electronics

MDPI

Article

Design of Secure Communication Networks for UAV Platform Empowered by Lightweight Authentication Protocols

Muhammet A. Sen *, Saba Al-Rubaye and Antonios Tsourdos

School of Aerospace, Transportation and Manufacturing, Cranfield University, Bedford MK43 0AL, UK
* Correspondence: muhammet.sen@cranfield.ac.uk

Abstract

Flying Ad Hoc Networks (FANETs) formed by cooperative Unmanned Aerial Vehicles (UAVs) require formally proven secure and resource-efficient authentication because open wireless channels allow active adversaries to inject commands, replay traffic, and impersonate nodes. Conventional certificate-based mechanisms impose key management overhead and remain vulnerable under device capture, while existing lightweight and Physical Unclonable Function (PUF)-assisted proposals commonly assume stable connectivity, lack formal adversarial verification, or are evaluated only through simulation. This paper presents a lightweight PUF-assisted authentication protocol designed for dynamic multi-hop FANET operation. The scheme provides mutual UAV–Ground Station (GS) authentication and session key establishment and further enables secure UAV–UAV communication using an off-path ticket mechanism that eliminates continuous infrastructure dependence. The protocol is constructed through verification-driven refinement and formally analysed under the Dolev–Yao model, establishing authentication and session key secrecy and resistance to replay and impersonation attacks. Implementation-oriented latency measurements on Raspberry-Pi-class embedded platforms demonstrate that cryptographic processing time can be further reduced with hardware improvements, while the overall end-to-end delay is still largely determined by channel conditions and connection behaviour. Comparative evaluation shows reduced communication cost and broader security coverage relative to existing UAV authentication schemes, indicating practical deployability in large-scale FANET environments.

Keywords: flying Ad Hoc Networks; FANET; UAV authentication; Physically Unclonable Functions; PUF; lightweight security protocol; Internet of Drones; mutual authentication; session key establishment; formal verification; Scyther

Check for updates

Academic Editor: Alexandra Bousia

Received: 23 December 2025
Revised: 8 February 2026
Accepted: 9 February 2026
Published: 12 February 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Unmanned Aerial Vehicles (UAVs) are increasingly deployed as cooperative autonomous systems in surveillance, monitoring, delivery, and disaster response operations [1]. Rather than operating in isolation, modern missions rely on Flying Ad Hoc Networks (FANETs) in which UAVs dynamically exchange sensing data and control commands with ground stations and neighbouring drones [2–4]. Because communication occurs over an open wireless medium, an adversary can attempt eavesdropping, command injection, replay, or impersonation, any of which may directly affect flight safety and mission integrity [5]. Authentication therefore becomes a primary safety mechanism rather than a supporting security feature. Conventional certificate-based cryptographic approaches provide strong theoretical protection but impose storage, key management, and computational overhead

unsuitable for small aerial platforms with limited battery and processing resources. Moreover, physical capture of a UAV exposes stored credentials, creating a persistent vulnerability that software-only key protection cannot fully mitigate [6–8].

The recent literature has explored lightweight authentication designs and hardware-assisted techniques, particularly Physical Unclonable Functions (PUFs), which derive secrets from intrinsic manufacturing variations in electronic components [9–12]. PUFs offer tamper-resistant identity binding and remove long-term key storage, making them attractive for resource-constrained aerial devices. Nevertheless, existing UAV authentication research exhibits three persistent shortcomings. First, many schemes focus on single-hop UAV–ground station interaction and implicitly assume stable connectivity, whereas FANET topology changes frequently and peer communication often occurs without continuous infrastructure support [13–15]. Second, a considerable portion of proposals rely on informal threat arguments rather than systematic symbolic verification under a standard adversarial model, leaving their security guarantees uncertain against active attackers [13,15–17]. Third, feasibility is typically demonstrated only through simulation or analytical cost estimation, while implementation-level performance on representative embedded platforms is rarely examined, making practical deployability unclear.

To address these limitations, this work proposes a lightweight PUF-assisted authentication protocol tailored for dynamic FANET environments. The protocol establishes mutual authentication and session key agreement between UAVs and the ground station, and further enables secure UAV–UAV communication through an on-demand ticket mechanism that removes the need for continuous infrastructure involvement during peer interaction. The design follows a verification-driven refinement approach in which weaknesses identified during formal modelling guide incremental protocol improvement. As a result, the final construction binds authentication to device hardware while remaining suitable for highly mobile multi-hop aerial networks. The key contributions of the work are stated below:

1. We redesign and extend the baseline PUF-based UAV authentication scheme [18] through verification-guided refinement, where Scyther-identified vulnerabilities are eliminated by nonce masking, explicit key confirmation, and context binding, and further introduce an off-path ticket-based UAV–UAV authenticated key establishment protocol enabling secure multi-hop peer communication without continuous GS participation or long-term pairwise secret exposure.
2. We formally model the UAV–GS sub-protocol in the Scyther symbolic framework under the Dolev–Yao adversary, proving injective agreement, mutual entity authentication, and session key secrecy, and demonstrating resistance against replay, impersonation, man-in-the-middle, and key-compromise impersonation attacks through automated trace analysis.
3. We provide both analytical and implementation-level evaluation by benchmarking the protocol against state-of-the-art PUF-based UAV authentication schemes and deploying it on the Raspberry Pi 4B and 5 platforms, where latency decomposition and distribution statistics confirm sub-dominant cryptographic cost and show end-to-end delay is governed primarily by link establishment and channel dynamics, validating real-world employability in dynamic FANETs.

The remainder of this paper is organised as follows: Section 2 reviews related work on UAV security and PUF-based authentication. Section 3 describes the threat model and security goals. Section 4 details the proposed protocol, including the initial registration phase and the authentication phase. Section 5 presents the formal security analysis using the Scyther tool and discusses the identified vulnerability and its resolution. Section 6 provides findings and a comparative performance evaluation. Section 7 discusses practical implications and Section 8 concludes the paper and suggests future research directions.

2. Background and Related Work

The development of secure and efficient authentication protocols for UAVs is a rapidly evolving field, driven by the increasing deployment of UAVs in critical applications and the unique challenges posed by Flying Ad Hoc Networks. This section reviews the existing literature relevant to our work, focusing on three key areas: the inherent security challenges of FANET environments, the application of PUFs in lightweight security, and the role of formal verification in protocol design.

2.1. FANET Characteristics and Security Challenges

FANETs represent a specialised class of Mobile Ad Hoc Networks (MANETs) distinguished by the unique operational environment and characteristics of Unmanned Aerial Vehicles [19]. These characteristics fundamentally influence the security landscape and render many conventional security protocols inadequate. The primary properties [20] of FANETs—such as node mobility, topology change rate, node density, energy constraints, use of position information, and routing requirements—are illustrated in Figure 1. These features differentiate FANETs from terrestrial MANETs and directly impact network stability, scalability, and communication performance.

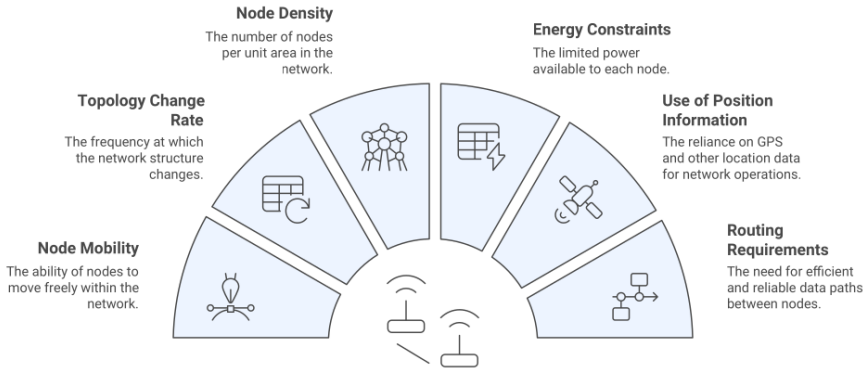


Figure 1. Key FANET characteristics including node mobility, topology dynamics, node density, and resource constraints.

The primary distinguishing features of FANETs include the following:

- High Mobility and Dynamic Topology: UAVs operate at high speeds and in three-dimensional space, leading to frequent and unpredictable changes in network topology. This high dynamism results in rapid link breakage and re-establishment, making routing and key management significantly more complex than in terrestrial ad hoc networks [21].
- Resource Constraints: Small-scale UAVs are constrained by limited battery life, processing power, and memory. Security solutions must therefore be ultra-lightweight to avoid excessive computational and energy overhead that would compromise mission endurance.
- Open and Hostile Environment: UAVs operate in open airspace, making them highly susceptible to physical capture, jamming, eavesdropping, and spoofing attacks from any direction. The lack of a fixed infrastructure or centralised trusted authority exacerbates these vulnerabilities.

The security requirements for FANETs are framed by the four pillars of information security: Confidentiality, Integrity, Availability (CIA), and Non-Repudiation. Data-related threats are analysed across communication scenarios such as Drone-to-Drone (D2D) and Drone-to-Ground Control Station (D2GCS). The lightweight nature of FANET protocols often leads to inter-layer vulnerabilities where an attack targeting a lower layer (e.g., physi-

cal or MAC) can compromise a higher-layer security mechanism (e.g., authentication at the network layer) [22]. This complex threat model necessitates the development of protocols that are not only computationally lightweight but also robust against a sophisticated, multi-faceted adversary.

Beyond their inherent characteristics, FANETs face a diverse set of security threats distributed across multiple layers of the system architecture. Due to their deployment in open and often adversarial environments, UAVs are vulnerable to attacks targeting hardware components, onboard firmware, communication links, data handling processes, command-and-control (C2) channels, and operational procedures. These vulnerabilities extend beyond isolated layers; an attack on a lower layer—such as hardware capture or firmware modification—can cascade upward and compromise higher-level security functions such as routing, authentication, and mission control. A structured overview of the main FANET vulnerabilities in six key layers is presented in Figure 2. In addition to the layered vulnerability landscape shown in Figure 2, FANET security is also strongly shaped by the underlying network architecture; accordingly, the main FANET architectural classifications are summarised in Figure 3.

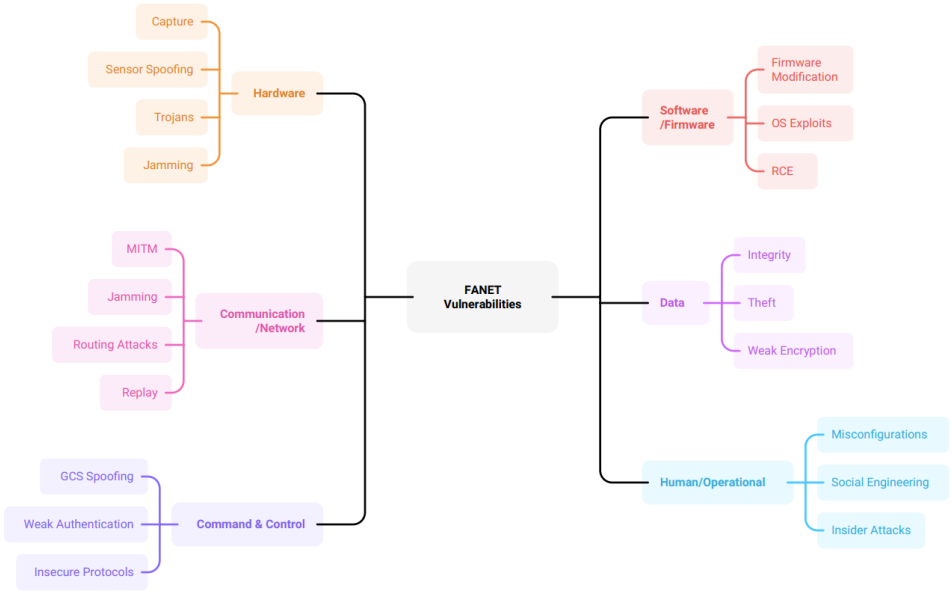


Figure 2. Layer-based classification of FANET vulnerabilities, covering hardware, software/firmware, communication/network, data, command-and-control, and human/operational threats.

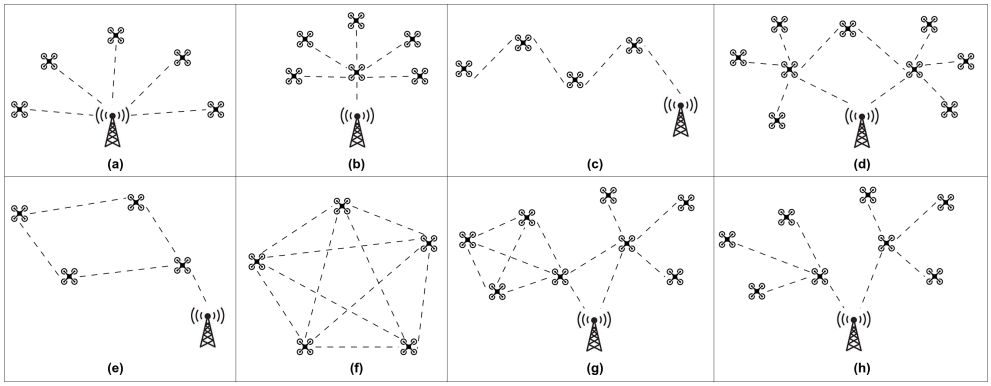


Figure 3. FANET architectural classification: (**a**) centralised single hop, (**b**) leader–follower star, (**c**) multi-hop relay, (**d**) clustered hierarchical, (**e**) single-layer cluster, (**f**) peer-to-peer mesh, (**g**) hybrid, and (**h**) centralised star.

2.2. Authentication Protocols in UAV and FANET Environments

Traditional security solutions, such as those based on PKI or complex certificate management, are often deemed unsuitable for UAV and FANET environments due to the inherent resource constraints of the aerial nodes [23]. Therefore, much of the recent research has focused on developing lightweight and efficient authentication schemes. Early work in FANET security primarily adapted protocols from MANETs, but these often failed to account for the high mobility, dynamic topology, and frequent link disruptions characteristic of FANETs [24]. More recent proposals have focused on identity-based cryptography (IBC) and elliptic curve cryptography (ECC) to reduce computational overhead [25,26]. Beyond security, contemporary multi-UAV systems increasingly operate as coordinated multi-agent infrastructures, such as cooperative sensing, mobile relays, and gateway-style data collection. For instance, multiple mobile LoRa gateways have been studied to enable large-scale low-power data transmission in dynamic environments, highlighting the need for scalable, relay-friendly security bootstrapping in multi-node deployments [27]. While ECC-based schemes offer better performance than traditional PKI, they still rely on software-based key storage, leaving them vulnerable to physical attacks or side-channel analysis if the UAV is captured [28]. A considerable portion of the literature focuses on the importance of mutual authentication between the UAV and the ground station, as well as secure communication between UAVs (UAV-UAV) during cooperative missions. Several protocols have been proposed that use hash functions and XOR operations to achieve lightweight authentication. However, formal analyses have shown that many of these approaches may be susceptible to attacks such as replay, impersonation, or man-in-the-middle [29]. Moreover, most of the proposed protocols have been evaluated only through simulation and have not been tested extensively on actual hardware. Designing a protocol that is both computationally efficient and strong against advanced adversarial models therefore remains an ongoing challenge.

2.3. PUF-Based Authentication for Lightweight Security

Physical Unclonable Functions have become a reliable alternative to traditional cryptographic key storage, offering a hardware-based solution for generating unique, tamper-resistant identifiers [11]. PUFs exploit the random manufacturing variations in silicon to create a Challenge–Response Pair (CRP) mechanism, where a given challenge (C) always produces a unique, repeatable response (R) for a specific device. This approach eliminates the need to store secret keys in non-volatile memory, thereby providing a strong defence against physical attacks [12]. The application of PUFs in resource-constrained environments, such as the Internet of Things (IoT) and Wireless Sensor Networks (WSNs), has been widely studied [30]. In the context of UAVs, PUF-based protocols are particularly attractive because they offer the following: (1) Lightweight Operation: PUF operations (challenge application and response generation) are typically fast and require minimal computational resources; (2) Tamper Resistance: Any attempt to physically clone or tamper with the PUF results in a change in its CRP behaviour, rendering the stored authentication data useless. Recent works have proposed various PUF-based protocols for UAVs, often combining PUFs with hash functions and nonces to achieve mutual authentication and session key agreement. Zhang et al. [31], for their PRLAP-IoD protocol, proposed an ECC-based, lightweight authentication and key negotiation scheme for IoD environments and provided a security analysis using the ROR model and AVISPA. While the inclusion of formal analysis is commendable, their model did not capture certain attack vectors. Subsequent cryptanalyses by Kwon et al. [32] and Choi et al. [33] showed that PRLAP-IoD remains vulnerable to impersonation, ephemeral secret leakage (ESL), and stolen verifier attacks. Moreover, their analyses indicated that the protocol fails to provide perfect forward secrecy (PFS). The protocol proposed by Choi et al. [33] offers a PUF-based authentication

scheme with strong formal verification using BAN logic, ROR, and AVISPA. It successfully addresses security flaws in a preceding scheme, PSECAS [34]. However, its architecture is fundamentally limited to single-hop UAV-GS interaction. The protocol does not provide any mechanism for secure peer-to-peer (UAV-UAV) communication, implicitly assuming that all interactions are mediated by the GS. This makes it impractical for multi-hop relaying or autonomous swarm operations where continuous GS connectivity cannot be guaranteed. The work of Pratap et al. [35] on the REHAS protocol is a strong contribution, featuring formal verification with Scyther and support for multi-hop communication. The authors provide a detailed analytical evaluation of computational and communication costs. However, their performance analysis is confined to theoretical calculations and simulations. Their study does not include any deployment on actual hardware, such as Raspberry Pi or similar embedded platforms. Our research extends this foundation by concentrating on the formal verification and development of the protocol.

2.3.1. PUF Reliability and Key Extraction in Practice

While PUFs are often abstracted as deterministic challenge–response primitives in protocol models, practical implementations are subject to response noise and bias caused by temperature/voltage variations, ageing, and readout uncertainty. Therefore, raw PUF responses are typically not used directly as cryptographic keys. Instead, standard fuzzy extractors or, equivalently, helper-data algorithms combine error correction with privacy amplification to derive stable, uniformly distributed keys from noisy PUF outputs, while storing only public helper data [36,37]. A related practical approach is the reverse fuzzy extractor, which shifts most of the error-correction workload to the verifier side and is therefore attractive for highly constrained devices [38]. In our setting, such stabilisation mechanisms are orthogonal to the message flow: wherever a PUF output R_i is used inside a hash or key-derivation expression, the implementation can replace R_i with a reproducible key derived from the noisy response using a fuzzy extractor, without changing the protocol structure [39,40].

2.3.2. Modelling Attacks and Controlled PUF Access

A known practical risk for so-called strong PUFs is machine learning-based modelling, where an attacker attempts to approximate the PUF function after collecting a sufficiently large labelled CRP dataset [41]. This risk is strongly coupled to the adversary's ability to obtain oracle-like access to the PUF interface and to query many challenges. The concept of controlled PUFs mitigates such risks by placing access-control logic around the PUF, restricting challenge exposure and enforcing protocol-mediated access [42]. Complementary implementation measures such as challenge rate limiting and reconfiguration can further reduce the feasibility of large-scale CRP harvesting.

2.4. Formal Verification of Security Protocols

Given the complexity and critical nature of authentication protocols, especially in military or industrial UAV applications, informal security analysis is generally insufficient. Formal verification techniques are essential to carefully verify the security properties of a protocol and identify minor vulnerabilities that might be missed by manual cryptanalysis [43]. Tools for formal verification can be broadly categorised into two main approaches: model checking and theorem proving. For security protocols, symbolic model-checking tools are particularly popular. Among these, Scyther and ProVerif are widely recognised [44].

- ProVerif: Based on the applied π-calculus, ProVerif is primarily used to prove secrecy and authentication properties under the Dolev–Yao threat model. It is highly effective for proving correctness but can sometimes be overly conservative.

- Scyther: Scyther is a push-button tool that automatically searches for attacks on security protocols. It is known for its efficiency and ability to quickly find attacks (falsification) or prove the absence of attacks (verification) up to a certain number of sessions [44].

Our prior research [18], similar to many other studies, relied on informal security analysis. The current work leverages the power of the Scyther tool to formally analyse our existing protocol, which led to the discovery of a critical vulnerability. The subsequent refinement and re-verification using Scyther ensure that the final proposed protocol is formally verified in the symbolic model for the UAV-GS sub-protocol within the analysed bounds, which, against the identified threats, is a crucial step often overlooked in the initial design of lightweight protocols. A summary of the addressed areas, associated challenges, relevant literature, and our contributions is provided in Table 1.

Table 1. Summary of contributions.

Area	Key Challenge	Relevant Literature	Our Contribution
UAV/FANET Auth.	High mobility, resource constraints, vulnerability to impersonation.	ECC-based schemes, Hash/XOR protocols.	Lightweight design, formally verified security, enhanced scalability for FANETs.
PUF-Based Security	Eliminating key storage, providing tamper resistance.	PUF applications in IoT/WSN, PUF-UAV protocols.	Formal verification and security hardening of a PUF-based protocol.
Formal Verification	Rigorous proof-of-security properties.	Scyther, ProVerif.	Use of Scyther to discover and eliminate a critical vulnerability, providing formal assurance for the UAV–GS phase within the symbolic model.

3. Threat Model and Security Goals

3.1. Threat Model

We assume a Dolev–Yao adversary that has full control over the wireless channel: the adversary can eavesdrop, replay, modify, inject, and drop messages. The adversary can also attempt impersonation and man-in-the-middle attacks. Cryptographic primitives (hash and PUF outputs) are assumed secure: $h(\cdot)$ is one-way and collision-resistant, and PUF responses are device-unique and computationally infeasible to predict without physical access to the specific device.

We further consider a realistic leakage model for UAVs: an adversary may read non-volatile memory contents (e.g., stored ID_{Ui}, current TID_i, and C_0). However, the adversary cannot clone the PUF behaviour to another device, and cannot obtain correct PUF responses for unseen challenges without the genuine hardware.

3.2. Security Goals

The proposed protocol aims to satisfy the following goals:

- Mutual authentication: UAV_i and GS authenticate each other (UAV–GS), and UAV_i and UAV_j confirm possession of the same fresh key (UAV–UAV).
- Session key secrecy: Session keys SK_i and SK_{ij} remain secret from any eavesdropper and intermediate relay nodes.
- Key freshness: Established session keys are fresh and bound to newly generated nonces.
- Replay resistance: Replayed messages are detected using nonce freshness checks and one-time TID usage.
- Impersonation and MITM resistance: Without valid PUF-derived secrets and GS-established session keys, an attacker cannot forge valid authentication hashes.
- Untraceability (privacy): Temporary identities TID_i prevent long-term tracking of UAVs on the air interface.

3.3. PUF Model and Operational Assumptions

PUF Instantiation and Ideal Assumption: We utilise the Ultracompact Switching-Voltage-Based Fully Reconfigurable RRAM PUF proposed in [40] for UAV deployment. This architecture is characterised by its low native instability and high density. Specifically, the device is capable of generating a response of at least 256 bits (up to 320 bits) within 0.4 µs when queried with a 32-bit challenge. Due to the high stability of this specific hardware, we adopt an ideal PUF assumption for the security analysis in this study. We assume that for a given challenge C_i, the PUF reliably reproduces the exact response R_i without bit errors. While we acknowledge this, typical silicon PUF outputs in practice may be affected by temperature, voltage variation, and aging; therefore, raw responses are commonly stabilised using helper-data-based mechanisms, such as fuzzy extractors or reverse fuzzy extractors or error correction algorithms, such as BCH, before being consumed as cryptographic material [38,45,46]. The integration of such mechanisms is reserved for future work. Consequently, the protocol analysis focuses on the cryptographic strength of the exchange, treating the PUF output as a stable unique identifier.

CRP Management and Scalability: The protocol requires fresh challenges to avoid CRP reuse. In our experimental validation, we utilised a set of 20 Challenge–Response Pairs to demonstrate the protocol's functionality on hardware. However, this is not a limitation of the system architecture. In real-world deployment, the ground station possesses a sufficient storage capacity to provision and manage a massive database of CRPs. This ensures that fresh challenges are available for long-term operation, preventing the reuse of challenges, which could compromise session unpredictability.

Resistance to Modelling via Protocol Design: Although we assume an ideal PUF, practical implementations must consider machine learning (ML) modelling attacks where adversaries attempt to predict responses based on collected CRPs. In our proposed framework, raw PUF responses are never transmitted over the public channel. Machine learning-based modelling attacks become relevant when an adversary can collect a large number of labelled CRPs [47,48]. The authentication transcripts contain cryptographic hashes computed over the PUF outputs. This significantly reduces the feasibility of transcript-based modelling attacks for a passive adversary observing the wireless channel. Modelling becomes relevant primarily under physical capture or extensive oracle access; controlled-PUF-style access control and rate limiting can further reduce this risk. To further reduce oracle-style access in the case of device capture or invasive probing, controlled-PUF-style access control and rate limiting can be implemented at the hardware/firmware level [42].

4. The Proposed Lightweight Authentication Protocol

This section details the proposed PUF-based authentication protocol, which is an enhanced and formally verified version of the one in our previous work, designed to secure communication within a large-scale FANET environment. The protocol is divided into two main phases: the registration phase and the authentication phase. The authentication phase is further subdivided into the UAV–ground station (UAV-GS) mutual authentication and the UAV–UAV session key establishment protocols.

4.1. Preliminaries and Notation

The protocol involves two main entities: the UAV and the GS. We assume the following cryptographic primitives and notations summarised in Table 2:

Table 2. Key notations.

Notation	Description
$h()$	A one-way cryptographic hash function.
$\oplus$	The bitwise XOR operation.
$PUF()$	Generates a response R for a given challenge C.
C_i	Challenges used in the PUF mechanism.
R_i	Responses generated by the PUF.
ID_{Ui}, ID_{Gi}	Identity of the UAV_i and the ground station, respectively.
T_{idi}	Temporary identity of UAV_i.
N_{ui}, N_{gi}	Fresh nonces generated by the UAV and GS, respectively.
SK_i	The established session key.
SK_{ij}	Session key established between UAV_i and UAV_j.
K_{ij}	Random key seed generated by the GS for the pair (UAV_i, UAV_j).
$Ticket_j$	Masked key seed intended for UAV_j (recoverable using SK_j or K_j^{LT}).
X_i	Masked version of K_{ij} sent from the GS to UAV_i.
N_{u3}, N_{u4}, N_{u5}	Fresh nonces generated by UAVs during the UAV–UAV protocol.
N_{g2}	Fresh nonce generated by the GS for the UAV–UAV key distribution.
$\|$	Concatenation operation.

4.2. Registration Phase

The registration phase is performed once in a secure environment before the UAV is deployed. The goal is to securely store the CRPs and initial secret parameters on both the UAV and the GS. Registration Steps:

1. The GS selects a set of challenges $C_0, C_1, C_2, \ldots$, and C_{10} and sends them to the UAV_i.
2. The UAV_i applies the challenges to its PUF to generate the corresponding responses: $R_0 = PUF(C_0)$, $R_1 = PUF(C_1)$, $R_2 = PUF(C_2), \ldots$, and $R_{10} = PUF(C_{10})$.
3. The UAV sends the generated responses back to the GS, which then stores each response together with its corresponding challenge in the GS database as a CRP entry.
4. The GS generates unique temporary IDs, $TID_1, TID_2, TID_3, \ldots$, and TID_{10}, for the UAV_i. The GS securely stores the following parameters in its database:

   $ID_{Ui} : (TID_1, TID_2, TID_3, \ldots, TID_{10})$
   $CRP_{U_i} : (C_0 - R_0, C_1 - R_1, \ldots, C_{10} - R_{10})$
5. The UAV$_i$ securely stores the following parameters in its non-volatile memory: IDU$_i$, $\{TID_1, TID_2, \ldots, TID_m\}$, and C_0. The remaining challenges and PUF responses are not stored but are generated on-the-fly during the authentication phase.

4.3. Authentication Phase

The authentication phase is initiated whenever a UAV_i attempts to communicate with the GS or another UAV_j.

4.3.1. UAV–Ground Station (UAV-GS) Mutual Authentication

This three-message protocol is used for mutual authentication and session key establishment between a UAV_i and the GS. This protocol incorporates the security fix identified through the Scyther analysis.

Step 1: $UAV_i \rightarrow GS$

Upon initiating the authentication process, the UAV_i generates a fresh nonce N_{u1}, and selects a temporary ID from its stored memory, which is then removed from the memory to prevent reuse. Using the stored parameter C_0, the UAV produces the on-the-fly PUF response R_0 and computes the initial authentication hash H_1 based on a combination of the generated nonce, the selected temporary ID, and the PUF response.

$$R_0 = PUF(C_0), \quad (1)$$

$$H_1 = h(ID_{Ui}|TID_i|R_0|N_{u1}), \tag{2}$$

$$UAV_i \rightarrow GS : (H_1, TID_i, N_{u1}) \tag{3}$$

Step 2: $GS \rightarrow UAV_i$

Upon receiving the message from the UAV, the GS first identifies the UAV by locating the corresponding ID_{Ui} based on the received TID_i. If the received TID_i has been used previously, the authentication process is terminated; otherwise, the TID_i is marked as used in the list. The GS then retrieves the PUF response R_0 corresponding to C_0 from the UAV's stored CRP list. The freshness of the received nonce N_{u1} is verified and recorded in the used nonce list. Using these values, the GS computes the expected initial authentication hash H_1 and compares it with the received H_1 for verification. If the verification fails, the session is terminated. Otherwise, the GS proceeds to prepare its response: it generates a fresh nonce N_{g1} and selects two challenges from the UAV's CRP list, for instance C_3, C_7. Subsequently, it computes the verification hash H_2 and the masked identity T_1, and sends the response to the UAV.

$$T_1 = ID_{Ui} \oplus ID_G, \tag{4}$$

$$N_h = N_{u1} \oplus N_{g1}, \tag{5}$$

$$H_2 = h((R_3 \oplus R_7)|T_1|(N_h)), \tag{6}$$

$$GS \rightarrow UAV : (C_3, C_7, H_2, T_1, N_h) \tag{7}$$

Step 3: $UAV_i \rightarrow GS$ (Key Confirmation)

Upon receiving the message from the GS, the UAV first validates the freshness of the GS nonce N_{g1} (recovers it from N_h). The UAV then derives the ground station identity as $ID_G = T_1 \oplus ID_{U_i}$ and verifies the received hash H_2. Using the received challenges C_3 and C_7, the UAV generates the corresponding on-the-fly PUF responses $R_3 = PUF(C_3)$ and $R_7 = PUF(C_7)$. If H_2 verification fails, the session is terminated. Otherwise, the UAV generates a fresh nonce N_{u2}, computes the masked nonce T_2, derives the session key SK_i, and finally sends an explicit key-confirmation hash H_{KC} to the GS.

$$N_{g1} = N_h \oplus N_{u1}, \tag{8}$$

$$H_2^* = h((R_3 \oplus R_7) \,|\, T_1 \,|\, N_h), \tag{9}$$

$$\text{verify } H_2^* \stackrel{?}{=} H_2. \tag{10}$$

$$T_2 = (N_{u2} \oplus N_{g1} \oplus N_{u1}), \tag{11}$$

$$SK_i = h((R_3 \oplus N_{g1}) \,|\, (R_7 \oplus N_{u2})), \tag{12}$$

$$H_{KC} = h(SK_i \,|\, N_{u1} \,|\, N_{g1} \,|\, N_{u2}), \tag{13}$$

$$UAV_i \rightarrow GS : (T_2, H_{KC}) \tag{14}$$

Upon receiving (T_2, H_{KC}), the GS recovers N_{u2} as $N_{u2} = T_2 \oplus N_{g1} \oplus N_{u1}$, computes SK_i using its stored CRP responses for C_3 and C_7, and verifies H_{KC}. If the verification holds, the GS confirms that UAV_i has derived the same session key and the authentication session is completed successfully.

Session Key Establishment:

Both parties can now compute the session key SK_i:

$$SKi = h((R_3 \oplus N_{g1})|(R_7 \oplus N_{u2})) \tag{15}$$

The GS recovers N_{u2} from T_2 using the received nonces: $N_{u2} = T_2 \oplus N_{g1} \oplus N_{u1}$. The GS further verifies, via the H_{KC} validation, that the UAV has generated the same SK_i key; thus, explicit key confirmation is achieved. Figure 4 represents the data flow between parties during the authentication process.

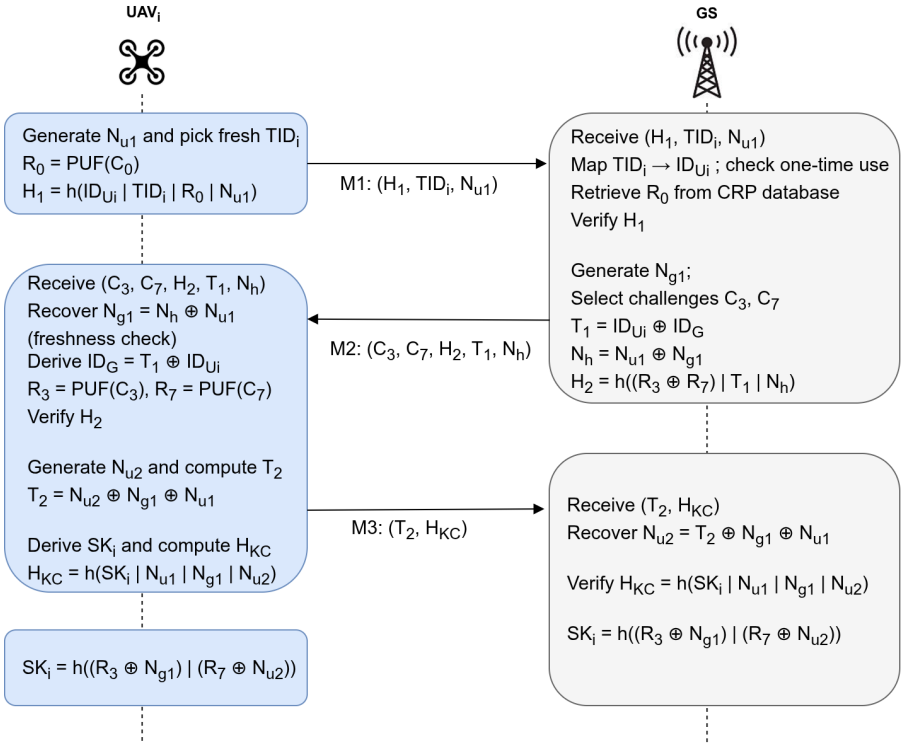


Figure 4. Message flow of the proposed three-message UAV-GS mutual authentication protocol.

4.3.2. UAV-UAV Session Key Establishment

In practical FANET deployments, UAV-to-UAV interactions can occur under a variety of architectural patterns, including single-layer, multi-layer, multi-group, head-drone-based clusters, and fully connected ad hoc formations. In our design, we prioritise scalability while deliberately avoiding architectural assumptions that would introduce unnecessary overhead or inefficiencies. Specifically, the proposed scheme is structured so that a configuration designed for a small swarm of 8–10 UAVs can seamlessly scale to larger formations of 100 or more UAVs without modification. Rather than assuming that every UAV must be capable of establishing secure links with all other UAVs—a requirement that becomes impractical as the network grows—we adopt a lightweight neighbour-centric model in which each UAV maintains secure communication only with a small set of nearby peers. This design choice reduces computational and communication load, preserves scalability, and enables efficient key establishment in large, dynamic FANET environments.

In large-scale FANET deployments, a UAV may not always have a direct single-hop link to the GS due to distance, obstacles, or mobility. Therefore, we assume that messages to the GS can be forwarded over multiple hops via a lead/head drone (or any already-authenticated UAV acting as a relay). Importantly, intermediate UAVs only forward packets and do not learn any secret material since all keying information is protected by per-UAV secrets established with the GS.

Goal and Rationale.

When two UAVs, UAV_i and UAV_j, wish to communicate securely, they must establish a fresh pairwise session key SK_{ij}. A naive approach would pre-distribute pairwise keys for all UAV pairs, which scales as $O(N^2)$ and becomes impractical as N grows. Instead, we

adopt an on-demand key establishment approach where the GS acts as a lightweight Key Distribution Centre (KDC): it generates a fresh session key seed and provides each UAV with recoverable key material using only its own GS-established secrets. This yields scalability proportional to the number of active links rather than the total number of UAV pairs.

Precondition.

Both UAV_i and UAV_j successfully completed the UAV-GS mutual authentication protocol described in the previous subsection (possibly via multi-hop forwarding). As a result, the keys UAV_i and the GS share SK_i, and UAV_j and the GS share SK_j are used solely to protect the distribution of the UAV-UAV session key seed; intermediate relays cannot recover the seed.

Protocol Messages.

The proposed UAV-UAV key establishment requires four messages and provides (i) fresh session key derivation and (ii) key confirmation between UAV_i and UAV_j.

Step 1: $UAV_i \rightarrow GS$ (Key Request)

UAV_i generates a fresh nonce N_{u3} and requests a secure link with UAV_j using temporary identifiers.

$$H_3 = h(SK_i \mid TID_i \mid TID_j \mid N_{u3}), \tag{16}$$

$$UAV_i \rightarrow GS : (TID_i, TID_j, N_{u3}, H_3). \tag{17}$$

The hash H_3 authenticates the request to the GS and binds it to TID_i, TID_j, and the fresh nonce N_{u3}.

Step 2: $GS \rightarrow UAV_i$ (Key Seed + Ticket)

Upon receiving the request, the GS verifies H_3 using SK_i. If verification fails, the session is terminated. Otherwise, the GS generates a fresh nonce N_{g2} and a fresh key seed K_{ij} (uniform random). It then prepares (i) a masked key seed for UAV_i and (ii) a ticket for UAV_j (decryptable only by UAV_j using SK_j).

$$X_i = K_{ij} \oplus h(SK_i \mid N_{u3} \mid N_{g2}), \tag{18}$$

$$Ticket_j = K_{ij} \oplus h(SK_j \mid TID_j \mid N_{g2}), \tag{19}$$

$$H_4 = h(K_{ij} \mid TID_i \mid TID_j \mid N_{u3} \mid N_{g2}), \tag{20}$$

$$GS \rightarrow UAV_i : (N_{g2}, X_i, Ticket_j, H_4). \tag{21}$$

Here, $Ticket_j$ is safe to be forwarded by any relay (including UAV_i) because recovering K_{ij} requires SK_j.

Step 3: $UAV_i \rightarrow UAV_j$ (Ticket Forward + Key Confirmation 1)

UAV_i recovers the seed and verifies integrity:

$$K_{ij} = X_i \oplus h(SK_i \mid N_{u3} \mid N_{g2}), \tag{22}$$

$$\text{verify } H_4 \stackrel{?}{=} h(K_{ij} \mid TID_i \mid TID_j \mid N_{u3} \mid N_{g2}). \tag{23}$$

If valid, UAV_i generates a fresh nonce N_{u4} and sends the ticket to UAV_j with a key-confirmation hash:

$$H_5 = h(K_{ij} \mid TID_i \mid TID_j \mid N_{u3} \mid N_{g2} \mid N_{u4}), \tag{24}$$

$$UAV_i \rightarrow UAV_j : (TID_i, Ticket_j, N_{u3}, N_{g2}, N_{u4}, H_5). \tag{25}$$

Step 4: $UAV_j \rightarrow UAV_i$ (Key Confirmation 2)

Upon receiving the message, UAV_j recovers the seed using its GS session key:

$$K_{ij} = Ticket_j \oplus h(SK_j \mid TID_j \mid N_{g2}), \tag{26}$$

It then verifies H_5. If verification succeeds, UAV_j generates a fresh nonce N_{u5} and returns the following:

$$H_6 = h(K_{ij} \mid TID_j \mid TID_i \mid N_{u4} \mid N_{u5}), \tag{27}$$

$$UAV_j \rightarrow UAV_i : (N_{u5}, H_6). \tag{28}$$

Finally, UAV_i verifies H_6 to complete mutual key confirmation.

UAV-UAV Session Key Derivation.

After successful confirmation, both UAVs derive the following final pairwise session key:

$$SK_{ij} = h(K_{ij} \mid N_{u3} \mid N_{g2} \mid N_{u4} \mid N_{u5}). \tag{29}$$

This construction ensures freshness and binds SK_{ij} to nonces from both UAVs and the GS. The seed K_{ij} is never transmitted in plaintext; it is only recoverable by the intended UAVs. Figure 5 illustrates the UAV–UAV key establishment message flow.

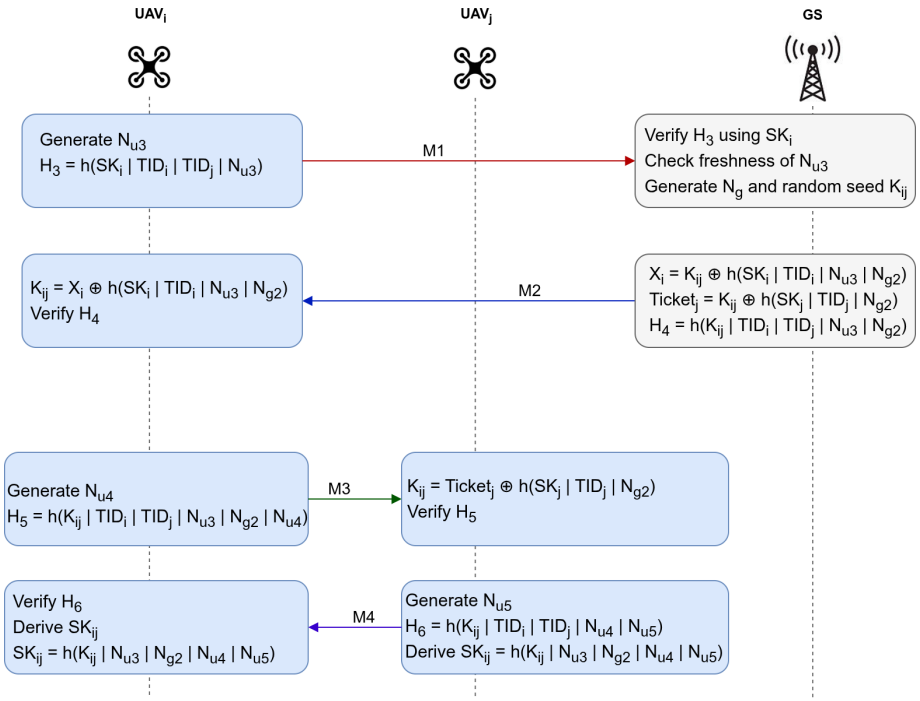


Figure 5. UAV–UAV session key establishment.

Unlike full-mesh pre-distribution, the GS does *not* assign $N-1$ keys to each UAV. Instead, a pairwise SK_{ij} is established only when needed. Therefore, memory requirements scale with the number of active neighbours (typically small in FANETs), and signaling overhead scales with the number of requested links rather than N^2. The head drone (or any relay UAV) can forward the messages between the GS and distant UAVs. However, since K_{ij} is masked using SK_i and SK_j, relays cannot recover K_{ij} or SK_{ij}. Although relay UAVs cannot perform eavesdropping, message tampering, or key recovery due to cryptographic protection, they may still disrupt communication by dropping or delaying packets. Such availability-oriented attacks are acknowledged but are not directly mitigated by the proposed authentication mechanism.

If a UAV is temporarily offline with respect to the GS, we optionally define a long-term per-UAV key encryption parameter:

$$K_j^{LT} = h(ID_{Uj} \mid R_0), \tag{30}$$

where $R_0 = PUF(C_0)$ is already available on-the-fly at the UAV and stored as a CRP entry at the GS. In this optional mode, the GS computes $Ticket_j = K_{ij} \oplus h(K_j^{LT} \mid TID_j \mid N_{g2})$ instead of using SK_j.

4.4. CRP Lifecycle and Scalability Management

The ground station maintains an epoch-based CRP pool for each UAV to ensure long-term sustainability. Within a given epoch, challenge pairs are selected according to a strict scheduling policy that ensures no deterministic repetition of the same challenge, thereby preventing replay attacks. The lifecycle is managed as follows:

Replenishment: When an epoch's CRP pool is near depletion, the GS triggers a secure maintenance window (re-enrollment phase) to query the UAV and populate a fresh set of CRPs for the next epoch.

Scalability: By centralising CRP management at the GS, the system avoids the $O(N^2)$ storage complexity associated with pairwise key pre-distribution in mesh networks. Each UAV only needs to maintain its own PUF interface, while the GS handles the database of CRPs, keeping the per-UAV on-board storage bounded and minimal.

5. Security Analysis and Formal Verification

This section provides a comprehensive security analysis of the proposed protocol. We first present an informal analysis against common attacks and then detail the formal verification process using the Scyther tool, which was instrumental in identifying and resolving a critical vulnerability in the initial protocol design.

5.1. Informal Security Analysis

The proposed protocol is designed to resist several common security threats prevalent in wireless communication environments. Mutual Authentication: The protocol ensures mutual authentication between the UAV_i and the GS. In Step 1, the GS authenticates the UAV_i by verifying H_1, which includes the UAV's unique PUF response (R_0) and a fresh nonce (N_{u1}). In Step 2, the UAV_i authenticates the GS by verifying H_2, which includes the GS's stored PUF responses ($R_3 \oplus R_7$), cryptographically linking the GS's response to the UAV's initial request. Session Key Secrecy: The session key SK_i is derived from two fresh nonces (N_{g1}, N_{u2}) and two PUF challenges (C_3', C_7'). Since N_{g1} and N_{u2} are fresh and known only to the GS and UAV_i, respectively, and the PUF challenges are not secret but are used as inputs to the hash function, the resulting session key is computationally infeasible for an adversary to derive. Replay Attack Resistance: The use of fresh nonces (N_{u1}, N_{g1}, N_{u2}) in every session ensures that an adversary cannot successfully replay a previously captured message. Any replayed message will fail the verification checks due to the nonces being outdated. Impersonation Attack Resistance: An adversary attempting to impersonate the UAV_i would need to know the correct PUF response R_0 to compute H_1 in Step 1, which is computationally infeasible due to the physical nature of the PUF. An adversary attempting to impersonate the GS would need to know the correct PUF responses R_3 and R_7 to compute H_2 in Step 2, which is also infeasible. Furthermore, the inclusion of H_1 in H_2 prevents an adversary from using a legitimate H_2 from a previous session to impersonate the GS in the current session. Physical Tampering Resistance: The reliance on PUFs ensures that the protocol is inherently resistant to physical attacks. If an adversary attempts to physically probe the UAV to extract the secret parameters, the

PUF mechanism will be altered, rendering the stored challenges and the corresponding responses useless for authentication.

5.2. Security Analysis

This subsection presents a qualitative analysis of the proposed protocol with respect to common attacks in UAV and FANET environments. We focus on the secrecy and robustness of the session keys SK_i and SK_{ij}, and on the resistance against eavesdropping, replay, impersonation, man-in-the-middle (MITM), and denial-of-service (DoS) attacks.

5.2.1. Resistance to Eavesdropping and Key Disclosure

In the proposed design, neither the UAV–GS session key SK_i nor the UAV–UAV session key SK_{ij} is ever transmitted in clear form. Instead, both keys are derived from random nonces and PUF-based secrets using the one-way hash function $h(\cdot)$.

For the UAV–GS protocol, the session key SK_i is computed as

$$SK_i = h((R_3 \oplus N_{g1}) \,|\, (R_7 \oplus N_{u2})),$$

where the PUF responses R_3 and R_7 are never stored on the UAV and appear only as outputs of the PUF for specific challenges selected by the GS. An external eavesdropper observing all protocol messages only sees hashed and XOR-masked values; recovering SK_i would require inverting $h(\cdot)$ or predicting the PUF responses, both assumed infeasible.

For the UAV–UAV protocol, the GS generates a fresh seed K_{ij}, which is hidden as

$$X_i = K_{ij} \oplus h(SK_i \,|\, N_{u3} \,|\, N_{g2}), \quad Ticket_j = K_{ij} \oplus h(SK_j \,|\, TID_j \,|\, N_{g2}),$$

and finally transformed into the session key

$$SK_{ij} = h(K_{ij} \,|\, N_{u3} \,|\, N_{g2} \,|\, N_{u4} \,|\, N_{u5}).$$

An eavesdropping adversary who does not know SK_i or SK_j cannot extract K_{ij} from X_i or $Ticket_j$, and thus cannot obtain SK_{ij}. Intermediate relay UAVs only see these masked values as well, so they also cannot learn any key material.

5.2.2. Replay Attack Resistance

The protocol uses multiple nonces and one-time temporary identities to provide freshness and replay protection. In the UAV–GS phase, each run uses a fresh N_{u1} and N_{g1}, and the GS maintains a list of used nonces and consumed temporary identities TID_i. Any attempt to replay an old message will either contain a previously used TID_i or an old nonce N_{u1}, causing the GS (or UAV) to abort the session.

In the UAV–UAV phase, the GS and UAVs use fresh nonces N_{u3}, N_{g2}, N_{u4}, and N_{u5}. The integrity hashes H_3–H_6 bind these nonces together with TID_i, TID_j, and the random seed K_{ij}. Replaying an old $(Ticket_j, N_{u3}, N_{g2}, N_{u4}, H_5)$ to UAV_j will fail, because either the GS will not confirm the corresponding context, or the receiving UAVs will detect mismatches in the hash verification (e.g., H_5 and H_6 no longer match fresh nonces). Therefore, the adversary cannot force two UAVs to reuse an old session key by replaying messages.

5.2.3. Impersonation Attacks

To successfully impersonate a UAV towards the GS, an attacker would have to generate a valid H_1 or H_3 without knowing the true PUF responses or the GS-established session keys. The values

$$H_1 = h(ID_{Ui} \,|\, TID_i \,|\, R_0 \,|\, N_{u1}), \qquad H_3 = h(SK_i \,|\, TID_i \,|\, TID_j \,|\, N_{u3}),$$

rely on secrets that are not observable on the communication channel. Since PUF responses are unique and not clonable to another device, and SK_i is only known to UAV_i and the GS, an attacker with only public transcripts cannot create valid hashes. Consequently, impersonating UAV_i to the GS (or to another UAV via H_5 and H_6) is computationally infeasible.

Similarly, impersonation of the GS to a UAV requires knowledge of the correct CRPs and secret nonces in order to compute valid values such as H_2, H_4, or the masked nonces and identities. Without access to the GS database, an attacker cannot generate matching hashes that survive verification on the UAV side. This prevents both GS impersonation and rogue "fake base station" attacks.

5.2.4. Man-in-the-Middle (MITM) Attacks

The UAV–GS and UAV–UAV protocols provide implicit key confirmation and bind the session keys to identities and nonces via the hash values. In the UAV–UAV phase, both UAV_i and UAV_j verify integrity hashes

$$H_5 = h(K_{ij} \mid TID_i \mid TID_j \mid N_{u3} \mid N_{g2} \mid N_{u4}),$$

$$H_6 = h(K_{ij} \mid TID_j \mid TID_i \mid N_{u4} \mid N_{u5}),$$

before accepting the session key. An MITM adversary attempting to interpose between UAV_i and UAV_j would need to know K_{ij} to forge a consistent H_5 and H_6, which is impossible without SK_i or SK_j.

Moreover, even if the adversary forwards messages through itself or reorders them, it cannot change any field without causing a hash mismatch at one of the endpoints. As a result, the attacker cannot cause both UAVs to end up with different keys, nor can it negotiate a key with each UAV separately without detection; any such deviation breaks the hash checks and terminates the protocol.

5.2.5. Denial-of-Service (DoS) and Node Capture

As with most wireless protocols, an active attacker can perform jamming or packet dropping, which results in denial of service but does not compromise cryptographic secrets. The use of multiple nonces, one-time TID_i, and PUF responses ensures that partially completed or aborted sessions do not leak reusable key material.

If an attacker physically captures a UAV and reads its non-volatile memory, they may obtain ID_{Ui}, some stored TID_i, and the challenge C_0. However, the attacker still cannot clone the PUF behaviour on another device and cannot predict responses for unseen challenges held only at the GS. Additionally, the UAV–UAV key distribution relies on SK_i or (in the optional offline mode) on $K_i^{LT} = h(ID_{Ui} \mid R_0)$, which requires access to the genuine PUF hardware to be used correctly. Therefore, while node capture can enable local misuse of that particular device, it does not directly reveal other UAVs' keys or the GS's secrets, and compromised identifiers and temporary IDs can be revoked by updating the CRP lists and TID pools at the GS.

5.3. Formal Verification Using Scyther

We modelled the improved UAV–GS mutual authentication protocol in Scyther's Security Protocol Description Language (SPDL) under the standard Dolev–Yao adversary model, where the attacker has full control over the communication channel (eavesdropping, replay, modification, injection, and message blocking), while cryptographic primitives are assumed ideal. The UAV and the GS are represented as roles I and R, respectively.

We verified (i) the secrecy of the established session key material SK_i, and (ii) authentication/liveness properties using `Nisynch` and `Alive` claims for both roles. The Scyther

analysis reported *"No attacks"* for all secrecy and authentication claims, as shown in Figure 6. This indicates that within the analysed bounds, no execution trace violates session key secrecy or the intended mutual authentication properties, supporting the protocol's resistance against replay, impersonation, and man-in-the-middle attacks in the symbolic model.

```
muhammet@ground-station:~/Scyther$ python3 scyther.py  uavpufprotocolfinal-fixed3.spdl

Verification results:
claim id [UAVPUFProtocolFINAL,I1], Secret(conc(xor(PUF(I,R,C1),Ng1),xor(PUF(I,R,C2),Nu2)))      : No attacks.
claim id [UAVPUFProtocolFINAL,I2], Nisynch      : No attacks.
claim id [UAVPUFProtocolFINAL,I3], Alive        : No attacks.
claim id [UAVPUFProtocolFINAL,R1], Secret(conc(xor(PUF(I,R,C1),Ng1),xor(PUF(I,R,C2),Nu2)))      : No attacks.
claim id [UAVPUFProtocolFINAL,R2], Nisynch      : No attacks.
claim id [UAVPUFProtocolFINAL,R3], Alive        : No attacks.
```

Figure 6. Scyther verification results showing no attacks for all secrecy and authentication claims.

Scyther-Guided Refinement

We modelled the UAV–GS mutual authentication protocol in Scyther's SPDL under the standard Dolev–Yao adversary model, where the attacker can eavesdrop, replay, modify, inject, and block messages. In the initial baseline model, Scyther reported violations of secrecy and authentication-related claims, indicating that the protocol messages were not sufficiently protected against replay/synchronisation traces in the symbolic setting.

Guided by these findings, we refined the protocol in three key aspects:

(i) The GS nonce is no longer exposed directly and is instead conveyed through the masked value $N_h = N_{u1} \oplus N_{g1}$;
(ii) The session key derivation is bound to PUF-derived responses mixed with fresh nonces (Equation (15)), ensuring that public transcripts alone cannot reconstruct SK_i;
(iii) An explicit key-confirmation hash H_{KC} (Equation (13)) is introduced so that the GS can confirm that UAV_i derived the same SK_i.

We then re-modelled the improved protocol and re-ran Scyther. As shown in Figure 6, Scyther reports *No attacks* for all secrecy and authentication claims within the analysed bounds, supporting the protocol's resistance against replay, impersonation, and man-in-the-middle attacks in the symbolic model.

The Scyther model specifically targets the UAV–GS mutual authentication and key confirmation phase, operating under the standard symbolic Dolev–Yao adversary model with idealised cryptographic primitives. Consequently, the verification results validate the secrecy and agreement properties specifically for this sub-protocol within the analysed session bounds. The subsequent UAV–UAV ticket-based key establishment relies on standard Key Distribution Centre-style reasoning (where trust is anchored in the GS) and explicit hash-based key confirmation; however, this phase is not explicitly modelled in Scyther in the current manuscript.

5.4. Security of UAV-UAV Session Key Establishment

The UAV–UAV protocol inherits trust from the UAV–GS phase, where UAV_i and UAV_j establish individual session keys SK_i and SK_j with the GS. For a requested pair (i, j), the GS generates a fresh random seed K_{ij} and delivers it in a recoverable form to each endpoint: X_i is recoverable only with SK_i, and $Ticket_j$ is recoverable only with SK_j (or K_j^{LT} in the optional offline mode). Therefore, any relay UAV forwarding $(X_i, Ticket_j)$ cannot recover K_{ij} and cannot derive SK_{ij}.

Key confirmation is achieved via the integrity hashes H_5 and H_6, which bind K_{ij} to the identities and fresh nonces $(N_{u3}, N_{g2}, N_{u4}, N_{u5})$. An adversary without K_{ij} cannot forge valid H_5/H_6, preventing impersonation and MITM attacks. Freshness follows from the nonces contributed by both UAVs and the GS, ensuring that replayed transcripts are

rejected. Finally, the session key $SK_{ij} = h(K_{ij}|N_{u3}|N_{g2}|N_{u4}|N_{u5})$ is never transmitted in plaintext and is computationally infeasible to recover from observed traffic.

6. Results and Analysis

6.1. Initial Protocol Vulnerability Analysis (The "Flawed" Protocol)

To validate the robustness of the proposed authentication approach, we first modelled an intuitive baseline variant of the UAV–GS protocol in Scyther. In this initial design, (i) the GS nonce was transmitted without masking, and (ii) the session key construction included a nonce-only component that could be reconstructed from public transcripts under certain traces. The SPDL model was analysed under the Dolev–Yao adversary model. Table 3 summarises the root causes and their security implications.

Table 3. Critical vulnerabilities.

Claim (Target)	Result	Analysis of the Flaw
claim (R, Secret, Ng1)	FAIL (At least 1 attack), Secret (Ng1): At least 1 attack.	The ground station's nonce, Ng1, was transmitted in plaintext in Message 2. This allows an attacker to capture it.
claim (R, Secret, Ski)	FAIL (At least 2 attacks), Secret (Ski): At least 2 attacks.	The session key, Ski, was compromised. An attacker can derive the second half of the key (Ng1$\oplus$ Nu2) by capturing Ng1 from M2, Nu1 from M1, and T2 from M3.
claim (I, Nisynch)	FAIL (At least 1 attack), Nisynch: At least 1 attack.	The UAV (I) cannot verify that the Ng1 it received is fresh (non-replayed), leading to synchronisation attacks.
claim (I, Alive)	FAIL (At least 1 attack), Alive: At least 1 attack.	The protocol is vulnerable to a replay attack where the adversary impersonates the ground station.
claim (I, Secret, Ski)	OK (No attacks), Secret (Ski): No attacks.	This is a "false positive" success; the UAV believes the key is secure, even though the GS role (R) knows it has been compromised.

Analysis Conclusion: The initial protocol design is fundamentally insecure due to the leakage of Ng1 and the flawed construction of the Ski, which relies on publicly available nonces rather than solely on secret PUF responses.

6.2. Formal Verification of the Improved Protocol

The vulnerability analysis in Section 5.1 revealed that the initial version of the protocol was fundamentally insecure. The Scyther analysis identified two critical flaws:

Nonce Leakage: The ground station's nonce, $Ng1$, was transmitted in an unencrypted way in M2, which caused Secret(Ng1): At least one attack.

Session Key Compromise: An attacker could passively intercept $Nu1$ from M1, $Ng1$ from M2, and $T2$ from M3 to derive $Nu2$. This allowed the attacker to compute the entire second half of the session key $(Ng1 \oplus Nu2)$, and this caused Secret(Ski): At least two attacks.

To address these severe vulnerabilities, we implemented a crucial modification to the transferred messages and session key derivation formula. This is the scheme adopted in the final design:

$$\text{Ski} = (\text{PUF}(\text{C3}) \oplus \text{Ng1})|(\text{PUF}(\text{C7}) \oplus \text{Nu2})$$

This revised construction directly mitigates the identified flaws. The session key is no longer composed of a secure half (PUF-based) and an insecure half (nonce-based). Instead, both halves of the key are now cryptographically bound to the secret, unclonable PUF responses.

Even if an attacker successfully intercepts all public nonces (Nu1, Ng1, and derives Nu2), they cannot compute the session key. The attacker only knows Ng1 and Nu2, and lacks the secret PUF(C3) and PUF(C7) values required to reverse the XOR operation. The

nonces (Ng1, Nu2) now serve as "salts" to the PUF secrets, ensuring the session key is unique for every session while remaining anchored to the hardware's root of trust.

We implemented this new key derivation formula within the same Scyther model and re-ran the verification. The results for the improved protocol are presented in Table 4.

Table 4. Summary of Scyther verification claims and results for the improved UAV–GS mutual authentication protocol.

Claim (Target)	Role	Result
Secret (Ski)	I (UAV)	No attacks.
Secret (Ski)	R (GS)	No attacks.
Nisynch	I (UAV)	No attacks.
Alive	I (UAV)	No attacks.
Nisynch	R (GS)	No attacks.
Alive	R (GS)	No attacks.

Analysis of Results: The verification of the improved protocol confirms that all previously identified attacks are successfully mitigated.

Session Key Secrecy (I1, R2): The session key secrecy claims for both roles now hold under formal verification. By binding the nonces to the secret PUF responses, the session key can no longer be derived from publicly observable protocol messages, thereby eliminating the critical vulnerability identified in the initial design.

Replay Attack Mitigation (I2, I3): The protocol now ensures resistance to replay and synchronisation attacks for the UAV role. The implicit verification mechanism—requiring the UAV to reuse its original Nu1 when validating the hash H2 in Message 2—prevents an adversary from successfully replaying or desynchronising protocol messages, assuming the session key derivation remains secure.

This formally verified model demonstrates that the proposed protocol achieves robust mutual authentication and secure session key establishment, validating its suitability for the target UAV environment. The security properties and attack resistance of the proposed protocol are summarised in Table 5.

Table 5. Security properties of the proposed protocol.

Attack	UAV–GS Protection	UAV–UAV Protection
Eavesdropping	$SK_i = h((R_3 \oplus N_{g1})\|(R_7 \oplus N_{u2}))$; keys never transmitted; PUF responses not stored on the UAV.	$SK_{ij} = h(K_{ij}\|N_{u3}\|N_{g2}\|N_{u4}\|N_{u5})$; K_{ij} is masked in X_i and $Ticket_j$ with SK_i and SK_j.
Replay	Fresh nonces N_{u1}, N_{g1} and one-time TID_i; GS and UAV maintain lists of used nonces and temporary IDs.	Fresh nonces N_{u3}, N_{g2}, N_{u4}, N_{u5} bound into H_3–H_6; out-of-context or repeated messages are rejected.
Impersonation (UAV)	Adversary must know R_0 and CRPs to forge H_1; PUF uniqueness and unclonability prevent cloning to another device.	Adversary must know SK_i or SK_j and K_{ij} to forge H_5 and H_6; these values are not derivable from observed traffic.
Impersonation (GS)	Forging H_2 requires access to the GS CRP database and correct R_3, R_7; infeasible for an external attacker.	A fake GS cannot generate consistent $(X_i, Ticket_j)$ and H_4 without knowing SK_i and SK_j.
Man-in-the-Middle (MITM)	Mutual verification of H_1 and H_2 with fresh nonces; both sides derive the same SK_i or abort on mismatch.	Mutual verification of H_5 and H_6 before accepting SK_{ij}; any message modification breaks the hashes and is detected.
Denial-of-Service (DoS)/Jamming	Attacker may jam or drop messages, causing service disruption but no key disclosure; each new run uses fresh nonces.	Same as UAV–GS; aborted or incomplete pairwise sessions do not reveal K_{ij} or SK_{ij}.
Node capture	CRPs are stored only at the GS; the PUF behaviour cannot be cloned. Compromise is limited to the captured UAV and can be revoked.	Compromise of a single UAV does not reveal other UAVs' SK_{ij} or GS secrets; new keys and TIDs can be provisioned by the GS.

6.3. Empirical Performance Results

This subsection evaluates the efficiency of the proposed UAV–GS and UAV–UAV protocols using an analytical cost model. Since the protocol is designed for resource-constrained UAV platforms, we quantify the number of lightweight operations (PUF evaluations, hash computations, and XOR operations) and the total communication payload in bits.

6.3.1. Performance and Cost Analysis

To quantify the efficiency of the proposed schemes, we evaluate (i) communication overhead, (ii) computational overhead, and (iii) energy cost. For a fair comparison, we use standard bit-length assumptions widely adopted in lightweight UAV/IoD authentication studies: $|ID| = |N| = |h(\cdot)| = 160$ bits, $|TID| = 64$ bits, and $|C| = 32$ bits. XOR costs are considered negligible compared to hash and PUF operations.

Communication Overhead

Table 6 reports the message-wise overhead for both the UAV–GS and UAV–UAV phases. Notably, the UAV–UAV key establishment is executed on demand (per active neighbour link), which prevents the quadratic $O(N^2)$ pre-distribution overhead of full-mesh keying.

Table 6. Communication overhead of the proposed protocols.

Phase	Msg.	Transmitted fields	Bits
UAV–GS	M_1	(H_1, TID_i, N_{u1})	$160 + 64 + 160 = 384$
	M_2	$(C_3, C_7, H_2, T_1, N_h)$	$32 + 32 + 160 + 160 + 160 = 544$
	M_3	(T_2, HKC)	$160 + 160 = 320$
Total			**1248**
UAV–UAV	M_1	$(TID_i, TID_j, N_{u3}, H_3)$	$64 + 64 + 160 + 160 = 448$
	M_2	$(N_{g2}, X_i, Ticket_j, H_4)$	$160 + 160 + 160 + 160 = 640$
	M_3	$(TID_i, Ticket_j, N_{u3}, N_{g2}, N_{u4}, H_5)$	$64 + 160 + 160 + 160 + 160 + 160 = 864$
	M_4	(N_{u5}, H_6)	$160 + 160 = 320$
Total			**2272**

Computational Overhead

We count the dominant primitives (PUF evaluations and hash computations). XOR and concatenation are lightweight and omitted from timing. Let T_h and T_{PUF} denote the execution time of one hash and one PUF evaluation, respectively. The UAV–GS phase requires three PUF evaluations and four hashes on the UAV side, as well as four hashes on the GS side. The UAV–UAV phase is PUF-free (it relies on the SK_i and SK_j already established with the GS) and mainly consists of hash computations for ticket recovery and key confirmation. The resulting per-session computational cost, expressed as the number of PUF evaluations, hash computations, and XOR operations for each entity, is summarised in Table 7.

Table 7. Per-session computational cost of the proposed authentication protocols.

Protocol	Entity	PUF	Hash	XOR
UAV–GS Mutual Auth.	UAV_i	3	4	7
UAV–GS Mutual Auth.	GS	0	4	7
UAV–UAV Key Establish.	UAV_i (initiator)	0	6	1
UAV–UAV Key Establish.	UAV_j (responder)	0	4	1
UAV–UAV Key Establish.	GS (KDC)	0	4	2

Using typical lightweight parameters (e.g., $T_h = 0.054$ ms and $T_{PUF} = 0.0023$ ms), the total computation time is approximately 0.439 ms per UAV–GS and 0.756 ms per UAV–UAV link.

Energy Cost

To highlight practical impact, we also report energy under the following common model: $E_h = 0.00046$ mJ, $E_{PUF} = 0.0108$ mJ, and per-byte radio costs $E_s = 5.9\mu$J (send), $E_r = 4.7\mu$J (receive). Under this model, computation energy is dominated by communication energy. For UAV–GS, the total radio energy is approximately 1.654 mJ per authentication run, while computation energy is about 0.036 mJ. For UAV–UAV, the total radio energy is about 3.010 mJ per pairwise link establishment.

6.3.2. Experimental Setup and Measurement Methodology

The experimental setup consists of one ground station and one UAV node per run. The GS was implemented on a laptop equipped with an Intel Core i7-10750H processor (6 cores, 12 threads) with 32 GB RAM and running Ubuntu 24.04 LTS (x86_64). The UAV node was implemented on either a Raspberry Pi 4 Model B (4 GB RAM) or a Raspberry Pi 5 Model B (8 GB RAM), both running 64-bit Debian GNU/Linux. All nodes communicated via their onboard IEEE 802.11 wireless interfaces.

Client-side end-to-end latency includes TCP connection establishment and the execution of the three-message authentication exchange. Server-side timing was measured as the processing window from receiving M1 to receiving M3. The GS hardware and software environment was kept fixed across all runs to isolate the impact of the UAV-side platform.

Figure 7 depicts the experimental testbed, showing the ground station on the laptop and the UAV nodes on Raspberry Pi 4B and Raspberry Pi 5.



Figure 7. Experimental testbed used for latency evaluation. The ground station runs on a laptop, while the UAV node is implemented on Raspberry Pi 4B or Raspberry Pi 5 in separate experimental runs. All nodes communicate over IEEE 802.11.

Hardware Comparison

The proposed three-message UAV–GS mutual authentication protocol was evaluated on two UAV hardware platforms (Raspberry Pi 4B and Raspberry Pi 5), keeping the ground station fixed on the same laptop. For each UAV platform, three independent runs of 1000 consecutive successful sessions were collected (n = 3000 per platform). On the UAV side, the end-to-end authentication latency (Total_Auth_Time_ms) was decomposed into connection

establishment time (Connect_Time_ms) and post-connection protocol execution time (Protocol_Time_ms). On the GS side, we report the session processing window (Total_Server_ms), measured from receiving M1 to receiving M3.

Table 8 shows that end-to-end latency is dominated by connection establishment on both platforms: Connect_Time_ms accounts for approximately 90–91% of the mean Total_Auth_Time_ms. The computation-bound part of the protocol remains lightweight; notably, Raspberry Pi 5 reduces Protocol_Time_ms from 12.93 ms to 10.33 ms on average (and improves tail latency, e.g., p99 from 39.49 ms to 28.41 ms). In contrast, the tail of the end-to-end latency is still largely governed by transport and wireless variability, as reflected by the Connect_Time_ms distribution; consequently, Total_Auth_Time_ms p95/p99 does not necessarily improve with the hardware upgrade.

Table 8. Client-side latency statistics over three runs (n = 3000 per platform).

Metric	Pi 4B				Pi 5			
	Mean	**Median**	**p95**	**p99**	**Mean**	**Median**	**p95**	**p99**
Total_Auth_Time_ms	132.31	162.14	208.23	230.88	121.48	113.77	223.98	232.63
Connect_Time_ms	119.05	149.08	193.65	206.41	111.04	103.04	211.51	217.27
Protocol_Time_ms	12.93	10.58	23.71	39.49	10.33	9.05	18.77	28.41

GS-side results in Table 9 remain in the single-digit-millisecond regime and improve under the Pi 5 condition (mean Total_Server_ms decreases from 6.80 ms to 5.27 ms). Figure 8 and Figure 9 compare the empirical distributions (CDF and boxplots) of the Total, Connect, Protocol, and Total_Server metrics across Raspberry Pi 4B and Raspberry Pi 5, further highlighting that the primary bottleneck is connection/channel overhead rather than cryptographic computation.

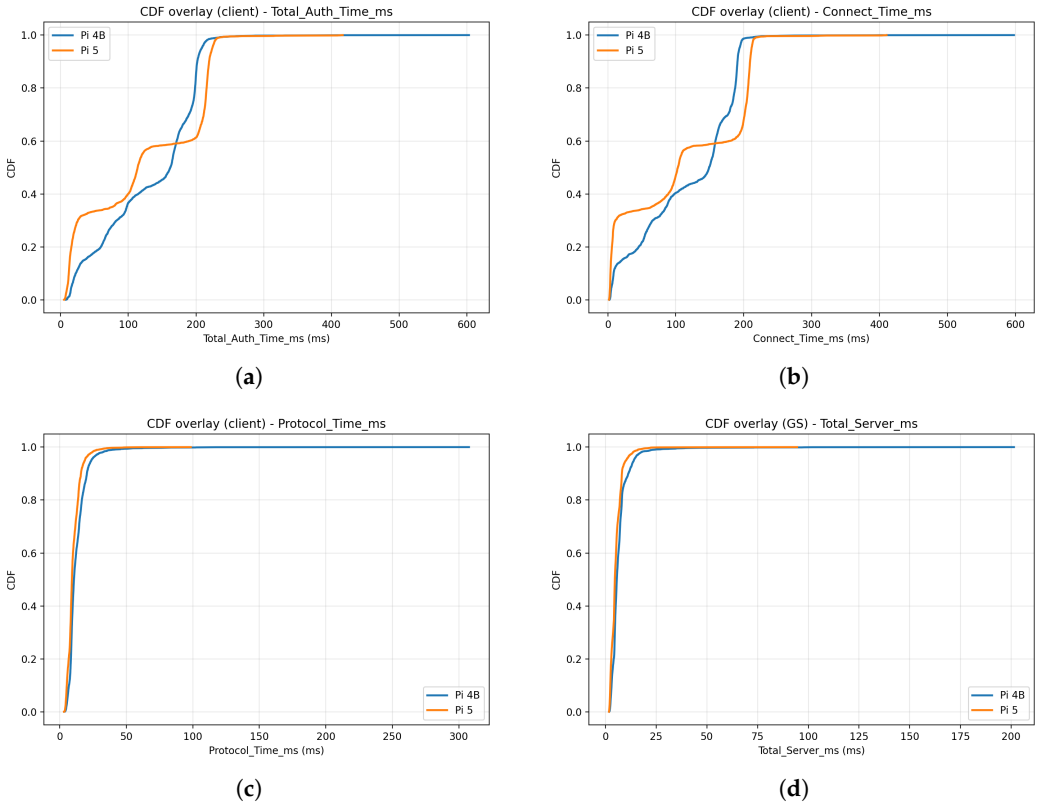


Figure 8. Empirical CDF comparison (Pi 4B vs. Pi 5) for latency metrics (n = 3000 per platform). (**a**) Total_Auth_Time_ms (UAV); (**b**) Connect_Time_ms (UAV); (**c**) Protocol_Time_ms (UAV); (**d**) Total_Server_ms (GS).

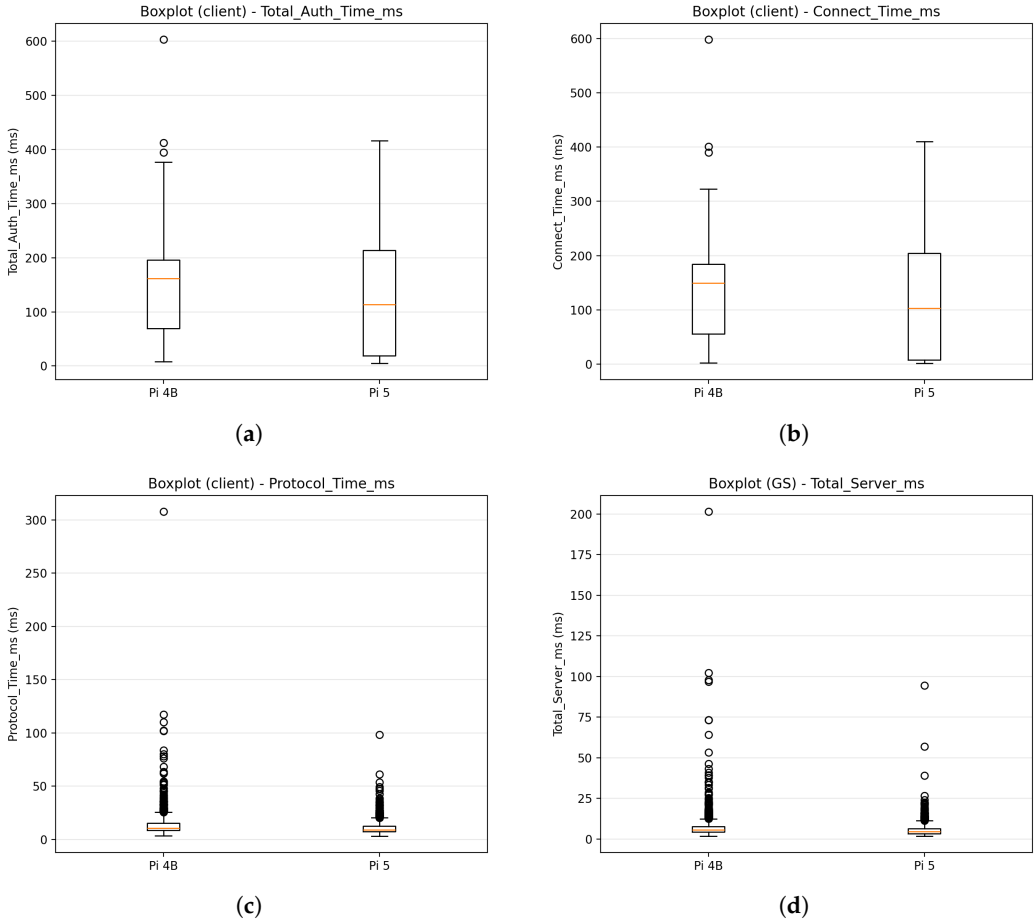


Figure 9. Boxplot comparison (Pi 4B vs Pi 5) for client- and server-side latency metrics over successful sessions (n = 3000 per platform). (**a**) Total_Auth_Time_ms (UAV); (**b**) Connect_Time_ms (UAV); (**c**) Protocol_Time_ms (UAV); (**d**) Total_Server_ms (GS).

Table 9. GS-side session window statistics over three runs (n = 3000 per platform).

Metric	Pi 4B Condition				Pi 5 Condition			
	Mean	**Median**	**p95**	**p99**	**Mean**	**Median**	**p95**	**p99**
M1_Recv_ms	0.110	0.008	1.303	0.434	0.082	0.008	1.643	0.055
M3_Recv_ms	6.44	5.12	6.36	13.76	4.95	4.16	2.99	9.58
Total_Server_ms	6.80	5.58	13.99	23.52	5.27	4.60	10.17	15.92

6.3.3. Scalability and Comparative Overhead

A naive full-mesh pre-distribution approach requires each UAV to store $(N-1)$ pairwise keys and forces the infrastructure to manage $O(N^2)$ keys. In contrast, our UAV–UAV design is on-demand: the GS generates the seed K_{ij} only when UAV_i and UAV_j need a secure link. Thus, storage and signalling scale with the number of active neighbour links rather than the total number of UAV pairs. In typical FANET operations where each UAV maintains secure links with a small number of nearby peers, the effective overhead becomes approximately $O(Nd)$ with a small constant degree $d \ll N$.

To contextualise this scalability advantage, Table 10 compares the communication payload sizes of representative UAV/IoD authentication schemes under the common bit-length assumptions. Since attacker models, PUF reliability assumptions, and verification coverage differ across the literature, Table 11 additionally provides a qualitative feature comparison (where "NR" denotes *not reported* in the corresponding reference). Together,

these two views highlight that our design reduces UAV–GS signalling overhead while still providing relay-friendly operation and explicit key confirmation.

Table 10. Communication overhead comparison with representative PUF-based schemes.

Scheme	Of Messages	Total Overhead (Bits)
Pratap et al. [35]	5	2496
Zhang et al. [31]	4	2272
Tanveer et al. [49]	4	2272
Sharma et al. [34]	4	2688
Choi et al. [33]	4	2112
This work (UAV–GS)	3	**1248**

Table 11. Qualitative comparison of representative UAV/IoD authentication schemes.

Scheme	Cryptographic Primitive	PUF Noise Handling	Formal Verification	UAV–UAV Support	Relay/Multi-Hop	Explicit Key Confirmation
Zhang et al. [31]	ECC, Hash	NR	Yes (ROR, AVISPA)	Yes	NR	NR
Tanveer et al. [49]	Biometrics, Hash	Fuzzy Extractor	Yes	NR	NR	NR
Sharma et al. [34]	PUF, XOR, Hash	NR	NR	NR	NR	NR
Choi et al. [33]	PUF, ECC, Hash	NR	Yes (BAN, ROR, AVISPA)	NR	NR	NR
Pratap et al. [35]	HECC, Hash	NR	Yes (Scyther)	Yes	Yes	Yes
This work	PUF, Hash, XOR	FE/RFE compatible	Scyther (UAV–GS)	Yes	Yes	Yes (HKC)

7. Discussion

The embedded evaluation on Raspberry Pi 4B and Raspberry Pi 5 demonstrates that the proposed UAV–GS protocol is computation-lightweight in practice. As reported in Table 8, the protocol-only component (Protocol_Time_ms) remains in the single-digit-to-low-tens-of-milliseconds regime and improves on the Pi 5 condition (mean 12.93 ms $\rightarrow$ 10.33 ms; p99 39.49 ms $\rightarrow$ 28.41 ms). However, the end-to-end authentication latency is dominated by transport connection setup and channel behaviour: Connect_Time_ms accounts for approximately 90–91% of the mean Total_Auth_Time_ms across both platforms. Consequently, while a faster UAV platform improves computation stability, the tail behaviour of Total_Auth_Time_ms (p95/p99) can still be governed by wireless and transport variability. This indicates that further reductions in total authentication time are more likely to come from persistent control links, session/connection reuse, and controlled RF conditions than from micro-optimising hash/PUF computations.

From a deployment perspective, long-term FANET operation requires lifecycle management of enrolment material and pseudonyms. The protocol assumes fresh challenge usage to avoid CRP reuse and supports GS-side epoch-based CRP scheduling and re-enrolment when a pool approaches depletion. Similarly, temporary identifiers (TIDs) should be treated as one-time pseudonyms and be replenished during scheduled maintenance windows (or via a protected management channel after a secure GS session is established) to sustain untraceability and scalability over extended missions.

Finally, we explicitly delimit the scope of the formal guarantees. In this manuscript, Scyther verification targets the UAV–GS mutual authentication and explicit key confirmation phase under the Dolev–Yao symbolic model. The UAV–UAV ticket-based mechanism is analysed by construction and informal reasoning, and a full symbolic verification of the UAV–UAV phase (including concurrent sessions and long-term operations) remains future work. Availability-oriented attacks such as RF jamming and packet dropping, as well as

organisational risks such as GS single-point-of-failure, are orthogonal to the cryptographic protocol and should be addressed via system-level measures.

8. Conclusions

This paper presents a lightweight PUF-based authentication framework for large-scale FANET deployments. The proposed design combines a UAV–GS mutual authentication protocol with explicit key confirmation and an on-demand, GS-assisted UAV–UAV key establishment mechanism in which the GS is not on-path for UAV–UAV data traffic and relay/multi-hop forwarding can be supported without exposing pairwise keys to intermediate UAVs.

The UAV–GS sub-protocol was formally modelled and analysed using Scyther under the Dolev–Yao adversary model. Within the analysed bounds, Scyther reports no attacks for the secrecy and agreement properties claimed for the UAV–GS phase. The UAV–UAV phase is analysed by construction and informal reasoning; extending symbolic verification to the UAV–UAV protocol, multi-session concurrency, and long-term operation is an important direction for future work.

We further validated the implementation on UAV class-embedded platforms using a fixed laptop ground station. Over three runs, the mean end-to-end authentication latency decreases from 132.31 ms on Raspberry Pi 4B to 121.48 ms on Raspberry Pi 5, while the mean protocol-only time decreases from 12.93 ms to 10.33 ms. On the GS side, the session window remains in the single-digit-millisecond regime and improves on the Pi 5 condition. These results confirm that cryptographic computation is not the dominant contributor to end-to-end delay; instead, connection establishment and channel variability drive the overall latency distribution, suggesting that persistent connections and transport-level optimisation are key levers for further reducing authentication latency in dynamic FANET deployments.

Future work will focus on integrating error-tolerant PUF stabilisation mechanisms such as fuzzy/reverse fuzzy extractors and the BCH algorithm; expanding formal verification coverage to the UAV–UAV phase and concurrent sessions; and evaluating robustness under realistic mobility, interference/jamming conditions, and GS redundancy models.

Author Contributions: Conceptualisation, M.A.S., S.A.-R. and A.T.; methodology, M.A.S.; validation, M.A.S.; formal analysis, M.A.S.; writing—original draft preparation, M.A.S.; writing—review and editing, S.A.-R. and A.T.; supervision, S.A.-R. and A.T. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data are contained within the article.

Acknowledgments: The first author acknowledges the Republic of Turkey's Ministry of National Education for supporting his PhD studies.

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

Abbreviations

The following abbreviations are used in this manuscript:

UAV	Unmanned Aerial Vehicle
GS	Ground Station
FANET	Flying Ad Hoc Networks
PKI	Public Key Infrastructure

CRP	Challenge–Response Pair
PUF	Physically Unclonable Functions
MANET	Mobile Ad Hoc Networks
IBC	Identity-Based Cryptography
ECC	Elliptic Curve Cryptography
IoT	Internet of Things
WSN	Wireless Sensor Network
SPDL	Security Protocol Description Language

References

1. Tsao, K.Y.; Girdler, T.; Vassilakis, V. A survey of cyber security threats and solutions for UAV communications and flying ad-hoc networks. *Ad Hoc Netw.* **2022**, *133*, 102894. [CrossRef]
2. Yaacoub, J.P.; Noura, H.; Salman, O.; Chehab, A. Security analysis of drones systems: Attacks, limitations, and recommendations. *Internet Things* **2020**, *11*, 100218. [CrossRef] [PubMed]
3. Diockou, J.; Wu, D.; Lu, X.; Jing, Y.; Shabut, A.; Barwood, M. A Review of Security Frameworks in Flying Ad-hoc Networks. In Proceedings of the 2024 IEEE International Conference on e-Business Engineering (ICEBE), Shanghai, China, 11–13 October 2024; pp. 183–190. [CrossRef]
4. Aloseel, A.; Al-Rubaye, S.; Zolotas, A.; He, H.; Shaw, C. A Novel Approach for Detecting Cyberattacks in Embedded Systems Based on Anomalous Patterns of Resource Utilization-Part I. *IEEE Access* **2021**, *9*, 103204–103229. [CrossRef]
5. Rodrigues, M.; Amaro, J.; Osório, F.S.; Branco, K.R.J.L. Authentication Methods for UAV Communication. In Proceedings of the 2019 IEEE Symposium on Computers and Communications (ISCC), Barcelona, Spain, 3–29 June 2019; pp. 1210–1215. [CrossRef]
6. Pandey, G.K.; Gurjar, D.S.; Nguyen, H.; Yadav, S. Security Threats and Mitigation Techniques in UAV Communications: A Comprehensive Survey. *IEEE Access* **2022**, *10*, 112858–112897. [CrossRef]
7. Höglund, J.; Lindemer, S.; Furuhed, M.; Raza, S. PKI4IoT: Towards Public Key Infrastructure for the Internet of Things. *Comput. Secur.* **2019**, *89*, 101658 . [CrossRef]
8. Michailidis, E.T.; Vouyioukas, D. A Review on Software-Based and Hardware-Based Authentication Mechanisms for the Internet of Drones. *Drones* **2022**, *6*, 41 . [CrossRef]
9. Shamsoshoara, A.; Korenda, A.; Afghah, F.; Zeadally, S. A survey on physical unclonable function (PUF)-based security solutions for Internet of Things. *Comput. Netw.* **2020**, *183*, 107593. [CrossRef]
10. Sun, D.Z.; Gao, Y.N.; Tian, Y. On the Security of a PUF-Based Authentication and Key Exchange Protocol for IoT Devices. *Sensors* **2023**, *23*, 6559. [CrossRef]
11. Maes, R.; Verbauwhede, I. Physically Unclonable Functions: A Study on the State of the Art and Future Research Directions. In *Towards Hardware-Intrinsic Security*; Springer: Berlin, Germany, 2010; pp. 3–37. [CrossRef]
12. Halak, B.; Zwolinski, M.; Mispan, M.S. Overview of PUF-based hardware security solutions for the internet of things. In Proceedings of the 2016 IEEE 59th International Midwest Symposium on Circuits and Systems (MWSCAS), Abu Dhabi, United Arab Emirates, 16–19 October 2016; IEEE : Piscataway, NJ, USA, 2016; pp. 1–4. [CrossRef]
13. Alladi, T.; Naren; Bansal, G.; Chamola, V.; Guizani, M. SecAuthUAV: A Novel Authentication Scheme for UAV-Ground Station and UAV-UAV Communication. *IEEE Trans. Veh. Technol.* **2020**, *69*, 15068–15077. [CrossRef]
14. Roy, S.; Das, D.; Mondal, A.; Mahalat, M.H.; Sen, B.; Sikdar, B. PLAKE: PUF-Based Secure Lightweight Authentication and Key Exchange Protocol for IoT. *IEEE Internet Things J.* **2023**, *10*, 8547–8559. [CrossRef]
15. Alkatheiri, M.S.; Saleem, S.; Alqarni, M.A.; Aseeri, A.O.; Chauhdary, S.H.; Zhuang, Y. A Lightweight Authentication Scheme for a Network of Unmanned Aerial Vehicles (UAVs) by Using Physical Unclonable Functions. *Electronics* **2022**, *11*, 2921. [CrossRef]
16. Alladi, T.; Chamola, V.; Naren; Kumar, N. PARTH: A two-stage lightweight mutual authentication protocol for UAV surveillance networks. *Comput. Commun.* **2020**, *160*, 81–90. [CrossRef]
17. Tian, C.; Jiang, Q.; Li, T.; Zhang, J.; Xi, N.; Ma, J. Reliable PUF-based mutual authentication protocol for UAVs towards multi-domain environment. *Comput. Netw.* **2022**, *218*, 109421. [CrossRef]
18. Sen, M.; Alrubaye, S.; Tsourdos, A. Securing UAV Flying Ad Hoc Wireless Networks: Authentication Development for Robust Communications. *Sensors* **2025**, *25*, 1194. [CrossRef] [PubMed]
19. Bekmezci, İ.; Sahingoz, O.K.; Temel, Ş. Flying Ad-Hoc Networks (FANETs): A survey. *Ad Hoc Netw.* **2013**, *11*, 1254–1270. [CrossRef]
20. Pasandideh, F.; da Costa, J.P.J.; Kunst, R.; Islam, N.; Hardjawana, W.; Pignaton de Freitas, E. A Review of Flying Ad Hoc Networks: Key Characteristics, Applications, and Wireless Technologies. *Remote Sens.* **2022**, *14*, 4459. [CrossRef]
21. Chriki, A.; Touati, H.; Snoussi, H.; Kamoun, F. FANET: Communication, mobility models and security issues. *Comput. Netw.* **2019**, *163*, 106877. [CrossRef]

22. Liu, X.; Wang, J.; Guo, S.; Wang, H. A Survey on Cross-Layer Authentication in Wireless Communication Networks. *J. Netw. Netw. Appl.* **2024**, *4*, 21–30. [CrossRef]
23. Khan, M.F.; Yau, K.L.A.; Noor, R.M.; Imran, M.A. Routing Schemes in FANETs: A Survey. *Sensors* **2020**, *20*, 38. [CrossRef]
24. Ceviz, O.; Şen, S.; Sadioglu, P. A Survey of Security in UAVs and FANETs: Issues, Threats, Analysis of Attacks, and Solutions. *IEEE Commun. Surv. Tutor.* **2023**, *27*, 3227–3265. [CrossRef]
25. Gupta, A.; Barthwal, A.; Vardhan, H.; Kakria, S.; Kumar, S.; Parihar, A. Evolutionary study of distributed authentication protocols and its integration to UAV-assisted FANET. *Multimed. Tools Appl.* **2023**, *82*, 1–20. [CrossRef]
26. Khan, A.S.; Sattar, M.A.; Nisar, K.; Ibrahim, A.A.A.; Annuar, N.B.; Abdullah, J.B.; Karim Memon, S. A Survey on 6G Enabled Light Weight Authentication Protocol for UAVs, Security, Open Research Issues and Future Directions. *Appl. Sci.* **2023**, *13*, 277. [CrossRef]
27. Chen, C.; Luo, J.; Xu, Z.; Xiong, R.; Shen, D.; Yin, Z. Enabling large-scale low-power LoRa data transmission via multiple mobile LoRa gateways. *Comput. Netw.* **2023**, *237*, 110083. [CrossRef]
28. Lee, S.W.; Safkhani, M.; Le, Q.; Ahmed, O.; Hosseinzadeh, M.; Rahmani, A.; Bagheri, N. Designing secure PUF-based authentication protocols for constrained environments. *Sci. Rep.* **2023**, *13*, 21702. [CrossRef]
29. Yilmaz, Y.; Gunn, S.R.; Halak, B. Lightweight PUF-Based Authentication Protocol for IoT Devices. In Proceedings of the 2018 IEEE 3rd International Verification and Security Workshop (IVSW), Costa Brava, Spain, 2–4 July 2018; pp. 38–43. [CrossRef]
30. Sajadi, A.; Shabani, A.; Alizadeh, B. DC-PUF: Machine learning-resistant PUF-based authentication protocol using dependency chain for resource-constraint IoT devices. *J. Netw. Comput. Appl.* **2023**, *217*, 103693. [CrossRef]
31. Zhang, Z.; Hsu, C.; Au, M.; Harn, L.; Cui, J.; Xia, Z.; Zhao, Z. PRLAP-IoD: A PUF-based Robust and Lightweight Authentication Protocol for Internet of Drones. *Comput. Netw.* **2023**, *238*, 110118. [CrossRef]
32. Kwon, D.; Park, Y. Security Analysis and Mitigation of "PRLAP-IoD: A PUF-Based Robust and Lightweight Authentication Protocol for Internet of Drones". In Proceedings of the 2025 Sixteenth International Conference on Ubiquitous and Future Networks (ICUFN), Lisbon, Portugal, 8–11 July 2025; pp. 756–758. [CrossRef]
33. Choi, J.; Son, S.; Kwon, D.; Park, Y. A PUF-Based Secure Authentication and Key Agreement Scheme for the Internet of Drones. *Sensors* **2025**, *25*, 982. [CrossRef]
34. Sharma, M.; Narwal, B.; Anand, R.; Mohapatra, A.K.; Yadav, R. PSECAS: A physical unclonable function based secure authentication scheme for Internet of Drones. *Comput. Electr. Eng.* **2023**, *108*, 108662. [CrossRef]
35. Pratap, B.; Singh, A.; Mehra, P.S. REHAS: Robust and Efficient Hyperelliptic Curve-Based Authentication Scheme for Internet of Drones. *Concurr. Comput. Pract. Exp.* **2025**, *37*, e8333. [CrossRef]
36. Dodis, Y.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. In *Proceedings of the Advances in Cryptology—EUROCRYPT 2004*; Cachin, C., Camenisch, J.L., Eds.; Springer: Berlin/Heidelberg, Germany, 2004; pp. 523–540.
37. Yang, J.; Chen, S.; Cao, Y. A PUF-Based Key Storage Scheme Using Fuzzy Vault. *Sensors* **2023**, *23*, 3476. [CrossRef]
38. Herrewege, A.V.; Katzenbeisser, S.; Maes, R.; Peeters, R.; Sadeghi, A.R.; Verbauwhede, I.M.R.; Wachsmann, C. Reverse Fuzzy Extractors: Enabling Lightweight Mutual Authentication for PUF-Enabled RFIDs. In Proceedings of the Financial Cryptography, Kralendijk, The Netherlands, 27 February–2 March 2012.
39. Cui, Y.; Li, J.; Gu, C.; Wang, C.; Liu, W. A Multi-Mode Configurable Physical Unclonable Function Based on RRAM With Adjustable Programmable Voltage. *IEEE Trans. Nanotechnol.* **2025**, *24*, 166–177. [CrossRef]
40. Zhao, X.; Zhao, Q.; Liu, Y.; Zhang, F. An Ultracompact Switching-Voltage-Based Fully Reconfigurable RRAM PUF With Low Native Instability. *IEEE Trans. Electron Devices* **2020**, *67*, 3010–3013. [CrossRef]
41. Rührmair, U.; Sölter, J. PUF modeling attacks: An introduction and overview. In Proceedings of the 2014 Design, Automation & Test in Europe Conference & Exhibition (DATE), Dresden, Germany, 24–28 March 2014; pp. 1–6. [CrossRef]
42. Gassend, B.; Clarke, D.; van Dijk, M.; Devadas, S. Silicon physical random functions. In Proceedings of the 9th ACM Conference on Computer and Communications Security, CCS '02, New York, NY, USA, 18–22 November 2002; pp. 148–160. [CrossRef]
43. Blanchet, B. Automatic Verification of Security Protocols in the Symbolic Model: The Verifier ProVerif. In *Foundations of Security Analysis and Design VII: FOSAD 2012/2013 Tutorial Lectures*; Springer International Publishing: Cham, Switzerland, 2014; pp. 54–87. [CrossRef]
44. Cremers, C.J.F. The Scyther Tool: Verification, Falsification, and Analysis of Security Protocols. In Proceedings of the Computer Aided Verification, Princeton, NJ, USA, 7–14 July 2008; Springer: Berlin/Heidelberg, Germany, 2008; pp. 414–418.
45. Dodis, Y.; Ostrovsky, R.; Reyzin, L.; Smith, A. Fuzzy Extractors: How to Generate Strong Keys from Biometrics and Other Noisy Data. *SIAM J. Comput.* **2008**, *38*, 97–139. [CrossRef]
46. Maes, R.; Tuyls, P.; Verbauwhede, I. Low-Overhead Implementation of a Soft Decision Helper Data Algorithm for SRAM PUFs. In *Proceedings of the Cryptographic Hardware and Embedded Systems—CHES 2009*; Clavier, C., Gaj, K., Eds.; Springer: Berlin/Heidelberg, Germany, 2009; pp. 332–347.

47. Rührmair, U.; Sehnke, F.; Sölter, J.; Dror, G.; Devadas, S.; Schmidhuber, J. Modeling attacks on physical unclonable functions. In Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS '10, Chicago IL, USA, 4–8 October 2010; ACM: New York, NY, USA, 2010; pp. 237–249. [CrossRef]
48. Rührmair, U.; Sölter, J.; Sehnke, F.; Xu, X.; Mahmoud, A.; Stoyanova, V.; Dror, G.; Schmidhuber, J.; Burleson, W.; Devadas, S. PUF Modeling Attacks on Simulated and Silicon Data. *IEEE Trans. Inf. Forensics Secur.* **2013**, *8*, 1876–1891. [CrossRef]
49. Tanveer, M.; Aldosary, A.; Kumar, N.; Aldossari, S.A. SEAF-IoD: Secure and efficient user authentication framework for the Internet of Drones. *Comput. Netw.* **2024**, *247*, 110449. [CrossRef]

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.

DSPACE

https://dspace.org/

Design of secure communication networks for UAV platform empowered by lightweight authentication protocols

Sen, Muhammet A.; Al-Rubaye, Saba; Tsourdos, Antonios

2026-02-02
MDPI
https://doi.org/10.3390/electronics15040785
Sen MA, Al-Rubaye S, Tsourdos A. (2026) Design of secure communication networks for UAV platform empowered by lightweight authentication protocols. Electronics, Volume 15, Issue 4, February 2026, Article number 785

Downloaded from a DSpace repository. https://dspace.org/