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

Frederico Baptista, Marina Dehez-Clementi, Jonathan Detchart

drones

MDPI

Article

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

Frederico Baptista , Marina Dehez-Clementi * and Jonathan Detchart

ISAE-SUPAERO, Université de Toulouse, 31055 Toulouse, France;
frederico.paramos-merino-freire-baptista@student.isae-supaero.fr (F.B.); jonathan.detchart@isae-supaero.fr (J.D.)
* Correspondence: marina.dehez-clementi@isae-supaero.fr

Abstract: The integration of Unmanned Aircraft Systems (UASs) into the current airspace poses significant challenges in terms of safety, security, and operability. As an example, in 2019, the European Union defined 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, operability among UTM stakeholders, and legal compliance. As such, we propose DFly, a publicly auditable and privacy-preserving UAS traffic management system on Blockchain, with 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 enables 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

Citation: Baptista, F.; Dehez-Clementi, M.; Detchart, J. DFly: A Publicly Auditable and Privacy-Preserving UAS Traffic Management System on Blockchain. *Drones* **2024**, *8*, 410. https://doi.org/10.3390/drones8080410

Academic Editor: Carlos Tavares Calafate

Received: 15 July 2024
Revised: 7 August 2024
Accepted: 17 August 2024
Published: 21 August 2024

1. Introduction

Updated and efficient manufacturing processes along with the integration of cutting-edge technologies have spawned democratization of the use of Unmanned Aircraft Systems (UASs) and paved the way for civilian and commercial applications over the years [1–3]. Safety concerns have risen along with the increased number of UASs in the airspace [4]. The integration and monitoring of UAS systems pose significant challenges due to the anticipated high volume of drones in the airspace.

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 reinforces distrust against trusted third parties, like the service providers defined by the U-Space regulations, due to the lack of operational agility and transparency. As an example, currently, the time required to obtain a decision after submitting a UAS flight request to the French Civil Aviation Authority (DGAC) varies. For routine operations, the processing time typically ranges from 5 to 10 business days, whereas for complex operations, it can take several weeks. In addition, centralized UTM systems are not well-suited to the network topology and connectivity requirements of drones. In this paper, we explore the decentralized paradigm to contribute to the existing framework like the one proposed by Lappas et al. [7]. Indeed, the absence of a 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. As such, the blockchain technology offers a decentralized solution capable of adapting to a large number of drones. Indeed, *Ethereum* has a theoretical maximum throughput of 119 transactions per second (tps) which enables 4 billion potential drones to issue requests simultaneously experiencing a delay of treatment of approximately 9 hours. This would represent a huge gain in operability compared to existing processes.

Moreover, 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 a drone's 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 civilian) operations. As such, the system supports standard and special flights, meaning that both civilian and commercial UAS flights can be monitored within the same system, increasing 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' operation. Section 3 presents some background knowledge on 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. Finally, Section 9 concludes the paper.

2. Related Work and Beyond

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 to be 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, the 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 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.

Table 1 summarizes the related work and proposes a classification based on the blockchain type (BC Type), the blockchain solution (Solution), the way data are stored (Storage: on-chain/off-chain), the proposed services (Services), and the security properties they provided. This classification offers a clear overview of how our work compares to the closest literature.

The reviewed papers collectively highlight the potential of blockchains in enhancing the security, trust, and operational efficiency of drone management systems. Khan et al. [12] and Alkadi and Shoufan [14] emphasize the integration of blockchain with, respectively, fog computation and *Ethereum*-based decentralized solutions. Yet, both propositions face scalability and cost challenges on public blockchains. Both solutions proposed to use arrays as a means of storing the information on-chain. Indeed, arrays can be used to store data in

smart contracts, but their suitability for handling large amounts of data depends on several factors, including gas costs and performance considerations. For large datasets (as the ones expected for a UTM service), it is crucial to consider alternative approaches such as optimizing the type of data structures used to store information in order to decrease both memory occupation (and ultimately and more importantly) cost. Ralitera and Gurcan [13] introduce trust graphs for BVLOS operations, enhancing data tracking and registration. However, the lack of details regarding the implementation makes it difficult to evaluate whether authors were able to overcome the challenges related to, firstly, the implementation and maintenance of the trust graphs, and, secondly, their effective integration with blockchains. Finally, the UASChainSec [15] and UTM-Chain [4] frameworks focus on secure communication channels and traffic management using permissioned blockchains.

Despite these encouraging advancements, there are common issues attached to the aforementioned research works such as high transactional costs, limited scalability, and inadequate or simply no confidentiality guarantees for special flight operations. Our research aims to address these gaps by proposing a more scalable, cost-effective solution that also incorporates confidentiality for flight operations.

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 the 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 by Alkadi and Shoufan [14]. This analysis enables us to derive our main conclusions on the scalability and 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
Ours	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. We begin with an overview of the UTM (Unmanned Aircraft System Traffic Management) environment, focusing specifically on U-Space regulations and services, which are crucial for understanding the operational framework and regulatory requirements governing drone operations in European airspace. Next, we delve into the fundamentals of blockchain technology, exploring its principles, functionalities, and the inherent limitations that might affect its application to U-Space

services. Finally, we introduce the concept of zero-knowledge proofs, including their theoretical underpinnings and practical implications. We will explain the role of zero-knowledge proofs in enhancing privacy and security, and how these techniques can be integrated into blockchain systems to overcome their limitations.

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 analyzes the legality of a flight request. The regulation defines a workflow for this service that is detailed in Figure 1:

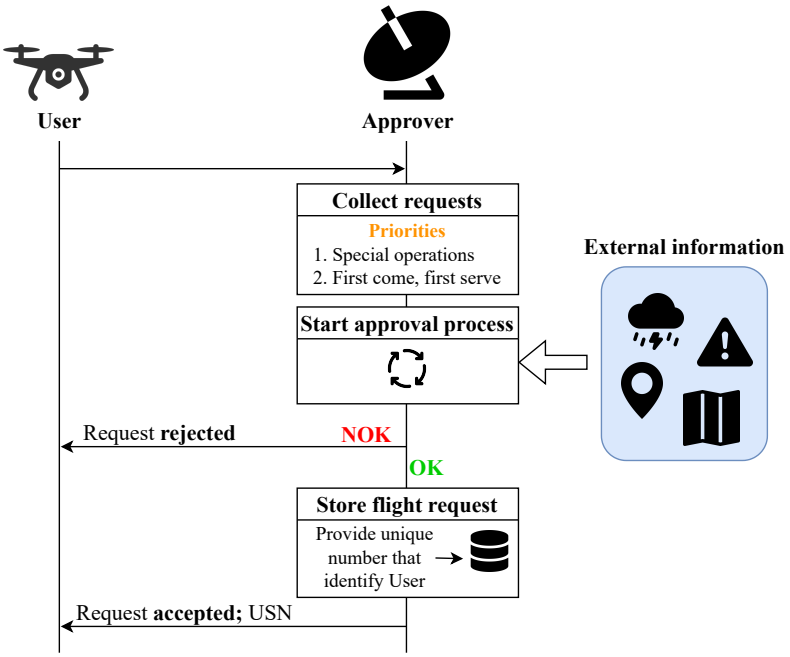

Figure 1. Workflow for flight authorization service.

For a flight authorization 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 Authorization 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, 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 their usage: every piece of 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 induces great transactional costs and reduces 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 Shoufan'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], a 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 are more complex to verify but are 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], which 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 (with respect to 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 detail 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 subsections, we define the entities and roles, adversarial model, and security guarantees, and present the sequence diagrams of both services.

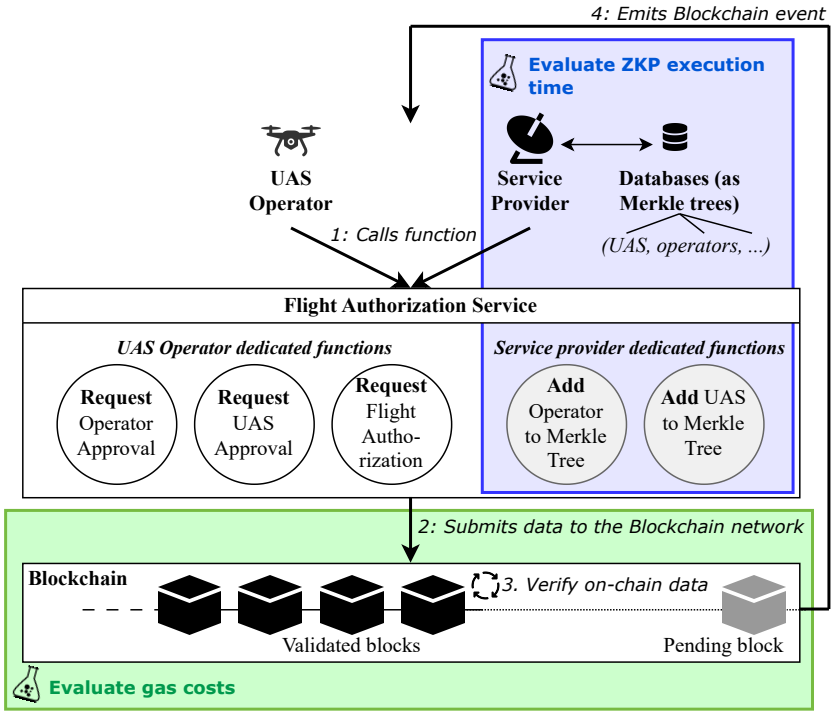

Figure 2. Use case diagram of the proposed U-Space-compliant Blockchain-based ZKP-enhanced flight authorization service and identification of the items relevant for performance evaluation.

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. A *UAS operator* is characterized by their operator number and the corresponding approval ZKP.
- **Service Provider:** is the owner of the service with respect to all legal effects, and 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 authorization 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 as we acknowledge the necessity of granting the access right control to the USSPs for legal stakes. The semi-distributed UAS management system involves several key entities: UAS operators, service providers, and block producers. The model makes specific assumptions about the trustworthiness and potential malicious behavior of these entities.

- **UAS operators**
  - *Behavior:* Considered malicious.
  - *Potential Actions:* They can submit fake *Registration* or *Flight Authorization* requests to the system.
  - *Rationale:* This assumption is based on the possibility that operators might try to deceive the system for unauthorized access or operations.
- **Service Providers:**
  - *Behavior:* Considered honest by default.
  - *Potential Deviation:* Although generally honest, the system includes mechanisms to handle scenarios where service providers might deviate from the protocol, such as:
    - Refusing to process valid registration requests.
    - Transmitting false information to users.
  - *Mitigation:* Minor functional additions to the current architecture are proposed to ensure robustness against such deviations, which will be detailed in Section 4.5.
- **Block producers:**
  - *Behavior:* Assumed to follow the same security model as the underlying *Ethereum* network.
  - *Security Assumption:*
    - For Proof-of-Work (PoW)-based *Ethereum*, it is assumed that at least 51% of the computational power is controlled by honest nodes [8].
    - Similarly, for Proof-of-Stake (PoS)-based *Ethereum*, it is assumed that the majority of the total amount of cryptocurrency staked in the network is held by honest validators who follow the protocol rules correctly.
  - *Rationale:* By relying on the established security model of *Ethereum*, the system inherits its security guarantees against attacks such as double-spending or majority attacks.

This adversarial model provides a clear framework for understanding the potential threats and the trust assumptions in the semi-distributed UAS management system.

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.

Claim 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.

Claim 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 as follows:

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 and are described in Section 6.

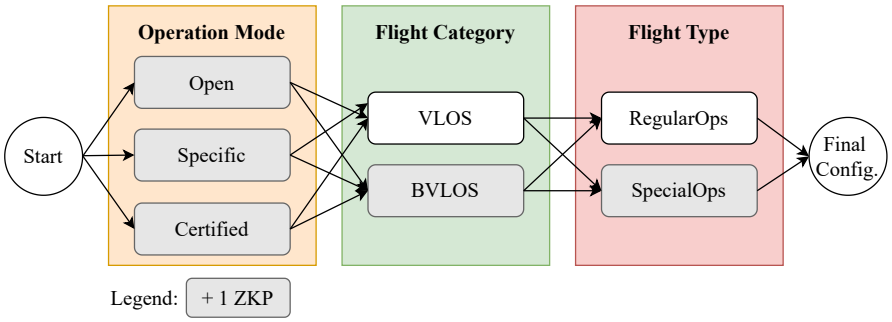

Figure 3. All possible configuration combinations of a flight request.

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.

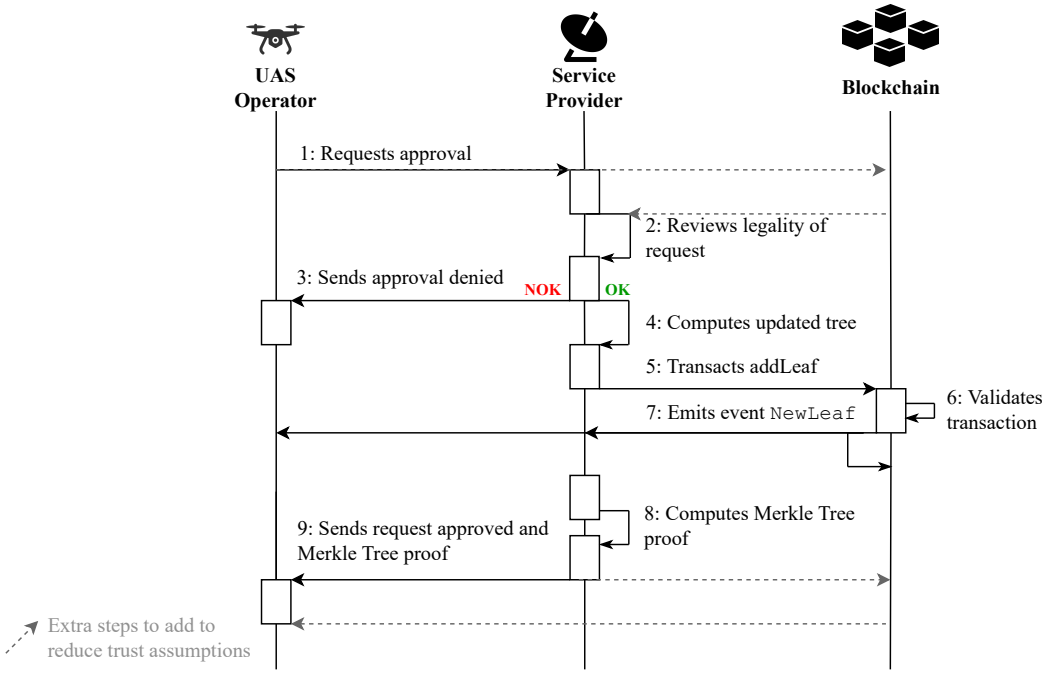

Figure 4. Sequence diagram of the request UAS/operator functionality.

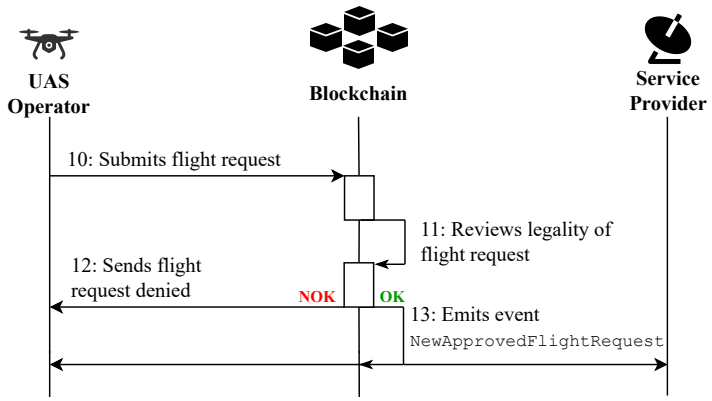

Figure 5. Sequence diagram of the flight request functionality.

4.5.1. Registration of UASs/UAS Operators

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

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 the case where 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.

4.5.2. 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.

4.5.3. 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 that the UAS belongs to the correct Merkle Trees, but in addition, due to the specific format of the proofs, that the UAS's 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.

4.6. A Word about Connectivity

Even though we consider that the networking and connectivity aspects of UASs are out of our scope, we want to highlight a few elements that support the claim that considering a blockchain-based UTM system for the future integration of drones is still realistic considering the current and future connectivity technologies.

The proposed architecture alleviates the role of the service provider by automating the *Registration* and *Flight Authorization* services. In addition to these two services (comprised in the U1 and U2 phases predicted by EUROCONTROL [30]), this architecture opens new opportunities when it comes to the future connectivity of the Internet. To further automatize the air traffic management system, it is necessary to implement tracking capabilities of the dynamic characteristics of the airspace (i.e., real UAS flights). For this to happen, two important technical challenges arise as suggested in [31]: *Flight Reporting* and *Data Transmission*. The *Flight Reporting* challenge is solved by the proposed system, which implements the blockchain as the single reliable source of truth in the UAS airspace. Once this single source of truth is defined, the role of the air traffic controller is to compare real flight data against the data saved on the blockchain, hence raising the second technical challenge: *Data Transmission*.

Communications within the scope of UAVs are realized differently than what is performed with conventional aircraft. These transmissions can include location, remaining flight time, distance and location to target, and many other parameters depending on the applied regulation, effectively making the data content more variable and larger when compared to traditional aircraft data packages. New technologies like 5G technology are set to serve these high load demands perfectly. Specifically in the case of our system, having the blockchain as the single source of truth means that drones connected to a 5G network can communicate with the service provider in am Ultra-Reliable Low-Latency Communication (URLLC) manner, with latencies as low as 1 millisecond and robust reliability mechanisms. Therefore, 5G can facilitate real-time data exchange and immediately identify drones violating the approved flight request stored in the blockchain.

Although 5G technology brings good prospects for the proposed system, and despite being in rapid growth, with 5G mobile subscriptions exceeding 5.3 billion in 2029 [32], universal adoption of these networks will not occur tomorrow. When it comes to 3G and 4G connectivity, the major downside that comes with these already established connectivity

technologies with respect to 5G is the increased delay in communications with the service provider despite still being possible to communicate and access blockchain information, i.e., to verify correct adherence to the flight plan.

In more extreme cases where no internet connection is possible, it was demonstrated in 2014 by Kryptoradio (https://kryptoradio.koodilehto.fi/ accessed on 19 august 2024) that Bitcoin transaction data can be encoded into radio signals using one-way digital broadcast networks like Digital Video Broadcasting – Terrestrial (DVB-T). This approach could, in theory, be adapted for *Ethereum* transactions, massively increasing the delay of the communications between drone and service provider but possibly making the proposed system more resilient to different connectivity scenarios.

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 conducted 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 improve efficiency while asserting the conformity of UAS attributes to U-Space rules.

Table 3 specifies the different functions, smart contracts, and events. In addition, the complete implementation is available at https://github.com/FredericoBaptista/DFly/ (accessed on 19 august 2024).

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 explained 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`, `BVLOSMerkleTree`, `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

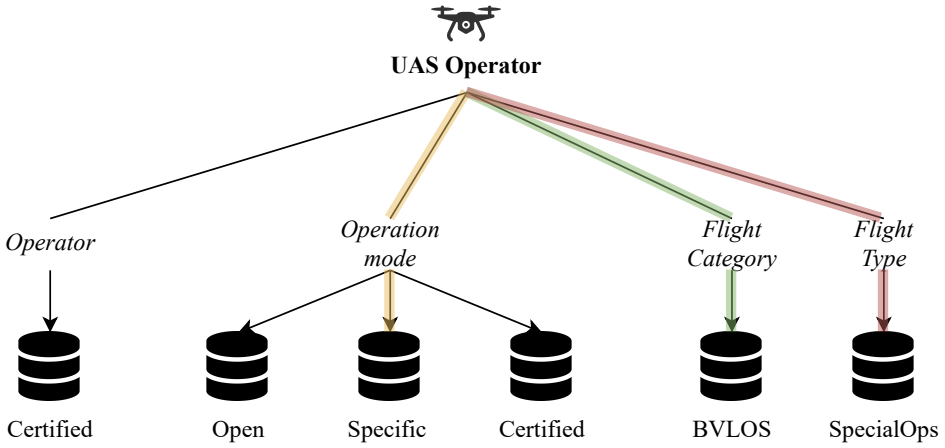

Figure 6. Management of different Merkle Trees in the architecture.

5.2. Flight Authorization Service

Now the UAS and UAS operator have been registered, it is possible to issue a flight request and have 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.

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

Algorithm 1 Create Flight Request

```
function CREATEFLIGHTREQUEST(current_request, current_info)
    current_request.approved ← false
    if ISVALID(current_request) and current_request._flight_type == false then
        current_request.approved ← true
        _createdAt ← block.timestamp
        emit NewFlightRequest (check Table 3)
    else if ISVALID(current_request) == false and current_request._flight_type ==
false then
        _createdAt ← block.timestamp
        emit NewFlightRequest(check Table 3)
    else if ISVALID(current_request) and current_request._flight_type then
        current_request.approved ← true
        _createdAt ← block.timestamp
        emit NewPrivateFlightRequest(check Table 3)
    else            ▷ isValid(current_request) == false and current_request._flight_type
        _createdAt ← block.timestamp
        emit NewPrivateFlightRequest(check Table 3)
    end if
end function
```

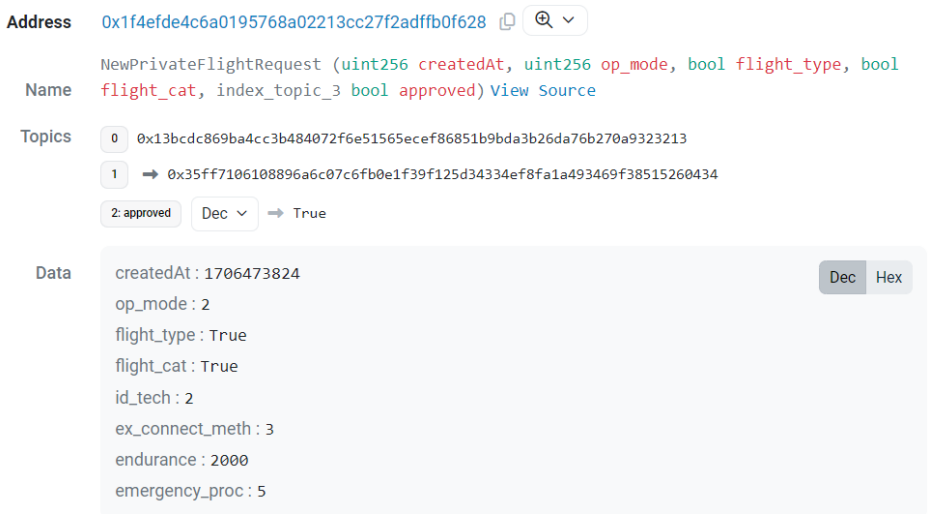

Figure 7. Event Information on Etherscan Block Explorer.

5.3. Zero Knowledge with Circom

As mentioned before, our implementation uses 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 [33], 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 = 59,281).

We implemented two versions based on two hash functions: Poseidon [34] ($n_H = 243$) and Anemoi ($n_H = 105$) [35] (denoted `HashFunction` in Appendix 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 and verify the flight request, which corresponds 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 performed 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. For this first series of tests, the goal is to assess the scalability of the proposed system. The idea is to illustrate how the execution time and the number of constraints in our circuit are influenced by the depth of the tree.

6.1.1. Methodology

For these tests, we compute the required number of hashes for each Merkle Tree depth. Therefore, for a Merkle Tree depth of 2, we compute 2 hashes; if depth is equal to 3, 3 hashes were computed. In our work, we compared 2 hash algorithms: Poseidon [34] and Anemoi [35].

6.1.2. Results

Figure 8 shows the execution time to generate a proof of membership depending on the Merkle Tree depth and the hash algorithm used. Alongside is Figure 9, which illustrates the number of constraints in the circuit according to the Merkle Tree depth and the hash function used. It leads to three remarks: (1) the number of constraints depends on both the hash function used and the Merkle Tree depth; (2) the growth is linear in the tree depth but remains inferior to 8000 for 2^{32} drones (4,000,000); and (3) if we change the hash function from Poseidon to Anemoi, the slope decreases by half. Thus, optimizing the number of constraints per hash function optimizes our solution.

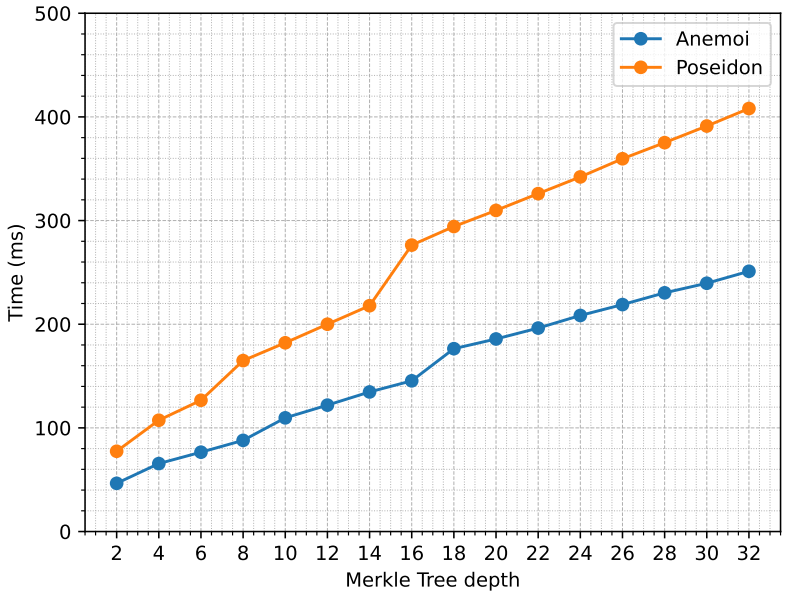

Figure 8. Proof generation time.

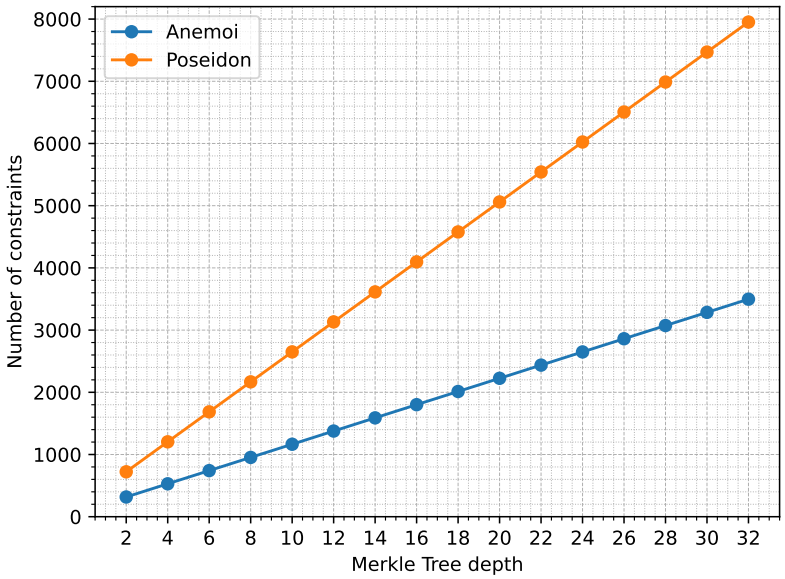

Figure 9. Number of constraints according to Merkle Tree depth and the hash function used (here, Poseidon and Anemoi).

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 largest 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, these values have been measured by using SnarkJS [36], which uses WebAssembly to perform computation. Other implementations like RapidSNARK [37], 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 analyze the two following situations:

- **Merkle Tree-related costs/initialization costs (Table 5)**

- **Flight request costs (Table 6)**

6.2.1. Methodology

We interpreted the system as a black-box and, therefore, checked the consistency between inputs and expected outputs for some tests. A total of 578 tests were performed, resorting to 12 test subjects that covered all the configuration possibilities as seen in Figure 3. These test subjects were added to the respective Merkle Trees while automatically checking if the event information corresponded to what was expected. Then, a list of all the 289 possible combinations of flight requests and test subjects was performed twice while verifying the correctness of the event information emitted. In total, 100% of the tests were successful and the system behaved as expected, emitting the correct event information and also approving the correct flights.

The performance analysis was conducted while checking for validation of the results of the system. While performing the 578 tests at different days and times of day, the gas cost (in units of gas) of each flight request was measured according to the number of ZKPs needed in each of them. The same method was applied to the cost of inserting a new leaf, measuring twice (although the gas units required do not change for the same transaction) the gas cost needed to insert the 12 test subjects (drones) with the different configurations in the Merkle Trees.

Thus, 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 [38], the maximum and minimum values for the period mentioned above are:

- **Max:** 155.84 (Gwei) in May 2023
- **Min:** 14.85 (Gwei) in Oct 2023
- **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.

6.2.2. Results

Table 5 specifies the gas cost of the operator's initial registration.

Table 5. Initialization cost.

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

'+ Operator' refers to the operation of registering a new operator.

In Alkadi and Shoufan [14], the gas price for the Register Drone function was measured as 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.

The last line of 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. Transaction cost.

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

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 billion UASs.

Instead, the 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 the 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 [39].

7. Security Discussion

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

Argument 1 (Public auditability)**.** *The public auditability of the UTM services developed in this paper unfolds from 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 are recorded, they 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.

Argument 2 (Privacy preservation of sensitive data)**.** *The privacy preservation of sensitive data unfolds from 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 consist of the drone's Serial Number and a list of zero-knowledge proofs. By the definition of a 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, sensitive information. Linking the drone's Serial Number to this knowledge may lead to further leaks of personally identifying information such as the country of origin, eventually the name of the company operating the drone if it is a commercial flight, etc. For this case, we suggest masking of the drone's Serial Number SN (i.e., $Enc(pk_{SP}, SN)$ where $Enc(\cdot)$ is a secure asymmetric encryption scheme, and pk_{SN} is 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 the 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 they reveal 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.

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.

8.1. 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 [33]. 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.

8.2. 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.

8.3. 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 the 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.

8.4. 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 their integration into the smart city context, the need for efficient and secure air traffic management solutions becomes essential. Enforcing rules and regulations is paramount to ensuring safety, security, and operability in increasingly crowded airspaces. Given the vast number of drones currently in operation, their accessibility to the general public, and their inherent mobility, a decentralized solution emerges as the most suitable approach.

In this paper, we proposed DFly, a publicly auditable and privacy-preserving UAS traffic management system on Blockchain, specifically designed to address the upcoming U-Space regulations in the EU. Our solution leverages smart contracts on the Ethereum blockchain to manage two critical services: *Registration* and *Flight Authorization*. By utilizing blockchain technology, we ensure the public auditability of these services and the actions of corresponding service providers, fostering greater transparency and trust among UTM stakeholders.

Furthermore, DFly facilitates comprehensive and distributed monitoring of airspace occupation and is designed to support the integration of additional functionalities, such as a live UAS tracker. The combination of blockchain with zero-knowledge proofs allows for the deployment of an automated, distributed, transparent, and privacy-preserving flight authorization service, executed on-chain.

Our implementation deployed on the Ethereum Sepolia testnet, along with the use of the Groth16 ZK-SNARK protocol, demonstrates the practical viability of our approach. Performance analyses, both on-chain (gas cost) and off-chain (execution time), confirm that DFly is an efficient alternative in the spirit of digitalization, providing enhanced security guarantees compared to current centralized models.

Author Contributions: Conceptualisation, investigation, software, writing—original draft preparation, F.B.; software, methodology validation, supervision, writing—review and editing, J.D.; conceptualisation, methodology, validation, supervision, writing—review and editing, M.D.-C. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by the Defense Innovation Agency (AID) of the French Ministry of Defense through the Research Project DRAGOON: Dependable distRibuted storAGe fOr mObile Nodes (2022 65 0082).

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

Acknowledgments: We would like to thank Thomas Lavaur 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 [34] or Anemoi [35].

Circom Pseudocode Inclusion Proof Circuit

```
template InclusionProof(levels)
    signal input leaf
    signal input nonce
    signal input root
    signal input pathElements[levels]
    signal input pathIndices[levels]
    signal output out

    component selectors[levels]
    component hashers[levels]
    component hashleaf

    signal computedPath[levels]
    signal hashedleaf

    hashleaf = HashFunction(2)
    hashleaf.inputs[0] <== leaf                    ▷ User inputs Serial Number
    hashleaf.inputs[1] <== nonce
    hashedleaf <== hashleaf.out

for i = 0 to levels − 1 do
        selectors[i] = PositionSwitcher()
        selectors[i].in[0] <== i == 0 ? hashedleaf : computedPath[i − 1]
        selectors[i].in[1] <== pathElements[i]
        selectors[i].s <== pathIndices[i]

        hashers[i] = HashFunction(2)
        hashers[i].inputs[0] <== selectors[i].out[0]
        hashers[i].inputs[1] <== selectors[i].out[1]
        computedPath[i] <== hashers[i].out
end for

out <== computedPath[levels - 1]
(root − computedPath[levels − 1]) === 0  ▷ Assert that the computed root equals the provided root

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. In Proceedings of the 2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI), Singapore, 31 July–2 August 2018; pp. 32–37. https://doi.org/10.1109/SOLI.2018.8476785.
2. Alladi, T.; Chamola, V.; Sahu, N.; Guizani, M. Applications of blockchain in unmanned aerial vehicles: A review. *Veh. Commun.* **2020**, *23*, 100249. https://doi.org/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. *Comput. Electr. Eng.* **2022**, *99*, 107847. https://doi.org/10.1016/j.compeleceng.2022.107847.
4. Allouch, A.; Cheikhrouhou, O.; Koubâa, A.; Toumi, K.; Khalgui, M.; Nguyen Gia, T. UTM-Chain: Blockchain-Based Secure Unmanned Traffic Management for Internet of Drones. *Sensors* **2021**, *21*, 3049. https://doi.org/10.3390/s21093049.
5. FAA-NASA. Version 2.0 of the Unmanned Aircraft Systems (UAS) Traffic Management (UTM) Concept of Operations. Available online: https://www.faa.gov/sites/faa.gov/files/2022-08/UTM_ConOps_v2.pdf (accessed on 21 august 2024).
6. 2021/664, C.I.R.E. On a regulatory framework for the U-space. *Off. J. Eur. Union* **2021**, *64*, 190.
7. Lappas, V.; Zoumponos, G.; Kostopoulos, V.; Lee, H.I.; Shin, H.S.; Tsourdos, A.; Tantardini, M.; Shomko, D.; Munoz, J.; Amoratis, E.; et al. 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. Available online: http://dx.doi.org/10.2139/ssrn.3440802 (accessed on 21 august 2024).

9. De Aguiar, E.J.; Faiçal, B.S.; Krishnamachari, B.; Ueyama, J. A survey of blockchain-based strategies for healthcare. *ACM Comput. Surv. (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 Comput. Surv. (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 Comput. Surv. (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*. https://doi.org/10.3390/atmos12111525.
13. Ralitera, T.; Gurcan, O. On Using Blockchains for Beyond Visual Line of Sight (BVLOS) Drones Operation: An Architectural Study. In Proceedings of the System Engineering for Constrained Embedded Systems, New York, NY, USA, 23 June 2022; pp. 27–32. https://doi.org/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. https://doi.org/10.1109/TNSM.2022.3201817.
15. Kacem, T. UASChainSec: A Blockchain Based Framework for Secure 5G-Capable UAS Communication. In Proceedings of the 2023 10th International Conference on Recent Advances in Air and Space Technologies (RAST), Istanbul, Turkiye, 7–9 June 2023; pp. 1–6. https://doi.org/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. In Proceedings of the International Workshop on Managing Insider Security Threats, Dallas, Texas, USA, 30 October 2017. https://doi.org/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. In Proceedings of the Workshop on Research, Education and Development of Unmanned Aerial Systems (RED-UAS), Linköping, Sweden, 3–5 October 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 Netw.* **2019**, *86*, 72–82. 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 Commun. Mag.* **2019**, *57*, 26–32. https://doi.org/10.1109/MCOM.2019.1800722.
20. European Union. Annex IV of Regulation (EU) 2021/664. 2021. Available online: https://eur-lex.europa.eu/legal-content/EN/TXT/HTML/?uri=CELEX:32021R0664&from=EN#d1e3576-61-1 (accessed on 29 April 2024).
21. de la Transition écologique et de la Cohésion des territoires, M. AlphaTango. 2022. Available online: https://www.ecologie.gouv.fr/alphatango (accessed on 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. Available online: https://eur-lex.europa.eu/eli/reg/2016/679/oj (accessed on 21 august 2024).
23. Goldwasser, S.; Micali, S.; Rackoff, C. *The Knowledge Complexity of Interactive Proof Systems*; Society for Industrial and Applied Mathematics: Philadelphia, PA, USA, 1985.
24. Blum, M.; Feldamn, P.; Micali, S. *Non-Interactive Zero-Knowledge and Its Applications*; Association for Computing Machinery: New York, NY, USA, 1988.
25. Parno, B.; Gentry, C.; Howell, J.; Raykova, M. Pinocchio: Nearly Practical Verifiable Computation. Cryptology ePrint Archive, Paper 2013/279, 2013. Available online: https://eprint.iacr.org/2013/279 (accessed on 19 august 2024).
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. Available online: https://eprint.iacr.org/2018/280 (accessed on 19 august 2024).
27. Groth, J. On the Size of Pairing-based Non-interactive Arguments. Cryptology ePrint Archive, Paper 2016/260, 2016. Available online: https://eprint.iacr.org/2016/260 (accessed on 19 august 2024).
28. iden3, G. Circom: ZkSnark Circuit Compiler. Available online: https://github.com/iden3/circom (accessed on 16 March 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): Piscataway, NJ, USA, 2022. https://doi.org/10.36227/techrxiv.19374986.v1.
30. and Single European Sky ATM Research 3 Joint Undertaking. *U-Space—Blueprint*; Publications Office: 2017. Brussels, Belgium Available online: https://european-union.europa.eu/institutions-law-budget/institutions-and-bodies/search-all-eu-institutions-and-bodies/sesar-3-joint-undertaking_en (accessed on 21 august 2024).
31. Sándor, Z. Challenges caused by the unmanned aerial vehicle in the air traffic management. *Period. Polytech. Transp. Eng.* **2019**, *47*, 96–105.
32. Ericsson Mobility Report. 2020. Available online: https://www.ericsson.com/en/reports-and-papers/mobility-report/reports (accessed on 7 august 2024).

33. Benaloh, J.; de Mare, M. One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In *Workshop on the theory and application of cryptographic techniques on Advances in cryptology*; Helleseth, T., Ed.; Springer: Berlin/Heidelberg, Germany, 1994; pp. 274–285.
34. Grassi, L.; Khovratovich, D.; Rechberger, C.; Roy, A.; Schofnegger, M. Poseidon: A new hash function for {Zero-Knowledge} proof systems. In Proceedings of the 30th USENIX Security Symposium (USENIX Security 21), 2021; August 11-13, virtual pp. 519–535.
35. 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. Available online: https://eprint.iacr.org/2022/840 (accessed on 21 august 2024).
36. Snarkjs Contributors. Snarkjs. Available online= https://docs.iden3.io/circom-snarkjs/ (accessed on 21 august 2024).
37. Rapidsnark Contributors. Rapidsnark. Available online: https://github.com/iden3/rapidsnark (accessed on 21 august 2024).
38. Etherscan. Ethereum Average Gas Price Chart. 2023. Available online: https://etherscan.io/chart/gasprice (accessed on 16 November 2023).
39. Frederico Baptista, Marina Dehez-Clementi, J.D. DFly. 2024. Available online: https://github.com/FredericoBaptista/DFly/tree/main (accessed on 19 august 2024).