μTesla-Based Authentication for Reliable and Secure Broadcast Communications in IoD Using Blockchain

Julio César Pérez García, Abderrahim Benslimane, *Senior Member, IEEE*, An Braeken, and Zhou Su, *Senior Member, IEEE*

***Abstract*—The Internet of Drones (IoD) manages and coordinates communications between drones in Internet of Things (IoT) applications. Ensuring security and privacy in unmanned aerial vehicles (UAVs) networks, i.e., drones, is essential to protect data from cyber attacks. In this context, providing authentication is a major challenge due to the fact that drones are devices limited in power capabilities. The problem is aggravated by the dynamism of IoD networks due to the high mobility of drones, being sensitive to packet loss and handovers. Blockchain technology is attractive to address the problem of centralization of existing authentication protocols. In this article, we provide a decentralized, secure, and efficient authentication protocol, based on μTesla, that relies on Blockchain to manage drone authentication. We analyze the security and performance of the proposed solution. Simulation results show that the proposed solution outperforms several approaches in the literature, achieving an authentication delay of less than 250 ms with a low information exchange of 1024 bits for 128-bit security level while maintaining low computational requirements.**

***Index Terms*—μTesla, authentication, blockchain, Internet of Drones (IoD), unmanned aerial vehicle (UAV).**

I. Introduction

DRONES allow access to hard-to-reach places with low energy, time, and manpower consumption. As a result, drone applications have expanded rapidly in various branches of human development, ranging from supply chain and agriculture applications, and rescue operations to military applications. The Internet of Drones (IoD) paradigm employs a layered network architecture to facilitate communication and coordination among drones. As leakage of sensitive information in IoD applications could result in significant economic and social losses, ensuring the security and privacy of exchanged data is imperative. Particularly, the authentication of legitimate IoD devices plays a critical role [1], [2].

Manuscript received 5 October 2022; revised 28 April 2023; accepted 22 May 2023. Date of publication 26 May 2023; date of current version 9 October 2023. *(Corresponding author: Abderrahim Benslimane.)*

Julio César Pérez García and Abderrahim Benslimane are with the Laboratoire Informatique d'Avignon, Avignon University, 84000 Avignon, France (e-mail: abderrahim.benslimane@univ-avignon.fr).

An Braeken is with the Department of Engineering Technology, Vrije Universiteit Brussel, 1050 Brussels, Belgium.

Zhou Su is with the School of Mechatronic Engineering and Automation, Shanghai University, Shanghai 200444, China.

Digital Object Identifier 10.1109/JIOT.2023.3280124

Drones have limited power, computational, and storage resources. The battery is used simultaneously for flight functions, communications, and onboard processing systems. These limitations, combined with the high mobility of drones, are the reasons why ensuring authentication is a significant challenge. Therefore, optimizing the power and time consumption of authentication protocols maximizes flight time and the level of operability in the missions.

Authentication is generally addressed through a centralized solution, where digital certificates are issued by a certificate authority (CA). Conventional centralized solutions have scalability issues as the network expands. In addition, they often have a single point of failure and are susceptible to disruption in the event of Denial-of-Service (DoS) attacks or technical failures [3], [4].

Due to the limitations of centralized solutions, several authentication protocols incorporating blockchain technology have been proposed in the literature to address centralization and security concerns in conventional solutions [5]. Blockchain is a type of distributed ledger technology (DLT) that uses cryptographic techniques to create an immutable record of data. The data is stored in blocks, which are replicated on all nodes in a peer-to-peer network. Peers in the network do not need to trust each other and maintain a local copy of the ledger. The consensus algorithm, developed by the peers, is responsible for adding new blocks of data to the blockchain in a distributed manner across the network.

A. Research Motivation and Contribution

Most communications in IoD networks are conducted over a public channel in a broadcast fashion. As a consequence, it is essential that all source devices and corresponding communication messages are properly authenticated in the network. In addition, given the high mobility of drones, it is possible to lose some packets. In the case where any of the lost packets contain authentication handshake information, the authentication will fail and must be restarted. Moreover, drones can change from one fly zone (domain) to another, i.e., handover, which results in some cases in the need for reauthentication. Therefore, given the high cost of the authentication process for the drones, the authentication protocol must also consider handover operations in the authentication and support packet losses.

Taking into account the limitations of drones and the fact that many IoD applications are sensitive to delays, it is essential that the authentication protocol is efficient in terms of energy consumption, authentication time, and computational complexity. Several existing works have proposed authentication protocols for IoD networks [3], [6], [7], [8], [9], [10], [11]. However, these proposals are not resilient to packet loss during communication, do not integrate handover solutions, and in some cases are vulnerable to certain attacks.

μTesla [12] is a well-known lightweight authentication protocol designed especially for resource-constrained devices that are both computationally lightweight and robust to packet loss. In this article, we propose an authentication scheme inspired by μTesla, using Blockchain technology to solve the limitations of the original protocol. The main contributions of the present protocol are summarized as follows.

1) Our approach uses Blockchain to support drone authentication information. Additionally, we use lightweight cryptographic operations, e.g., like hash function and exclusive OR (XOR), making our protocol computationally lightweight. Our solution is resistant to different security attacks and robust to packet loss and handover events.
2) We provide a security analysis of the proposed protocol against the main attacks to which IoD networks are vulnerable and compare it with some existing works in the literature.
3) We propose a storage optimization algorithm to address the storage requirements in μTesla.
4) In addition to the security analysis, we evaluate by simulation the performance of the protocol in terms of storage, communication overhead, and consumption of energy. The proposed protocol shows a very good tradeoff between security and efficiency.

B. Organization of This Article

The remainder of this article is organized as follows. Section II presents some work related to authentication protocols and Blockchain-based existing solutions for broadcast authentication. Section III presents some preliminaries related to the cryptographic protocols involved in our approach. In Section IV, we introduce the proposed scheme, and in Sections V and VI, we provide the security and performance evaluation, respectively. Finally, we conclude this article in Section VII.

II. Related Work

A. Broadcast Authentication

Broadcast communications are critical in many applications because multiple receivers can be reached with the same packet, enabling rapid and efficient information exchange. Unfortunately, packet injection attacks and eavesdropping are easy to implement in a wireless environment, hence source authentication is necessary to avoid these security issues.

In the special context of IoD, an authentication scheme should provide resilience against various attacks, including eavesdropping, replay attacks, impersonation attacks, and man-in-the-middle (MITM) attacks. Most point-to-point solutions are not secure against these attacks in broadcast transmissions and in some cases are not efficient enough. In IoD networks, authentication schemes must consider their dynamic and heterogeneous nature, as well as the resource constraints of drones. Therefore, it is necessary to authenticate the source of broadcast packets efficiently. A broadcast authentication protocol in IoD networks must meet the following performance and security requirements [13].

1) Secure and attack-resistant.
2) Low computational cost for generation and verification of authentication information.
3) Low communication overhead, and robust to packet loss and handover.
4) Scalable for a large number of receivers.
5) Decentralized architecture.

Different authentication protocols have been proposed in the literature to address the requirements of IoD networks with broadcast communications. In the following section, we survey several of these existing solutions.

B. Authentication Protocols for IoD

Depending on the architecture of key or certificate management, existing solutions for authentication can be classified as centralized or decentralized. Several protocols have been proposed in the literature for authentication in Internet of Things (IoT) and IoD networks that present a centralized architecture [6], [7], [14], [15]. In [16], the timed efficient stream loss-tolerant authentication (Tesla) protocol is introduced. Tesla allows all receivers to check the integrity and authenticate the source of each packet in broadcast transmission. The protocol does not require trust between receivers, uses low-cost computational operations at both the sender and receiver, and can tolerate the loss of packets without the need for retransmissions. The Tesla protocol achieves asymmetric properties by delaying the disclosure of secret keys while relying on symmetric message authentication codes (MACs).

Perrig et al. proposed the Tesla-inspired protocol μTesla in [12], designed for resource-constrained networks. This protocol communicates the initial key in the key chain to all receivers, reducing the size of transmitted packets compared to the Tesla protocol, and saving time and energy. In addition, it restricts the number of authenticated senders by not storing the one-way key chain in all the nodes. Unlike the original Tesla, where a digital signature is used for initial packet authentication, it instead sends the initial key commitment to all receivers by unicasting. In our solution, we utilize μTesla for ensuring the authentication of the drones. The μTesla protocol will be discussed in-depth in Section III-C.

Centralized services possess the problem of a single point of failure, which makes them vulnerable to DoS attacks or technical failures that could disable the network. In addition, centralized solutions present scalability problems due to the deterioration in performance when the number of users (drones) that the server has to serve simultaneously increases.

To overcome the centralization drawbacks, Blockchain has been applied to distribute services. It replaces trusted

entities with a publicly verifiable, tamper-proof, peer-to-peer distributed data storage that maintains its integrity. Several decentralized solutions for authentication employ a blockchain to store and verify the validity of the identity and public key of devices. Using blockchain alleviates public-key infrastructure (PKI) management without a third party while ensuring the security and privacy of the system [17].

Multiple blockchain-based solutions for authentication have been proposed in different applications, including wireless sensor networks (WSNs) [8], [17], [18], [19], Smart Home [20], Industrial IoT (IIoT) [9], [10], [21], [22], and IoV [23]. These protocols proposed for IoT are generally lightweight and could be adapted to IoD networks. However, recently, novel authentication protocols for IoD have been proposed. In [24], a secure and low-latency authentication of drones using blockchain-based security is addressed. The proposed architecture provides a transparent and efficient mechanism for data security as well as ensures the secure migration of drones between different zones.

In addition, a cross-domain authentication scheme for 5G-enabled unmanned aerial vehicles (UAVs) based on blockchain is proposed [25]. The identity of each drone is dynamically managed by applying a multisignature smart contract. Entities from different domains can authenticate each other without knowing their true identities. The blockchain enables security auditing and the establishment of an accountability mechanism for the involved entities.

In [3], a blockchain-assisted authentication service for industrial UAVs is designed. The distributed peer nodes in the blockchain keep together the ledger that stores authentication information. Industrial drones can call Smart Contract APIs to access the ledger to facilitate their authentication process, which is performed on the basis of ECC to ensure security.

The solution presented in [11] involves the implementation of a secure and efficient distributed authentication mechanism. The approach utilizes a multisignature smart contract to facilitate mutual authentication between terminals operating in a distributed environment. This is achieved through the utilization of consortium blockchain technology.

Based on the above literature review, we observe that many practical authentication mechanisms have been designed for IoT and specifically for IoD networks. Most of them consider the privacy preservation of the participants and successfully resist different attacks caused by internal or external attackers, while they are suitable for authentication for resource-constrained devices due to their computational efficiency. However, not all of these solutions offer resistance to the packet losses that occur in wireless networks and to the possibility of handover mechanisms, either due to technical problems or mobility.

Table I gives an overview of some existing solutions with demonstrated effectiveness in many contexts, highlighting their differences in terms of security requirements provided (or not) by these existing protocols. The last column shows the proposed protocol, which is based on μTesla to guarantee the authentication of broadcast packets in an efficient and packet loss-resistant way while providing decentralized and secure management of the key via the blockchain. In contrast to other existing works, our approach provides all the analyzed security requirements which will be verified in Section V.

TABLE I
COMPARISON OF AUTHENTICATION SCHEMES

Req.	[6]	[7]	[8]	[9]	[10]	[11]	[12]	[25]	Ours
EA	●	●	●	●	●	●	●	●	●
DII	●	●	●	●	●	●	●	●	●
DIF	●	●	●	●	●	●	●	●	●
ESL	●	●	●	●	●	●	●	●	●
MITM	●	●	●	●	●	●	●	●	●
DoS	●	●	●	●	●	●	●	●	●
Dec	○	●	●	●	●	●	●	●	●
SKB	○	●	○	○	○	○	○	○	●
HA	○	○	○	○	○	○	●	●	●
MLT	○	○	○	○	○	○	○	○	●

● Provides ○ Does not provide

Requirements: **EA:** Eavesdropping Attack, **DII:** Drone Identity Impersonation, **DIF:** Drone Identity Forgery, **ESL:** Ephemeral Secret Leakage Attack, **MITM:** Man-in-the-Middle Attack, **DoS:** Denial of Service Attack, **Dec:** Decentralization, **SKB:** Symmetric Key Based, **HA:** Handover Authentication, **MLT:** Message Loss-Tolerant.

III. PRELIMINARIES

In this section, some background is given on the different building blocks of the proposed solution, being the blockchain, the cryptographic hash operation, and the μTesla protocol.

A. Blockchain Considerations

A blockchain is essentially a distributed ledger on a peer-to-peer network that allows transactions to be securely stored and verified without the need for any centralized authority. Transaction serves as the fundamental unit of information exchange between entities. Blockchain allows the transfer of data of any type directly from an externally owned account to another account on the blockchain platform [26].

A typical transaction comprises several fields, including *from*, *to*, *value*, *data*, and other fields related to mining. The *from* field contains the address of the sender, the *to* field contains the destination address to which the information is transmitted, and the *value* field represents the digital currency transmitted value. The *data* field is used to store attachment information for the transaction and is usually left empty. In our proposed protocol, we utilize the *data* field to store the necessary information required for the authentication of the drones and set to zero the field *value* due to the blockchain model being used only to transmit and store the information required for authentication.

In the blockchain structure, the information is stored in blocks and each block is linked to the immediately preceding block through a hash pointer. Thus, it is not possible to modify a block without being detected, since the hash value of the modified block is significantly different from that of the same block without modifications. Moreover, since the blockchain is distributed among all peers in the network, any local change made by a dishonest node to the data in a block can be easily

discovered by other nodes in the network. The process of adding a new block of information to the existing blockchain involves a consensus protocol that is developed by all the peers in the network. This protocol enables the validation of the reliability and authenticity of the block within a decentralized and untrusted peer-to-peer environment, without the need for a trusted third party.

The proposed scheme incorporates the distinctive chain structure of a blockchain for information flow, following the typical blockchain structure but block contents differ slightly. In general, each block contains several fields, including *Version number*, *Timestamp*, *Previous hash*, and *Merkle root*. In our scheme, the *Version number* field is utilized to document the identifier of the flying zone of the drone, and the *Timestamp* field records the block's generation time.

Smart contracts, as an added functionality to the blockchain, are executable programs whose instances and states are stored in the blockchain. Smart contracts allow for the automation of code execution without intermediaries and are executed in a decentralized way by the peers in the network and the results are validated via the consensus protocol.

In our scheme, we use two Smart Contracts *RegisterUAV* and *RevokeUAV*. When *RegisterUAV* is invoked, it automatically generates a special transaction adding the authentication information of a drone to the list of Registered drones (White List) allowing the drone to authenticate with the other elements of the network. On the other hand, the invocation of *RevokeUAV* rewrites the White List without including the drone's authentication information, thus disabling the authentication of the drone.

B. One-Way Cryptographic Hash Functions

Hash functions are an important cryptographic primitive and are widely used in security protocols to guarantee the integrity of information. They compute a digest of a message, which represents a short, fixed-length string of bits. For a particular message, the message digest, or hash value, can be thought of as the fingerprint of a message, i.e., a unique representation of a message. A small change in the input string results in a completely different output string. One-way cryptographic hash functions are defined in [6] as follows.

Definition 1: A cryptographic one-way hash function $H : \{0,1\}^* \rightarrow \{0,1\}^n$ is a deterministic function that takes input data of arbitrary size and produces a fixed length output string.

A cryptographic one-way hash function must have some properties.

1) The hash value is easy and fast to compute and has a low hardware implementation cost.
2) *Preimage Resistance or Unidirectionality:* For a given y no polynomial time algorithm exists for finding a value x such that $H(x) = y$.
3) *Second Preimage Resistance or Weak Collision Resistance:* For a given x no polynomial time algorithm exists for finding a value $x' \neq x$ such that $H(x') = H(x)$.

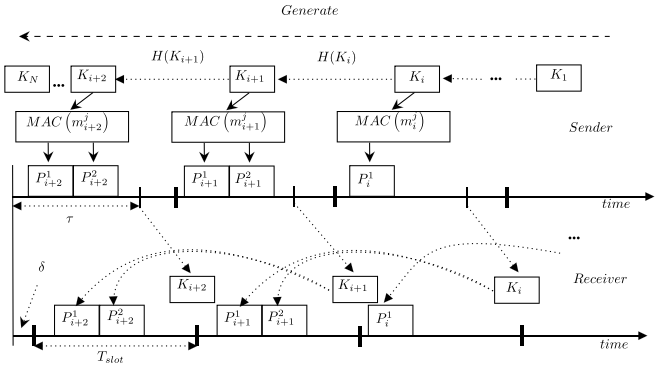


Fig. 1. μtesla authentication protocol.

4) *Collision Resistance or Strong Collision Resistance:* No polynomial time algorithm exists for finding two distinct values $x' \neq x$ such that $H(x') = H(x)$.

C. μTesla Authentication

μTesla improves the performance of the Tesla protocol, making it possible to implement it in devices with limited resources. μTesla is lighter because of the elimination of digital signatures, which are computationally expensive. The main idea of μTesla is to broadcast an authenticated packet through a MAC protocol and a short period of time later publish the key used to compute the MAC. In this way, it is impossible to forge the broadcast packets before the key is published.

Fig. 1 shows an example of μTesla. First, the sender generates a sequence of secret keys (or key chain), for which it chooses the last key K_n randomly, and generates the remaining values by successively applying a one-way function H, that satisfies Definition 1. Hence, the key in the interval i can be obtained by applying i times the function H to K_1, i.e., $K_i = H^i(K_1)$. The one-way function gives the key chain the characteristic that anyone can compute in one direction, i.e., $K_{j+1}, K_{j+2}, \ldots, K_N$ for a given K_j, while it is impossible to compute in the other direction, i.e., $K_1, K_2, \ldots, K_{j-1}$ for a given K_j.

The time is divided into intervals of equal length (T_{slot}) and the sender associates each key of the one-way key chain with a time interval. In this way, the sender uses the key K_i in the interval i to calculate the MAC code of all the packets in this interval. However, in order for the receivers to verify the MAC code of each received packet q in the interval i (P^q_i), they must eventually know the key K_i. The sender publishes K_i during the next interval after some time (τ) has elapsed. From that moment, the receivers can verify the packet P^q_i without the risk of an impersonation attack since the Sender uses in this interval the key K_{i-1} to calculate the new packet.

In order for μTesla to operate effectively, time synchronization between the sender and receivers, as well as knowledge of the key distribution schedule, are necessary prerequisites. However, μTesla does not need the strong time synchronization properties that sophisticated time synchronization protocols provide [27], but only requires loose time synchronization, and the receiver knows an upper bound on the local time of the sender. The time delay interval (τ) utilized in

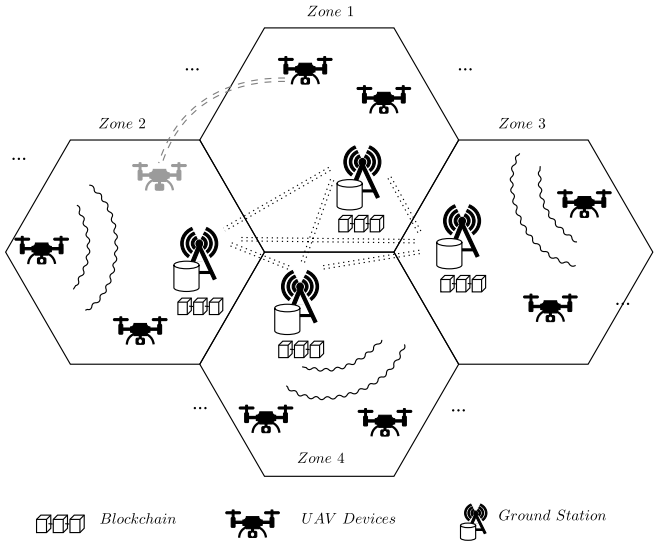


Fig. 2. Blockchain-enabled IoD architecture.

the key revelation must exceed any round-trip time between the network's sender and receivers, as well as any potential synchronization error (δ).

Time synchronization is established in μTesla by a mechanism that provides strong freshness and point-to-point authentication [12]. For this purpose when joining the network each receiver sends a random nonce (N_A) in the request packet to the sender (D_{req}). The sender responds with the message ($T_S|K_i|T_i|T_{\text{slot}}|\tau$) and its corresponding MAC, which contains its actual time (T_S) (allowing synchronization), the corresponding key (K_i) of the one-way key chain for interval i, and the start time (T_i) of the interval i, the duration of a time interval (T_{slot}), and the disclosure delay (τ). Note that those last three values are sufficient to unambiguously determine the timing of the key disclosure.

In broadcast communication, the sender node may not have a preshared key with each receiver. In that case, the sender must send a unicast packet to each receiver with the authentication information, which brings network overhead problems. In addition, the number of keys is finite and consequently the number of packets, so a mechanism for refreshing the keys is required.

IV. Proposed Authentication Protocol

This section presents the proposed scheme, which adapts μTesla to be used in an IoD scenario. We present the considered network architecture and the processes involved in the proposed protocol, which include configuration, drone registration, communication authentication, and authentication revocation. Finally, we address the problem of limited storage in drones.

A. Network Architecture

We consider a network architecture as shown in Fig. 2, in which drones coexist in different flight zones. These zones may represent different domains, zones in a smart city, or sectors in a disaster zone. We assume that drones can communicate with each other and with a ground station (GS) through broadcast messages, as long as they are in the same zone.

Each GS is responsible for controlling and managing the drones, including the processes of registration, authentication revocation, handover management, and drone communication with the blockchain. We assume that in each flight zone, there is at least one GS with which the drones flying over that zone can communicate, and in turn, the GSs communicate with each other through the P2P network provided by the blockchain.

We assume drones to be limited in terms of energy, processing, and storage capacity. On the other hand, we assume that GSs are provided with much more computational and energy resources than drones and they will be the entities in charge of blockchain storage. In this model, the blockchain stores the information necessary for the authentication of each entity in the network, we consider a permissioned blockchain that anyone can access but only a registered GS can add new blocks.

B. Setup and Registration

In this phase, the public parameters for the subsequent authentication process are generated and all drones must be registered to obtain blockchain-assisted authentication services in the respective flight zones.

System Setup: In this phase, each drone is prepared for a specific mission. It happens offline and without the risk that any attacker can have access to the data generated during the course of the mission. Before each mission, the user who owns a drone must generate a hashchain of length n in a similar way as in μTesla. The following procedure is used to perform this calculation.

1) Choose a random number K_n of 128 bits from the set $\{1, 2, \ldots 2^{128} - 1\}$ as private key.
2) Choose a cryptographic one-way hash function H (see Definition 1), which meets the properties discussed in Section III-B.
3) Compute and store $K_i = H^{n-i}(K_n), i \in (1, n)$, where K_0 will be stored in the blockchain doing public-key functions.

Here, $H^i(x)$ denotes applying i times the hash function H to the message x. Note that this process does not consume power from the drone because it can be executed on another computing medium, e.g., laptop. It can also be done with the drone on the ground, where the battery charge completes again before take-off. Once the key chain has been stored in the drone and the mission has been programmed, it proceeds to register it in the blockchain through one of the GSs.

Registration: The drone owner must receive through a secure channel a unique identifier, i.e., pseudonym, from the GS and sends the information generated in the Setup phase, i.e., K_0 and the hash function H used to compute the hashchain. When the GS receives and validates this information, it invokes the smart contract *RegisterUAV*. As a result, the information K_0, H, and the pseudonym of the drone is sent in the *data* field of a transaction and added to

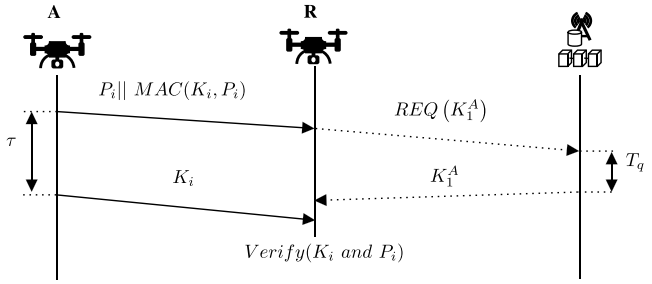


Fig. 3. Blockchain-assisted μtesla authentication protocol.

Algorithm 1 Authentication Algorithm

Input: $t, i, K_i, K_l, P_i,$ MAC(K_i, P_i)
Output: **bool** A

```
if: t < i * τ                      % Check security condition
    Verify MAC(K_i, P_i)
    if: K_l == F^(i−l)(K_i)        % Key verification
        A = True                   % Successfully authentication
        Store K_i                  % K_l = K_i
    else: A = False
return A
```

the Blockchain, and the public key K_0 of the drone is added to the White List.

Once validated and included in the blockchain of all GS, the drone can start the mission in any area. The keys of each drone can be accessed by a simple query to any of the GSs in the different zones, which is responsible for searching the local copy of the blockchain and responding to the query.

C. Broadcast Authentication

Once the mission of each drone starts, all sent packets will be authenticated by a MAC code added to each message. As in μTesla, in each time slot i a different key K_i will be used to calculate the MAC code of all messages transmitted by the drone during this time slot. Fig. 3 shows an example of the authentication protocol of a packet P_i sent by the drone A.

When a receiver R receives the packet (P_i), coming from A, with its corresponding MAC, it needs to make sure that the packet could not have been spoofed by an adversary. The threat is that the adversary already knows the key revealed for this time slot and could therefore forge the packet since it knows the key used to calculate the MAC. Therefore, the receiver needs to be sure that the sender has not yet revealed the key that corresponds to an incoming packet, which implies that no adversary could have spoofed the content. This is called the security condition, in which receivers check for all incoming packets.

In addition, each receiver must verify which key corresponds to the current time slot l, which is calculated from the last key of the drone A stored in drone R, denoted as K_l^A. The verification process is performed using Algorithm 1. In the case that some receiver has never had communication with the drone A, it must make a request to the blockchain through the GS of its zone, which will respond with the value of the key of A stored in the BC during the Registration phase.

D. Drone Revocation

In case of expiration of the information provided for drones in the Registration phase or the detection of malicious behaviors of some drones, the smart contract *RevokeUAV* is invoked. Consequently, the authentication information of the corresponding drone is disabled by rewriting the While List without including the public key of the drone. As a result, the drone will not be able to compute valid MAC codes for the messages and will lose the authentication with the others.

The White List can be publicly accessed with a simple query to the blockchain via the GSs. In addition, when malicious behaviors have been detected, the GS proposes disabling the authentication of the drone immediately by invoking the *RevokeUAV* smart contract. Note that the GSs will record the White List (based on the consensus mechanism) into the blockchain for complete auditing. Many existing mechanisms for intrusion detection could be applied for the automatic detection of malicious drone behaviors, some of them are studied in [28].

E. Handover and Loss of Packets Analysis

In our scheme, where a drone required a handover, it does not require to be reauthenticated. The legitimate drones know the corresponding time slot number and will use the appropriate key. Since the public key is stored in the blockchain, the GS of the new zone can respond to the queries of the receivers, and the authentication of the drone is successful.

No need for reauthentication makes the protocol efficient in case of handover events. Without loss of generality, if there is a node handover in time slot number i, it sends the next packet in the new zone after the handover in time slot number $i+j$. So, the legitimate drone will be able to use in this case the correct key K_{i+j} to calculate the MAC of the message sent, and the receivers will be able to validate the key via the blockchain following Algorithm 1.

By the same token, thanks to the properties of the μTesla hashchain, the protocol is resistant to packet loss. If a receiver received a packet from a drone in slot number i and then several packets containing the key to validate the MAC code of previous packets are lost, eventually, with the key of the next received packet in slot $i+j$ and thanks to the properties of the hashchain, it is possible to validate all the keys from K_i to K_{i+j} (via Algorithm 1) thus guaranteeing the authentication of all previous packets. By combining μTesla and public-key storage in the blockchain, the proposed protocol is efficient during handover events and resilient to packet loss.

F. Limited Storage Considerations

Since the number of packets to be transmitted in a mission and thus the length of the hashchain can be considerably large, a large amount of data needs to be stored in each drone. Even when the keys have a length of 32 bytes for a 128-bit security level, some devices may not be able to store the whole hashchain needed for a mission. We next consider the problem that the drone does not have the memory capacity required to store all the values of a hashchain of length N and can store only a number S of the values. From these stored values

Algorithm 2 Key Storage Algorithm

Input: $N, S, t, [x_1^{t-1}, x_2^{t-1} \dots x_S^{t-1}]$
Output: x_S^t

```
if (x_S^{t-1} == t) #Update storage
    if (D(x_{S-1}^{t-1}, x_S^{t-1}) > 1) #not at distance one
        x_S^t = floor((x_S^{t-1} - x_{S-1}^{t-1}) / 2)
    else
        x_S^t = x_{S-1}^t
        x_{S-1}^t = floor((x_{S-1}^{t-1} - x_{S-2}^{t-1}) / 2)
else
    x_S^t = x_S^{t-1} #Keep the same value
return x_S^t
```

it must be able to reconstruct the hashchain completely and unambiguously, by successively applying the hash function. Note that applying the hash function consumes device time and power, so it is desired to minimize the number of times the hash function is applied.

Let x_u^t be the index of the key stored in memory location u at time slot number t. During time slot number i, the key K_i is used. If it is not stored, the hash must be calculated from the last stored key with a subindex less than i. The value K_1 must always be stored, i.e., $x_1^t = 1; \forall t$, or it would be impossible to calculate it. In turn, once the stored value has been used in the corresponding time slot, this memory space can be used. The main goal is to minimize the number of times the hash function is applied, which leads to the following optimization problem:

$$\begin{aligned} \min_{x_u^t} \quad & \sum_{u=2}^{S}\sum_{t=N}^{2}\left(x_S^t - t\right) + \left(x_u^t - x_{u-1}^{t-1}\right) \\ \text{s.t.} \quad & x_u^{t+1} \le x_u^t \\ & 1 \le x_u^t \le N \\ & 1 \le u \le S; 2 \le t \le N. \end{aligned} \tag{1}$$

Note that the first term of the objective function is how many times the function should be applied in slot t given that the nearest value is x_S^t and the second term is how much it costs to store a more convenient value in the already used, i.e., available, memory slots.

The problem can be solved using an integer linear programming algorithm, but it would be computationally expensive and therefore impractical for drones. We propose an algorithm (shown in Algorithm 2) for dynamic memory management, which allows for improving memory utilization and thus minimize the computational cost of the proposed scheme.

The idea of the algorithm is to start from the storage of S keys evenly spaced among the N keys of the hashchain and update the memory location with the largest index of the stored keys, i.e., x_S^t. The value to be stored will be equidistant between x_S^t and x_{S-1}^t (as shown in Algorithm 2) according to Theorem 1. Let us define the distance function $\mathbb{D}$ between two stored values as the number of times the hash function must be applied to compute the other value, for example, between $\mathbb{D}(K_{10}, K_4) = 6$ because $K_4 = H^6(K_{10})$.

A toy example could be when a hashchain has a length of seven ($N = 7$) and only two values can be stored ($S = 2$). Following the proposed algorithm, in the beginning, K_1 and K_4 are stored. In the first slot, key K_7 is required but is not stored and must be calculated from K_4, via applying three times the function H, $\mathbb{D}(K_4, K_7) = 3$. The next two slots require K_6 and K_5, respectively, which are not in memory and are calculated from K_4 by applying twice and once the function H, respectively. Subsequently, K_4 is used, which is already in memory (zero cost). In the next slot, K_3 is used, which is not in memory and should be calculated from K_1, for which it is necessary to calculate $K_2 = H(K_1)$. At this moment, the value of K_2 is stored in the memory space where K_4 was stored. Finally, in the following slots, K_2 and K_1 are required, which are in memory and have zero cost. Therefore, a total of 8 times the calculation of function H is required. This means that one less time is required than if the allocations were static, i.e., without reusing the memory spaces. As the values of N and S increase, the gain is higher as verified later in Section VI-H.

Theorem 1: If the distance between two consecutive stored values x_i^t and x_j^t, with respective indexes i and j, is equal to d at time t, (i.e., $\mathbb{D}(K_i, K_j) = d$), then the value x_y^{t+1} to be stored, which minimizes the number of operations to reconstruct the hashchain, is when $y = \lfloor d/2 \rfloor$.

Proof: It is impossible to use the value x_j^t to calculate x_i^t because the problem of calculating the inverse of the hash function is intractable. Therefore, the hashchain is reconstructed from the stored value with a lower index x_i^t. Then, the total number of operations T required to reconstruct the hashchain is given by

$$T = \sum_{t=i}^{y} \mathbb{D}\left(x_i^t, x_y^t\right)\Big) + \sum_{t=y}^{j} \mathbb{D}\Big(x_y^t, x_j^t\Big)\Big). \tag{2}$$

For simplicity and without loss of generality, we use $i = 1$ and $j = d$, as $\mathbb{D}(x_t^i, x_t^x) = y - i$ then

$$T = (1 + 2 + \dots + y) + \left[(d - y) + (d - y - 1) + \dots + y\right] \tag{3}$$

$$T = \frac{y(y+1)}{2} + \frac{(d-y)(d-y+1)}{2}. \tag{4}$$

Solving $\partial T/\partial y = 0$, i.e., $2y - d = 0$, results in $y = d/2$ and with $\partial^2 T/\partial y^2 = 2$ (positive), it is the minimum value. ■

V. Security Analysis

This section provides a security analysis of the proposed scheme, analyzing the resilience to potential attacks and providing a formal security analysis. It also develops the analysis of compliance with the requirements for an authentication protocol addressed in Section II-A. A comparison with existing authentication solutions for IoD networks is also provided.

A. Attack Model

In the proposed blockchain-based authentication scheme, the following assumptions are applied to analyze the security against existing attacks.

1) The private blockchain is jointly maintained by the authorized GSs in the whole network so that the transaction and smart contract data are transparent to all participating GSs.
2) Attackers can intercept communication data and compromise system security by impersonating drones or GS. In the worst case, for example, the drone could be taken over and its data acquired.
3) We consider the Dolev–Yao (DY) model, which involves communications over an insecure channel and an untrusted nature between the parties. Thus, an attacker can eavesdrop, modify, and replay messages exchanged over the public channel. With the illicitly acquired information, the attacker can also obtain some ephemeral secrets including secret keys.

Based on the attack model described above the possible attacks that the protocol must resist are listed and analyzed below.

Eavesdropping Attack: An adversary can record all the messages exchanged during the communication in any flight zone and coming from both drones and GSs. This means that the attacker can obtain any message with its respective MAC code and the messages containing the keys used in the previous slots. Suppose that the packet P_i is captured with its respective MAC ($MAC(K_i, P_i)$) and then the keys K_i, which are broadcasted in another packet τ seconds later. Note that if the message is modified using the K_i keys, no receiver will accept the spoofed packet because the temporary key security condition is not met. On the other hand, it could try to calculate the key K_{i+1} that should be fresh. For this, besides having a short time, it would have to calculate the inverse operation to the hash function H since $K_i = H(K_{i+1})$. Note that the probability of success for a brute-force attack to try to compute the inverse hash function is very low. For example, for the SHA-256 hash function, due to birthday attacks, 128-bit security level is guaranteed and eavesdropping attacks can be resisted.

Drone Identity Impersonation: If an attacker wants to impersonate a drone, it must generate valid MAC codes for messages with the correct key corresponding to the current time slot. However, due to the intractability of calculating the inverse of the hash function, the impersonation attack can be prevented with this scheme.

Drone Identity Forgery: An attacker who wishes to generate an illegal MAC code in a packet to fool the verifying party, must use the key corresponding to the current time instant, which results in solving the intractable problem of calculating the inverse of the hash function.

Drone Cloning Attack: An attacker could attempt to clone a drone if he knows the physical address and IP of the drone, but he would not be able to access the hashchain generated in the Setup and Registration process and therefore would not be able to impersonate the identity of the drone. As a consequence, resistance against the Drone Cloning attack is obtained.

Ephemeral Secret Leakage Attack: The attacker can guess one of the keys that will be used in one of the time slots. With this, he could generate valid MAC codes during this time interval. In the next time interval, to generate valid MAC codes he must calculate the next key, for which he would need to solve the intractable problem of calculating the inverse of the hash function. However, the probability of guessing the key in an instant of time is very low given the large size of the search space.

MITM: An attacker could secretly relay and possibly alter communication between drones and GS or other drones. However, as discussed above, it would be impossible for an adversary who does not know the valid key to be used in the current slot to generate valid MAC codes. Thus, the scheme is not susceptible to MITM attacks.

DOS: Because authentication in μTesla is delayed for some time, receivers appear vulnerable to flooding attacks that can cause excessive packets in the buffer, even if they are eventually unauthenticated. In [29], some requirements that guarantee security against DoS attacks are shown. In the proposed scheme, we employ the same mechanisms described in [29] to eliminate the risk of a DoS attack which includes not reusing keys and checking that when a packet arrives if the key index is incorrect, it should not be buffered.

B. Formal Security Analysis

We use a formal method to prove the security of the proposed protocol. The message exchange is modeled by ProVerif [30], which is designed for the analysis of secrecy and authentication properties as well as additional properties, such as privacy, traceability, and verifiability. It has been used for the validation of a large number of protocols [31], [32].

Fig. 4 shows the ProVerif simulation code for the proposed scheme. The *ver_key* function verifies that the security condition holds and that the key corresponds to the current time slot number. Fig. 5 shows the results of the simulations in Proverif. As can be observed, the private key of every slot (*ki*) will not be obtained by the adversary, which formally evidences the security of the proposed scheme.

VI. Performance Analysis

In this section, we present an analysis of the main performance parameters on which the proposed scheme has a direct influence and which are desired to be optimized in an IoD authentication protocol. First, the computational cost and network overhead caused by the proposed scheme is analyzed. In addition, the authentication delay is estimated via simulation when the number of drones and the number of flight zones are increased.

A. Experimental Settings

To evaluate the computation and communication costs of the drones, the computation time of the cryptographic operations involved in our scheme is measured on a Raspberry Pi microcomputer, which has hardware properties similar to a drone. A Raspberry Pi 3B configuration consists of a Broadcom BCM2711, Quad-core Cortex-A72 (ARM v8) 64-bit SoC @ 1.5 GHz, 8G RAM, and 64G ROM. For encryption, we use the *pycrypto* library (version 2.6.1) written in Python language. We use AES128 as the symmetric encryption algorithm, SHA256 as the general hash function, and the code

```
type key .
type timeslot .
free c1: channel.
free c2: channel[private].
free n,ki:bitstring[private].
free slot:timeslot.
free pk:key.

(*---Cryptographic functions---*)
fun H(bitstring):bitstring. (*One-way Hash function*)
fun MAC(bitstring,key):bitstring.
reduc forall m:bitstring, k:key ; getMessage(MAC(m,k)) = m
     .

(* functions for generating and verifying keys *)
fun gen_keys(bitstring,key) : bitstring.
fun get_key(bitstring, timeslot): key.
fun ver_key(key,timeslot,key): bitstring.

(*---Queries---*)
query attacker (ki).

(* Drone A*)
let UAV_Tx(sk:key, Hashchain:bitstring)=

   let ki = get_key(Hashchain,slot) in
   new m:bitstring;
   let HMAC = MAC(m,ki) in
   out ( c1 ,(HMAC,m) );
   out ( c2 ,ki ).

(* Drone B*)
let UAV_Rx(pk:key) =
   in (c1,(ki:key, HMAC:bitstring)) ;
   let Valid = ver_key(ki,slot,pk) in
   let M = getMessage(HMAC) in
   0.

process
new sk : key ;
let Hashchain = gen_keys(n,sk) in
((!UAV_Tx(sk,Hashchain))|(!UAV_Rx(pk)))
```

Fig. 4. Proverif code for the proposed scheme.

```
Verification summary:
Query not attacker(ki[]) is true.
```

Fig. 5. Proverif simulation results of the proposed scheme.

defined in [33] as MAC code, which is computed using

$$MAC(K, m) = H\big(\big((K' \oplus opad)\|H\big(K' \oplus ipad\big)\|m\big)\big). \quad (5)$$

Here, *opad* is the block-sized outer padding, consisting of repeated bytes valued $0x5C$. The variable *ipad* is the block-sized inner padding, consisting of repeated bytes valued at $0x36$. K' is a block-sized key derived from the secret key, K, either by padding to the right with zeros up to the block size or by hashing down to less than or equal to the block size first and then padding to the right with zeros. Like in [9], we use 64 and 384 bytes for the operations on the additive cyclic groups, $\mathbb{G}_1$ and $\mathbb{G}_T$, respectively. Both groups have order q, where q is a large prime number of length 128 bits. The time cost of the average of 1000 executions for each of the cryptographic operations analyzed in this article is shown in Table II. For a confidence level of 95%, this number of runs ensures that the maximum error in the estimation of the metrics is less than 3%.

B. Computational Overhead

To estimate the computational overhead of the proposed scheme, a theoretical analysis of the most time-consuming operations is performed, neglecting lighter operations, such as string concatenation and XOR. Table III shows the results of the proposed scheme for packet authentication and packet verification at one of the receivers. A comparison with existing authentication solutions is also shown, the works used in the comparison were selected based on their good performance in this type of scenario. In our scheme, to authenticate a message, which may or may not have been previously encrypted, it is required to calculate the MAC code of the message (T_{MAC}) by the transmitting drone. To verify the key, it is required to calculate the hash of the previous key (T_H). So, in total, ($T_H + T_{MAC}$) is required to authenticate the message. Note that the previous encryption of the message is not essential for authentication, which could be achieved even if the original message is in plaintext. Thus, if we consider the time to encrypt and decrypt (T_S) in the proposed scheme, a total of ($T_H + T_{MAC} + T_S$) is required. In the third column of Table III, the computation time required to complete authentication, for a security level of 128 bits, is provided.

TABLE II
TIME COST OF DIFFERENT CRYPTOGRAPHIC OPERATIONS

Notation	Description	$Time$(ms)
T_{hm}	Hyper-elliptic curve multiplication	9.899
T_H	SHA256 Hash function	0.026
T_S	AES128 encryption and decryption	1.975
T_m	Scalar Multiplication in $\mathbb{G}_1$	0.031
T_e	Exponentiation in $\mathbb{G}_T$	7.682
T_{bp}	Bilinear pairing in $\mathbb{G}_T$	8.128
T_{MAC}	MAC Code	0.053

TABLE III
COMPUTATION AND COMMUNICATION COST

Schemes	Time Complexity	$Time$(ms)	Comm.($bits$)
[6]	$12T_{hm}$	118.788	1280
[7]	$23T_H + 2T_S$	2.548	2304
[9]	*	12799.200	16912
[12]	$2T_H + 5T_S$	4.927	7168
Ours	$T_H + T_{MAC} + T_S$	1.054	1024

* $14T_H + 14T_m + T_e + 3T_{bp} + 3T_S$

C. Communication Overhead

Since the message length varies depending on the task and application, only the payload related to authentication is calculated. The communication overhead is calculated based on the total number and size (in bits) of messages used for authentication. Note that the MAC code to be added to each message is 256 bits long if SHA256 is used as the hash function. In addition, 256 bits are needed to send the key, thus 512 bits are needed to authenticate a packet. The last column of Table III (Comm.) shows the total number of bits required by some existing solutions. Our protocol requires little information to achieve authentication due to the lightness of the HMAC code.

D. Blockchain Latencies

The latencies of the different smart contracts involved in the protocol are estimated. For this purpose, a blockchain network is implemented using Hyperledger Fabric (v2.2) in

TABLE IV
TIME COST OF BLOCKCHAIN OPERATIONS (S)

Operation	Min	Max	Average
Invoke RegisterUAV	1.342	1.721	1.418
Invoke RevokeUAV	1.541	1.726	1.692
Query for White List	0.364	0.621	0.481
Query for UAV	0.127	0.182	0.150

a Docker environment (v20.10.6). The network consists of five organizations that function as validators and three peers in each organization. The validators process, store, validate transactions, and add new blocks to the blockchain using the consensus mechanism, as well as handle queries from peers. Hyperledger Caliper(v0.4.2) is installed as a benchmark to obtain the average delays for query and writing of 300 independent invocations to the smart contracts (chaincodes in Hyperledger Fabric).

Write latency is the duration from the time the write operation code is invoked to the time the data is uploaded into the blockchain. The query latency is the duration from the time the read operation code is invoked to the time the result is returned. Note that the query operation does not require transaction validation and consensus, and the data can be queried from a local copy of the blockchain. Table IV shows the result of the minimum, maximum, and average time of the main operations of the smart contracts and the queries to the blockchain. The results are consistent with the average block generation time of 2 s level established for the implementation. For this, 10^4 runs of each operation were performed through the API included in the Hyperledger Fabric Python SDK.[1]

We have estimated these parameters using the practical Byzantine fault tolerance (PBFT) consensus mechanism with a transaction size of 64 kB, which is sufficient to transmit key information and update the flight zone of the handover drones. Both the MATLAB simulations and the previous experiments with Hyperledger Fabric were run on a laptop with Intel Core $i7-7700$ CPU 3.60 GHz $\times$ 8, and 16 GB of RAM.

E. Implementation and Performance Evaluation

In this section, we present the results of the simulation of the proposed authentication scheme for a varying number of zones and drones in the network traffic, as well as for different levels of instability of the wireless links. For this purpose, several scenarios are simulated in which the influence of the packet arrival rate of each drone, the key disclosure time, and the probability of packet loss on the average authentication time of each packet and the network throughput are varied. The performance metrics are defined as follows.

1) *Average Message Authentication Delay* (A_d)*:* This corresponds with the average time it takes to authenticate a message, from the time a message is received (T_r^i) until the information to verify the MAC code and the key used to calculate it are obtained at the receiver (at time T_a^i). Denote by n_p the total number of packets, then the average authentication delay is calculated as

$$\bar{A_d} = \sum_{i=1}^{n_p} (T_r^i - T_a^i)/n_p. \tag{6}$$

2) *Average authenticated throughput* (T_p) is defined as the average amount of information per unit of time exchanged in the authenticated packets and is calculated as follows:

$$T_p = n_p * |p^i|/T_S. \tag{7}$$

For the analysis of the performance of the proposed scheme, a discrete-event simulator is developed in MATLAB, using Table III values for the processing times on the drones. Table V shows the simulation setup of the experiments. Cross-domain device authentication is achieved using smart contracts. To measure the efficiency of key management, the latency of the smart contract operation is recorded.

[1] Available: https://github.com/hyperledger/fabric-sdk-py.

TABLE V
SIMULATION SETUP

Parameter	Value
Simulation time	30 minutes
Zones (GS)	$[5, 10]$
Drones/Zone (ρ)	$[5, 10]$
Mobility model	Random
Average Block time	2 seconds
Transaction size	64kB
Packet size	512 Bytes
Data Rate (802.11b)	11 MBps

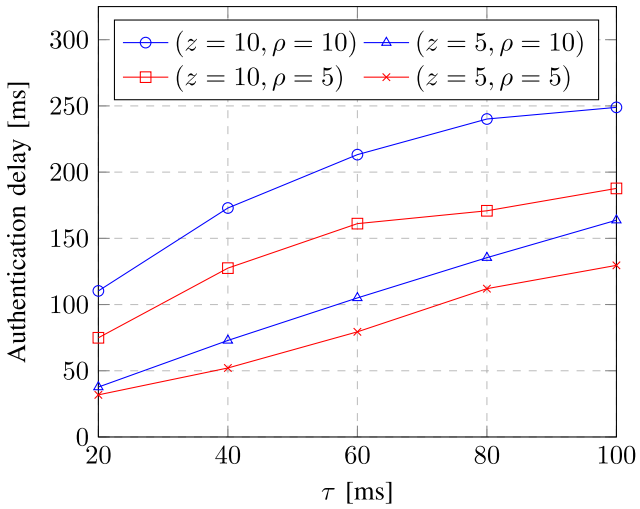


Fig. 6. Average authentication time for different key disclosure times.

F. Average Authentication Delay

Fig. 6 shows the results of the simulations for different scenarios, in which the number of zones (z) and the density (ρ) of drones per zone, i.e., the ratio between the total number of drones and the number of zones, are modified. We here consider four scenarios, corresponding to the values 5 and 10 for ρ and z. For each of these scenarios, we modified the time in which the drone discloses the key (τ), taking in each case a value between 20 and 100 ms with an increment of 20 ms.

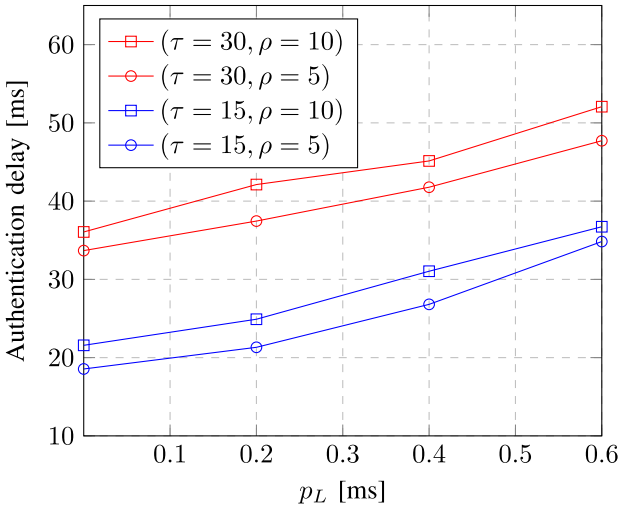


Fig. 7. Average authentication time for different packet loss probabilities (p_L) in a network with ten zones.

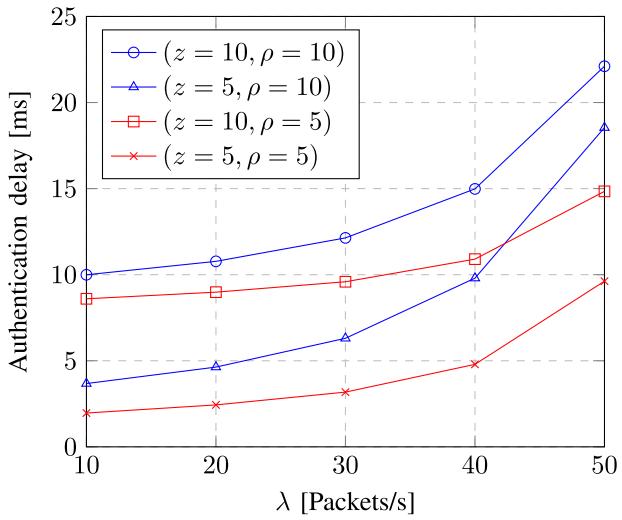


Fig. 8. Average authentication time for different arrival rates.

As shown in Fig. 6, in all scenarios, the average time required to complete the authentication is higher with an increase of τ. This is due to the fact that the receivers must wait longer to receive the key with which the MAC code of each message was calculated.

In addition, for scenarios with equal node density, when the number of zones increases, the authentication time increases because the number of drones will be higher and a higher number of queries to the Blockchain will be required when a previous key is not stored.

Similarly for scenarios with an equal number of total drones, for example, ($z = 10$, $\rho = 5$) and ($z = 5$, $\rho = 10$), both having 50 drones in total, the scenario where there are fewer zones, i.e., ($z = 5$, $\rho = 10$), presents better performance in terms of authentication time. This is because with more zones there is a higher probability that the drones did not have previous communication and do not have a previous key. As a consequence, a query to the Blockchain is necessary and thus requires additional time.

To conclude, for τ values below 100 ms, the authentication time does not exceed 250 ms for each of the four scenarios, which is useful for most drone applications.

1) Impact of Packet Loss: Since μTesla is resistant to packet loss due to the hashchain characteristics of the keys, the impact of packet loss on the average authentication time is studied. To achieve this, the channel stability is varied by modifying the packet loss probability (p_L), taking values between 0 and 0.6 with increments of 0.1 in scenarios with ten zones and different densities (5 and 10), as well as modifying the value of τ, with values 15 (blue lines in Fig. 7) and 30 (red lines in Fig. 7), respectively.

Fig. 7 shows that as P_L increases the average authentication time increases, which is due to the loss of packets. It takes longer for the receivers to receive the respective keys that allow verifying the authenticity of the received packets. On the other hand, for a given value of τ, the higher the density of the network, the higher the number of drones and therefore the higher the authentication time. Although the increase in the probability of packet loss increases the authentication time, the protocol is able to authenticate messages with keys received subsequently, which makes it resilient to the loss of messages containing the keys to verify authentication.

2) Impact of Network Traffic: An important parameter to consider in the performance analysis of every protocol is network traffic. Fig. 8 shows the results of several simulated scenarios in which the packet sending rate varies. We consider again four scenarios with the number of z zones and density ρ equal to {5, 10}. The packet generation of each drone follows a Poisson distribution with rate parameter λ and the probability of observing k events in a time period is calculated using (8). In the simulated scenarios, the value for λ ranges between 10 and 50 in increments of 10 packets/s

$$p(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}. \tag{8}$$

As shown in Fig. 8, as the network traffic increases, the average authentication time increases because the drones must process more packets for authentication. Note also that for two scenarios with the same number of total drones, for example, scenarios ($z = 10$, $\rho = 5$) and ($z = 5$, $\rho = 10$), the scenario where there are fewer zones present a worse performance in the authentication time for values of 40 packets/s or less, but from this value, the scenario with more zones present worse performance. This behavior is due to the fact that with more areas it is more likely that queries are made to the blockchain because the receivers are less likely to store a previous key from a given transmitter.

G. Average Authenticated Throughput

Finally, we analyze the behavior of the Average Authenticated Throughput (T_p) in several scenarios in which the time to discover the key (τ) is varied. Again, we consider the same four scenarios with the number of zones $z \in \{5, 10\}$ and the network density $\rho \in \{5, 10\}$. The value of τ changes in the range of 20 to 100 with increments of 20 ms. As seen in Fig. 9, as τ increases the T_p of the network decreases in all scenarios.

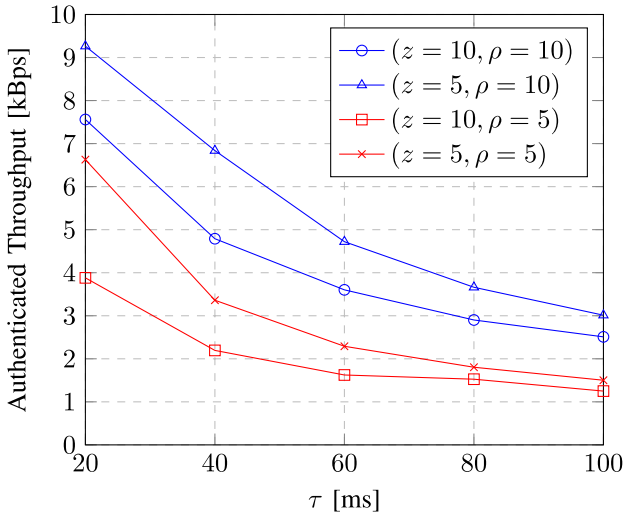


Fig. 9. Average authenticated throughput for different key disclosure times.

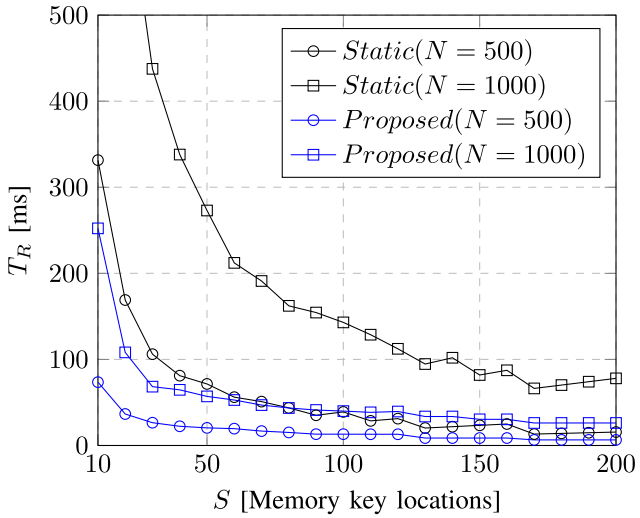


Fig. 10. Number of operations to reconstruct the hashchain for different memory availabilities.

This is because more time is required before authentication of the message becomes possible, affecting the denominator in the calculation of the T_p. Note also that for the same density in the network, for example, scenarios $(z = 10, \rho = 5)$ and $(z = 10, \rho = 5)$, the scenario with a higher number of zones, i.e., $(z = 10, \rho = 5)$, has worse performance in terms of authenticated Throughput. This follows from the fact that as shown above, the increase in the number of zones worsens the authentication time and therefore the amount of information authenticated per unit of time decreases. These results allow the choice of a value of τ that guarantees that a secure condition is maintained with the maximum possible T_p in the network.

H. Storage Limited Scenario

Although μTesla requires little storage (2 kb) of the program and 32 byte keys for 128 bits security levels, in our proposal, it is necessary to store multiple keys considering the total number of packets sent in a mission. In some cases, this may result in insufficient memory available in the drone to store the entire hashchain. In this section, results are presented for scenarios where drones cannot store the entire hashchain and can only store S keys of the total length N of the hashchain. To evaluate the performance of the proposed memory management algorithm, the time required to reconstruct the hashchain from the S values stored in memory is determined. For this purpose, the number of times the hash function is applied when reconstructing the hashchain is calculated and multiplied by the time required to complete a hash operation. To this end, we consider the computed time (T_H) for a Raspberry Pi 3B, shown in Table II.

Fig. 10 shows the results in terms of the Key Reconstruction delay, denoted as T_R, which is applied in order to reconstruct the entire hashchain of length 500 and 1000, which is realistic for a flight of 40 min, allowing sending more than 2 and 4 packets/s, respectively. In each case, the amount of available memory, i.e., S, is varied, taking values between 10 and 200 with increments of 10. We include the comparison with scenarios where static values are stored (without reusing the memory locations). The results of the static and proposed algorithm are shown in the graph with black and blue curves, respectively.

As shown in Fig. 10, as the amount of available memory space increases, the time required to rebuild the hashchain decreases due to the decrease in total number of required hash operations. At the same time, in all scenarios, as the total length of the hashchain increases (from 500 to 1000) the number of operations required to reconstruct the hashchain increases for an identical amount of available memory.

On the other hand, in all simulated scenarios, the proposed solution matches the obtained optimal solution and is better than the solution with static memory utilization, with a minor improvement as the amount of available memory increases. For example, for values of S equal to 50, the proposed algorithm requires 2.5 times fewer operations than if a static allocation is used, but for $S = 200$, the improvement is 23%. These results show that the proposed algorithm allows efficient use of the available memory in the drone in case the storage capacity is very limited.

VII. Conclusion

Ensuring security and privacy in IoD networks is critical to protect data from cyber attacks. However, establishing secure authentication is challenging because drones are power-limited devices with high mobility. Typical authentication solutions present several drawbacks due to their centralized architecture. We use Blockchain technology to address the centralization problem and contribute to improving the security level. We provide a decentralized, secure, and very lightweight authentication protocol based on μTesla and supported by Blockchain to store, manage, and control the information needed to authenticate drone-to-drone communication. We address the packet loss problem in broadcast authentication, which is a novelty considering the existing works. Simulation

and security analysis results show that the proposed solution outperforms other recent authentication solutions while ensuring security.

REFERENCES

[1] M. Yahuza et al., "Internet of Drones security and privacy issues: Taxonomy and open challenges," *IEEE Access*, vol. 9, pp. 57243–57270, 2021.

[2] M. A. Khan et al., "A machine learning approach for blockchain-based smart home networks security," *IEEE Netw.*, vol. 35, no. 3, pp. 223–229, May/Jun. 2021.

[3] Y. Tan, J. Wang, J. Liu, and N. Kato, "Blockchain-assisted distributed and lightweight authentication service for industrial unmanned aerial vehicles," *IEEE Internet Things J.*, vol. 9, no. 18, pp. 16928–16940, Sep. 2022.

[4] O. Samuel, N. Javaid, A. Khalid, M. Imrarn, and N. Nasser, "A trust management system for multi-agent system in smart grids using blockchain technology," in *Proc. IEEE Global Commun. Conf.*, 2020, pp. 1–6.

[5] S. Aggarwal, N. Kumar, and S. Tanwar, "Blockchain-envisioned UAV communication using 6G networks: Open issues, use cases, and future directions," *IEEE Internet Things J.*, vol. 8, no. 7, pp. 5416–5441, Apr. 2021.

[6] M. A. Khan et al., "An efficient and secure certificate-based access control and key agreement scheme for flying ad-hoc networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 5, pp. 4839–4851, May 2021.

[7] M. W. Akram et al., "A secure and lightweight drones-access protocol for smart city surveillance," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 10, pp. 19634–19643, Oct. 2022.

[8] T. Hewa, A. Braeken, M. Ylianttila, and M. Liyanage, "Blockchain-based automated certificate revocation for 5G IoT," in *Proc. IEEE Int. Conf. Commun.*, 2020, pp. 1–7.

[9] C. Lin, D. He, N. Kumar, X. Huang, P. Vijayakumar, and K.-K. R. Choo, "HomeChain: A blockchain-based secure mutual authentication system for smart homes," *IEEE Internet Things J.*, vol. 7, no. 2, pp. 818–829, Feb. 2020.

[10] C. Lin, D. He, X. Huang, K.-K. R. Choo, and A. V. Vasilakos, "BSeIn: A blockchain-based secure mutual authentication with fine-grained access control system for industry 4.0," *J. Netw. Comput. Appl.*, vol. 116, pp. 42–52, Aug. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804518301619

[11] C. Feng, B. Liu, Z. Guo, K. Yu, Z. Qin, and K.-K. R. Choo, "Blockchain-based cross-domain authentication for intelligent 5G-enabled Internet of Drones," *IEEE Internet Things J.*, vol. 9, no. 8, pp. 6224–6238, Apr. 2022.

[12] A. Perrig, R. Szewczyk, J. D. Tygar, V. Wen, and D. E. Culler, "SPINS: Security protocols for sensor networks," *Wireless Netw.*, vol. 8, no. 5, pp. 521–534, 2002.

[13] M. Wazid, A. K. Das, and J.-H. Lee, "Authentication protocols for the Internet of Drones: Taxonomy, analysis and future directions," *J. Ambient Intell. Humanized Comput.*, vol. 31, pp. 1–10, Aug. 2018.

[14] Y. Tian, J. Yuan, and H. Song, "Efficient privacy-preserving authentication framework for edge-assisted Internet of Drones," *J. Inf. Security Appl.*, vol. 48, Oct. 2019, Art. no. 102354. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2214212618307038

[15] M. Barbareschi, V. Casola, A. De Benedictis, E. L. Montagna, and N. Mazzocca, "On the adoption of physically unclonable functions to secure IIoT devices," *IEEE Trans. Ind. Informat.*, vol. 17, no. 11, pp. 7781–7790, Nov. 2021.

[16] A. Perrig, R. Canetti, J. Tygar, and D. Song, "Efficient authentication and signing of multicast streams over lossy channels," in *Proc. IEEE Symp. Security Privacy*, 2000, pp. 56–73.

[17] L. Wang, Y. Tian, and D. Zhang, "Toward cross-domain dynamic accumulator authentication based on blockchain in Internet of Things," *IEEE Trans. Ind. Informat.*, vol. 18, no. 4, pp. 2858–2867, Apr. 2022.

[18] X. Yang et al., "Blockchain-based secure and lightweight authentication for Internet of Things," *IEEE Internet Things J.*, vol. 9, no. 5, pp. 3321–3332, Mar. 2022.

[19] S. Boutalbi, J. C. P. García, and A. Benslimane, "Blockchain-based secure handover for IoT using zero-knowledge proof protocol," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, 2021, pp. 1–6.

[20] Z. Haddad, M. Baza, M. M. Mahmoud, W. Alasmary, and F. Alsolami, "Secure and efficient AKA scheme and uniform handover protocol for 5G network using blockchain," *IEEE Open J. Commun. Soc.*, vol. 2, pp. 2616–2627, 2021.

[21] Y. Yu, Y. Zhao, Y. Li, X. Du, L. Wang, and M. Guizani, "Blockchain-based anonymous authentication with selective revocation for smart industrial applications," *IEEE Trans. Ind. Informat.*, vol. 16, no. 5, pp. 3290–3300, May 2020.

[22] M. Shen et al., "Blockchain-assisted secure device authentication for cross-domain industrial IoT," *IEEE J. Sel. Areas Commun.*, vol. 38, no. 5, pp. 942–954, May 2020.

[23] D. S. Gupta, A. Karati, W. Saad, and D. B. Da Costa, "Quantum-defended blockchain-assisted data authentication protocol for Internet of Vehicles," *IEEE Trans. Veh. Technol.*, vol. 71, no. 3, pp. 3255–3266, Mar. 2022.

[24] A. Yazdinejad, R. M. Parizi, A. Dehghantanha, H. Karimipour, G. Srivastava, and M. Aledhari, "Enabling drones in the Internet of Things with decentralized blockchain-based security," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6406–6415, Apr. 2021.

[25] B. Liu, K. Yu, C. Feng, and K. K. R. Choo, "Cross-domain authentication for 5G-enabled UAVs: A blockchain approach," in *Proc. 4th ACM MobiCom Workshop Drone Assist. Wireless Commun. 5G Beyond*, 2021, pp. 25–30.

[26] C.-M. Chen, X. Deng, W. Gan, J. Chen, and S. H. Islam, "A secure blockchain-based group key agreement protocol for IoT," *J. Supercomputing*, vol. 77, pp. 9046–9068, Feb. 2021.

[27] A. Perrig and J. Tygar, "TESLA broadcast authentication," in *Secure Broadcast Communication: In Wired and Wireless Networks*. Boston, MA, USA: Springer, 2003, pp. 29–53.

[28] G. Choudhary, V. Sharma, I. You, K. Yim, I.-R. Chen, and J.-H. Cho, "Intrusion detection systems for networked unmanned aerial vehicles: A survey," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, 2018, pp. 560–565.

[29] A. Perrig, D. Song, R. Canetti, J. Tygar, and B. Briscoe, "Timed efficient stream loss-tolerant authentication (TESLA): Multicast source authentication transform introduction," IETF, RFC 4082, 2005.

[30] B. Blanchet, V. Cheval, and V. Cortier, "ProVerif with lemmas, induction, fast subsumption, and much more," in *Proc. IEEE Symp. Security Privacy (SP)*, 2022, pp. 69–86.

[31] H. L. Alaoui, A. El Ghazi, M. Zbakh, A. Touhafi, and A. Braeken, "A highly efficient ECC-based authentication protocol for RFID," *J. Sensors*, vol. 2021, Jul. 2021, Art. no. 8876766.

[32] M. T. Damir, T. Meskanen, S. Ramezanian, and V. Niemi, "A beyond-5G authentication and key agreement protocol," in *Network and System Security*, X. Yuan, G. Bai, C. Alcaraz, and S. Majumdar, Eds. Cham, Switzerland: Springer Nat., 2022, pp. 249–264.

[33] H. Krawczyk, M. Bellare, and R. Canetti, "HMAC: Keyed-hashing for message authentication," IETF, RFC 2104, 1997.

Julio César Pérez García received the B.E. degree in telecommunications and electronics engineering and the M.Sc. degree in telematics from the University of Central Las Villas, Santa Clara, Cuba, in 2015 and 2019, respectively. He is currently pursuing the Ph.D. degree in Internet of Things security and blockchain with the Laboratoire Informatique d'Avignon, Avignon University, Avignon, France.

His research interests include distributed technologies, network optimization, and the development of security and privacy protocols.

Abderrahim Benslimane (Senior Member, IEEE) received the B.S. degree in computer science from the University of Nancy, Nancy, France, in 1987, and the DEA (M.S.) and Ph.D. degrees in computer science from the Franche-Comte University of Besaŋon, Besaŋon, France, in 1989 and 1993, respectively.

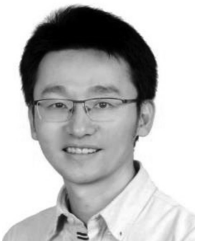
He has been a Full Professor of Computer Science with Avignon University, Avignon, France, since 2001, where he is also the Vice Dean of the Faculty of Sciences and Technology. He was an International Expert with the French Ministry of Foreign and European Affairs from 2012 to 2016. He has been an Associate Professor with the University of Technology of Belfort-Montbéliard, Belfort, France, since September 1994. He has more than 280 refereed international publications (books, conference proceedings, journals, and conferences) and more than 20 special issues. All publications are in my research topics. He supervised more than 20 Ph.D. thesis and more than 42 M.Sc. research thesis. For more details, see my complete CV: http://abderrahimbenslimane.org/.

Prof. Benslimane obtained the title to supervise research (HDR 2000) from the University of Cergy-Pontoise, Cergy, France. He has the French Award for Doctoral Supervision and Research in 2021–2025. He has been nominated as the IEEE ComSoc Steering Chair of Multimedia TC from 2022 to 2024. He is the Past Chair of the ComSoc Technical Committee of Communication and Information Security from 2017 to 2019. He is currently the Editor-in-Chief of *International Journal of Multimedia Intelligence and Security* (Inderscience), an Editorial Board Member of IEEE INTERNET OF THINGS JOURNAL, and an Editorial Member of IEEE TRANSACTIONS ON MULTIMEDIA, *IEEE Wireless Communication Magazine*, IEEE SYSTEMS JOURNAL, *Ad Hoc Networks* (Elsevier), and *Wireless Networks* (Springer). He is the Co-Founder and has been serving as the General Chair/Steering Chair of the IEEE WiMob since 2005. He served as the General Chair of IEEE CNS 2020, an Executive Forum Co-Chair at IEEE Globecom 2020, the Program Vice Chair of IEEE TrustCom 2020 and iThings 2020, and the Symposium Co-Chair/Leader in many IEEE international conferences, such as ICC, Globecom, iCoST, MOWNET, AINA, and VTC. He has been nominated in 2020 as an IEEE VTS Distinguished Lecturer.

An Braeken received the M.Sc. degree in mathematics from Gent University, Ghent, Belgium, in 2002, and the Ph.D. degree in engineering science from the Research Group Computer Security and Industrial Cryptography (COSIC), KU Leuven, Leuven, Belgium, in 2006.

She became a Professor with the Erasmushogeschool Brussel, Vrije Universiteit Brussel, Brussels, Belgium, in 2007, where she has been with the Department of Engineering Technology since 2013. She is the (co)author of over 200 publications. Her current interests include security and privacy protocols for IoT, cloud and fog, blockchain, and 5G security.

Dr. Braeken has been a member of the program committee for numerous conferences and workshops, such as LANMAN2023, ICWMC 2023, and ICNS 2023 and has been a member of the editorial board for Security and Communications Magazine. She has also been a member of the organizing committee for the IEEE IECON 2023 Conference, Cloudtech 2018 Conference, and the Blockchain in IoT Workshop at Globecom 2018. She has cooperated and coordinated more than 12 national and international projects. She has been an STSM Manager in the COST AAPELE Project (2014–2017) and a member of the management committee of the COST RECODIS Project (2016–2019) and COST OPENSENSE Project (2021–2025).



Zhou Su (Senior Member, IEEE) received the Ph.D. degree from Waseda University, Tokyo, Japan, in 2003.

He has published technical papers, including in top journals and top conferences, such as IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS, IEEE TRANSACTIONS ON INFORMATION FORENSICS AND SECURITY, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING, IEEE TRANSACTIONS ON MOBILE COMPUTING, IEEE/ACM TRANSACTIONS ON NETWORKING, and INFOCOM. His research interests include multimedia communication, wireless communications, and network traffic.

Dr. Su received the Best Paper Award of International Conference IEEE ICC 2020, IEEE BigdataSE 2019, and IEEE CyberSciTech 2017. He is an Associate Editor of IEEE INTERNET OF THINGS JOURNAL, IEEE OPEN JOURNAL OF THE COMPUTER SOCIETY, *IET Communications*, and *IEICE Transactions on Communications*.