{}_{b}$. The probability of correctly detecting one key used for a single transaction is:

$$P_{d,\,Key} = \frac{1}{2^{N_b}} \tag{1}$$

When M transactions are encrypted, each with a different key, then the probability of detecting all of the keys to decrypt the whole chain is:

$$P_{d,\,Chain} = \left(\frac{1}{2^{N_b}}\right)^{M} \tag{2}$$

It should be noted that the probability in (2) does not take into account the difficulty of detecting the initialization vector.

For example, using N_b = 128 and M = 10, then (1) leads to $P_{d,Key} = 2.94 \times 10^{-39}$ and (2) leads to $P_{d,\,Chain} = 4.8 \times 10^{-386}$. This makes it impossible to recover the data without access to the private key of the Control Center, which is safely stored away from the UAV.

In Equation (1), we considered the probability of guessing one key. This can allow decoding one transaction, say transaction i, given that the encrypted previous transaction $(i - 1)$ is accessible, in order to reverse the XOR process shown in Figure 3. However, the proposed approach can be amended to make this previous transaction inaccessible, especially in areas covered several times a day with DTN connectivity by multiple UAVs. The details can be explained as follows:

- The UAV that performed the delivery (let us denote it by UAV j) does not store the encrypted transaction (although it performs the encryption process). It stores only the symmetric keys, encrypted with the Control Center's public key. It needs only to store the encrypted version of the last transaction to use it with the next XOR operation, and then it can discard it. For example, it stores encrypted transaction i to use it for encrypting transaction $(i + 1)$, then discards it, keeping encrypted transaction $(i + 1)$ only to use it for encrypting transaction $(i + 2)$, and so on.
- However, the encrypted block of each transaction is stored at that transaction's location (the actual location where the delivery happened, e.g., BS, AP, or CPE at a house, etc.), to be "picked up" later.
- The next UAV $(j + 1)$ that will be hovering over the area provides DTN connectivity and collects the odd-numbered transactions of UAV j.
- The one after it, UAV $(j + 2)$, provides DTN connectivity and collects the even-numbered transactions of UAV j.
- It should be noted that each of UAVs $(j + 1)$ and $(j + 2)$ could also perform delivery operations, encrypt the corresponding transactions, and store them at the corresponding locations to be picked up by subsequent UAVs, similar to what they did for UAV j.
- Thus, in fact, UAV $(j + 2)$ would be collecting the even-numbered transactions of UAV j and the odd-numbered transactions of UAV $(j + 1)$.

This approach would completely confuse any attacker: Even if the attacker maliciously accesses a drone and guesses one of the keys (which would happen with an extremely low probability as shown by (1)), they will not be able to obtain the initial transaction data. In fact, to do so, they would have to perform the reverse XOR process with the previous transaction; however, this previous transaction is carried by a completely different drone. Hence, to decrypt a single transaction, the malicious attacker needs to (i) be aware of the encryption/collection scheme described above to conceal the encrypted data, (ii) take down two consecutive drones and access their data successfully, and (iii) guess the key used for encrypting the transaction.

5. Discussion

In this section, we provide an analysis of the proposed approach, described in Figures 3 and 4, to show its robustness and the strength of its security features. We start by describing the key generation approach using only physical layer methods, without having to carry the keys from the Control Center. Then, we discuss how the hybrid symmetric/asymmetric approach ensures data confidentiality. By adopting a hybrid approach combining the advantages of symmetric and asymmetric cryptography, the proposed approach provides data confidentiality in the harsh conditions considered, where the UAV performs multiple stops in remote areas without permanent internet connectivity. Afterwards, we analyze how the proposed approach allows for dealing with physical attacks on the UAV. Lastly, we clarify how the proposed approach allows cipher block chaining, physical layer security, and symmetric and asymmetric encryption techniques, to work harmoniously together to track goods delivery and secure the transaction logs. The purpose is to show the need for each of these techniques to provide the required levels of security in the absence of connectivity.

5.1. Physical Layer Security (PLS) Key Generation

The PLS key generation methods could vary. For example, in orthogonal frequency division multiple access (OFDMA) communications, channel state information (CSI) of

multiple subcarriers can be exchanged to select the best group of subcarriers for communication. This CSI exchange could be used to generate a secret key: If the CSI level of a certain subcarrier is above a pre-defined threshold, this corresponds to a bit value of "1" in the key and to a value of "0" otherwise. Concatenating these 1/0 values for a large number of subcarriers leads to a long key that can be used for symmetric encryption.

Another approach is to use some indicative feature about the state of the UAV hardware at a given instant and use it as a seed for a pseudorandom number generator, which could lead to the generation of a pseudorandom key for encrypting a specific transaction.

These PLS key generation methods rely on randomness, depend on the particular state of the UAV at the time of the transaction, and they are used at the UAV only, without being exchanged or transmitted over any wireless link. Consequently, they are extremely hard to guess by a malicious attacker. In fact, an eavesdropper attempting to guess the key obtained through CSI thresholding cannot measure the signal level between the destination device and the drone. It can only measure the signal strength it receives at its current location. Due to fading fluctuations over the OFDMA subcarriers, even assuming it knows the threshold agreed upon for deciding the 1/0 values of the key bits, the device-eavesdropper and drone-eavesdropper links will have significant differences compared to the device-drone link. To be able to correctly capture the key bits, the eavesdropper must come too close to the device so that its channel characteristics become similar to the device-drone link, which would lead to its discovery and detection.

For example, considering PLS key generation using the CSI values for the subcarriers in an OFDMA system, we define d_D as the distance between a legitimate device and the drone, and d_E as the distance between the device and a malicious eavesdropper. It should be noted that, although the drone delivers goods at the exact location of the device, communication to generate the encryption key using PLS can start when the drone is still at a distance d_D.

We assume the eavesdropper attempts to guess the key values using the signal strength it receives at its own location, and we plot the percentage of correctly guessed "1/0" values of the key bits for various values of d_D and d_E. The results are shown in Figure 5.

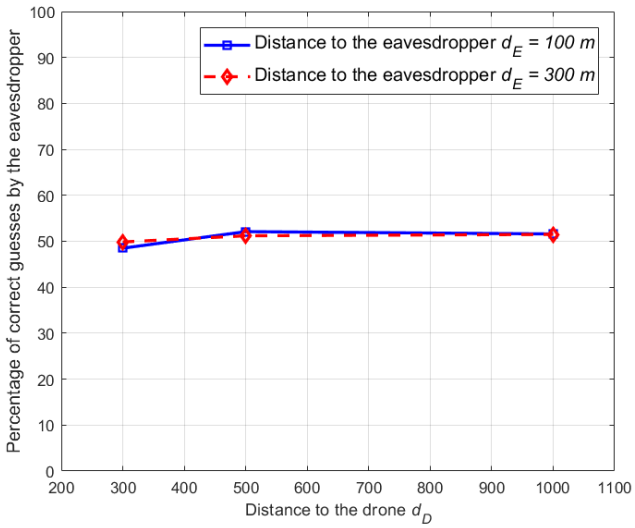

Figure 5. Percentage of eavesdropper's correct "guesses" of the bits of a given key.

Figure 5 shows that the correct guesses are around 50%, which leads to maximum confusion for the eavesdropper. In fact, if the approach was known to lead to a much lower percentage, e.g., 10%, the eavesdropper would simply have to invert its decisions (if it guesses a "0" bit, flip it to "1" and vice versa), in order to reach a guess-rate of 90%! However, a 50% guess rate leads to maximum confusion [24,25] as the eavesdropper would not know the positions in the key in which bits are correctly guessed and which ones are not.

Inverting the decisions for all of them will still leave the eavesdropper with a 50% guess rate, thus maximizing uncertainty. Since the key is used only once for encrypting a single transaction (as per the proposed approach of Figure 3), the eavesdropper will not get a second chance to try to enhance its estimates. Encryption at a new location is performed with a completely new key.

5.2. Data Confidentiality

As mentioned in the previous Section, the key generation process is robust to attacks. Moreover, each key is used to encrypt only one transaction. In addition, the keys are not stored directly in the UAV's memory. However, the keys are needed by the Control Center in order to decrypt the transactions' data. Therefore, they are stored in a secure way using asymmetric encryption. They are first encrypted using the Control Center's public key, and then this encrypted version is stored in the UAV. The only way to recover each initial key is to decrypt this stored value using the Control Center's private key, which is only known to the Control Center.

It should be noted that asymmetric encryption could have been used alone to encrypt all of the data. However, due to being slower and requiring more computational power, a tradeoff was adopted by resorting to a hybrid symmetric/asymmetric method. The bulk of the data is encrypted efficiently and swiftly using symmetric keys, while only the keys themselves are protected using asymmetric encryption. This provides a reasonable tradeoff between security, encryption speed, and computational power consumption.

To get an estimate of the computational time required, we experimented with the implementation of an symmetric encryption algorithm, Advanced Encryption Standard (AES) [26,27], using a Raspberry Pi 4 Model B with a 1.8 GHz processor. To mimic the encryption of the transaction logs, we performed encryption of text data, which was completed using AES with an average time of around 0.4 ms. The encryption of image data (e.g., to mimic a scenario where the drone might take a picture confirming goods delivery) was completed in an average time of 1.5 ms, using a 650 $\times$ 375 jpeg image, of size 70 kbytes. To cater to the scenario of implementing the proposed approach with lightweight and computing-limited UAVs, it is reasonable to assume that if the drone can carry some weight over long distances (for multiple deliveries), it can carry a small Raspberry PI attached to it. Thus, the Raspberry PI used for this experiment was powered by a separate battery and was attached to a commercial off-the-shelf drone (Parrot Anafi), which was flown successfully with the Raspberry Pi attached to it. In this way, we decoupled the burden of encryption and communication, handled by the Raspberry PI (with its separate power source), from the navigation and drone control activities, handled by the drone's processor.

In [28], asymmetric encryption using RSA was shown to be around 100 times faster than RSA decryption, which in turn is around 10 times faster than RSA key generation. The results of the experiment performed in [28] showed that for a 2048 bit RSA key, 5 s are needed for key generation, 500 ms for decryption (with the private key), and around 5 ms only for encryption (using the public key). Thus, by having the asymmetric key generation and decryption tasks performed at the Control Center, our proposed approach requires the drone to only perform RSA encryption for the symmetric AES keys, which is much faster and computationally efficient.

Consequently, the total time required for encrypting each transaction at the drone does not exceed a few milliseconds.

5.3. Resilience to Tampering and Physical Attacks

Based on the above analysis, if a malicious user captures the drone and physically accesses its memory, they will not be able to recover any of the keys. By following this hybrid (symmetric/asymmetric) method, the proposed approach allows secure logging of transactions that can be only decrypted at the Control Center. The UAV itself will not be able to decrypt the data after encrypting it.

Moreover, the UAV can encrypt and store information about any suspicious activity (e.g., unscheduled activity in the initial trip plan to access the drone's data or steal the carried items, etc.) as it does with any other transaction, as long as the malicious act does not consist of destroying the drone. It will be one additional block in the chain that can be decrypted and analyzed at the Control Center. Then, adequate measures can be taken, since the time and location of the suspicious activity are known, as they are stored in the corresponding block.

A slight modification to the proposed approach can be adopted to account for the extreme scenario of destroying the UAV itself (with the impossibility of reporting that immediately, in the absence of connectivity): A copy of the encrypted block of each transaction can be stored at that transaction's location (the actual location where the delivery happened), in addition to storing it at the UAV. If the drone is destroyed between stops no. i and $i + 1$ for example, a subsequent investigation finds CBC blocks 1 to i at their respective locations, while finding nothing at location $i + 1$ onwards. This would help narrow down the location of the culprits. By storing only the encrypted block of a given transaction at the corresponding location, without any keys or any of the previous blocks, it will be impossible to decrypt the data. An example is shown in Figure 6 with $i = 4$. The same approach presented in Figure 6 works in other attack scenarios where the drone is rendered un-operational although not necessarily taken down, e.g., jamming attacks. However, it should be noted that in the case of attacks like eavesdropping and jamming, other physical layer techniques can be used to maintain operations. Such techniques include beamforming between the UAV and the target destination, beam nulling in the direction of the jamming signal whenever it can be estimated, or the use of free space optical communications, where data are transmitted over a line of sight between the sender and receiver. Although there is abundant research on each of these techniques, their integration with the proposed approach constitutes an interesting topic for future research.

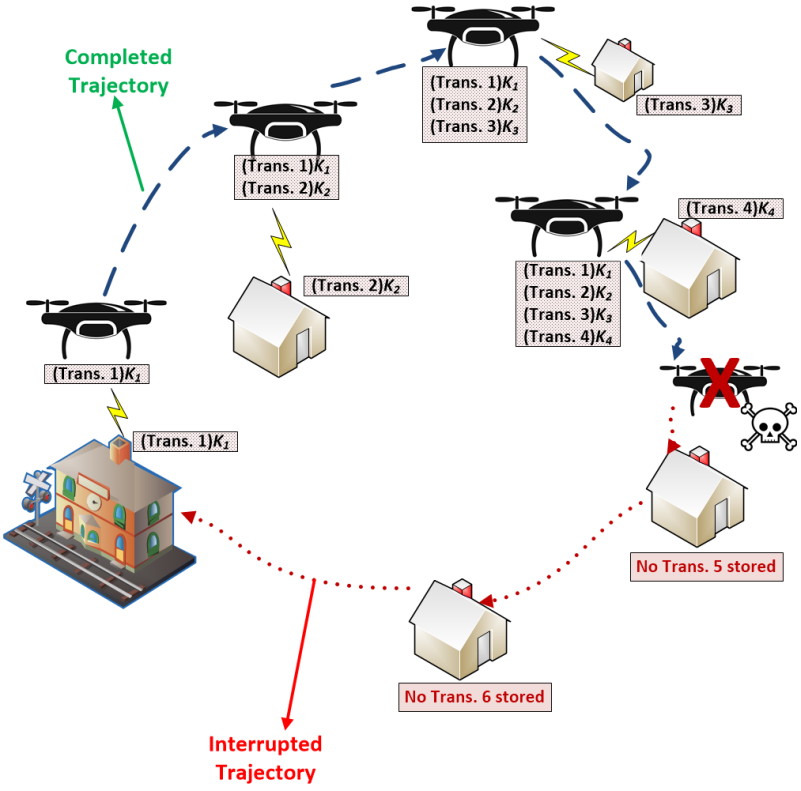

Figure 6. Example of trajectory interruption.

5.4. Interplay between the Various Techniques Used in the Proposed Approach

The proposed approach in this paper combines the benefits of multiple techniques: cipher block chaining, physical layer security, and symmetric and asymmetric encryption techniques. The purpose is not to perform a dull combination of these techniques. Quite the opposite; they operate in tandem in order to secure the transactions' data in the absence of internet connectivity. In fact:

- DTN is a precondition for the proposed approach to be applicable: If internet connectivity is available, then the drone can communicate directly with the Control Center, encrypt the transaction using any traditional technique, and send it immediately (in encrypted form) to the center without any need for storing it on the drone or having to carry this information through multiple destination/stops. The proposed approach is unnecessary in the presence of connectivity.
- Asymmetric cryptography (e.g., RSA) is needed to encrypt the keys so that they can be securely carried by the drone until its return to the Control Center. Since the private key is only stored at this Control Center, then an attacker accessing the memory location of the encrypted keys on the drone will not be able to decrypt them.
- Symmetric encryption following the diagram of Figure 3 is used with different keys, although traditional CBC might use the same key. This is necessary for the scenario considered in this paper, where the drone must make multiple stops at different destinations. In fact, if the same key is used, once the drone lands at a given destination, the data of the previous destinations can be recovered and decoded, which violates the confidentiality of the data and the privacy of the previous users. Using a different key per destination, such that this key is unknown to the other destinations, provides protection against such a scenario.
- In the proposed approach, each one of these keys is used once, immediately after being generated, then stored after being encrypted using the Control Center's public key. Thus, after using a key K_i for one of the blocks shown in Figure 3, the drone itself cannot recover its value after storing it in encrypted form (decryption can be performed only with the private key of the Control Center). To be able to use these symmetric keys without carrying them in an accessible form, the framework proposed in this paper suggests generating them using physical layer security. In this case, the key for a given transaction can be generated immediately at the corresponding location, used for encrypting the corresponding information block, and then stored securely in the drone without having to be exchanged/transmitted over the wireless channel.
- The use of CBC adds an extra layer of security to this symmetric encryption approach. This is achieved by linking the transactions at the multiple stops of the drone in a blockchain-like approach, showing the order of delivery along the followed trajectory, not just the separate encoding of the data as standalone delivery operations. Furthermore, it is necessary in case the modified version of the proposed approach, discussed in Section 4 while analyzing Equations (1) and (2), is adopted.

Finally, it should be noted that the proposed approach is a framework that does not depend on a specific algorithm. It is a suggested approach for protecting transaction data logs in autonomous multiple-delivery scenarios in the absence of connectivity. The proposed approach can be used with any symmetric encryption algorithm that can be implemented in the "Encryption" blocks of Figure 3: DES, 3DES, AES, AES-GCM, etc. It can also be used with any asymmetric encryption approach to encrypt the keys K_i and store them in the drone's memory. In practice, the selected encryption algorithm could depend on the capabilities of the drone, such as the computational power available, amount of energy that can be spared for cryptographic operations, etc. For example, large drones with a large payload carrying heavy items to multiple destinations could probably afford more computationally involved algorithms than lighter drones.

Thus, the proposed approach complements, instead of competing with, other encryption techniques. To provide more experimental data supporting this fact, in Appendix A, we provide the MATLAB code for an example where the method of [29] (AES-256 algo-

rithm with CBC for internal encryption of the 256-bit data blocks) is implemented internally inside each "Encryption" block of Figure 3, while our CBC blockchain-like approach is implemented externally to link the different ciphertexts of the various transactions. The symmetric encryption keys were encrypted and decrypted using RSA. We used this code to perform an experiment where we measured the time needed for each of the encryption/decryption operations. The results were averaged over 1000 iterations, performed on a laptop with an Intel Core i7 1.7 GHz processor with 16 GB RAM and running the Windows 11 operating system. They are displayed in Table 1 for AES and Table 2 for RSA.

Table 1. CPU time for AES and XOR operations.

AES Encryption	XOR Operation	AES Decryption
32.24 ms	2.3 µs	62.45 ms

Table 2. CPU time for RSA Operations.

RSA Key Generation (p = 13, q = 113)	RSA Key Generation (p = 5381, q = 9319)	RSA Encryption (p = 13, q = 113)	RSA Encryption (p = 5381, q = 9319)	RSA Decryption (p = 13, q = 113)	RSA Decryption (p = 5381, q = 9319)
0.295 ms	688.8 ms	0.173 ms	0.347 ms	0.183 ms	0.407 ms

In the approach of Figure 3, the encryption of a given transaction consists of: (i) XOR-ing the input with the previously encrypted block, and (ii) performing symmetric encryption (in this case using AES) of the result. Table 1 shows that the XOR operation is orders of magnitude faster than the encryption operation, which nevertheless takes, on average, 32.24 ms. The decryption takes nearly double that time. However, the decryption operation is not performed by the drone's processor. It is performed at the Control Center after the UAV's return.

In RSA, the public and private keys are generated from two prime numbers p and q. The key generation operation is known to be the most computationally intensive phase [28]. Therefore, in Table 2, we performed the experiment with two scenarios: in the first one, the prime numbers were relatively small (p = 13, q = 113), while they were significantly larger in the second scenario (p = 5381, q = 9319). Table 2 shows that the time for all three operations (key generation, encryption, and decryption) increases when larger primes are used. Although the time roughly doubles for encryption and decryption, it increases by orders of magnitude for key generation. Given that the RSA key generation and decryption operations are performed at the Control Center, whereas only encryption is performed by the UAV, the proposed approach does not impose a high computational burden on the drone. The Control Center is expected to have ample resources to handle the computationally intensive operations.

5.5. Summary

In this section, we summarize the steps needed to implement the proposed approach in a UAV delivery scenario with multiple stops, particularly in remote areas without internet connectivity.

- First, the trajectory of the UAV is planned at the Control Center. It can be programmed on the UAV, the goods can be loaded, and the drone can be sent on its way.
- Then, the UAV can deliver the goods following the optimized path, e.g., following the approach suggested in [22] and outlined in Figure 1.
- At each location, the recipients of the carried goods can identify themselves through an app or code, then get the item dedicated to them. The removal of the item at the given location is detected using, for example, the approach summarized in Figure 2, which

is described in more detail in [23]. A notification is sent to the user in case the wrong item is removed, and they are informed to take only the item(s) dedicated to them.

- The outcome of this transaction (whether success, taking a wrong item, or failing to find the recipient after a certain time), is encrypted using the approach proposed in this paper and summarized in Figure 3: (i) a symmetric encryption key is generated using physical layer security techniques, (ii) this key is used to encrypt the current transaction using a symmetric encryption algorithm, such as AES, after XOR-ing it with the output of the previous transaction (the very first block is encrypted at the Command Center, indicating, for example, the trip number, date and time, planned trajectory, etc.), and (iii) the key is safely stored after encrypting it with the Control Center's public key, using an asymmetric algorithm such as RSA.
- The drone then moves on its way to the other target destinations, repeating the above two steps (delivery and encryption) at each stop, until returning to base.
- After the drone's return, the Control Center's private key is used to decrypt the symmetric encryption keys. Then, each key is used to decrypt the corresponding transaction in the chain. The data can then be analyzed, processed, and stored. Appropriate action can be taken as needed: Correct deliveries could be marked as such. If payment is required, it can be made by charging, for example, a corresponding credit card number of the customer, stored safely at the Control Center. Incorrect deliveries can be detected and identified. Suitable measures can be taken (e.g., legal action).

6. Conclusions and Future Research Directions

In this paper, we considered the use of UAVs for autonomous goods delivery to multiple destinations. The scenario studied corresponded to remote areas, where internet connectivity is not available. We proposed an approach for recording and securing the transaction logs of a multihop delivery process. The proposed approach relies on a combination of cipher block chaining and physical layer security, in addition to symmetric and asymmetric cryptography. It represents a framework that can be implemented with existing encryption algorithms. It was shown to provide unparalleled levels of security in harsh conditions that usually make the UAV vulnerable to attack. The features of the proposed approach were discussed and analyzed in detail, showcasing the security levels provided. The proposed approach was shown to be robust to brute-force attacks. Moreover, it could record events related to unauthorized access to the drone or data (thus providing forensic evidence). Furthermore, in the extreme case of perpetrated physical damage on the drone, it could allow for measures to recover the data and guide forensic investigators toward the area where the incident happened.

Future work could consist of adapting the proposed framework dynamically to the drone's capabilities. In fact, the proposed approach is a framework that works with many algorithms with different levels of complexity and computational requirements. The symmetric and asymmetric encryption algorithms used with the proposed approach could be selected dynamically in an optimized way, depending on the available energy, carried load, and remaining destinations for the UAV. Another direction for future research would be to consider scenarios with intermittent connectivity: In the absence of connectivity, the proposed approach is implemented. When connectivity returns, the UAV can use traditional techniques to save power and memory space. Last but not least, an important area for future work is to implement the proposed approach in actual deployments and perform practical case studies with drone-based delivery to multiple destinations in the absence of connectivity.

Author Contributions: Conceptualization, E.Y., K.A. and M.M.; methodology, E.Y. and M.M.; software, E.Y.; validation, E.Y., K.A. and M.M.; investigation, E.Y. and K.A.; writing—original draft preparation, E.Y.; writing—review and editing, E.Y., K.A. and M.M.; visualization, E.Y. and K.A. All authors have read and agreed to the published version of the manuscript.

Funding: This publication was supported by Qatar University grant no. QUCD-IRCC-CENG-24-349. The findings achieved herein are solely the responsibility of the authors.

Institutional Review Board Statement: Not Applicable.

Informed Consent Statement: Not Applicable.

Data Availability Statement: All data are contained within the article. The original contributions presented in the study are included in the article—Appendix A. Further inquiries can be directed to the authors.

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

Appendix A

This appendix presents a toy example showing how the proposed approach can be used in conjunction with existing encryption techniques. Table A1 shows the MATLAB code used to implement the proposed approach, where the encryption algorithm used in each block (rectangle) labeled "Encryption" in Figure 3 is obtained from [29] (AES-256). Table A2 shows the results: Clearly, the messages were encrypted into a form that cannot be understood by an attacker, and then they were successfully recovered to their original form after decryption.

Table A1. Code for toy example showing how the proposed approach can be used with the encryption method of [29].

```
%--------------------------------------------

% README

%--------------------------------------------

% This is file "Test.m" used to test the proposed approach.

% The proposed approach is a framework that does not depend on a specific algorithm. It is a suggested approach for protecting transaction data logs in autonomous multiple delivery scenarios in the absence of connectivity.

% The proposed approach can be used with any symmetric encryption algorithm in the indicated encryption blocks: DES, 3DES, AES, AES-GCM, etc. It can also be used with any asymmetric encryption approach to encrypt the keys Ki and store them in the drone's memory.

% Moreover, the proposed approach complements and does not compete with other methods, for example, AES.

% AES itself can (and generally does) use CBC in its internal operation to encrypt data (inside each one of the "Encryption" rectangles shown in Figure 3).

% Beyond this "internal" CBC inside a given "Encryption" block, what is proposed in the paper is to link the ciphertexts of each transaction via an "external" CBC in a blockchain-like fashion.

% To show how the proposed method complements, instead of competing with, other encryption techniques, we implement the proposed "external" CBC approach in conjunction with AES-256 using CBC internally (from [29]) in each "Encryption" block of Figure 3:

%
```

Table A1. *Cont.*

```
% To run this file:

% 1) Download the AES-256 Encryption code from:

% [29] David Hill (2024). AES-256 GUI using CBC mode

% (https://www.mathworks.com/matlabcentral/fileexchange/73427-aes-256-gui-using-cbc-mode) (accessed on 14 March 2024),

% MATLAB Central File Exchange.

% 2) Extract the code of the zip file from [29] into a single folder

% 3) Copy this Test file to the same folder

% 4) Save the functions rsa_keys, rsa_enc, and rsa_dec found below in the same folder

% 5) Run the file "Test.m"
%-----------------------------------------------

% Examples of potential transaction texts, to be encrypted with the

% proposed approach

message(1,:) = 'Departing from Base Command Center 123. Date 22 February 2024; Time 8:12 AM; Nb of Locations: 4; Nb of Items to deliver: 4';

message(2,:) = 'This is the transaction at Location 1. Date 22 February 2024; Time 8:47 AM; Item: COVID Vaccine Pack; Delivery: Successful';

message(3,:) = 'This is the transaction at Location 2. Date 22 February 2024; Time 9:20 AM; Item: Antibiotic; Delivery: Successful';

message(4,:) = 'This is the transaction at Location 3. Date 22 February 2024; Time 9:56 AM; Item: Polio Vaccine; Delivery: Successful';

message(5,:) = 'This is the transaction at Location 4. Date 22 February 2024; Time 10:35 AM; Item: Flu Medicine; Delivery: Successful';



% Examples of encryption keys (in hexadecimal), to be used for encrypting

% the above messages

key(1,:) = 'ABCDEFFEDCBA1234567890098765432 1ABCDEFFEDCBA12345678900987654321';

key(2,:) = '1234567890098765432 1ABCDEFFEDCBA12345678900987654321ABCDEFFEDCBA';

key(3,:) = 'AAAAA678900987654321ABCDEFFEDCBA12345678900987654321ABCDEFFEDCBA';

key(4,:) = 'ABCDEFFEDCBA12345678900987654321ABCDEFFEDCBA12345678900987CDECDE';

key(5,:) = '012DEFFEDCBA12345678900987654321ABCDEFFEDCBA12345678900987654FFF';

% Encrypt the first message with the first key while calculating the CPU time needed:

t = cputime;

em(1,:) = Crypt(message(1,:), key(1,:));

te(1) = cputime -t; % calculate the CPU time needed
```

Table A1. *Cont.*

```
% Encrypt the rest of the messages using the proposed CBC approach: each plaintext message is

% XORed with the ciphertext of the previous message, before encrypting the result

for i=2:size(message,1)

t = cputime;

lm = length(message(i,:));xm = bitxor(typecast(uint8(message(i,:)),'int8'), typecast(uint8(em(i-1,1:lm)),'int8'));

em(i,:) = Crypt(char(xm), key(i,:));

te(i) = cputime -t;

end




%----------RSA Encryption and Decryption of the keys----------

% The authors wrote the functions rsa_keys, rsa_enc, and rsa_dec after

% adapting/modifying the file rsa_code.m found at:

% suriyanath (2024). RSA algorithm (https://www.mathworks.com/matlabcentral/fileexchange/46824-rsa-algorithm), MATLAB
Central File Exchange. Retrieved 26 February 2024.

p = 13; q = 113; % select two prime numbers

[e, d, n]= rsa_keys(p, q); % generate public and private keys


% Encrypt the symmetric encryption keys using RSA (with the Center's public

% key (e,n)):

for i=1:size(key,1)

ek(i,:)= rsa_enc(key(i,:), e, n);

end

% After the drone returns to the Control Center:

% Decrypt the symmetric encryption keys using RSA (with the Center's

% private key (d,n)):

for i=1:size(ek,1)
dk(i,:)= rsa_dec(ek(i,:), d, n);

end

%-------------------------------------------------------------------
```

Table A1. *Cont.*

```
% Decrypt the messages using CBC: each decrypted message is

% XORed with the ciphertext of the previous message, before obtaining

% the actual result (which would correspond to the initial message):

for i=size(em,1):-1:2

t = cputime;

dm(i,:) = Decrypt(dk(i,:),em(i,:));

lm = length(message(i,:));

xm = bitxor(typecast(uint8(dm(i,:)),'int8'), typecast(uint8(em(i-1,1:lm)),'int8'));

dm(i,:) = char(xm);

td(i) = cputime -t;

end

% decrypt the last message with the last key while calculating the CPU

% time needed:

t = cputime;
dm(1,:) = Decrypt(dk(1,:),em(1,:));

td(1) = cputime -t;

% Display the results:

fprintf('The initial messages to be encrypted are:\n');

message

fprintf('\n\n');

fprintf('The encrypted messages are:\n');

em

fprintf('\n\n');

fprintf('The decrypted messages are:\n');

dm

fprintf('\n\n');

fprintf('Average CPU time for Encryption:');
```

Table A1. *Cont.*

```
sum(te)/length(te)

fprintf('\n');

fprintf('Average CPU time for Decryption:');

sum(td)/length(td)

fprintf('\n');

function [e, d, n]= rsa_keys(p, q)

% This function takes two prime numbers as input and produces two keys as

% output:
% Public key: e,n

% Private key: d, n

% e.g.,:

% p= 13;

% q= 113;

% n=1469

% phi(1469) is 1344

% d=367

% Public key is (271,1469)

% Private key is (367,1469)

n=p*q;

phi=(p-1)*(q-1);

val=0;

cd=0;

while(cd~=1||val==0)

n1=randi(n,1,1);

e=randi([2 n1],1,1);

val=isprime(e);

cd=gcd(e,phi);

end
```

Table A1. *Cont.*

```
val1=0;
d=0;
while(val1~=1);
d=d+1;
val1=mod(d*e,phi);
end



function [c1]= rsa_enc(m, e, n)
% This function takes a message m and a public key (e,n) as input and
% generates the RSA encrypted ciphertext c1 as output

m1=m-0;
over=length(m1);
o=1;
while(o<=over)
m=m1(o);
diff=0;
if(m>n)
diff=m-n+1;
end
m=m-diff;


qm=dec2bin(e);
len=length(qm);
c=1;
xz=1;
while(xz<=len)
if(qm(xz)=='1')
```

Table A1. *Cont.*

```
c=mod(mod((c^2),n)*m,n);

elseif(qm(xz)=='0')

c=(mod(c^2,n));

end

xz=xz+1;

end

c1(o)=c;

o=o+1;

end



function [nm1]= rsa_dec(c1, d, n)

% This function takes an RSA encrypted message c1 and a private key (e,n)

% as input and generates the decrypted message nm1 as output

over=length(c1);

o=1;

while(o<=over)

c=c1(o);

diff=0;

if(c>n)
diff=c-n+1;

end

c=c-diff;

qm1=dec2bin(d);

len1=length(qm1);

nm=1;

xy=1;

while(xy<=len1)

if(qm1(xy)=='1')
```

Table A1. *Cont.*

```
nm=mod(mod((nm^2),n)*c,n);

elseif(qm1(xy)=='0')

nm=(mod(nm^2,n));

end

xy=xy+1;

end

nm=nm+diff;

nm1(o)=char(nm);

o=o+1;

end
```

Table A2. Results after running the code of Table A1.

```
>> clear all; Test

The initial messages to be encrypted are:

message =

5×117 char array

'Departing from Base Command Center 123. Date 22 February 2024; Time 8:12 AM; Nb of Locations: 4; Nb of Items to deliver: 4'

'This is the transaction at Location 1. Date 22 February 2024; Time 8:47 AM; Item: COVID Vaccine Pack; Delivery: Successful'

'This is the transaction at Location 2. Date 22 February 2024; Time 9:20 AM; Item: Antibiotic; Delivery: Successful''This is the
transaction at Location 3. Date 22 February 2024; Time 9:56 AM; Item: Polio Vaccine; Delivery: Successful'

'This is the transaction at Location 4. Date 22 February 2024; Time 10:35 AM; Item: Flu Medicine; Delivery: Successful'


The encrypted messages are:

em =

5×234 char array


'1b81f0e814482ffb0b9196187fadd43a1b2bd0d859c4b529f14efd8dafbb6bd5d06ed35899e561bdfcd90f4bac411d9b48a423d885ec8426dd4a
507da03755ccb5ecc552de2cd83ea20d27648350308ac425bc664a5975fc337dfda3a565b329c9dc8372ea2a647b9e667afbb80c06a9bf90a166f5'
```

Table A2. *Cont.*

'b5cd8cc8c460e1dc35d086f12340c89bc5623fee151652836e2579c19a7b765593ceadbd6763a83e83c3703468d19688570f632a2d2eff35b6192c3e82b3d65fe0233b1ea17ee556178ad7471ff148799c53af88a2610754c231d29a78571a5878d29e2a9751cb3987f16532e6b9bbcab175344aa3'

'159075f08fc1ecfe70dfa03fa5475cb696d3285aaf2172001bd9a995a0b9042124597bb504cb8315c399860d3e3dd373338703c116387c397f9aff9997457e596080db66d2e1ea94a3b56f66e5d954d52ed3d458488b272c97a8e5480cef4b2fa4a97e2decf14511894376f830d145dbae98c25a91'

'99bcb582f35abba2b8e0b66143a9be29d70c68e82c1061a945b118b0f762b7670d8338bab3534619df494450ba414020d779462f54c3a0da37fa03dc9f9b8904b29e1b90324bdf008ad9329c9dac14f1cba418f8267c0163aa6b00fd19912b7c8208d1ef7b50f70a7b7d5b07f2ab9e949ee3386a2e'

'073b99983997356f6849dc02095627d7cca42e1f169eea7971ac6942613bb64788ddfb8f7368253bc043b4969f481bffdf28eeb182b02e63208eb8ceba9bb523abb33bc510aee576114a4965c8ab319bc30a8f2826f7e8fb865e93f22f5e503f274182ddaea77e084211589b093ba200489c5cbccd'

The decrypted messages are:
dm =

5×117 char array

'Departing from Base Command Center 123. Date 22 February 2024; Time 8:12 AM; Nb of Locations: 4; Nb of Items to deliver: 4'

'This is the transaction at Location 1. Date 22 February 2024; Time 8:47 AM; Item: COVID Vaccine Pack; Delivery: Successful'

'This is the transaction at Location 2. Date 22 February 2024; Time 9:20 AM; Item: Antibiotic; Delivery: Successful''This is the transaction at Location 3. Date 22 February 2024; Time 9:56 AM; Item: Polio Vaccine; Delivery: Successful'

'This is the transaction at Location 4. Date 22 February 2024; Time 10:35 AM; Item: Flu Medicine; Delivery: Successful'

Average CPU time for Encryption:

ans =

0.0437

Average CPU time for Decryption:

ans =

0.0656

References

1. Asghar, M.Z.; Memon, S.A.; Hämäläinen, J. Evolution of Wireless Communication to 6G: Potential Applications and Research Directions. *Sustainability* **2022**, *14*, 6356. [CrossRef]
2. Onireti, O.; Imran, M.A.; Qadir, J.; Sathiaseelan, A. Will 5G See Its Blind Side? Evolving 5G for Universal Internet Access. In Proceedings of the Workshop on Global Access to the Internet for All (GAIA), Florianopolis, Brazil, 22–26 August 2016; ACM: New York, NY, USA, 2016; pp. 1–6.
3. Dangi, R.; Choudhary, G.; Dragoni, N.; Lalwani, P.; Khare, U.; Kundu, S. 6G Mobile Networks: Key Technologies, Directions, and Advances. *Telecom* **2023**, *4*, 836–876. [CrossRef]
4. Suraci, C.; Pizzi, S.; Montori, F.; Di Felice, M.; Araniti, G. 6G to Take the Digital Divide by Storm: Key Technologies and Trends to Bridge the Gap. *Future Internet* **2022**, *14*, 189. [CrossRef]
5. Saad, W.; Bennis, M.; Chen, M. A Vision of 6G Wireless Systems: Applications, Trends, Technologies, and Open Research Problems. *IEEE Netw.* **2020**, *34*, 134–142. [CrossRef]
6. Tong, W.; Zhu, P. *6G: The Next Horizon—From Connected People and Things to Connected Intelligence*, 1st ed.; Cambridge University Press: Cambridge, UK, 2021; pp. 1–462; ISBN 9781108839327.

7. Yaacoub, E.; Alouini, M.-S. A Key 6G Challenge and Opportunity—Connecting the Base of the Pyramid: A Survey on Rural Connectivity. *Proc. IEEE* **2020**, *108*, 533–582. [CrossRef]
8. Chiaraviglio, L.; Amorosi, L.; Blefari-Melazzi, N.; Dell'Olmo, P.; Natalino, C.; Monti, P. Optimal design of 5G networks in rural zones with UAVs, optical rings, solar panels and batteries. In Proceedings of the 20th International Conference on Transparent Optical Networks (ICTON), Bucharest, Romania, 1–5 July 2018; pp. 1–4.
9. Amorosi, L.; Chiaraviglio, L.; D'Andreagiovanni, F.; Blefari-Melazzi, N. Energy-efficient mission planning of UAVs for 5G coverage in rural zones. In Proceedings of the IEEE International Conference on Environmental Engineering, Milan, Italy, 12 March 2018; pp. 1–9.
10. Kusakana, K.; Vermaak, H.J. Hybrid renewable power systems for mobile telephony base stations in developing countries. *Renew. Energy* **2013**, *51*, 419–425. [CrossRef]
11. Alsharif, M.H.; Kim, J. Optimal solar power system for remote telecommunication base stations: A case study based on the characteristics of South Korea's solar radiation exposure. *Sustainability* **2016**, *8*, 942. [CrossRef]
12. Kassem, M.M.; Marina, M.K.; Radunovic, B. DIY mdel for mobile network deployment: A step towards 5G for all. In Proceedings of the 1st ACM SIGCAS Conference on Computing and Sustainable Societies (COMPASS), Menlo Park and San Jose, CA, USA, 20–22 June 2018; pp. 1–5.
13. Schmitt, P.; Iland, D.; Zheleva, M.; Belding, E. HybridCell: Cellular connectivity on the fringes with demand-driven local cells. In Proceedings of the 35th Annual IEEE International Conference on Computer Communications (INFOCOM), San Francisco, CA, USA, 10–14 April 2016; pp. 1–9.
14. Soares, V.; Farahmand, F.; Rodrigues, J. A layered architecture for vehicular delay tolerant networks. In Proceedings of the IEEE Symposium on Computers and Communications, Sousse, Tunisia, 5–8 July 2009; pp. 122–127.
15. Pentland, A.; Fletcher, R.; Hasson, A. DakNet: Rethinking connectivity in developing nations. *IEEE Comput.* **2004**, *37*, 78–83. [CrossRef]
16. Asano, H.; Okada, H.; Naila, C.B.; Katayama, M. Flight Model Using Voronoi Tessellation for a Delay-Tolerant Wireless Relay Network Using Drones. *IEEE Access* **2021**, *9*, 13064–13075. [CrossRef]
17. Iranmanesh, S.; Raad, R.; Raheel, M.S.; Tubbal, F.; Jan, T. Novel DTN Mobility-Driven Routing in Autonomous Drone Logistics Networks. *IEEE Access* **2020**, *8*, 13661–13673. [CrossRef]
18. Koetsier, J. Drone Delivery Is Live Today, and It's 90% Cheaper Than Car-Based Services. Forbes.com. 18 August 2021. Available online: https://www.forbes.com/sites/johnkoetsier/2021/08/18/drone-delivery-is-live-today-and-its-90-cheaper-than-car-based-services/ (accessed on 26 February 2024).
19. Baur, S.; Hader, M. Cargo Drones: The Future of Parcel Delivery. Roland Berger. 19 February 2020. Available online: https://www.rolandberger.com/en/Insights/Publications/Cargo-drones-The-future-of-parcel-delivery.html (accessed on 26 February 2024).
20. Ueland, S. 8 Commercial Drone Delivery Companies. PracticalCommerce.com. 25 January 2021. Available online: https://www.practicalecommerce.com/8-commercial-drone-delivery-companies (accessed on 26 February 2024).
21. Drones, A. High Payload Drone for Delivery & Transport Heavy Lifting UAV. Available online: https://www.airbornedrones.co/delivery-and-transport/ (accessed on 26 February 2024).
22. Yaacoub, E. *Travel Hopping Enabled Resource Allocation (THEResA) and Delay Tolerant Networking through the Use of UAVs in Railroad Networks*; Ad-Hoc Networks; Elsevier: Amsterdam, The Netherlands, 2021; Volume 122, pp. 1–10.
23. Araji, C.; Aburajouh, H.; Al Hail, M.; Yaacoub, E. Secure Automated Delivery of Critical Goods with RFID-based Tracking and Authentication. In Proceedings of the 10th International Symposium on Networks, Computers and Communications (ISNCC'23), Doha, Qatar, 23–26 October 2023; pp. 1–6.
24. Jeon, H.; Choi, J.; McLaughlin, S.; Ha, J. Channel Aware Encryption and Decision Fusion for Wireless Sensor Networks. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 619–625. [CrossRef]
25. Yaacoub, E.; Chehab, A.; Al-Husseini, M.; Abualsaud, K.; Khattab, T.; Guizani, M. Joint Security and Energy Efficiency in IoT Networks Through Clustering and Bit Flipping. In Proceedings of the IEEE International Wireless Communications and Mobile Computing Conference (IWCMC 2019), Tangier, Morocco, 24–28 June 2019; pp. 1385–1390.
26. Cecchinato, N.; Toma, A.; Drioli, C.; Oliva, G.; Sechi, G.; Foresti, G.L. Secure Real-Time Multimedia Data Transmission from Low-Cost UAVs with A Lightweight AES Encryption. *IEEE Commun. Mag.* **2023**, *61*, 160–165. [CrossRef]
27. Su, J.; Gu, N.; Bai, Q.; Lin, C. Parallel Implementation of AES-GCM with High Throughput and Energy Efficiency. In Proceedings of the 2018 International Conference on Networking and Network Applications (NaNA), Xi'an, China, 12–15 October 2018; pp. 251–256.
28. Schmitt, J.P. Cryptography Algorithm RSA—Rivest-Shamir-Adleman. wordpress.com. 17 September 2018. Available online: https://joaoschmitt.wordpress.com/2018/09/17/cryptography-algorithm-rsa (accessed on 22 February 2024).
29. Hill, D. AES-256 GUI Using CBC Mode. MATLAB Central File Exchange. 2024. Available online: https://www.mathworks.com/matlabcentral/fileexchange/73427-aes-256-gui-using-cbc-mode (accessed on 22 February 2024).