

Article

Not peer-reviewed version

DFly: A Publicly Auditable and Privacy-Preserving UAS Traffic Management System on Blockchain

[Frederico BAPTISTA](#), [Marina DEHEZ-CLEMENTI](#)*, [Jonathan DETCHART](#)

Posted Date: 16 July 2024

doi: 10.20944/preprints202407.1340.v1

Keywords: U-Space; UAS Traffic Management; Public blockchain; Zero-knowledge proof; Auditability; Privacy-preservation; Security



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Article

DFly: A Publicly Auditable and Privacy-Preserving UAS Traffic Management System on Blockchain

Frederico Baptista, Marina Dehez-Clementi *  and Jonathan Detchart 

ISAE-SUPAERO, Toulouse, France

* Correspondence: marina.dehez-clementi@isae.fr

Abstract: The integration of Unmanned Aircraft Systems (UAS) into the current airspace poses significant challenges in terms of safety, security and operability. As an example, the European Union has defined in 2019 a set of rules to support the digitalization of UAS Traffic Management (UTM) systems and services, namely the U-Space regulations. Current propositions opted for a centralized and private model, concentrated around governmental authorities (e.g., AlphaTango provides the Registration service and depends on the French government). In this paper, we advocate in favor of a more decentralized and transparent model in order to improve safety, security and operability among UTM stakeholders. As such, we propose DFly, a Publicly Auditable and Privacy-preserving UAS Traffic Management system on Blockchain, and developed two initial services: Registration and Flight Authorization. We demonstrate that the use of a blockchain guarantees the public auditability of the two services and corresponding service providers' actions. In addition, it facilitates the comprehensive and distributed monitoring of airspace occupation and the integration of additional functionalities (e.g., the creation of a live UAS tracker). The combination with zero knowledge proofs enable the deployment of an automated, distributed, transparent and privacy-preserving flight authorization service, performed on-chain thanks to the blockchain logic. In addition to its construction, this paper details the instantiation of the proposed UTM system with the Ethereum Sepolia's testnet and the Groth16 ZK-SNARK protocol. On-chain (gas cost) and off-chain (execution time) performance analyses confirm that the proposed solution is a viable and efficient alternative in the spirit of digitalization and offers additional security guarantees.

Keywords: U-Space; UTM; public blockchain; auditability; zero-knowledge; airspace management; data collection; security; privacy

1. Introduction

Updated and efficient manufacturing processes along with the integration of cutting-edge technologies have spawned the democratization of the use of Unmanned Aircraft Systems (UAS) and paved the way for military but more importantly civilian and commercial applications over the years [1–3]. Safety concerns rise along with the increased number of UASs in the airspace [4].

Thus, efforts have been deployed to monitor, control, and automate UAS Traffic Management (UTM) which is currently in a state of transformation. In the United States, the National Aeronautics and Space Administration (NASA) has partnered with the US Federal Aviation Administration (FAA) to establish a UTM system for the safe integration of UASs into low-altitude airspace and the management of drone traffic [5]. The quality of these UTM systems is highly dependent on users obeying the rules. The recent introduction of U-Space regulations by the European Union Aviation Safety Agency (EASA) is set to change this paradigm [6] as policies will likely leverage new drones' capabilities to their advantage.

Due to the intertwined nature of safety and security in Cyber-Physical Systems like the UTM system, ensuring drones' security has become as essential as providing safety guarantees [1]. Existing UTM constructions, like EuroDRONE [7], are leaning towards centralization which reinforce distrust against trusted third parties, like the Service Providers defined by the U-Space regulations, due to the lack of operational agility and transparency. In this paper, we explore the decentralized paradigm to contribute to existing framework like the one proposed by Lappas et al. [7]. Indeed, the absence of single point of failure make decentralized systems more resilient to failures and attacks compared to

centralized ones; in addition, they experience greater resilience and are less prone to the concentration of data in the hands of a few powerful entities. However, one of the key challenges in decentralized systems is to foster trust among untrustworthy participants.

The Blockchain technology (BCT) addresses the trust issue by providing three essential properties: Distribution (data are disseminated to all the nodes), Transparency (data are verified by all the nodes), and Immutability (validated data are recorded in an immutable ledger). The answer to this issue is also augmented by the unique combination of its data structure and consensus mechanisms.

Since 2009, the BCT has evolved from cryptocurrency applications [8] to viable solutions that improve security and privacy levels in various domains such as Healthcare [9], the future of IoT [10], and research towards blockchain interoperability suggests that the use of blockchains is broader than just monetary [11]. In particular, several works have proposed to use blockchains in the context of aviation. They can be classified into two main categories: theoretical works and practical implementations that adapt the blockchain concept to data collection [12–15], and theoretical/practical proposals to embed blockchain nodes within flying objects [16–18].

Our work falls into the first category. In this paper, we propose the foundations for UAS-specific services to be deployed securely and in a privacy-preserving yet publicly auditable way (e.g., coordination between commercial UAS services on top of blockchains; dynamic path construction for collision-free movement of UASs, cooperation and fast synchronization [2], autonomous UAS networks with traceability and privacy-preservation, NDN-based data delivery like in [19]). We propose a blockchain-based UTM system that is compliant with U-Space regulations in terms of drone registration and flight request management while enabling competent authorities to have control over the airspace occupation and monitor the evolution of the airspace conditions. Unlike previous works, we acknowledge the necessity of hiding information related to special (e.g., commercial or military) operations. As such, the system supports standard and special flights, meaning that both civilian and commercial/military UAS flights can be monitored within the same system, which increases safety and operability. We complement the construction with an evaluation of its instantiation in terms of gas consumption and execution time.

The remainder of the paper is organized as follows. Section 2 presents five of the most representative works on blockchain-based systems for drones' operations. Section 3 presents some background knowledge on the U-Space regulations, blockchains, and zero-knowledge proofs. Section 4 presents the scenario and explains our design methodology. Section 5 describes the instantiation of our blockchain-based construction. Section 6 is dedicated to the performance analysis of our proof-of-concept. Section 7 gives the security arguments. Section 8 expands on future work. And finally, Section 9 concludes the paper.

2. Related Work

Khan et al. (2021) [12] proposed a blockchain approach to fog computation, focusing on drone management in land changes. Their system allows for a safe and distributed transfer of the fog node data while being able to register and manage the drones collecting the data via blockchain and smart contracts (i.e., the blockchain serves as truth keeper for cloud transactions and drone registration). While improving trust in a large distributed system, we do infer, from the available implementation, that the current registration process is highly costly to deploy on public and more established blockchains like *Ethereum*. In addition, there is no consideration to confidential flight operations, which makes their registration process too specific for being applied in a broader commercial context.

The work of Ralitera and Gurcan (2022) [13] focuses on the Beyond Visual Line of Sight (BVLOS) drone flights and proposes to include blockchain into the data flow of BVLOS operations. They develop a blockchain-based service architecture composed of ground stations (full nodes) and drones (lightweight nodes). In order to track registered drones, authors take advantage of the use of trust graphs to register entities (e.g., drones) that can perform flights and data transfers. The role of these trust graphs is similar to the role that Merkle trees take in our proposed architecture.

Closer to our work, Alkadi and Shoufan (2023) [14] propose a decentralized solution based on *Ethereum* and smart contracts to provide confidentiality, integrity, and availability to the drone flight data. However, the implementation may suffer from the same problem as [12]. Indeed, the use of arrays increases not only on the gas cost of the solution per se but also the difficulty of iterating through the data (i.e., the solution does not scale). Compared to their work, our solution is more scalable and cheaper.

The emerging frameworks of UASChainSec by Kacem [15] and UTM-Chain by Allouch et al. [4] offer novel insights into the use of blockchain technology to improve the security and operational efficiency of UAS communications and traffic management, respectively. UASChainSec’s contribution lies in its provision of a secure communication channel for 5G-capable UAS, employing a cloud-hosted permissioned blockchain that not only streamlines cryptographic key distribution but also significantly enhances security against common cyber threats. This aligns with the imperative for scalable, secure communications infrastructure as drone fleets expand and their applications diversify. On the other hand, UTM-Chain proposes a Hyperledger Fabric-based framework focused on securing unmanned traffic management, particularly for low-altitude UASs. It underscores the importance of secure, unalterable traffic data sharing between UASs and control stations, a critical factor for ensuring public safety and operational reliability in increasingly congested skies. Both frameworks underline the evolving landscape of drone technology, stressing the significance of innovative, blockchain-powered solutions in overcoming the multifaceted challenges of scalability, security, and regulatory compliance. However, both works leverage permissioned blockchains, more specifically [4] implementing Hyperledger Fabric, which is very centralized.

In this paper, we present a design based on the *Ethereum* blockchain, which is a permissionless blockchain. We propose a performance evaluation of our solution, mainly in terms of gas costs and processing time of cryptographic activities (namely related to the ZKPs and Merkle Tree) and compare ourselves to the closest existing work proposed in Alkadi and Shoufan [14]. This analysis enables us to derive main conclusions on the scalability and the feasibility of the proposed architecture.

Table 1. Comparative summary of existing contributions.

Ref.	BC Type	Solution	Storage	Services	Properties
[12]	Private	NC	on-chain	Registration	Security and privacy issues related to fog-cloud-based nodes
[13]	NC	NC	on-chain	Registration	Secure and trustworthy exchanges of data, traceability and synchronization
[15]	Private	Hyperledger Fabric	on-chain	Registration	Resistance to Denial of service, Man-in-the-middle, ADS-B and internet-based attacks
[4]	Private	Geth	off-chain	Telemetry, Request mission	Availability, Flight Data Integrity, Privacy
[14]	Public	Ethereum	on-chain	Registration, Flight Authorization	Confidentiality, Integrity, Availability, Non-repudiation, Authentication and Authorization
Our	Public	Ethereum	off-chain	Registration, Flight Authorization	Public auditability (thus, authentication, non-repudiation, authorization, integrity, traceability), privacy (thus, confidentiality and data protection)

3. Preliminaries

In the following section, we present some background knowledge on the U-Space regulations, blockchains, and zero-knowledge proofs.

3.1. U-Space Regulation

Following up on the problems of scalability of the previous regulations, the EU has decided to present a new set of regulations that can withstand the Industry development of the upcoming years. U-Space is characterized by a set of services and procedures designed to ensure a safe and traceable access to airspace for a large number of drones, and which are based on high levels of digitization and automation. In this article, we focus on two services: *Registration* and *Flight-Authorization* [6].

3.1.1. Registration

If a drone is certified, it must be registered meaning that the UAS operator must declare it to the competent authorities. The REGULATION (EU) 2021/664 [20] specifies that member states are mandated to establish and maintain accurate registration systems for UAS; in France, the system is called AlphaTango [21]. In this context, UAS operators are asked to provide personally identifiable information such as their full name and date of birth, their postal and email addresses, phone number, or even insurance policy. Upon successful registration, the UAS operator has to display their registration number on their UASs.

3.1.2. Flight Authorization Service—Generalities

For the provision of this service, EASA introduces two new authorities into the system: the *Common Information Service Provider* (CISP) and the *U-Space Service Provider* (USSP). The CISP should become the single and reliable common source of information that will guarantee the spreading of the necessary information required to enable the operation of every U-Space airspace (left out of our scope). Accordingly, U-Space services will be locally provided by the USSPs. Each USSP is assigned to one specific area in the airspace where it is in charge of providing all the mandatory services, including the Registration and Flight Authorization services.

3.1.3. Flight Authorization Service—Description

This service requires a two-way communication channel between the UAS operator, denoted *user*, and the Service Provider, identified as the *approver*, who analyses the legality of a flight request. The regulation defines a workflow for this service that is detailed in Figure 1:

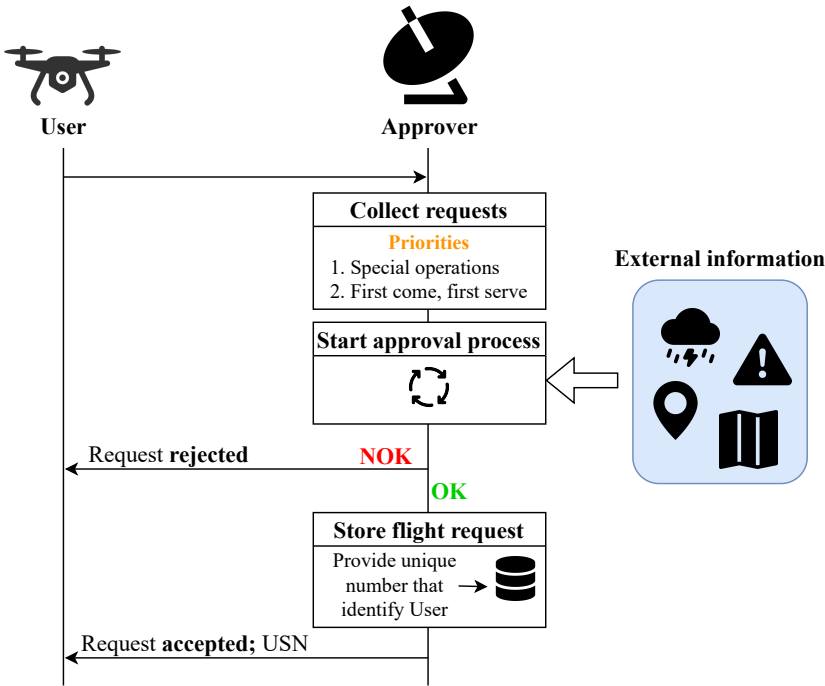


Figure 1. Workflow for Flight Authorisation Service

For a flight authorisation service to be approved, it has to provide the necessary information provided in Annex IV of the REGULATION (EU) 2021/664 [20]. According to the latter document, we provide a summarized table of the required data in Table 2.

Table 2. Required data to issue a flight request according to Annex IV of the REGULATION (EU) 2021/664 [20].

Action/Actor	Origin node
Flight Authorisation Request by UAS Operator	Unique SN of UAS
	Mode of operation
	Type of Flight
	Category of UAS operation
	4D trajectory
	Identification Technology
	Expected connectivity methods
	Endurance
	Applicable emergency procedure in case of loss of command and control link
	UAS operator number

Given the depth of the information provided in a flight request, and considering its complexity and core role in the realm of the U-Space services, we do believe that a blockchain approach can vastly help the distribution of information between USSPs, as the information will be safely stored while also being available for approved users only, and the automation of its processing. In addition, saving data on the blockchain helps with the task of implementing the other U-Space services in a decentralized way.

3.2. Limitations of Using Blockchains

As explained in Section 2, blockchains have been used to implement parts of UTM systems. The closest proposition, to which we compare our work to, is Alkadi and Shoufan's [14]. However, we observe two main limitations to their design that are inherent to the decentralized and public nature of blockchains and its usage: every data stored in this public ledger is visible by everyone. Thus,

1. In order to work properly (especially during the Flight Request verification), all necessary fields of information are stored on-chain, which induce great transactional costs and reduce the efficiency of the decision-making process.
2. Data stored on-chain are immutable. Therefore, it is impossible (with the current version of Ethereum) to erase data when permissions are changed or, more specifically, when the users' right to be forgotten [22] is expressed.

In this paper, we want to improve Alkadi and Soufan's proposition. We leverage the blockchain as a tool for increased auditability of the decentralized UTM system we propose while reducing the cost related to the use of blockchain and, proposing a confidentiality and privacy-preserving mechanism for the public verification of sensitive data without disclosing it. That is why we introduce in our new design the use of zero-knowledge proofs.

3.3. Zero-Knowledge Proof

Defined in [23], Zero-Knowledge Proof (ZKP) is a way of proving the correctness of a statement without revealing the statement itself.

In [24], the authors proposed a non-interactive zero-knowledge proof construction (NIZK). Such a system requires only one round of communication between participants. A *prover* passes the secret information to a special algorithm to compute a ZKP. This proof is sent to a *verifier*, who checks that the *prover* knows the secret information using another algorithm. Non-interactive proving reduces communication between the *prover* and *verifier*, making NIZK proofs more communication efficient.

Moreover, once a proof is generated, it is available for anyone else (having access to the shared key and verification algorithm) to verify.

Among the ZKPs, there are verifiable computation (VC) schemes. These constructions allow one to prove the correct execution of a program to a verifier without having to redo the computation. The first practicable system is [25]. Among all the verifiable computation protocols, there are two families used to build verifiable computation proofs. They work in the same way with some different nuances: **SNARKs** and **STARKs**. The first produces smaller proofs that are easier and cheaper to verify, while the latter provides longer proofs and more complex to verify, but more secure (post-quantum) given the fact that these proofs only rely on hash functions. Moreover, it is possible to add zero-knowledge on top of **SNARKs** and **STARKs** in order to hide some or all the inputs of the program to the verifier. They are respectively called **ZK-SNARKs** and **ZK-STARKs**.

In the proposed architecture, ZK-SNARKs [26] will be employed to prove the authenticity of UASs. More specifically, we chose the Groth16 ZK-SNARK protocol [27] because the proof size and the computation verification process are constant no matter the circuit's complexity. The authenticity of UAS is crucial for security and regulatory compliance, yet it is equally essential to maintain the privacy of sensitive UAS-related information. With ZK-SNARKs, it is possible to confirm the authenticity of a UAS, including the verification of metadata such as the country of origin or the mode of operation, without disclosing any proprietary or sensitive data about the system, its controls, or its operations. This approach ensures compliance with necessary regulatory checks without compromising the privacy of proprietary information. Furthermore, the use of NIZK proofs simplifies this process, by reducing both the need for constant communication between the *prover* and *verifier*, and the amount of personally identifying information to share, thus making the system more efficient and the proof easily verifiable by anyone with access to the shared key and verification algorithm. The balance of security, privacy, and efficiency makes verifiable computation proofs a highly suitable technology for UAS authentication in this architecture.

To design ZKP systems, we need to construct a circuit representing the program. This circuit is used to guarantee that the chosen constraints are met ensuring the correct execution. To write such a circuit, we used *Circom* [28] that is a *domain-specific-language* (DSL) and a compiler that allows programmers to design and create their own arithmetic circuits for ZK purposes. *Circom* language is designed as a low-level circuit language, close to the design of electronics that allows the programmers to define the constraints of an arithmetic circuit in a friendly way. In the case of the proposed architecture, it will be used as a tool to compute a ZKP correspondent to a Merkle tree insertion [29].

4. System Overview

The ultimate goal of the proposed architecture is to alleviate the role of the Service Provider and reduce the related trust assumptions by automating the Registration and Flight Authorization services. As a consequence, we will explain in Section 7 that the proposed system guarantees public auditability of the developed UTM services in a privacy-preserving way (w.r.t. the users). With the proposed architecture, every time a flight request is submitted, the data listed in Table 2 are publicly emitted on the blockchain with some exceptions that will be explored in more details in the following subsections.

Figure 2 illustrates the use case diagram of the proposed U-Space-compliant Blockchain-based ZKP-enhanced UTM System. The registration service supports the registration of a user enabling them to become part of the system (i.e., becoming one of the valid entities and acquiring the corresponding role within the system according to Section 4.1). The registration service operates off-chain via a secure channel between the user and a service provider node. Yet, the service provider commits on-chain to the former conversation after treatment of the registration request. The user then communicates exclusively with the blockchain network to access the Flight Authorization service. By calling the functions that will be defined in Section 4.5, the user can issue a flight request without the intervention of any trusted third party. The validation of their flight request is decentralized, automated, and privacy-preserving to the users of the proposed UTM system. In the following sub-sections, we define

the entities and roles, adversarial model, and security guarantees, and present the sequence diagrams of both services.

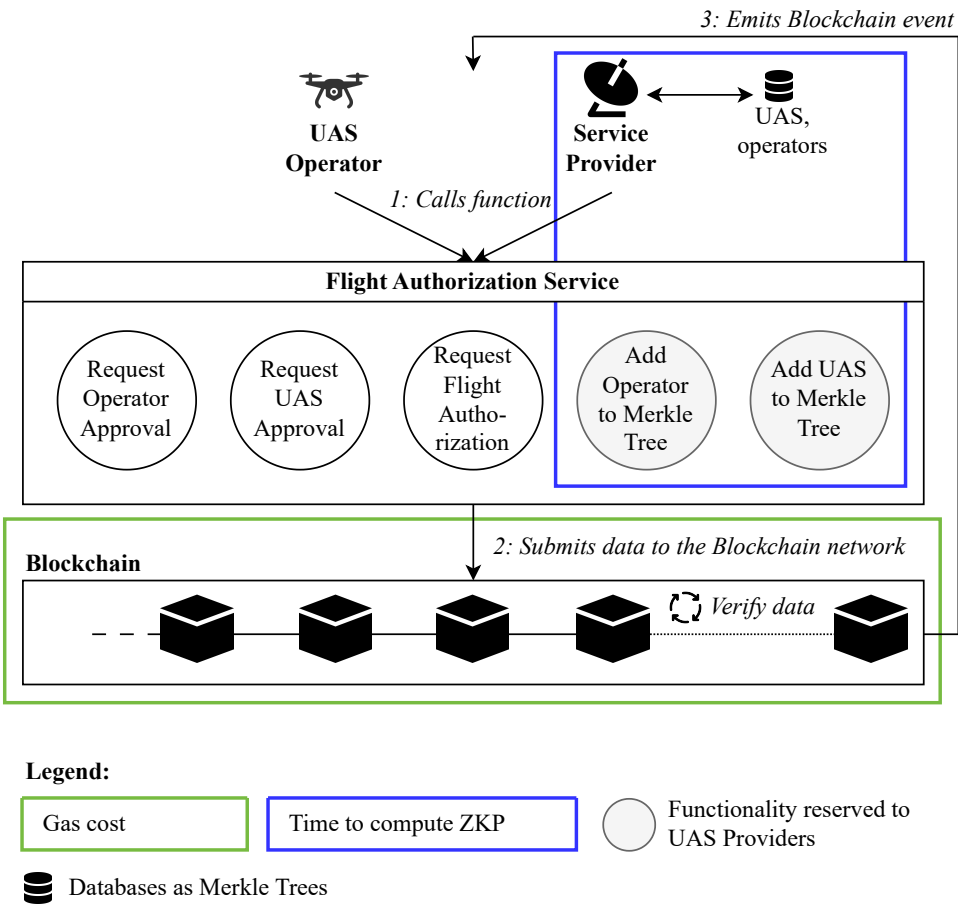


Figure 2. Use-case Diagram of the proposed U-Space-compliant Blockchain-based ZKP-enhanced Flight Authorisation Service and identification of the items relevant for performance evaluation.

4.1. Entities and Roles

The system’s architecture involves the following entities (Figure 2):

- **UAS operator:** interacts with the *Service Provider* through off-chain communications. An *UAS operator* is characterized by their operator number and the corresponding approval ZKP.
- **Service provider:** is the owner of the service w.r.t. all legal effects, is responsible for the maintenance and correct functioning of the system. In the proposed system, it approves the legality of the UASs and operators, and manages the data thanks to Merkle trees.
- **Blockchain:** represents the set of block producers and smart contracts deployed for the Flight Authorisation Service to be functional with all the desired functional and security properties.

4.2. Adversarial Model

In this paper, we opt for a semi-distributed model. We acknowledge the necessity of granting the access right control to the USSPs for legal stakes.

- **UAS operators** are considered malicious: they can submit fake registration or flight authorization requests.

- **Service providers** are considered honest. However, we will explain throughout the Sub-section 4.5 how minor functional additions in the current architecture can make the system robust against malicious Service Providers who deviate from the protocol (e.g., refusing to address a valid registration requests, transmitting false information to the users).
- **Block producers** are the ones provided by the Ethereum network. Thus we adopt the same security model as the underlying technology we used (e.g., for Proof-of-Work-based Ethereum, we assume that at least 51% of the computational power is controlled by honest nodes [8]).

4.3. Security Guarantees

The proposed blockchain-based ZKP-enhanced UTM system offers two main security guarantees. They are described as follows.

First, we defined the *auditability* property as the ability to track, verify, and validate all transactions and data entries recorded in the proposed UTM system. This property ensures that all actions related to the registration and flight authorization services performed can be reviewed and assessed by authorized parties.

Proposition 1 (Public auditability). *The proposed blockchain-based ZKP-enhanced UTM system guarantees that the UTM services, namely the Registration and the Flight Authorization services, are publicly auditable.*

Then, we define the *privacy* property as the protection of UAS and operators' data from unauthorized access, ensuring that sensitive information remains confidential and is only accessible to those with the appropriate permissions.

Proposition 2 (Privacy-preservation of sensitive data). *The proposed blockchain-based ZKP-enhanced UTM system guarantees the privacy-preservation of sensitive data in the use of UTM services, namely the Registration and the Flight Authorization services.*

4.4. Types of Flight Request

There exist 12 possible configurations for a flight as shown in Figure 3. These configurations depend on the Operation Mode (either Open, Specific or Certified), the Flight Category (either VLOS or BVLOS), and the Flight Type (either RegularOps or SpecialOps). For each attribute marked in gray, a ZKP is necessary as the configuration requires additional verification. Per default, each UAS operator must provide a proof that their UAS is registered (Operation Mode's proof). We chose to use a ZKP for several purposes:

1. It protects the operator's sensitive data (namely the Personally Identifying Information as defined in the General Data Protection Regulations [22]) during the use of the Flight Authorization service.
2. It reduces the amount of data to be transacted and stored in the blockchain for the Flight Authorization service (hence, increasing the system's efficiency in the long run).
3. It supports the verification of additional constraints on the underlying attributes (e.g., the country of origin based on the UAS operator's identification number) without revealing them.

As an example, for Visual Line of Sight (VLOS) flights, the U-Space regulations do not imply extra levels of verification, unlike Beyond VLOS (e.g., compatible license). In which case, the UAS operator is asked to submit a ZKP that they own such authorization without revealing their identity. This applies to Regular and SpecialOps. The same reasoning applies to the SpecialOps flight type.

By combining the different attributes in a sequential way, we obtain a total of 12 possible configurations, which were all tested in Section 6.

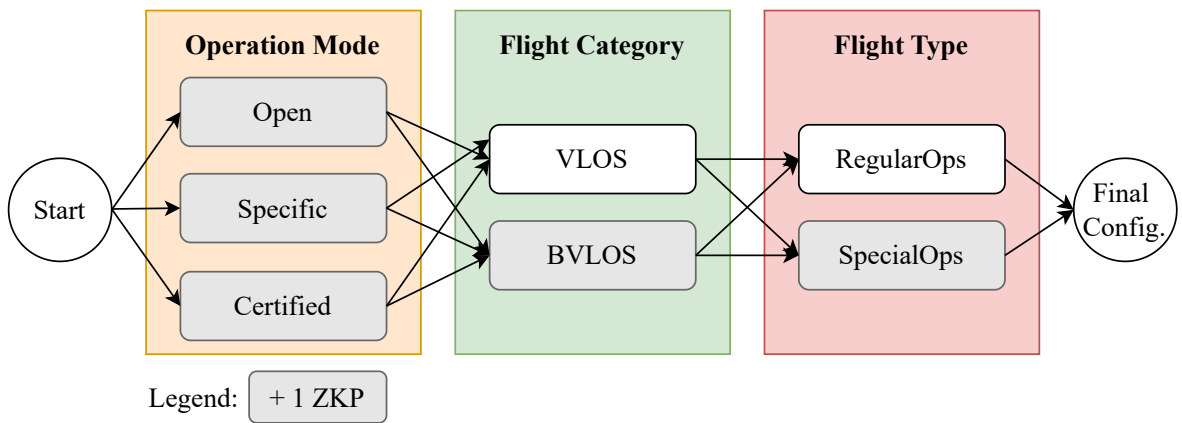


Figure 3. All possible configuration combinations of a Flight Request

4.5. Workflow

Figures 4 and 5 illustrate, respectively, the Registration and Flight Authorization services. With respect to the U-Space regulations, we acknowledge the necessity of maintaining a legal authority, here the Service Provider, in the system. As stated in Section 4.2, in the following description we consider the Service Provider to be honest. However, we will elaborate on how to reduce this trust assumption.

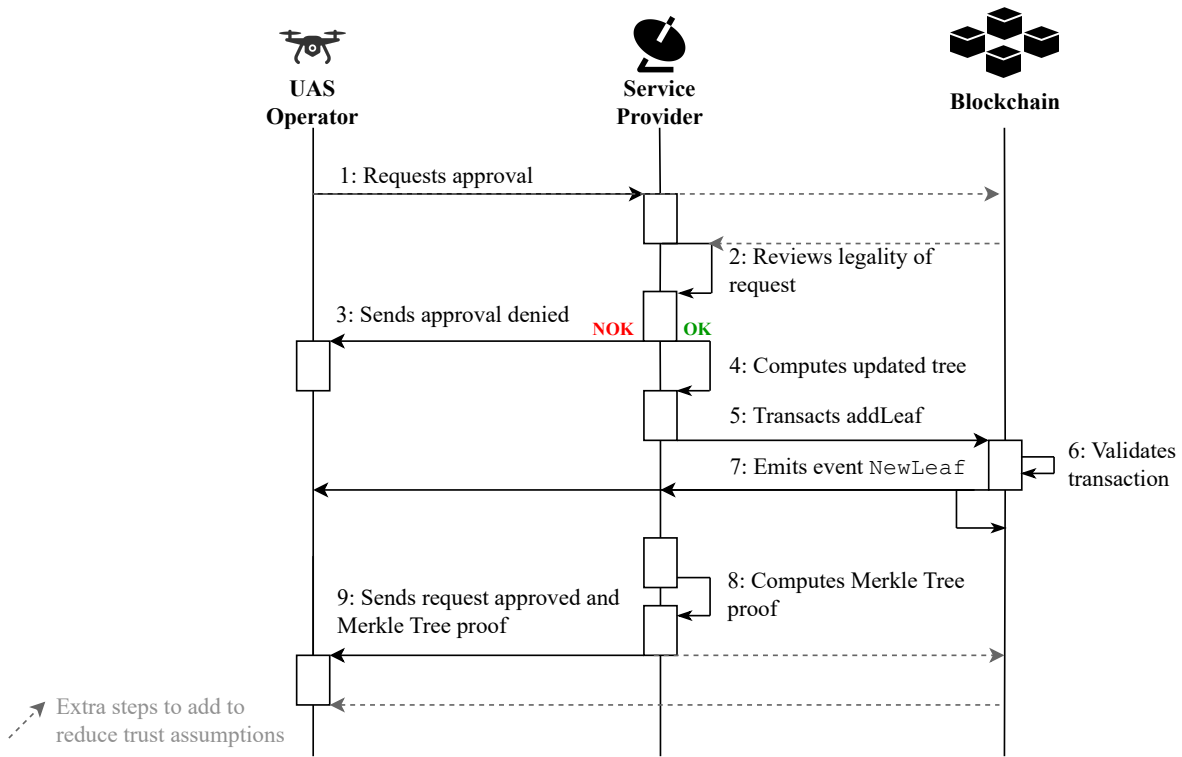


Figure 4. Sequence Diagram of the Request UAS/Operator functionality

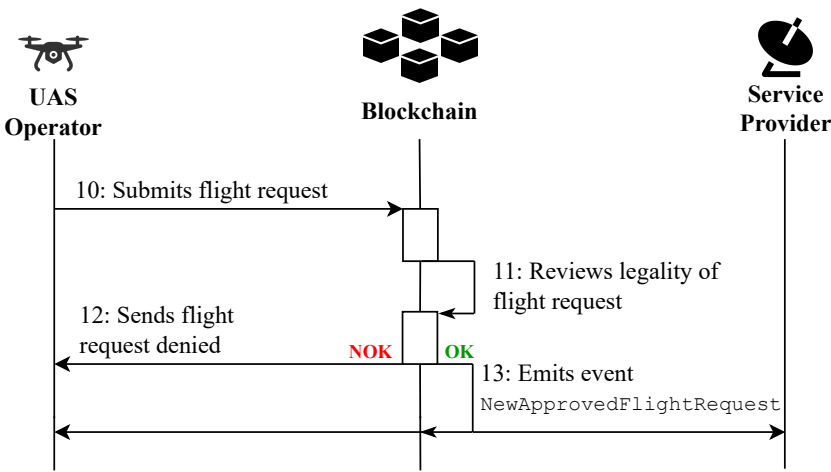


Figure 5. Sequence Diagram of the Flight Request functionality

Registration of UASs/UAS Operators.

Figure 4 illustrates the first service: the *Registration* process for UAS and their operators. The service provider is in charge of maintaining the various Merkle trees used to store authentication information (Figure 6).

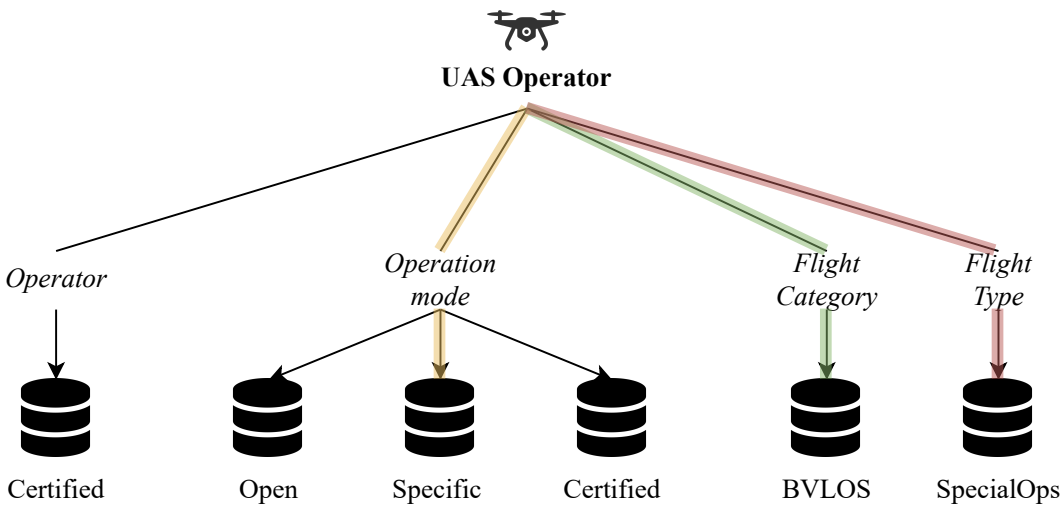


Figure 6. Management of different Merkle trees in the architecture

① First, the UAS operator issues a request for approval and submits it via a private and secure communication channel to the service provider. Being honest, the service provider does not deviate from the protocol and ② correctly reviews the legality of the request. ③ In case the request is invalid, the service provider rejects it. Otherwise, ④ it computes the updated tree corresponding to the categories (Figure 3) selected in the request. ⑤ The addition of the new element is passed onto the chain with a transaction. ⑥ The transaction is reviewed by the block producers within the blockchain network and ⑦ upon block validation, a notification is emitted. Finally, ⑧ the service provider computes the Merkle Tree proof of membership for the requesting UAS Operator and ⑨ sends it back to them in the ‘request approved’ notification.

Remark:

A way to reduce the trust assumptions regarding the Service Provider during the Registration service would be to initiate the registration request and to retrieve the Service Provider’s response via a blockchain transaction. By adding a voting-based exclusion/reporting mechanism, we can also detect malicious providers and exclude them.

Flight Request.

Figure 5 illustrates the second functionality: the flight authorization service.

⑩ The approved UAS operator starts by submitting their flight request to the blockchain network. To this end, they join the required number of proofs of membership according to the type of flight request to the list of mandatory data to fill the request. ⑪ The blockchain contains all the logic for verifying the legality of the flight request, i.e., for verifying the correctness of the submitted data, including the proofs of membership. The verification of the proof of membership implies not only to verify that the UAS belongs to the correct merkle trees, but in addition, due to the specific format of the proofs, the verification that the UAS’ attributes (e.g., UAS operator’s serial number, country of origin) respect the U-Space regulations (e.g., correct format, authorized country). After verification, the blockchain takes a decision, either ⑫ it rejects the flight request or ⑬ accepts it.

5. Implementation

As the verifiable computation proof needs to be verified on the blockchain, through a smart contract, we decided to implement our system on the Ethereum blockchain. Off-chain data management has been done in Python and TypeScript and uses SnarkJS to generate proofs before verifying them on the blockchain. In this section, we describe the instantiation of both the drone registration and the flight authorization services. Moreover, we explain how we use verifiable computation to ensure the privacy and and improve efficiency while asserting the conformity of UAS attributes to U-Space rules.

Table 3 specifies the different functions, smart contracts, and events.

Table 3. Transaction Table.

Contract	Origin node	Transaction name	Event name
FlightAuth.sol	User	createFlightRequest	NewFlightRequest OR NewPrivateFlightRequest
MerkleTree.sol	Provider	createTree	NewTree
		addLeaf	NewLeaf
		deleteLeaf	DeleteLeaf
		updateTree	UpdateTree
Verifier.sol	User	verify	None

5.1. Registration Service

As explain in Section 4.5, before being able to operate a UAS, the operator is requested to register their device.

They interact with the Service Provider off-chain. Upon validation, the UAS and UAS operator serial numbers are added into one of the available merkle trees depending on their requests. To this end, the service provider calls the insertLeaf solidity function from the MerkleTree.sol smart contract, performs the off-chain computations and emits a NewLeaf event, sharing the name of the tree and the hash value of the new leaf with the blockchain network.

Figure 6 exemplifies how this inclusion in different Merkle trees is decided in the architecture.

To fully register a UAS, the provider must insert the UAS information into the corresponding Merkle Tree, and send back to the user a merkle tree proof of the inclusion in the requested tree. For instance, a drone having the attributes listed in Table 4 gets added in the respective Merkle trees: OpenMerkleTree, BVL0SMerkleTree, SpecialOpsMerkleTree; and receives back one ZKP for each tree insertion (Figure 6). In parallel, the service provider calls the addLeaf function of the

MerkleTree.sol smart contract. This call triggers the emission of an event embedding the hashed value of the corresponding leaf. By doing so, the UAS operator can verify that their device has indeed been approved. Afterwards, the service provider updates the tree roots stored in the MerkleTree.sol smart contract with the updateTree function.

By comparing their proofs of membership to the corresponding updated tree roots, the user can prove and verify their inclusion in the correct databases by just reading the blockchain.

Table 4. Simple example for a UAS Configuration.

Operation Mode	Flight Category	Flight Type
Specific	BVLOS	SpecialOps

5.2. Flight Authorisation Service

Now UAS and UAS operator have been registered, it is possible to issue a flight request and make it verified automatically on-chain.

The Flight Authorization service is ensured by the on-chain createFlightRequest function from the FlightAuth.sol smart contract as described in Algorithm 1. The contract articulates the decision-making process involved in approving flight requests based on a combination of conditions such as the validity of the request and the type of flight, supporting full automation of the Flight Authorization service. The use of blockchain introduces a level of integrity and non-repudiation to the operation. Furthermore, emitting events based on the outcome of these checks allows for transparency and traceability. This mechanism illustrates a seamless integration between off-chain data validation and on-chain logic execution.

Algorithm 1 Create Flight Request

```
1: function CREATEFLIGHTREQUEST(current_request, current_info)
2:   current_request.approved  $\leftarrow$  false
3:   if ISVALID(current_request) and current_request._flight_type == false then
4:     current_request.approved  $\leftarrow$  true
5:     _createdAt  $\leftarrow$  block.timestamp
6:     emit NewFlightRequest (check Table 3)
7:   else if ISVALID(current_request) == false and current_request._flight_type == false then
8:     _createdAt  $\leftarrow$  block.timestamp
9:     emit NewFlightRequest(check Table 3)
10:  else if ISVALID(current_request) and current_request._flight_type then
11:    current_request.approved  $\leftarrow$  true
12:    _createdAt  $\leftarrow$  block.timestamp
13:    emit NewPrivateFlightRequest(check Table 3)
14:  else  $\triangleright$  isValid(current_request) == false and current_request._flight_type
15:    _createdAt  $\leftarrow$  block.timestamp
16:    emit NewPrivateFlightRequest(check Table 3)
17:  end if
18: end function
```

Once the event is emitted, the information is accessible from the blockchain and can be read in the block explorer (Figure 7).



Figure 7. Event Information on Etherscan Block Explorer

5.3. Zero-Knowledge with Circom

As mentioned before, our implementation is using the Circom domain specific language to generate and manipulate ZKP.

In our case, the circuit, through some constraints, ensures that a provided Merkle proof is valid and issues a proof of knowledge. This proof is called a Proof of Membership. However, the circuit also checks that some properties about the leaves are correct, such as the correctness of the serial number. Then, our proof is not only a proof of membership, but a proof of verifiable computation.

Without verification of these properties, we could use accumulators [30], which are more efficient for the proof of membership.

To verify on-chain if such a proof is correct, we use Circom and SnarkJS to generate a smart contract that is able to check if a given proof is valid or not. We use this smart contract to prove a UAS or its operator is correctly included in the corresponding Merkle tree (Figure 6).

The Inclusion Proof circuit we implemented is specified as the Inclusion Proof Circuit in Appendix A. It specifies a SNARK circuit for verifying a Merkle tree inclusion proof, detailing the computation steps from leaf hashing through to the final assertion that the computed path's root matches the provided root. It employs a hash function for hashing leaf nodes and intermediate nodes, utilizing selectors to correctly position sibling nodes and path elements based on the path indices, thereby reconstructing the path to the root from the provided leaf and nonce. In the last line, the circuit main is defined for a Merkle tree with depth = 32.

The proof of membership is generated by using the Circom domain specific language upon a list of hash values provided by the user (namely, the path of hash values that need to be used to compute the root of the tree).

Considering that the proposed architecture uses a binary Merkle tree (i.e., a tree with two leaves per final branches), let us first remark that the number of drones that can be handled is 2^{depth} (i.e., the number of leaves at the end of the tree), with $depth \in \mathbb{N}^*$ the depth of the tree. Then, we denote the number of constraints of the chosen hash component of Circom n_H for two inputs a and b representing a total of 512 bits of data. We could have used SHA algorithms for the Hash function, but, they are not ZK-friendly as the number of constraints needed to compute a hash is very high ($n_H = 59281$).

We implemented two versions based on two hash functions: Poseidon [31] ($n_H = 243$) and Anemoi ($n_H = 105$) [32] (denoted HashFunction in Circom Pseudocode A). We will evoke the performance gains in the next section.

6. Performance Evaluation

We deployed our solution on the Ethereum *Sepolia* testnet. We interpreted the system as a black-box and therefore checked the consistency between inputs and expected outputs for a series of tests. Based on Figure 3, we defined 12 possible drone configurations and 24 scenarios for each.

The tested functions can be separated into two categories:

- **Merkle tree related functions** in particular inserting a leaf into a tree, which corresponds to the ZKP generation (blue rectangle in Figure 2);
- **Flight Request related functions**: Create a Flight Request, Verify Flight Request, which correspond to the on-chain ZKP verification (green rectangle in Figure 2).

6.1. Execution Time Analysis of the Off-Chain Proof Generation

The proof generation is done by using a circom circuit. As explained in the previous section, this circuit computes a set of hashes and checks if the computed root is equal to the given Merkle root. This execution time mainly depends on the complexity of the hash function. In our work, we compared 2 hash algorithms: Poseidon [31] and Anemoi [32].

Figure 8 shows the execution time to generate a proof of membership depending on the Merkle Tree depth and the hash algorithm used.

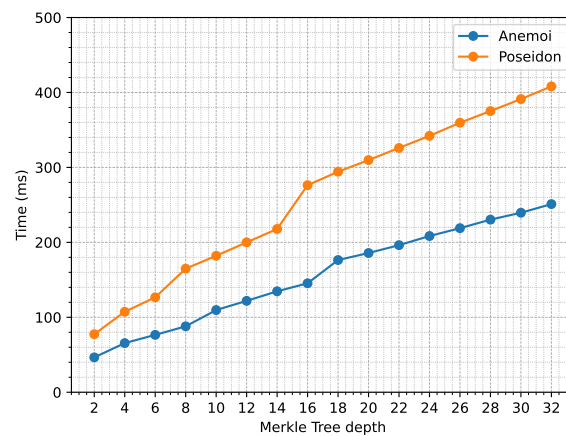


Figure 8. Proof Generation Time.

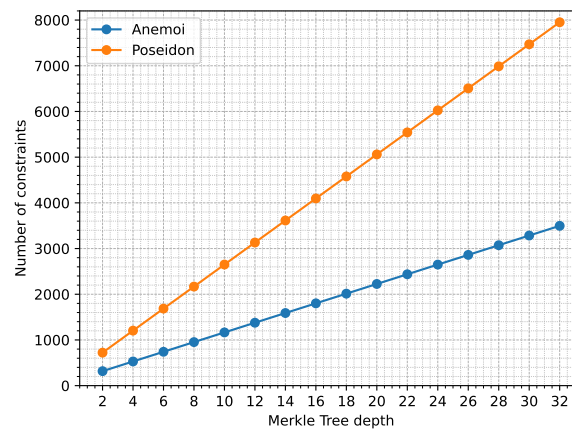


Figure 9. Number of constraints according to Merkle Tree depth and the hash function used (here, Poseidon and Anemoi).

Compared to Poseidon, using the Anemoi hash function can reduce by 40 % the time needed to generate a single proof.

If we take into consideration the drone configuration that requires the most amount of ZKPs (Figure 3), even with a Merkle tree of depth 32 (i.e., Merkle tree able to store the information of 4 billion UASs), we only need 250 ms per proof generation, which is minimal when considering the security benefits that it gives to the data.

Furthermore, this values have been measured by using SnarkJS [33], which use WebAssembly to perform computation. Other implementations like RapidSNARK [34], which is written in C++, could increase the performance.

6.2. Gas Cost Analysis for On-Chain Verification

In this section we will analyze the gas cost of smart contract calls and we analyse the two following situations:

- **Merkle tree related costs / Initialisation costs (Table 5)**
- **Flight Request costs (Table 6)**

To have a better understanding of the "real" price someone would have to pay while using this service, two columns were added taking in consideration the maximum and minimum historical gas prices in ETH for the year 2023/2024 (gas prices measured for the period of Feb 2023 to Feb 2024). According to *etherscan's* Ethereum Average Gas Price chart [35] the maximum and minimum values for the period mentioned above are:

- **Max:** 155.84 (Gwei) in May '23
- **Min:** 14.85 (Gwei) in Oct '23
- **Mean:** 33.7 (Gwei)

This gives us a range of prices for which the price of a Flight Request can vary, making it easier to understand the viability of the proposed system.

Table 5 specifies the gas cost of the operator's initial registration. In Alkadi and Shoufan [14], the gas price for the Register Drone function was measured to 114,320 gas. The closest scenario we implemented is configuration 1, and in our case, it costs 64,547 gas, which represents a 44% gain. We differentiate from them by adding extra layers of security by using zero-knowledge in configurations 2 and 3 (e.g., military use cases with private data). Indeed, unlike Alkadi and Shoufan [14], or even Khan et al. [12] the proposed architecture offers the possibility of registering drones with hierarchized access to the air space, while also being compliant with the U-Space legislation and compatible with special operations that require extra attention to privacy. This explains and justifies the cost overheads observed in Table 5 for configurations 2 and 3.

Table 5. Initialisation Cost Table.

Configuration	Gas Cost (gas)	Minimum Cost (ETH)	Mean Cost (ETH)	Maximum Cost (ETH)
1	64,547	0.0010	0.0021	0.0101
2	101,150	0.0015	0.0034	0.0158
3	160,984	0.0024	0.0054	0.0251
+ Operator	63,162	0.0009	0.0021	0.0098

The last line of the Table 5 specifies the costs for the register of the operator as well. This cost is taken separately because we assume the possibility of an operator being able to own more than one drone.

Table 6 refers to the costs of requesting a flight. In the work of Alkadi and Shoufan [14], the Request Mission Plan costs 219,648 gas for each request. In the proposed architecture, we present a minimum cost of 528,168 gas. This is justified due to a factor of scalability we provide, as we are able to handle up to 4 billions of UAS.

Table 6. Transaction Cost Table.

Configuration	Gas Cost (gas)	Minimum Cost (ETH)	Mean Cost (ETH)	Maximum Cost (ETH)
1	528,168	0.0078	0.0178	0.0823
2	909,851	0.0135	0.0307	0.1418
3	1,327,826	0.0197	0.0447	0.2069

Instead, existing literature chose to register drones inside an array stored in the smart contract. We argue that while testing with small arrays, the reviewed architectures pose no issue. However, when the system becomes bigger, the array grows and, each operation (e.g., searching, adding) in this array incurs a cost. As such, we believe that our system remains more scalable than existing works as we make use of ZK-SNARKs to reduce the on-chain computation.

Apart from the scalability issue, as implementations differ, we insist on the fact that the three propositions, [12,14] and ours, do not treat the information in the exact same way, differences that may impact the gas costs for calling smart contracts and further comparison. The full project is open source and available online [36].

7. Security Discussion

It remains to analyze the security of our blockchain-based ZKP-enhanced UTM system and provide an intuition why it guarantees both public auditability (Proposition 1) and privacy-preservation (Proposition 2).

Argument 1 (Public auditability). *The public auditability of the UTM services developed in this paper unfolds from to the use of an Ethereum-like blockchain. Indeed:*

- *The immutable ledger of the selected blockchain (i.e., Ethereum) grants data integrity (i.e., once data is recorded, it cannot be altered or deleted). This immutability ensures that historical records remain intact and unchanged, providing a reliable audit trail. In addition, by construction, blockchains are tamper-proof meaning that altering any data would require changing all subsequent blocks, which is computationally infeasible in a well-secured blockchain.*
- *The transparency ensures both open access and traceability. Public blockchains, like the one chosen here, allow anyone to view the transaction history and verify data without needing special permissions. Consequently, all entities in the UTM system can verify the correctness of the transactions and overall history. In addition, every transaction is time-stamped and linked, creating a comprehensive and transparent chain of events that can be traced back to the origin.*
- *The consensus mechanism provides decentralization and verification. The absence of a central authority reduces the risk of data manipulation and provides a more trustworthy and neutral verification process. In addition, the decentralized verification ensures that the recorded data are accurate and agreed upon by the majority.*
- *Finally, using smart contracts confers automated compliance.*

Thus, since all the proposed UTM services are backed by blockchain calls, the public auditability of UTM services is reduced to the public auditability of blockchains. \square

Argument 2 (Privacy-preservation of sensitive data). *The privacy-preservation of sensitive data unfolds from to both our use of the blockchain and the Groth16 ZK-SNARK protocol. Unlike previous propositions, we only store minimal amount of data on-chain. During registration, all personally identifying information (PII) like the operator's serial number or their identity are directly addressed off-chain (through a secured communication channel) to the competent authorities (i.e., the Service Providers). During flight authorization, we must distinguish two cases: 1) the Regular Flights; and 2) the SpecialOps flights.*

1. *For regular flights, the data shared on-chain consists in: the drone's Serial Number and a list of zero knowledge proofs. By definition of ZKP, the proofs do not leak any information except whether the drone is*

authorized to fly under a certain operation mode, flight category and flight type. The Serial Number acts as a pseudonym. If there is no other data leakage outside the UTM system, it is highly unlikely that, based on the aforementioned data, another user of the system is able to infer any personally identifying information about the drone's operator from on-chain data.

2. The treatment is slightly different for SpecialOps flights since the knowledge of the flight type is *per se* a sensitive information. Linking the drone's serial number to this knowledge may lead to further leak personally identifying information such as the country of origin, eventually the name of the company operating the drone if it is a commercial flight, ... For this case, we suggest the masking of the drone Serial Number SN (i.e., $Enc(pk_{SP}, SN)$ where $Enc(\cdot)$ is a secure asymmetric encryption scheme, and pk_{SN} the authentic public key of the Service Provider SP). As such, the Service Provider can still have a fair view of the airspace occupation and can audit users *a posteriori*. In addition, due to the inherent properties of the asymmetric encryption scheme (namely the confidentiality property), no malicious adversary can with a high probability access the underlying SN value without the knowledge of sk_{SP} the Service Provider's private key. The rest of the reasoning is similar to the Regular Flight case.

Consequently, considering our usage of the blockchain and the way we log data inside, the privacy-preservation of sensitive data inside our UTM system is ensured by the privacy-preservation property of the selected zero knowledge proofs. In case the serial numbers are crafted in such a way that it reveals personally identifying information, we suggest the use of asymmetric encryption to protect them. As such, in this case, the privacy-preservation property is reduced to the privacy-preservation property of the selected zero knowledge proofs and the confidentiality property of the asymmetric encryption scheme. \square

8. Future Work

Of course, this architecture is only the start that supports the development of lots of other services. As such, we suggest several ways to improve the current proposition.

Observation 1: Optimization of Merkle Tree Root Updates

One significant challenge observed is the high cost associated with updating Merkle tree roots within the smart contract, both in terms of gas fees and network traffic. To mitigate these expenses, future research could explore the application of cryptographic accumulators [37]. These structures offer a means to prove that a prior root and the corresponding user proof of membership remain valid without necessitating frequent updates to the tree, thus potentially reducing the operational costs and network load.

Observation 2: Enhanced Utilization of ZK-SNARKs

Currently, the use of ZK-SNARKs in our system is minimal. However, ZK-SNARKs present an opportunity to verify more intricate underlying data structures, such as the country of origin or specific serial number formats, without revealing sensitive information. Future work should focus on integrating ZK-SNARKs more extensively to enhance the privacy and security of the data verification processes. This integration could enable more complex and confidential proofs while maintaining the integrity and transparency of the system.

Observation 3: Reducing the Role of USSP

To minimize the trust assumptions within the system, it is imperative to further reduce the role of the USSP. This reduction could involve decentralizing functions currently handled by the USSP or distributing these tasks across multiple entities to diminish single points of failure and trust. A dedicated study on the modeling of USSP is recommended to identify which aspects can be further decentralized and how these changes can be effectively implemented while maintaining system integrity and performance.

Observation 4: Ensuring Transaction Fairness

Given that our system operates on the Ethereum blockchain, transaction fairness is inherently tied to Ethereum's consensus and operational protocols. Nonetheless, exploring mechanisms to enhance fairness within our specific use case remains crucial. Future research could investigate ways to mitigate the impact of Ethereum's transaction ordering and gas fee dynamics on our system's fairness. This might include off-chain solutions or layer 2 scaling techniques that could offer more predictable and equitable transaction processing.

By addressing these observations, we can significantly improve the efficiency, security, and trustworthiness of our system, paving the way for more robust and scalable blockchain-based UTM solutions.

9. Conclusions

With the growing trend of UAV applications and with its insertion in the smart city context, the need for efficient and secure air traffic management solutions becomes essential. For that to be feasible, enforcing rules and regulations is a task of the utmost importance. Given the amount of drones circulating in the world right now, their availability to the general public, and their great mobility, the decentralized solution suggests itself to be the most adequate. This article proposed a solution that deploys smart contracts on the *Ethereum* blockchain to manage aspects of the upcoming U-Space regulation in the EU, bringing to light the different measurements and metrics useful to understand the feasibility of web3 projects. Implementation details on the *Ethereum* blockchain and considerations on Merkle trees and Zero-Knowledge circuits were also presented.

Author Contributions: Conceptualisation, investigation, software, writing—original draft preparation, Frederico BAPTISTA.; software, methodology validation, supervision, writing—review and editing, Jonathan DETCHART; conceptualisation, methodology, validation, supervision, writing—review and editing, Marina DEHEZ-CLEMENTI. All authors have read and agreed to the published version of the manuscript.

Data Availability Statement: Our implementation is available at <https://github.com/FredericoBaptista/DFly/tree/main>.

Acknowledgments: We would like to thank Thomas LAVAUUR for his valuable remarks and insightful feedback on this work.

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

Appendix A. Circom Circuits

This appendix presents the Circom pseudocode for the Inclusion Proof circuit. `HashFunction` may refer either to Poseidon [31] or Anemoi [32].

Circum Pseudocode Inclusion Proof Circuit

```

1: template InclusionProof(levels)
2:   signal input leaf
3:   signal input nonce
4:   signal input root
5:   signal input pathElements[levels]
6:   signal input pathIndices[levels]
7:   signal output out
8:
9:   component selectors[levels]
10:  component hashers[levels]
11:  component hashleaf
12:
13:  signal computedPath[levels]
14:  signal hashedleaf
15:
16:  hashleaf = HashFunction(2)
17:  hashleaf.inputs[0] <== leaf           ▷ User inputs serial number
18:  hashleaf.inputs[1] <== nonce
19:  hashedleaf <== hashleaf.out
20:
21:  for  $i = 0$  to  $levels - 1$  do
22:    selectors[i] = PositionSwitcher()
23:    selectors[i].in[0] <==  $i == 0$  ? hashedleaf : computedPath[ $i - 1$ ]
24:    selectors[i].in[1] <== pathElements[i]
25:    selectors[i].s <== pathIndices[i]
26:
27:    hashers[i] = HashFunction(2)
28:    hashers[i].inputs[0] <== selectors[i].out[0]
29:    hashers[i].inputs[1] <== selectors[i].out[1]
30:    computedPath[i] <== hashers[i].out
31:  end for
32:
33:  out <== computedPath[levels - 1]
34:  ( $root - computedPath[levels - 1]$ ) == 0   ▷ Assert that the computed root equals the provided
    root
35:
36:  component main = InclusionProof(32)

```

References

1. Kuzmin, A.; Znak, E. Blockchain-base structures for a secure and operate network of semi-autonomous Unmanned Aerial Vehicles. 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), 2018, p. 32–37. doi:10.1109/SOLI.2018.8476785.
2. Alladi, T.; Chamola, V.; Sahu, N.; Guizani, M. Applications of blockchain in unmanned aerial vehicles: A review. *Vehicular Communications* **2020**, *23*, 100249. doi:10.1016/j.vehcom.2020.100249.
3. Abualsauod, E.H. A hybrid blockchain method in internet of things for privacy and security in unmanned aerial vehicles network. *Computers and Electrical Engineering* **2022**, *99*, 107847. doi:10.1016/j.compeleceng.2022.107847.
4. Allouch, A.; Cheikhrouhou, O.; Koubâa, A.; Toumi, K.; Khargui, M.; Nguyen Gia, T. UTM-Chain: Blockchain-Based Secure Unmanned Traffic Management for Internet of Drones. *Sensors* **2021**, *21*, 3049. doi:10.3390/s21093049.
5. FAA-NASA. Unmanned Aircraft System (UAS) Traffic Management (UTM).
6. 2021/664, C.I.R.E. On a regulatory framework for the U-space. *Official Journal of the European Union* **2021**.
7. Lappas, V.; Zoumponos, G.; Kostopoulos, V.; Lee, H.I.; Shin, H.S.; Tsourdos, A.; Tantardini, M.; Shomko, D.; Munoz, J.; Amoratis, E.; others. EuroDRONE, a European unmanned traffic management testbed for U-space. *Drones* **2022**, *6*, 53.
8. Nakamoto, S. Bitcoin: A Peer-to-Peer Electronic Cash System. 2008, 2008.
9. De Aguiar, E.J.; Faïçal, B.S.; Krishnamachari, B.; Ueyama, J. A survey of blockchain-based strategies for healthcare. *ACM Computing Surveys (CSUR)* **2020**, *53*, 1–27.
10. Lao, L.; Li, Z.; Hou, S.; Xiao, B.; Guo, S.; Yang, Y. A survey of IoT applications in blockchain systems: Architecture, consensus, and traffic modeling. *ACM Computing Surveys (CSUR)* **2020**, *53*, 1–32.
11. Belchior, R.; Vasconcelos, A.; Guerreiro, S.; Correia, M. A survey on blockchain interoperability: Past, present, and future trends. *ACM Computing Surveys (CSUR)* **2021**, *54*, 1–41.

12. Khan, A.A.; Shaikh, Z.A.; Laghari, A.A.; Bourouis, S.; Wagan, A.A.; Ali, G.A.A.A. Blockchain-Aware Distributed Dynamic Monitoring: A Smart Contract for Fog-Based Drone Management in Land Surface Changes. *Atmosphere* **2021**, *12*. doi:10.3390/atmos12111525.
13. Ralitera, T.; Gurcan, O. On Using Blockchains for Beyond Visual Line of Sight (BVLOS) Drones Operation: An Architectural Study. *System Engineering for Constrained Embedded Systems*; Association for Computing Machinery: New York, NY, USA, 2022; DroneSE and RAPIDO, p. 27–32. doi:10.1145/3522784.3522794.
14. Alkadi, R.; Shoufan, A. Unmanned Aerial Vehicles Traffic Management Solution Using Crowd-Sensing and Blockchain. *IEEE Trans. Netw. Serv. Manage.* **2023**, *20*, 14463–14479. doi:10.1109/TNSM.2022.3201817.
15. Kacem, T. UASChainSec: A Blockchain Based Framework for Secure 5G-Capable UAS Communication. 2023 10th International Conference on Recent Advances in Air and Space Technologies (RAST), 2023, pp. 1–6. doi:10.1109/RAST57548.2023.10197868.
16. Sharma, V.; You, I.; Kul, G. Socializing Drones for Inter-Service Operability in Ultra-Dense Wireless Networks using Blockchain. *International Workshop on Managing Insider Security Threats*, 2017. doi:10.1145/3139923.3139932.
17. Kapitonov, A.; Lonshakov, S.; Krupenkin, A.; Berman, I. Blockchain-based protocol of autonomous business activity for multi-agent systems consisting of UAVs. *Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS)*, 2018.
18. García-Magariño, I.; Lacuesta, R.; Rajarajan, M.; Lloret, J. Security in networks of unmanned aerial vehicles for surveillance with an agent-based approach inspired by the principles of blockchain. *Ad Hoc Networks* **2019**, *86*, 72–82. doi:https://doi.org/10.1016/j.adhoc.2018.11.010.
19. Lei, K.; Zhang, Q.; Lou, J.; Bai, B.; Xu, K. Securing ICN-Based UAV Ad Hoc Networks with Blockchain. *IEEE Communications Magazine* **2019**, *57*, 26–32. doi:10.1109/MCOM.2019.1800722.
20. European Union. Annex IV of Regulation (EU) 2021/664. <https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32021R0664&from=EN#d1e3576-61-1>, 2021. [Online; accessed 29-April-2024].
21. de la Transition écologique et de la Cohésion des territoires, M. AlphaTango. <https://www.ecologie.gouv.fr/alphatango>, 2022. [Online; accessed 30-April-2023].
22. European Commission. Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC (General Data Protection Regulation) (Text with EEA relevance), 2016.
23. Goldwasser, S.; Micali, S.; Rackoff, C. The knowledge complexity of interactive proof systems. *Society for Industrial and Applied Mathematics* **1985**.
24. Blum, M.; Feldamn, P.; Micali, S. Non-interactive zero-knowledge and its applications. *Association for Computing Machinery* **1988**.
25. Parno, B.; Gentry, C.; Howell, J.; Raykova, M. Pinocchio: Nearly Practical Verifiable Computation. *Cryptology ePrint Archive*, Paper 2013/279, 2013. <https://eprint.iacr.org/2013/279>.
26. Groth, J.; Kohlweiss, M.; Maller, M.; Meiklejohn, S.; Miers, I. Updatable and Universal Common Reference Strings with Applications to zk-SNARKs. *Cryptology ePrint Archive*, Paper 2018/280, 2018. <https://eprint.iacr.org/2018/280>.
27. Groth, J. On the Size of Pairing-based Non-interactive Arguments. *Cryptology ePrint Archive*, Paper 2016/260, 2016. <https://eprint.iacr.org/2016/260>.
28. iden3, G. circom: zkSnark circuit compiler. <https://github.com/iden3/circom>. (Accessed on 03/16/2024).
29. Muñoz-Tapia, J.L.; Belles, M.; Isabel, M.; Rubio, A.; Baylina, J. CIRCOM: A Robust and Scalable Language for Building Complex Zero-Knowledge Circuits. *Institute of Electrical and Electronics Engineers (IEEE)*, 2022. doi:10.36227/techrxiv.19374986.v1.
30. Benaloh, J.; de Mare, M. One-Way Accumulators: A Decentralized Alternative to Digital Signatures. *Advances in Cryptology — EUROCRYPT '93*; Helleseth, T., Ed.; Springer Berlin Heidelberg: Berlin, Heidelberg, 1994; pp. 274–285.
31. Grassi, L.; Khovratovich, D.; Rechberger, C.; Roy, A.; Schafneger, M. Poseidon: A new hash function for {Zero-Knowledge} proof systems. 30th USENIX Security Symposium (USENIX Security 21), 2021, pp. 519–535.
32. Bouvier, C.; Briaud, P.; Chaidos, P.; Perrin, L.; Salen, R.; Velichkov, V.; Willems, D. New Design Techniques for Efficient Arithmetization-Oriented Hash Functions: Anemoi Permutations and Jive Compression Mode. *Cryptology ePrint Archive*, Paper 2022/840, 2022. <https://eprint.iacr.org/2022/840>.

33. snarkjs contributors. snarkjs.
34. rapidsnark contributors. snarkjs.
35. Etherscan. Ethereum Average Gas Price Chart. <https://etherscan.io/chart/gasprice>, 2023. (Accessed on 16/11/2023).
36. Frederico Baptista, Marina Dehez-Clementi, J.D. DFLy. <https://github.com/FredericoBaptista/DFly/tree/main>, 2024.
37. Benaloh, J.; De Mare, M. One-way accumulators: A decentralized alternative to digital signatures. Workshop on the Theory and Application of Cryptographic Techniques. Springer, 1993, pp. 274–285.

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