Selective Authenticated Pilot Location Disclosure for Remote ID-enabled Drones

Pietro Tedeschi
pietro.tedeschi@tii.ae
Technology Innovation Institute

Siva Ganesh Ganti
s.g.t.ganti@student.tue.nl
Eindhoven University of Technology

Savio Sciancalepore
s.sciancalepore@tue.nl
Eindhoven University of Technology

ABSTRACT

Remote Identification (RID) regulations recently promulgated worldwide are forcing commercial drones to broadcast wirelessly the location of the pilot in plaintext. However, in many real-world use cases, the plaintext availability of such information leads to privacy issues, allowing the extraction of sensitive information about the pilot and confidential details about the drone's business. To address this issue, this paper proposes SNELL, a RID-compliant solution for selective authenticated pilot location disclosure. Using SNELL, a drone can disclose RID messages providing encrypted information about the pilot's location. At the same time, thanks to the smart integration of Ciphertext-Policy Attribute-Based Encryption (CP-ABE) techniques, the data about the pilot location can be decrypted only by receivers with a set of attributes satisfying an access control policy chosen by the drone at run-time. Thanks to an extensive experimental assessment carried out on a real medium-end drone (Lumenier QAV-R) and a constrained chip (ESP32), we demonstrate that SNELL can fulfil all the requirements imposed by RID and relevant standardization authorities in terms of pilot location update time and message size while also requiring negligible energy toll on RID-compliant drones.

KEYWORDS

Unmanned Aerial Vehicles, Security, Privacy, Location Privacy.

1 INTRODUCTION

The recent explosion of the number of drones used for commercial and entertainment purposes (9.64 million drones forecasted by 2029, according to [55]) led to the promulgation of several operational rules regulating drones integration into local airspaces [57], [2]. The most famous of such regulations is Remote ID (RID), promulgated by the US-based Federal Aviation Administration (FAA), forcing nearly all drones to regularly broadcast plaintext messages on the wireless channel, reporting their unique identity, location, speed, and pilot location, among the others. Initiatives similar to RID are following, applying to other airspaces such as EU, China, and India [6].

At the same time, RID also introduces security and privacy concerns. In particular, the availability of drones' identity, location, and pilot information as plaintext on the channel might threaten drone applications, exposing the devices to easy tracking and capture, as well as pilot privacy issues, although this is not the intended scope of the regulation. Scientific contributions such as the recent one in [51] confirm that it is possible to decode RID messages easily. Such findings increase the concerns of professionals and amateurs working in the field, as acknowledged by several sources [21].

In this context, recent works investigated privacy issues connected with drone identity disclosure [58], drone location disclosure [14], [53], and drone location privacy [59], [43]. However, RID messages also disclose the plaintext location of the pilot of the drone. In many real-life use cases, such as commercial drone operations and drone flights on country borders, the availability of such information to attackers may lead to new threats to the pilot's safety, e.g., the pilot might be localized and physically threatened, and private details about its location can be exposed. However, securing pilot information in RID messages is not straightforward. Besides intended message receivers, selected entities, e.g., aviation regulatory bodies, should always access pilot information without additional interactions with the drones, due to the broadcast nature of RID and the purely passive nature of many receivers. Drone messages should also be authentic to avoid impersonation attacks, requiring digital signatures. At the same time, to comply with regulations, drones should update the location of the pilot reported in RID messages at least once every 3 seconds [1], not taking more than one WiFi frame to convey such information and not draining too much energy from the available battery. A simple solution would consist of using hardwired keys, but, for many use cases (see Sec. 3.3), this would not provide enough flexibility to drones' operating conditions. Also, as discussed in Sec. 7, other solutions, such as attribute signcryption and techniques for drone anonymity, cannot satisfy all the discussed requirements at the same time. To the best of our knowledge, no solution currently exists that can fulfil all such privacy and system requirements (see Sec. 7 for more details).

Contribution. In this paper, we propose SNELL (an acronym for Selective autheNticatEd piLot Location disclosure), a novel protocol achieving selective authenticated disclosure of pilots' location information through standard-compliant RID messages. In a nutshell, SNELL allows drones to disclose the pilots' actual location only to selected receivers, i.e., the ones owing a set of attributes satisfying a specific access control policy chosen at run-time by the drone. Moreover, as required by relevant requirements on secure RID deployments [50], SNELL messages are also authenticated to avoid impersonation and spoofing attacks. To address all our requirements in a standard-compliant fashion, SNELL integrates smartly several building blocks with RID, such as the Ciphertext-Policy Attribute-Based Encryption (CP-ABE) cryptography technique (for location selective disclosure) and *Schnorr*-based signatures (for efficiently signing messages), and it manages to keep the overhead of such cryptography techniques manageable on drones, both in terms of message size and computational complexity. To show the

Proceedings on Privacy Enhancing Technologies 2024(3), 523–539

https://doi.org/10.56553/popets-2024-0091

viability of SNELL on actual drones, we deployed two proofs-of-concept. We first tested the performance of our solution on a drone Lumenier QAV-R, using a Raspberry Pi 4 as the mission computer. Our experiments show that SNELL allows using complex access control policies, including up to 22 attributes, while not exceeding the maximum pilot location update time of 3 s and a single WiFi frame, as required by RID. Also, the energy toll of SNELL stays very low, not affecting its lifetime and usability. Moreover, we deploy SNELL on the constrained device ESP32, used as part of many low-end drones [22]. We devise three implementation strategies, i.e., (*Fully Precomputed*, *Partially Precomputed* and *Parallel Computed*), and we show that we can always achieve selective pilot location disclosure within the time bounds defined by RID regulations (3 s) while posing a minimal energy toll on the device (0.0033% of the device's battery).

We also release the code of SNELL as open-source at [56] and [27] to foster the replicability of our results and encourage the deployment and extension of SNELL. Overall, our work demonstrates that we can achieve pilot location privacy and message authentication together in a single standalone solution, even on constrained devices, by considering all fundamental privacy and system requirements considered by the relevant RID-related standardization communities.

Roadmap. This paper is organized as follows. Sec. 2 introduces preliminaries, Sec. 3 outlines the assumed scenario and adversarial model, Sec. 4 provides the details of SNELL, Sec. 5 discusses the security considerations, Sec. 6 provides the extensive experimental evaluation of SNELL, Sec. 7 reviews related work and qualitatively compares SNELL to the literature and, finally, Sec. 8 concludes the paper and outlines future work.

2 PRELIMINARIES

2.1 RemoteID

The RID regulation was emitted first in 2020 by the US-based FAA and will be mandatory for all drones by September 2023. In brief, the RID regulation requires drones to emit wireless broadcast messages with a minimum rate of 1 per second, reporting information about their unique identity, location, speed, pilot location, and emergency status. The regulation applies to almost any drone independently of its weight, with very few exceptions [24]. Moreover, broadcast RID messages should be emitted via WiFi or Bluetooth Low Energy. When the drones do not feature wireless communication capabilities by design, they should be equipped with additional modules to support the broadcast of RID messages. Concerning security services connected to wireless messages, RID does not mandate message encryption or authentication.

Although the RID regulation strictly applies to the US airspace, similar initiatives are appearing worldwide. For instance, the EASA emitted a RID-inspired rule in the EU area, effective from Jan. 1, 2024 [6]. All such regulations describe the requirements for RID but do not specify how to implement the specification. To this aim, several initiatives arose worldwide involving companies and stakeholders. In the US, the ASTM provided more precise requirements to implement and deploy RID. Although the RID message generation rate is 1 second, static information reported in RID messages, such as the pilot location, should be updated at least once every 3 seconds [1]. At the same time, the absence of mandated security services led to the formation of a dedicated Working Group (WG) at the Internet Engineering Task Force (IETF), namely, Drone Remote Identification Protocol (drip) [50]. According to the charter [17], *DRIP's goal is to specify how RID can be made trustworthy and available in both Internet and local-only connected scenarios, especially in emergencies.* The *drip* WG published several documents considering various aspects of the RID ecosystem. This manuscript considers the baseline architecture of *drip* available at [50] to develop a fully drip-compliant solution.

2.2 CP-ABE

CP-ABE encryption schemes allow a data owner to encrypt a plaintext using an access control policy defined over a set of attributes. Receivers, i.e., data consumers, can decrypt the ciphertext only if they own a set of attributes satisfying the access control policy specified by the data owner [7]. Such techniques are commonly used in scenarios requiring restricted data access, such as for fine-grained access control in the Cloud [64].

In this paper, we build on top of the recent implementation of CP-ABE provided by the authors in [47], namely, *FABEO*, briefly outlined below. Let 1^λ a security parameter indicating the security level in bits, $(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$ asymmetric bilinear groups of prime order p, $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$ the bilinear pairing operation with generators g_1, g_2 for $\mathbb{G}_1, \mathbb{G}_2$, $\mathcal{U}$ the universe of attributes, and $\mathcal{H} : |\mathcal{U}|+1 \rightarrow \mathbb{G}_1$ a collision-resistant hashing function. Moreover, for a prime p, let $\mathbb{Z}_p$ the set $[0, p-1]$, where addition and multiplication are computed modulo p. The FABEO implementation of CP-ABE includes the following algorithms.

- CP_FABEO.Setup(1^λ). This algorithm outputs the cryptographic parameters $\mathcal{G} := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g1, g2)$, the master public key $\mathsf{mpk} := (\mathcal{G}, \mathcal{H}, e(g_1, g_2)^\alpha)$ and the master secret key $\mathsf{msk} : \alpha$, being $\alpha \leftarrow \mathbb{Z}_p$ a nonce generated uniformly and independently from $\mathbb{Z}_p$.
- CP_FABEO.KeyGen$(\mathsf{mpk}, \mathsf{msk}, \mathcal{S} \subseteq \mathcal{U})$. Using a sub-set $\mathcal{S}$ of $\mathcal{U}$ and a nonce $r \leftarrow \mathbb{Z}_p$, for an attribute $u \in \mathcal{S}$, this algorithm uses Eq.1 to output the user secret key $sk_u = (sk_1, sk_{2,u\in\mathcal{S}}, sk_3)$.

$$\begin{cases} sk_1 & = g_1^\alpha \cdot \mathcal{H}(|\mathcal{U}|+1)^r, \\ sk_{2,u} & = \mathcal{H}(u)^r, \\ sk_3 & = g_2^r. \end{cases} \tag{1}$$

- CP_FABEO.Encrypt$(\mathsf{mpk}, (\mathbf{M}, \pi))$. Let n_1 and n_2 the number of rows and columns of the Monotone Span Programs (MSP) matrix $\mathbf{M} \in \mathbf{Z}_p^{n_1 \times n_2}$, $\pi : [n_1] \rightarrow \mathcal{U}$ a function mapping each row of $\mathbf{M}$ to an attribute, i.e., the access control policy, τ the maximum usages of an attribute in $\mathbf{M}$, and $s_1 \leftarrow \mathbf{Z}_p$, $\mathbf{v} \leftarrow \mathbf{Z}_p^{n_2-1}$, $\mathbf{s'} \leftarrow \mathbf{Z}_p^\tau$. Denoting $\rho(i) := |z|\pi(z) = \pi(i), z \leq i|$ as the $\rho(i)$-th occurrence of the attribute $\pi(i)$, for $j \in [\tau]$, $i \in [n_1]$, this phase outputs the ciphertext $ct(ct_1, ct_{2,j}, ct_{3,i})$ and

the encapsulated key d, as per Eq.2.

$$\begin{cases} ct_1 & = g_2^{s_1}, \\ ct_{2,j} & = g_2^{s'[j]}, \\ ct_{3,i} & = \mathcal{H}(|\mathcal{U}|+1)^{\mathbf{M}_i(s_1||\mathbf{v})^\top} \cdot \mathcal{H}(\pi(i))^{s'[\rho(i)]}, \\ d & = e(g_1,g_2)^{\alpha s_1} \end{cases} \quad (2)$$

- CP_FABEO.Decrypt(mpk, $(\mathbf{M},\pi)$, $\mathcal{S}$, ct, sk). If $\mathcal{S}$ fulfils the policy $(\mathbf{M},\pi)$ and sk is the secret key for $\mathcal{S}$, there are constant values $\{\gamma_i\}_{i\in I}$ s.t. $\sum \gamma_i \mathbf{M}_i = (1,0,\ldots,0,0)_{i\in I}$. Then, this phase allows to recover the encapsulated key d as per Eq. 3.

$$d := e(\mathsf{sk}_1, \mathsf{ct}_1) \cdot \frac{\prod_{j\in[\tau]} e(\prod_{i\in[I],\rho(i)=j}(\mathsf{sk}_{2,\pi(\mathrm{i})})^{\gamma_i}), \mathsf{ct}_{2,\mathrm{j}})}{e(\prod_{i\in[I]}(\mathsf{ct}_{3,\mathrm{i}})^{\gamma_i}, \mathsf{sk}_3)} \quad (3)$$

Finally, note that FABEO adopts Type-3 pairing operations. Interested readers can find more details in [26].

2.3 Schnorr Signatures on Elliptic Curves

The Schnorr algorithm, based on the standard ISO/IEC:14888-3 [25], is a signature generation and verification algorithm well-known for generating very small signatures [54]. Assume a sender A and a receiver B, equipped with their respective Elliptic Curve (EC) public key-pair (x_A, x_AG) and (x_B, x_BG), being x_i the private key and x_iG the public key, with G generator point of the curve. We define the EC domain parameters $\mathcal{G} = (p, a, b, G, n, h)$, where p s the prime number, a and b are the parameters of the elliptic curve (expressed in the form $y^2 = x^3 + ax + b$), n is the order of the group and h is the co-factor. Assume a generic hashing function H and a plaintext m. The Schnorr algorithm defines two algorithms.

Schnorr.Sign($\mathsf{m}, \mathcal{G}, \mathsf{x_A}$). From the message m, the domain parameters $\mathcal{G}$ and the private key x_A, this algorithm outputs the signature (r, s), as a result of the following operations:

- Generate a random value $k \in_{\mathbb{R}} [1, n-1]$;
- Compute $Q = kG$, where $Q = (Q_x, Q_y)$;
- Compute $r = H(Q_x||Q_y||m)$, with $r \neq 0 \mod n$;
- Compute $s = (k + r \cdot x_A) \mod n$, with $s \neq 0$;
- Output the signature (r, s).

Schnorr.Verify($\mathsf{m}, \mathcal{G}, \mathsf{r}, \mathsf{s}, \mathsf{x_A G}$). From the signed message m, the domain parameters $\mathcal{G}$, the message signature (r, s) and the public key of the sender x_AG, this algorithm output the decision $d = [0, 1]$, as a result of the following operations:

- Compute $Q = sG - rx_AG$. If $Q = 0$, $d = 0$;
- Compute $v = H(Q_x||Q_y||m)$. If $v == r$, $d = 1$; otherwise, $d = 0$;
- If $d = 1$, the signature is valid; otherwise, the signature is invalid.

The generation of tiny signatures made the Schnorr algorithm very popular for applications in constrained domains [32]. However, to the best of our knowledge, it has never been adopted for broadcast scenarios, and even not in the drone ecosystem.

3 SCENARIO AND REQUIREMENTS

3.1 Scenario

We depict the scenario assumed in this work in Fig. 1. We take into account three entities: (i) the drone, (ii) the Unmanned Service Supplier (USS), and (iii) the observers. We consider a generic drone, namely, d_n, flying in an area to achieve the intended mission. d_n might be either remotely piloted, by either a user or a Ground Control Station (GCS), or (semi)-autonomous, performing a series of autonomous tasks. For convenience, independently from the remote control, we will refer to the GCS as the entity responsible for the drone's movements. In line with the equipment available on most commercial drones, we assume d_n features a Global Navigation Satellite System (GNSS) receiver, allowing it to obtain its real-time location. We consider the situation at drone deployment time, when the drone already features a stable GNSS connection, i.e., the current almanac, initial position, Ephemeris, and time information of the GPS are all set. Also, d_n is equipped with a Radio Frequency (RF) communication module, e.g., WiFi. Although drones can also use long-range communication technologies, we consider WiFi to keep compliance with the *Broadcast* RID rule. Thus, once every second, d_n has to emit wireless messages containing (at least) its identifier ID_n, the location at time t, namely, $lat_n(t)$, $lon_n(t)$, $alt_n(t)$, its speed $v_x(t)$, $v_y(t)$, $v_z(t)$, pilot location information $O_n(t)$, and an emergency status $ec(t)$. We assume the drone communicates securely with the GCS to receive the pilot's location. In the case of fully autonomous drones, in compliance with RID, pilot information should be pre-loaded into the drone. Based on the available maximum transmission power and environmental conditions, RID messages can be correctly received and decoded by any entity near the drone (theoretically) up to many kilometres away from the actual drone location (the official resources at [19] indicate a range of 3 km for Bluetooth-enabled RID modules, and WiFi is definitely higher). For compliance with the commercial products available in the market [20], we consider the use of WiFi as enabling communication technology for RID.

Figure 1: Scenario assumed in this work.

Our scenario also includes a Trusted Authority (TA), namely, the USS. As per its definition by the IETF WG drip, the USS is responsible for real-time traffic monitoring and planning, data archiving, and enforcing airspace and violation control [50]. Thus, whenever the USS receives reports about drones' misbehaviour, it can sanction the drones and their pilots. Also, the USS hosts two databases of drones' information. The *Private Information Registry (PvIR)* contains private information of drones active in the area for which the USS is responsible; it is stored locally and never shared with any parties. The *Public Information Registry (PbIR)* is publicly accessible

by all (authorized) entities and contains public information about locally-active drones. Note that any entities may connect to the USS before drone deployment, in line with the requirements of *drip* mentioned in Sec. 2.1, while no access to the USS is required during drone operations.

In line with the reference architecture proposed by *drip* at [50], we take into account the presence of multiple *observers* in the same area where d_n is operating. Observers are passive RF devices, equipped with (at least) a receiving antenna tuned on the same communication channel used by d_n to emit RID messages. They can detect and decode any RID messages, accessing all the information. Although *drip* distinguishes between Generic Public Observers (GPOs) and Public Safety Observers (PSOs), in this manuscript, we will refer to generic *observers*. To verify RID messages, observers can use a locally-stored copy of the PbIR, downloaded periodically, e.g., before the occurrence of a situation when the observer expects not to feature a persistent Internet connection. As observers might not feature persistent Internet connection, observers should be able to verify RID messages directly, without requiring additional interaction with the USS or other entities. For readers' convenience, we report the notation used in the paper in Tab. 2 (Annex).

3.2 Adversary and Threat Models

The adversary assumed in this work, namely, ϵ, features both passive and active capabilities. On the one hand, ϵ is equipped with one or more receiving antennas, tuned on the same channel drones use to broadcast RID messages. Thus, as a regular observer, it can receive and decode all RID messages. On the other hand, ϵ can also inject messages on the communication channel through dedicated TX antennas, either by replaying previously recorded or by creating rogue messages. Using such tools, ϵ might: (i) obtain information about the location of the pilot of the drones active in the area, and (ii) impersonate such drones in a way to let them appear as performing disallowed operations, such as invasion of no-fly areas. ϵ can also carry out offline data analysis to infer additional private information about the pilot and the drone deployment, such as retrieving the home/work location of the pilot and other related sensitive details. In Sec. 3.3, we provide some real use cases where such information might lead to privacy issues. We highlight that impersonation/spoofing attacks do not directly affect the operator's privacy. However, integrating authentication techniques (signatures) within RID messages, necessary to avoid such attacks, affects the message size and, thus, the space available in the packet for encrypted pilot information. Therefore, operator privacy and authentication must be tackled together in a single approach, demonstrating that we can address all major concerns through a unique solution.

3.3 Use cases

We introduce two real-world use cases motivating the need to protect selectively the pilot information available in RID messages.

Use Case 1. Consider a delivery service (e.g., Amazon, Walmart, etc.) using drones to carry goods to customers, using a GCS to issue commands and receive telemetry data. The plaintext broadcast of the location of the GCS would let anyone know where the GCS is. This knowledge might enable attackers to physically target the GCS to gain full control over the drone. To overcome this problem, the drone might encrypt the pilot data included in RID messages through a hardwired key. However, encrypting such data with a symmetric key would require a large set of entities (data consumers) to know such a key, increasing the chances of leakage significantly (whether such leakage is intentional or not). At the same time, every new receiver authorized to access such data would require such a key, making the system hardly scalable.
To handle this use case, the USS might setup an attribute `ROLE` and assign to each data consumer a value for this attribute based on the role of the entity deploying the receiver, e.g., `COMPANY`, `USER`, `GOVERNMENT`. Then, the drone might encrypt the pilot data, specifying a policy such that only a specific set of companies can decrypt the data, e.g., `ROLE = COMPANY AND GOVERNMENT`. Thus, generic untrusted data consumers cannot know where the ground station is, reducing the attack surface.

Use Case 2. Consider a drone flying across the borders between several countries. Such areas are notoriously sensitive, as each country would like to maintain full control over the disclosed information. Thus, the drone would like to allow only the receivers deployed in its home country to know the exact location of the pilot, while the receivers in other countries should not learn such information. Real-life scenarios matching this use case can be found, e.g., in the EU area, where several businesses are planning the deployment of cross-borders drone deliveries [46], [15], [31].
In such a scenario, the drone might use a hardwired key. However, using symmetric encryption requires the administrator of the drone to define a set of authorized receivers before the mission and to deploy the same key on each of the receivers. At the same time, devices possibly authorized to access such data but not known to be in the place at the time of the setup (namely, opportunistic receivers) cannot access such messages.
To handle this case, the USS might set an attribute named `COUNTRY`, assigning to each country different values, e.g., `COUNTRY_1` $\wedge$ `COUNTRY_2`. The drone might encrypt the pilot location using a policy `COUNTRY = COUNTRY_1`, allowing only the receivers in `COUNTRY_1` to decrypt it. However, deploying such a system into a drone and running it at run-time is challenging. Indeed, the drones should apply Attribute-Based Encryption (ABE) operations at run-time while fulfilling all the requirements of the RID regulation in terms of pilot location update rate and size of the messages while ensuring sufficient policy expressiveness.

3.4 Requirements

From the discussion in the previous subsections, we can extract several security and system requirements that a suitable solution should have. Overall, the use cases discussed above motivate us to provide *Selective Access to Only Pilot Data* (**R1**), to guarantee that only authorized data consumers can access pilot location data included in RID messages while remaining information in the RID messages stays available to all parties. However, being RID a broadcast technology, we also need a solution that allows for opportunistic access to pilot data by authorized parties, as it is not possible to carry out any explicit key establishment occurring at run-time

(*Opportunistic Pilot Data Encryption*, **R2**). Moreover, observers deployed in some regions might not feature persistent Internet access at deployment time. Thus, we need to account for *Offline Pilot Data Decryption*, (**R3**). Furthermore, unauthorized observers should not infer any information about the pilot from the protected RID messages, not even if the pilot of a given drone is changing (*Pilot Data Unlinkability*, **R4**). At the same time, we also have to consider requirements imposed by the RID regulation. Indeed, as per the ASTM regulation in [1], the pilot location reported in RID messages should be updated at least once every 3 s (*Short Pilot Location Update Time*, **R5**), through a single WiFi frame at the MAC-layer (*Single WiFi Frame*, **R6**). Also, messages emitted by the drone should be authentic, to avoid impersonation attacks (*Messages Authenticity*, **R7**). Finally, as some receivers may be fully passive, we also need a solution that allows authorized receivers to access the pilot's location without additional network messages (*Zero-Touch Pilot Location Disclosure*, **R8**). As discussed in Sec. 7, none of the solutions currently available addresses all the mentioned requirements, at the same time, motivating the design of our solution. Note that the requirements **R1**, **R2**, **R3**, **R7**, and **R8** map to the normative requirements *PRIV-1*, *PRIV-2*, *PRIV-3*, *GEN-2*, and *GEN-3* defined by IETF drip in RFC 9153 [49]. Also, the requirements **R5** and **R6** can be extracted directly from the standard RID specification.

4 SNELL

4.1 Rationale

SNELL includes two phases: the Registration Phase (Sec. 4.2) and the Deployment Phase (Sec. 4.3). During the registration phase, drones and observers register with the local USS and receive the private and public attribute set, defining which attributes they can use for encrypting messages and which messages they are allowed to decrypt, respectively. During the deployment phase, once every 1 sec, the drone transmits a RID message containing various information, such as the drone's identity, location, speed, emergency status, and pilot location. To protect the information about the pilot location, our proposed SNELL protocol extracts a nonce and encrypts it using CP-ABE. In particular, SNELL uses the FABEO variant by Riepel et al. [47] due to its reduced complexity and bandwidth overhead, which makes it ideal for applications on constrained drones. SNELL also uses a Key Derivation Function (KDF) to cast the resulting ciphertext into a 128-bit key, used as a key for AES to encrypt the pilot location into a one-time attribute-dependent ciphertext. To finally provide authentication of the whole RID message, SNELL uses the Schnorr-type signatures on the entire RID message, including the encrypted pilot location (via AES), the encrypted nonce (using CP-ABE), and all the other fields of the RID message. Every observer receiving the message can always authenticate the received message by verifying the Schnorr signatures attached to it. When it would like to know the pilot location, it uses its attribute set to decrypt the encrypted nonce in the message. The decryption succeds only if the observer possesses an attribute set compatible with the used policy. If this operation is successful, the observer can use the KDF to cast the nonce into the key to be used to decrypt the encrypted pilot location in the RID message.

4.2 Registration Phase

The aim of the *Registration Phase* is to register all the actors with the USS, to equip them with the cryptographic materials necessary to run SNELL. This phase is executed before the deployment through a regular Internet connection, e.g., secured via Transport Layer Security (TLS), and may be repeated when updating cryptography parameters. Figure 2 depicts the sequence diagram of the *Registration Phase* for both the drone d_n and the observers.

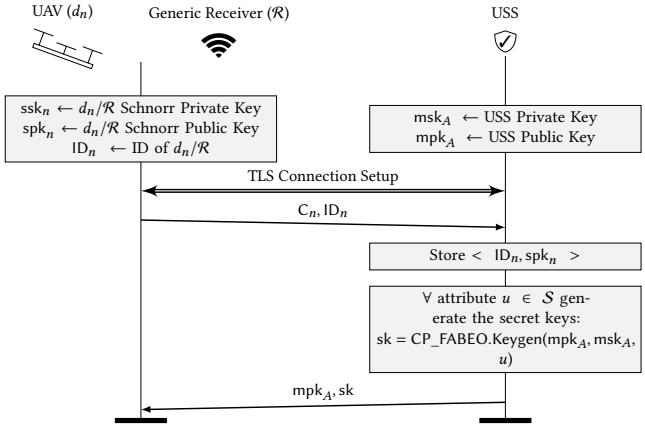

Figure 2: Sequence Diagram of the *Registration Phase* of SNELL.

This phase requires the following operations.

- Assume either the drone d_n or a generic observer $\mathcal{R}$, equipped with a Schnorr private-public key pair $(\text{ssk}_n, \text{spk}_n)$ and a public-key certificate C_n signed by a Certification Authority (CA). After the TLS connection establishment, d_n (or a generic receiver $\mathcal{R}$) provides to the USS the identity ID_n and the public key certificate C_n, including the public key spk_n.
- Assume that the USS owns a private-public key pair, namely, $(\text{msk}_A, \text{mpk}_A)$, and a public-key certificate C_A, signed by a CA. At the reception of the information from $d_n/\mathcal{R}$, the USS stores in the PvIR the entry for the UAV d_n or $\mathcal{R}$, i.e., the tuple $< \text{ID}_n, \text{spk}_n >$. Then, it provides back the public key mpk_A and the set of public parameters of the cryptographic system. According to the scheme described in Sec. 2.2, the public key mpk_A is computed as in Eq. 4:

$$\text{mpk}_A := (p, \mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T, e, g_1, g_2, \mathcal{H}, e(g_1, g_2)^\alpha), \tag{4}$$

  while the master secret key is $msk := \alpha$. Moreover, the USS generates and sends back the correspondent CP-ABE secret keys for each attribute, i.e., u, as per Eq. 5.

$$\text{sk} = \text{CP\_FABEO.Keygen}(\text{mpk}_A, \text{msk}_A, u). \tag{5}$$

- At message reception, the entity (d_n or $\mathcal{R}$) locally stores the master public key mpk_A, the set of secret keys, and the cryptographic parameters for the following *Deployment Phase*.

After completing this phase, d_n and $\mathcal{R}$ do not need any further communication with the USS.

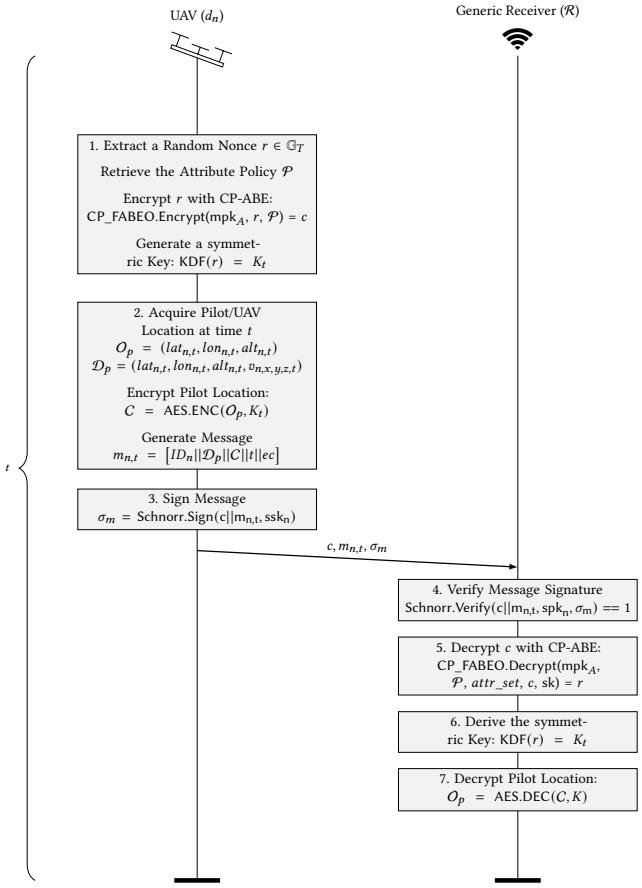

Figure 3: Sequence Diagram of the *Deployment Phase* of SNELL.

4.3 Deployment Phase

During the *Deployment Phase* of SNELL, the drone d_n generates and broadcasts *RID*-compliant messages, enabling only authorized observers to identify the pilot location. During this phase, the drone and possibly the observer(s) are offline, i.e., not connected to the Internet. Figure 3 shows the sequence diagram of the required operations for each new RID message while we report the details below.

(1) d_n first generates a secret nonce $r \in \mathbb{G}_T$. Then, it retrieves the access policy $\mathcal{P}$ and the master public key mpk. Using these parameters, d_n uses the CP-ABE encryption algorithm CP_FABEO.Encrypt to generate the ciphertext c, as per Eq. 6.

$$c = \text{CP\_FABEO.Encrypt}(\text{mpk}_A, r, \mathcal{P}). \tag{6}$$

From the nonce r, d_n also computes an ephemeral symmetric key K_t, using a KDF as in Eq. 7.

$$K_t = \text{KDF}(r). \tag{7}$$

(2) Assume that at the time t, d_n acquires via GNSS the current location, expressed as $lat_{n,t}, lon_{n,t}, alt_{n,t}$. It also computes its instantaneous speed, as $v_{x,n,t}, v_{y,n,t}, v_{z,n,t}$. We define $\mathcal{D}_p = \left[lat_{n,t}, lon_{n,t}, alt_{n,t}, v_{x,n,t}, v_{y,n,t}, v_{z,n,t})\right]$. d_n obtains the pilot location $O_p = \left[lat_{n,t}, lon_{n,t}, alt_{n,t}\right]$.
d_n encrypts the *pilot location* O_p through the Advanced Encryption Standard (AES) symmetric encryption algorithm, using the ephemeral key K_t, as per Eq. 8:

$$C = \text{AES.ENC}(O_p, K_t). \tag{8}$$

Note that the protocol does not use CP-ABE to encrypt directly the pilot location. Conversely, it uses CP-ABE to derive a key used to encrypt information via a lightweight symmetric encryption scheme. Such a construction is remarkably lightweight, as CP-ABE operations might be precomputed. We exploit such an advantage for our implementation on the constrained drone (see Sec. 6). We denote the RID message $m_{n,t}$ as in Eq. 9.

$$m_{n,t} = \left[ID_n||\mathcal{D}_p||C||t||ec\right]. \tag{9}$$

(3) d_n signs the string $c||m_{n,t}$ through the Schnorr signature generation algorithm Schnorr.Sign, as per Eq. 10, using its private key ssk_n, generating the message signature σ_m.

$$\sigma_m = \text{Schnorr.Sign}(\text{c}||\text{m}_{\text{n,t}}, \text{ssk}_n). \tag{10}$$

Then, d_n delivers a broadcast *RID* message containing the information string $m_{n,t}$ and the message signature σ_m.

(4) The observer $\mathcal{R}$ receives the *RID* packet, decodes it and first verifies its authenticity. Specifically, $\mathcal{R}$ applies the Schnorr signature verification algorithm Schnorr.Verify, using the public key spk_n of d_n, as per Eq. 11.

$$\text{Schnorr.Verify}(\text{c}||\text{m}_{\text{n,t}}, \text{spk}_n, \sigma_m). \tag{11}$$

If the algorithm Schnorr.Verify ends successfully ($d = 1$), $\mathcal{R}$ proceeds further. Otherwise, $\mathcal{R}$ discards the message.

(5) The observer $\mathcal{R}$ recovers the ephemeral nonce $\tilde{r}$. To do so, $\mathcal{R}$ executes Eq. 12, using the CP-ABE decryption algorithm CP_FABEO.Decrypt providing in input the ciphertext c, the master public key of the USS, namely, mpk_A, the attribute policy $\mathcal{P}$ delivered by d_n (included in c), its attribute set $attr\_set_{\mathcal{R}}$ and the private key $\text{sk}_{\mathcal{R}}$.

$$\tilde{r} = \text{CP\_FABEO.Decrypt}(\text{mpk}_A, \mathcal{P}, attr\_set_{\mathcal{R}}, c, \text{sk}). \tag{12}$$

Note that Eq. 12 $\neq \perp$ only if the attribute set possessed by the observer $\mathcal{R}$, namely, $attr\_set_{\mathcal{R}}$, satisfies the access control policy $\mathcal{P}$. Otherwise, Eq. 11 $= \perp$, meaning that $\mathcal{R}$ is not authorized to access the pilot location.

(6) $\mathcal{R}$ can derive the recovered ephemeral symmetric key $\tilde{K_t}$, using the KDF and the recovered ephemeral nonce $\tilde{r}$ previously obtained, as in Eq 13.

$$\tilde{K_t} = \text{KDF}(\tilde{r}). \tag{13}$$

(7) Finally, $\mathcal{R}$ can decrypt the encrypted *pilot location* c and obtain O_p using the AES decryption algorithm and the recovered ephemeral symmetric key $\tilde{K_t}$, via Eq. 14.

$$O_p = \text{AES.DEC}(c, \tilde{K_t}). \tag{14}$$

Note that SNELL is a standard-compliant extension to the RID specification. Therefore, it is not meant to replace RID but to work aside it, achieving additional location privacy for the pilot. Moreover, note that SNELL does not require all observers to register with

the USS. Assume an observer, not registered with the USS, receives a message secured using SNELL. The observer can successfully read all the information in the RID message, excluding the pilot location, as these pieces of information are in plaintext. Suppose the observer knows the public key of the USS (e.g., it downloads such key from an Internet web page). In that case, it can also verify the authenticity of the information listed above by executing step 4 of SNELL. However, as the observer did not register with the USS, it cannot successfully decrypt the pilot's location, similar to an unauthorized message receiver. Therefore, SNELL requires only authorized observers to execute the Registration Phase. We acknowledge that such registration is not mandated by the standard RID and neither by drip. We believe regulatory entities can easily maintain a dedicated USS specifically for the purpose.

5 SECURITY ANALYSIS

5.1 Security Considerations

Overall, SNELL achieves the following security objectives.

Selective Pilot Location Disclosure. RID messages emitted by drones equipped with SNELL do not include plaintext information about the pilot location. Instead, pilot location information fields include the ciphertext C generated through Eq. 8 and the policy $\mathcal{P}$ necessary to (obtain the key to) decrypt such a ciphertext. Only the observers in possession of a set of attributes $attr\_set_{\mathcal{R}}$ satisfying the access policy $\mathcal{P}$ can obtain the ephemeral key K_t and decrypt the location of the pilot O_p. We verify this security property of SNELL through ProVerif in Sec. 5.2. Thus, the success ratio of any attacks on the AES ciphertext depends on the capability of guessing the plaintext from one or multiple ciphertexts. An assumption the attacker can use is that the drone's pilot should be in the communication range of the drone to be able to issue commands and receive data from it. Modern drones can use WiFi with relatively long ranges, (theoretically) up to 15 km away from the actual drone location [30], creating a reasonably large location estimation error.

Robustness to Impersonation Attacks. SNELL messages include the signature σ_m, obtained using the Schnorr signature generation mechanism (Eq. 10). Thus, the drone d_n uses its private key ssk_n to authenticate RID messages. At the same time, observers use the corresponding public key spk_n to verify that d_n emitted a particular RID message. Provided that such key stays private on d_n, no other entities can use it to generate messages which would appear to be emitted by the legitimate drone, thus protecting against impersonation. We also verify this security property through ProVerif in Sec. 5.2. In principle, an attacker might achieve impersonation attacks also by impersonating the USS, to issue keys on its behalf. However, note that the entities involved in the protocol connect over the Internet to the USS in the *Registration Phase* using the well-known TLS protocol, and we rely on its security features to protect against impersonation over the public Internet.

Robustness to RID Messages Modification. As per Eq. 10 in Sec. 4, d_n generates the signature σ_m on the plaintext string $m_{n,t}$, including the identity of the drone, its location, speed, encrypted pilot location, timestamp, and emergency code. Thus, per the properties of digital signatures, modifications to any of these parameters would cause the signature verification operation in Eq. 11 to fail, leading to message discard. Therefore, using digital signatures also protects against RID message modification. We also verify this security property through ProVerif in Sec. 5.2.

Mitigation of Replay Attacks. The combined usage of the timestamp t within RID messages and the Schnorr signature verification algorithm mitigates replay attacks. When an observer receives an RID message, it can extract the timestamp t and match it with the current local time based on local criteria. The message is discarded when the difference between the message timestamp and the current local time is higher than a local threshold. Since RID is a broadcast-only communication scenario, providing formal protection against replay attacks is impossible. In principle, an attacker might replay a legitimate message immediately after having eavesdropped it, and such a message would still be accepted as legitimate by the receiver. On the one hand, we notice that the discussed attack is not due to SNELL but to the broadcast nature of the problem considered here. On the other hand, we highlight that attacks carried out in such a way have limited scope for the attacker. Indeed, replaying messages immediately after their legitimate disclosure would make the legitimate drone appear very close to where it is, likely not creating any safety-related concerns.

Robustness to Birthday Attacks. SNELL features a KDF, used in Eq. 7 to map the nonce r to a key K_t used within AES (Eq. 8). The KDF is similar to hashing functions. Thus, based on the features of the KDF, an attacker might be able to find a colliding hash, i.e., find a value $p \neq r$ leading to the same hash value K_t. Such attacks are well-known in the literature as *Birthday Attacks* [5]. When successful, the attacker would bypass the CP-ABE scheme, being able to decrypt the pilot location O although not in possession of a set of attributes allowing to obtain the actual value r used by the drone. To avoid such attacks, we need to integrate a robust KDF, minimizing the chances for an adversary to carry out successful birthday attacks. As detailed in Sec. 6.1, our proof-of-concept integrates the KDF Argon2 [9], which can be configured to make birthday attacks computationally hard.

Asynchronous Attributes Revocation. As SNELL integrates CP-ABE techniques, it also inherits common issues of CP-ABE, such as asynchronous attributes revocation. Suppose an observer gets an attribute revoked suddenly during a drone mission. In that case, the same observer can most likely still use the cryptography materials in its possession (e.g., the public key) to decrypt the ciphertext C delivered by a drone. The typical solution consists of updating the cryptography materials corresponding to a given attribute set to exclude data consumers with revoked attributes from the set of allowed ones [34]. SNELL offers several opportunities to do so. First, before starting a new mission (Registration Phase), d_n can download an updated list of cryptography materials, so considering the latest updates on revoked attributes and users. Also, the drone can use the communication link with the pilot to retrieve real-time updates on updated cryptography materials for specific attributes. However, when d_n does not feature a persistent Internet connection, such an issue cannot be resolved at run-time but only after the drone mission. Note that asynchronous attribute revocation is an issue only for the observers (receivers). Indeed, we can design the attribute trees so that the relevant authorities always have access to the pilot's location (e.g., forcing the use in the policy of an attribute specifically dedicated to them), preventing the abuse of asynchronous attribute revocation for malicious purposes.

5.2 Formal Verification

Formal Security Proofs. Proving formally the security of SNELL implies proving formally the security of its building blocks, i.e., Schnorr-type signatures for message authentication and the FABEO implementation of CP-ABE for pilot confidentiality. As for authentication, Pointcheval and Stern first proved in [54] the security of Schnorr-type signatures in the Random Oracle Model (ROM). In Sec. 4.2 of [54] they prove that if an existential forgery of the Schnorr-type signature scheme is possible with nonnegligible probability, then factorization of RSA moduli can be performed in polynomial time—which is not possible, so proving the hypothesis. As for pilot confidentiality, the adversary gets access to two pieces of information: (i) the ciphertext c result of CP-ABE encryption of a nonce r, and (ii) the AES encryption x of the pilot location O_p using as a key the output of the KDF of the nonce, namely K_t. Thus, the adversary could recover the pilot location either by breaking CP-ABE encryption or by breaking AES. As for CP-ABE, Riepel and Wee in [47] prove the security of the FABEO implementation of CP-ABE in the Generic Group Model (GGM), i.e., considering an adversary can perform group operations via oracle access. Specifically, in the same setting of our paper, consider $\lambda \in N$ the security parameter and A an adversary that on input $(1^\lambda, p)$ makes Q_{op} group operation queries to oracles O_{add} and O_{pair} (oracle for pairing operations), Q_{ct} queries to O_{ct} (oracle for ciphertexts), Q_{sk} queries to O_{sk} (oracle for secret keys) and Q_H queries to the random oracle H (oracle for hashing functions). Also, denote with $|S|$ the size of the attribute set and n_1 the number of rows queried to O_{ct}. In Sec. 6.1 of [47] the authors prove that CP-ABE is adaptively secure in the GGM such that the probability that the following Eq. 15 holds.

$$ADV^{GCM}_{CP-ABE,A}(\lambda) = \frac{3(Q_H + (n_1+3)Q_{ct} + (|\mathcal{S}|+2)Q_{sk} + Q_{op})^2}{p} \tag{15}$$

Finally, being AES a symmetric encryption scheme, there is no such formal security proof. However, as of today, the most successful attacks to AES reduce negligibly the complexity of the attack compared to the time to brute-force the key (e.g., see [11]). Thus, when used with keys of sufficient size (e.g., 128-bit), AES is widely considered secure.

Logic Verification via ProVerif. We formally verify some of the security features offered by SNELL, i.e., the confidentiality of the pilot location data, RID messages authenticity and robustness against RID messages modification, using ProVerif [10], in line with many recent works [38], [62]. We chose such a strategy since the security of the single building blocks used by SNELL, i.e., CP-ABE, Schnorr signatures, and AES, has been already proved [47], [4]. However, their logical combination into SNELL could generate new security issues. In such cases, using automated verification tools like ProVerif is the most suitable way to look for security issues.

Leveraging the Dolev-Yao attacker model, ProVerif allows the attacker to perform various actions on the shared communication channel, such as reading, modifying, deleting, and creating new packets. Once the user defines the security goals of the protocol, ProVerif carries out automated procedures to check whether the attacker can compromise these properties. If an attack is detected, ProVerif also provides a detailed description of it.

We modelled SNELL in ProVerif using a shared channel, where all entities, including the adversary, have access. In line with our adversary model, we also used a private channel accessible only by the drone and the USS during the registration. To prove the properties of our interest, we defined two events:

(1) *acceptUAV(id)* indicates that a drone with the identity *id* is running SNELL;
(2) *termUAV(id)* means that a receiver successfully executed SNELL and verified that the message was generated by an authentic drone with identity *id*.

Verification Summary

```
Query event(termUAV(id_3)) ==> event(acceptUAV(id_3))
is true.
Query not attacker latO[] is true.
Query not attacker lonO[] is true.
Query not attacker altO[] is true.
```

Figure 4: Excerpt of the output provided by the ProVerif tool.

As for the output messages, we used the following notation:

- *event(last_event ()) ⟹ event(previous_event ()) is true* denotes that the function *last_event* is executed only when *previous_event* is executed.
- *not attacker(elem[]) is true* demonstrates that the attacker does not own the value of *elem*.

Figure 4 shows the excerpt of the output of ProVerif. The tool confirms that (i) message authenticity is verified, and (ii) the attacker cannot obtain the pilot location, namely, $latO$, $lonO$, $altO$.

To allow the readers to reproduce our results and verify our claims, we also release the source code of SNELL in ProVerif at [56].

6 PERFORMANCE EVALUATION

6.1 Proofs-of-Concept Details

We implemented two proofs-of-concept of SNELL, i.e., on the Lumenier QAV-R commercial drone and on the ESP32.

Deployment on Lumenier QAV-R. We first implemented a proof-of-concept of SNELL using the commercial drone Lumenier QAV-R [35], integrating the embedded processing unit Raspberry Pi 4 [42] as the mission computer. It is an embedded platform equipped with a System on Chip (SoC) Broadcom BCM2711, a 64-bit SoC processor quad-core Cortex-A72 (ARM v8) running at 1.5 GHz GHz, 4 GB of RAM, and 8 GB of storage. As for the operating system, the Raspberry Pi 4 runs *Ubuntu 20.04 LTS Focal Fossa* [60]. The mission computer is connected to a *Pixhawk 4 Autopilot* integrating, in turn, a GPS module Holybro M9N. As for the observer, we used another standalone Raspberry Pi 4 device, so as to consider the case of a receiver device equipped with intermediate computational capabilities. As for the software, we implemented our protocol through the Python programming language. As Python is an interpreted programming language, this is a sub-optimal solution, providing worst-case performance metrics for integrating SNELL into commercial drones. We used the publicly-available implementation provided by the authors of FABEO at [48] for integrating

Figure 5: Drone used as part of our proof-of-concept.

CP-ABE cryptography operations; we adopted the pairing-friendly elliptic curve *BN254*, while for the Schnorr signature generation and verification algorithm, we used the implementation available at [37]; finally, we used the well-known KDF Argon2, robust to birthday attacks discussed in Sec. 5.1 [52]. As for the communication between the drone and the observer, we set up an ad-hoc IEEE 802.11 network. Then, we used the Coincise Binary Object Representation (CBOR) encoding scheme [12] to encode RID messages generated according to SNELL within custom IEEE 802.11b MAC-layer frames broadcasted over-the-air, integrating our code with the interactive packet manipulation tool *Scapy* [8]. For the readers' convenience, we report in Tab. 4 (Annex) the fields included in the RID messages generated through SNELL, together with their size when encoded into WiFi frames.

Deployment on ESP32. To test the viability of SNELL on more constrained devices, we deployed another proof-of-concept using the constrained device ESP32. It is a tiny microcontroller equipped with a dual-core Xtensa LX6 processor with up to 240 MHz clock frequency, 328 kB of Dynamic Random Access Memory (DRAM), and supporting various WiFi standards, e.g., IEEE 802.11b/g/n [23]. It also features a hardware accelerator that efficiently performs random number generation, AES and other ECC-based operations (e.g., Schnorr signature generation). We implemented SNELL on the ESP32 using the Arduino software tools, particularly integrating the *MIRACL Core* library for the cryptographic operations [39]. Note that the hardware accelerator of the ESP32 cannot be used to accelerate Pairing-based operations. To solve this issue, we devise three implementation strategies: (i) *Fully Precomputed*, where we precomputed the pairing-based cryptography elements on a fully-fledged laptop, while we executed ECC-based operations at runtime on the ESP32; (ii) *Partially Precomputed*, where we precomputed and stored a part of the pairing-based cryptography elements (ephemeral random nonce $r \in G_T$) on the device, while we executed the rest of pairing-based operation and ECC-based operations at runtime on the ESP32; and (iii) *Parallel Computed*, where we took advantage of the dual-core architecture of the ESP32 to accelerate pairing-based operations further so that all the cryptographic values are computed at runtime. Finally, for architectural reasons, the Maximum Transmission Unit (MTU) at the MAC-layer available on the ESP32 is 1,500 bytes [23]. Thus, we adopted the well-known ECC point compression mechanism [13] to include as much data as possible in a single wireless message (see Tab. 5 in Annex). Finally, we make our implementation of SNELL on the ESP32 publicly available at [27] to allow interested readers to use our work further.

6.2 Experiment Settings

To assess the feasibility of SNELL, we conducted several experiments through the two proofs-of-concept described in Sec. 6.1. Specifically, for the deployment of SNELL on the Lumenier QAV-R drone, we measured the bandwidth overhead, execution time, and energy consumption of our approach. For the bandwidth overhead, we experimentally measured the size of IEEE 802.11 frames delivered by the transmitting device while varying the number of attributes in the policy. We carried out such an investigation by using three different elliptic curves for the execution of the Schnorr signature scheme, i.e., *secp256k1*, *secp384r1*, and *secp521r1*. We also measured experimentally the execution time (in milliseconds) of the *Deployment Phase*, both on the transmitter (drone) and receiver (observer). Specifically, on the transmitter, we measured the time required from the nonce extraction process (step 1 in Fig. 3) to the delivery of the signed message (step 3 in Fig. 3). Instead, on the receiver, we measured the time required to execute the operations from the reception of the message (step 4 in Fig. 3) to the decryption of the pilot location (step 7 in Fig. 3). We also experimentally measured the energy consumption required by the *Deployment Phase* of SNELL, both on the transmitter and the receiver. Fig. 6 shows the testbed used for energy consumption measurements on the Lumenier QAV-R. We used a Keysight E36231A DC power supply

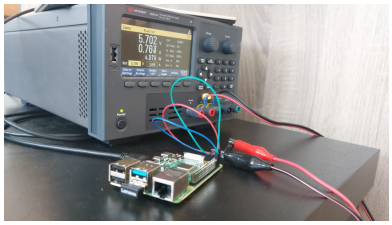

Figure 6: Experimental setup used for energy consumption measurements.

set to take the measurements, providing a voltage of 5.7 V to the mission computer Raspberry Pi (note that we disconnected the Raspberry from the drone frame to ease the connection of the wires to the pins for the measurements). Specifically, we measured the difference in the electrical current drained by the device under test between two different states: (i) idle and (ii) during the execution of our protocol. We computed an average difference in the electric current drained by the device over 1,000 runs between the two states equal 289.3 mA. We computed the actual energy consumption using Eq. 16, using such values and the time measurements obtained experimentally above.

$$E[mJ] = V \cdot \int_0^T i(t)dt, \tag{16}$$

where V is the input voltage (5.7 V), $i(t)$ is the instantaneous drained current, and T denotes the duration of the specific operation, measured through the tests described above. For all the tests described above, we executed them during regular drone operations and repeated them 1,000 times, for a total of 750,000 tests, reporting the corresponding average values and 95% confidence intervals, using the tool *paramci* provided by MATLAB.

We also investigated the performance of SNELL on the ESP32. Specifically, we selected a fixed configuration of the Schnorr signature generation algorithm (signature of 64 bytes), and we measured the time, energy and DRAM necessary to run SNELL using the three implementation strategies mentioned in Sec. 6.1.

6.3 Results

Experiments on the Lumenier QAV-R. Fig. 7 reports the RID message size after integration of SNELL while varying the number of attributes in the access control policy used by the drone, with a fixed attribute size of 2 B. We also considered three different elliptic curves to generate the Schnorr signature. Overall, we

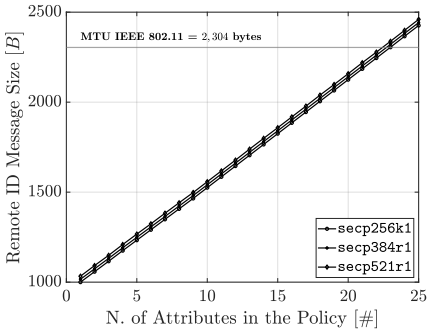

Figure 7: Size of the RID message after integration of SNELL, increasing the number of attributes in the policy, with various elliptic curves.

notice that using an increasing number of attributes in the access control policy increases linearly the size of the RID message generated through SNELL. Considering the requirement **R6** discussed in Sec. 3.4, we notice that we can include up to 22 attributes into the access control policy without exceeding the payload available in a single WiFi frame, i.e., 2,304 B. This finding does not depend on the elliptic curve used for the Schnorr signature generation algorithm. Indeed, the additional bandwidth required to send a larger signature always fits the remaining space available into an RID message secured with an access control policy of 22 attributes. Such a finding allows SNELL to feature at the same time the maximum possible level of security for the signature (512 bits), and very fine-grained access control policies, including up to 22 different attributes. Fig. 8 reports the average time required by the drone to generate an RID message secured through SNELL while increasing the number of attributes in the access control policy. Similarly to the previous experiment, we also investigate the usage of multiple curves to generate the Schnorr signature. On the one hand, using more attributes increases (linearly) the time necessary to generate secure RID messages. Such time also increases when considering a larger curve. On the other hand, even with the most demanding configurations (elliptic curve *secp521r1* and 22 attributes into the policy), the message generation time on our proof-of-concept is 306.855 ms, never exceeding 3 s, as per the requirement **R5** discussed in Sec. 3.4. Thus, SNELL can fulfil all requirements, including **R5** and **R6**, when using up to 22 attributes for the access control

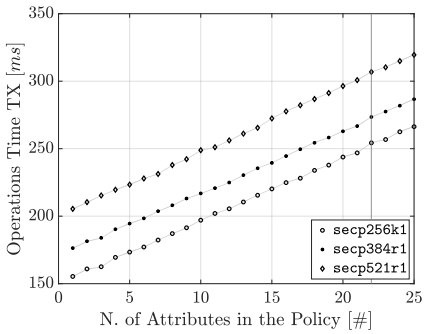

Figure 8: Messages generation time through SNELL, increasing the number of attributes into the access control policy, with various elliptic curves.

policy. We also measured the time necessary to verify and decrypt the message at the observer, as reported in Fig. 9. First, considering

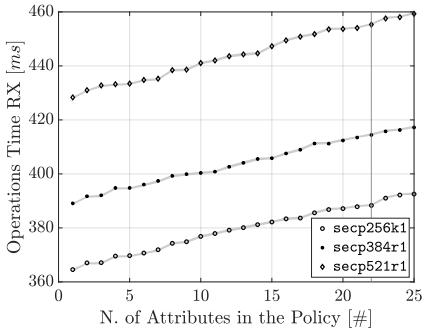

Figure 9: Messages verification and decryption time through SNELL, increasing the number of attributes into the access control policy, with various elliptic curves.

a given number of attributes in the policy and elliptic curve size, the message verification and decryption take longer than secure message generation (see Fig. 8). This result is not only expected but also intended. Indeed, the operations executed at the receiver side are more processing-intensive than the ones performed at the transmitter side, contributing to fulfilling the requirements **R5** and **R6** discussed above. At the same time, in the worst case, we require 455.303 ms at the receiver for such tasks, which is a reasonable delay for RID messages verification and decryption. Finally, we also evaluated the energy consumption required by the *Deployment Phase* of SNELL. We report the results in Fig. 10 and Fig. 11 for the transmitter and the receiver, respectively.

In line with previous measurements, energy consumption increases with the number of attributes in the policy and the elliptic curve size. However, the energy consumption, especially at the transmitter, stays very low. Assuming the worst case of 22 attributes in the policy and the usage of the curve *secp521r1*, our approach requires 505.948 mJ of energy per single secure RID message emitted

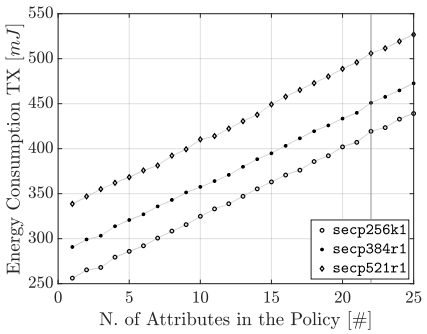

Figure 10: Energy required for messages generation through SNELL, increasing the number of attributes into the access control policy, with various elliptic curves.

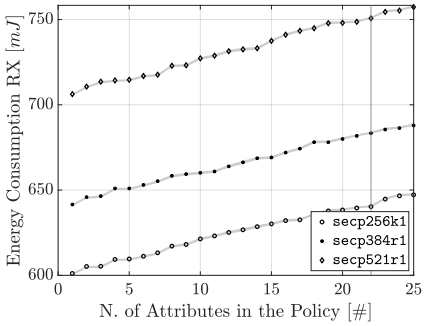

Figure 11: Energy required for messages verification and decryption through SNELL, increasing the number of attributes into the access control policy, with various elliptic curves.

by the drone. Considering that the drone Lumenier QAV-R is powered by a battery with an overall capacity of 266, 400 J (5, 000 mAh at 14.8 V), running SNELL for a mission of 30 minutes (with a message rate of 1 per second) would require $505.948 \cdot 1,800 = 9.1071 \cdot 10^5$ mJ, which is only the $\approx$ 0.34% of the battery capacity. Thus, SNELL minimally affects the energy availability and lifetime of the drone, being highly usable and very lightweight.

Experiments on the ESP32. We report in Tab. 5 the message format of SNELL messages on the ESP32. Note that, dedicating 64 bytes to the Schnorr signature, we can build and use policies including up to 23 attributes, i.e., one more compared to the previous deployment—although less space is available in the payload, the use of point compression techniques make it possible to use more attributes. Fig. 12 reports the average time required by the ESP32 to generate a RID message secured through SNELL while increasing the number of attributes in the access control policy. We report the performance for the three deployed implementation strategies, i.e., *Fully Precomputed*, *Partially Precomputed* and *Parallel Computed*, considering the adoption of the curve *secp256k1* for the generation of the Schnorr signature (signature size of 64 bytes).

Note that the *Fully Precomputed* strategy is the quickest implementation strategy, taking only 307.8 ms with 23 attributes. In the same configuration, the *Partially Precomputed* strategy requires 2, 962 s. In contrast, the *Parallel Computed* strategy reduces such execution time to only 1, 888 s, always less than the limit of 3 seconds required by RID. We also notice a slight dip in the time needed by the *Parallel Computed* version when working with 2 attributes, compared to the use of a single attribute: such unexpected behavior is due to the parallelization of some attribute-related operations when working with more attributes, which is not possible when working with only a single attribute. Using the same setup, we also investigated the energy consumption of SNELL on the ESP32. Fig. 13 summarizes our results. The *Fully Precomputed* strategy

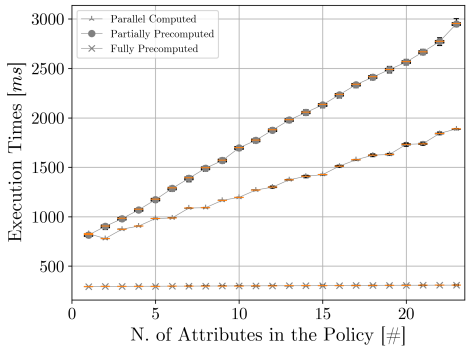

Figure 12: Messages generation time through SNELL on the ESP32, increasing the number of attributes, considering the three mentioned implementation strategies.

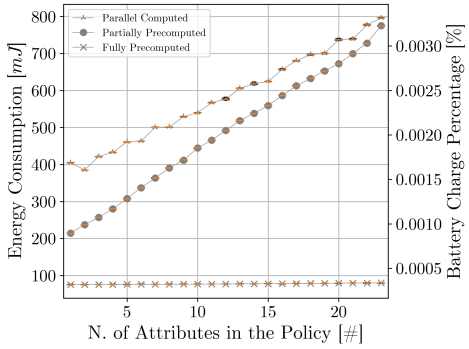

Figure 13: Energy required by SNELL on the ESP32. The y-axis on the right side shows the consumption w.r.t. to the capacity of the battery (900 mAh at 7.8 V).

emerges as the most energy-efficient solution, as it requires almost no runtime computations (less than 0.0005% of the battery capacity). Also, as all cryptographic operations are precomputed, the consumed energy increases very slowly with the considered number of attributes in the policy, differently from the *Partially Precomputed* and the *Parallel Computed* strategies. Activating two cores requires more energy, causing the overhead of the *Parallel*

Computed strategy to be generally larger. However, the difference between the two implementation strategies tends to decrease as the number of attributes increases. Indeed, by increasing the number of attributes, more operations can be parallelized, reducing execution time and compensating the energy overhead derived by running two cores simultaneously. The two strategies consume almost the same energy when considering the maximum number of attributes that can be included in a single WiFi frame (23), i.e., ≈ 0.0033% of the battery capacity. Such a limited overhead allows for executing SNELL approx. 33,000 times, i.e., for a flight duration of at least 27 hours (much more than the regular flight duration of constrained drones). Such a result confirms the negligible impact of SNELL on the usability of ESP32-based drones [22]. Finally, we also summarize in Fig. 14 the DRAM consumption of SNELL on the ESP32 while varying the number of attributes in the policy. The DRAM

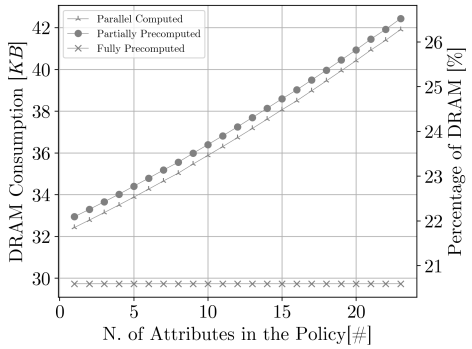

Figure 14: DRAM required by SNELL on the ESP32, increasing the number of attributes, considering the three mentioned implementation strategies. The y-axis on the right shows the consumption w.r.t. to the available DRAM (328 kB).

consumption of the *Fully Precomputed* strategy on the ESP32 is constant (29.72 kB), while for the other two strategies it increases with the number of attributes. With 23 attributes in the policy, the more memory-hungry approach is the *Partially Precomputed* version, which requires 42.43 kB, i.e., 12.93% of the DRAM available on the device. This result originates from the file access operation required by such a strategy. Conversely, the *Parallel Computed* strategy reduces the DRAM overhead for 23 attributes to 41.91 kB.
Finally, we experimentally measured the latency overhead of SNELL, i.e., the additional time it takes for the plaintext information in the RID message to become available at the receiver. Without SNELL, the communication latency is 0.0183 s. Worst case, i.e., with the maximum number of attributes in the policy, the communication latency increases to 0.785 sec using the QAV-R Lumenier and to 2.331 sec using the ESP-32. Overall, our results on the ESP32 confirm that SNELL is a technique suitable for providing selective pilot location disclosure even on very constrained drones.

7 RELATED WORK AND COMPARISON

A few recent works considered privacy issues connected with RID regulations, e.g., [58], [62] (identity privacy) and [14] (drone location privacy). Similarly, some works such as [3] and [29] considered pilot authentication using traffic analysis techniques. However, none of these works consider pilot location privacy. We might adapt some of these proposals, e.g., the ones in [62], to achieve pilot location privacy. However, such a solution cannot support opportunistic selective data encryption, and neither the decryption of the pilot location—it is possible to obtain and verify a random identifier chosen at runtime, but not the information to be protected.

However, the problem of pilot location privacy considered in this manuscript is very close to the issue of location privacy considered in other domains, such as for Automatic Dependent Surveillance - Broadcast (ADS-B) and Vehicular Ad Hoc Networks (VANETs). Many contributions considered privacy issues in ADS-B networks arising from the plaintext disclosure of the aircraft's location within broadcast messages. For instance, Yang *et al.* [63] proposed to preserve the privacy of the aircraft' identities and the associated GNSS location by encrypting the International Civil Aviation Organization (ICAO) aircraft address. However, it does not allow opportunistic decryption of the messages. Also, their proposal requires multiple messages, not fulfilling our requirement *R6*. Several contributions in the VANET domain propose privacy-preserving broadcast authentication protocols involving vehicles and Internet-connected Road-Side Units (RSUs). For example, Zhou *et al.* [65] propose using a multi-key secure outsourced computation scheme based on one-way trapdoor permutations. In the same domain, Lai *et al.* [33] present a fully privacy-preserving and revocable identity-based broadcast encryption scheme to protect the message confidentiality and the identity of receivers.Differently from our scenario, both the cited schemes do not consider requirements on the time taken to generate messages (*R5*) and need multiple MAC-layer frames (*R6*). Other proposals come from the healthcare domain, where only authorized entities (e.g., doctors) should retrieve private data, such as the proposal by Mandal [36]. However, these schemes do not consider requirements on the size (*R6*) and the authenticity (*R7*). The literature also provides several *theoretical* solutions, e.g., the work by Gay et al. [28], but they do not fulfil all our requirements.

Attribute-Based Signcryption (ABS) schemes cannot be used in our context since they require encrypting and signing the same plaintext, so not fulfilling simultaneously the requirements *R1* and *R7*. To keep compliance with RID requirements, we should encrypt the pilot location only using ABS and then apply Schnorr signatures to sign the whole message, so generating additional overhead. To provide further insights, we compare the encryption time required by SNELL and two reference approaches using ABS, i.e., [18] and [45]. Specifically, for each of the selected approaches, we considered two configurations for a total of four benchmarks: (i) non-standard-compliant, where we applied the ABS technique on the whole RID message, so violating requirement *R1*, and (ii) standard-compliant, where we first applied the ABS technique on pilot location and then sign the whole RID message through the Schnorr signature algorithm (using the curve *secp256*). We report in Fig. 15 the nominal encryption time required by SNELL and the four benchmarks explained above, i.e., the time to generate the ciphertext. For SNELL the nominal execution time reported in Fig. 15 does not match with the one reported in Fig. 8 since we consider only the time to generate the ciphertext. Our investigation shows that SNELL is always the most lightweight solution. Considering 23

Table 1: Qualitative comparison of SNELL against related work considering selective broadcast encryption. The symbol ● denotes that a specific requirement is fulfilled, while the symbol ○ denotes that the requirement is not fulfilled.

Ref.	Selective Access to Pilot Data **(R1)**	Opportunistic Pilot Data Encryption **(R2)**	Offline Pilot Data Decryption **(R3)**	Pilot Data Unlinkability **(R4)**	Short Pilot Location Update Time **(R5)**	Single WiFi Frame **(R6)**	Messages Authenticity **(R7)**	Zero-Touch Pilot Location Disclosure **(R8)**
[62]	○	○	●	●	●	●	●	●
[28]	○	○	●	○	○	○	○	○
[33]	●	○	●	●	○	○	○	○
[36]	●	●	○	●	○	○	○	○
[63]	○	○	○	●	●	○	●	○
[65]	○	●	○	●	●	○	○	●
[18]	○	●	●	●	●	○	●	●
[45]	○	●	●	●	●	○	●	●
SNELL	●	●	●	●	●	●	●	●

attributes, the processing overhead of SNELL is 32.5% less than the overhead required by the most lightweight competing approach.

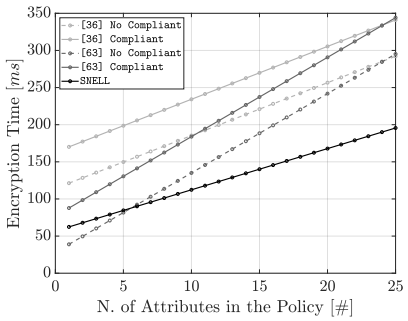

Figure 15: Nominal Encryption Time required by SNELL and the approaches in [18] and [45], used each in a RID-compliant and non-RID compliant fashion.

The only contribution specifically considering drone pilot privacy is the draft document in [44]. The authors propose encrypting the fields dedicated to the pilot location with a Format-Preserving Encryption (FPE) technique based on a pre-shared symmetric key. As a result, compared to SNELL, the solution in [44] requires less bandwidth, computation and energy overhead when encrypted nonces are not pre-computed. However, such a solution does not fulfil the requirements *R2*, *R7*, and *R8*. Our problem is different from the one considered by contributions in the area of privacy-preserving ride-hailing services, e.g., [41], [61] and [40], as such proposal require either intermediary entities or persistent Internet connection. We also notice that 0-RTT handshake protocols are not suitable for this context, as they are designed to work in a peer-to-peer session involving a client and a server [16]. Tab. 1 summarizes our discussion. To the best of our knowledge, SNELL is the only solution fulfilling, at the same time, all the considered requirements. Also, the design of SNELL solves several technical challenges: (i) designing a solution that can work in a broadcast scenario with no peer-to-peer connections and possibly no Internet connection; (ii) reducing the overhead of running CP-ABE on constrained devices; (iii) minimizing the communication overhead to fit a single RID message. Finally, SNELL solves a real problem, i.e., integrating a mechanism for the selective disclosure of the pilot location in challenging privacy-sensitive use cases (see Sec. 3.3).

8 CONCLUSION AND FUTURE WORK

In this paper, we proposed SNELL, a novel protocol allowing selective authenticated pilot location disclosure for Remote ID-enabled drones. By integrating SNELL, drones broadcast pilot location in a *protected* way, allowing only receivers in possession of a compatible set of attributes to retrieve the actual location of the pilot. We deployed SNELL both on an actual drone Lumenier QAV-R, using a Raspberry Pi 4 as the mission computer, and on a constrained device ESP32, used in many low-end micro-drones, and we conducted an extensive performance assessment. We demonstrate experimentally that SNELL meets all the requirements for a RID-compliant deployment, such as a pilot location update time well below the threshold of 3 s. At the same time, with SNELL, it is possible to construct complex access control policies, including up to 22 attributes on the QAV-R Lumenier and up to 23 attributes on the ESP32, using a single WiFi frame and posing a tiny energy toll on the involved devices (with the most demanding security configuration, ≈ 0.34 % of the battery capacity of the Lumenier QAV-R drone in 30 minutes and ≈ 0.0033% of the battery capacity of the ESP32). Future work involves optimising SNELL to cope with more advanced use cases.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers who helped improve the paper. The work has been supported by the Technology Innovation Institute, Abu Dhabi, United Arab Emirates, and the INTERSCT project, Grant ID NWA.1162.18.301, funded by Netherlands Organisation for Scientific Research (NWO). The findings reported herein are solely responsibility of the authors.

REFERENCES

[1] 2022. Standard Specification for Remote ID and Tracking. https://www.astm.org/f3411-22a.html
[2] Simeon Okechukwu Ajakwe, Dong-Seong Kim, and Jae Min Lee. 2023. Drone Transportation System: Systematic Review of Security Dynamics for Smart Mobility. *IEEE Internet of Things Journal* (2023), 1–1.
[3] Ruba Alkadi, Sultan Al-Ameri, Abdulhadi Shoufan, and Ernesto Damiani. 2021. Identifying Drone Operator by Deep Learning and Ensemble Learning of IMU and Control Data. *IEEE Transactions on Human-Machine Systems* 51, 5 (2021), 451–462. https://doi.org/10.1109/THMS.2021.3102508

[4] Paulo SLM Barreto, Marcos A Simplicio, Jefferson E Ricardini, and Harsh Kupwade Patil. 2020. Schnorr-based implicit certification: Improving the security and efficiency of vehicular communications. *IEEE Transactions on Computers* 70, 3 (2020), 393–399.

[5] Mihir Bellare and Tadayoshi Kohno. 2004. Hash function balance and its impact on birthday attacks. In *Advances in Cryptology-EUROCRYPT 2004: International Conference on the Theory and Applications of Cryptographic Techniques, Interlaken, Switzerland, May 2-6, 2004. Proceedings 23*. Springer, 401–418.

[6] Kais Belwafi, Ruba Alkadi, Sultan A Alameri, Hussam Al Hamadi, and Abdulhadi Shoufan. 2022. Unmanned Aerial Vehicles' Remote Identification: A Tutorial and Survey. *IEEE Access* 10 (2022), 87577–87601.

[7] John Bethencourt, Amit Sahai, and Brent Waters. 2007. Ciphertext-Policy Attribute-Based Encryption. In *IEEE Symposium on Security and Privacy*. IEEE, 321–334.

[8] Philippe Biondi. 2023. Scapy. https://scapy.net/. (Accessed: 2024-Feb-26).

[9] Alex Biryukov, Daniel Dinu, and Dmitry Khovratovich. 2016. Argon2: new generation of memory-hard functions for password hashing and other applications. In *2016 IEEE European Symposium on Security and Privacy (EuroS&P)*. IEEE, 292–302.

[10] Bruno Blanchet. 2009. Automatic Verification of Correspondences for Security Protocols. *Journ. of Comp. Security* 17, 4 (2009), 363–434.

[11] Andrey Bogdanov, Dmitry Khovratovich, and Christian Rechberger. 2011. Biclique cryptanalysis of the full AES. In *Advances in Cryptology–ASIACRYPT 2011: 17th International Conference on the Theory and Application of Cryptology and Information Security, Seoul, South Korea, December 4-8, 2011. Proceedings 17*. Springer, 344–371.

[12] C Bormann and P Hoffman. 2020. RFC 8949 Concise Binary Object Representation (CBOR). *Internet Engineering Task Force (IETF)* (2020).

[13] Joppe W Bos, J Alex Halderman, Nadia Heninger, Jonathan Moore, Michael Naehrig, and Eric Wustrow. 2014. Elliptic Curve Cryptography in Practice. In *18th International Conference on Financial Cryptography and Data Security*. Springer, 157–175.

[14] Alessandro Brighente, Mauro Conti, and Savio Sciancalepore. 2022. Hide and Seek: Privacy-Preserving and FAA-Compliant Drones Location Tracing. In *Proceedings of the 17th International Conference on Availability, Reliability and Security* (Vienna, Austria) *(ARES '22)*. Association for Computing Machinery, New York, NY, USA, Article 134, 11 pages.

[15] Commercial UAV News. 2021. FlyingBasket Gets Approval for Cross-Border Operations. https://www.commercialuavnews.com/drone-delivery/flyingbasket-gets-approval-for-cross-border-operations. (Accessed: 2024-Feb-26).

[16] Cas Cremers, Marko Horvat, Sam Scott, and Thyla van der Merwe. 2016. Automated Analysis and Verification of TLS 1.3: 0-RTT, Resumption and Delayed Authentication. In *IEEE Symposium on Security and Privacy (SP)*. 470–485. https://doi.org/10.1109/SP.2016.35

[17] D. Micault, M. Boucadair. 2022. *Drone Remote Identification Protocol (DRIP) Charter - charter-ietf-drip-01*. Technical Report. IETF.

[18] Ningzhi Deng, Shaojiang Deng, Chunqiang Hu, and Kaiwen Lei. 2020. An Efficient Revocable Attribute-Based Signcryption Scheme With Outsourced Unsigncryption in Cloud Computing. *IEEE Access* 8 (2020), 42805–42815.

[19] Drone Remote-ID. 2023. Tech specifications for Remote ID range. https://drone-remote-id.com/#tech_specifications_for_remote_id_range. (Accessed: 2024-Feb-26).

[20] Drone Remote-ID. 2023. Where can I buy a Remote ID module? How much does it cost? https://drone-remote-id.com/#what_is_remote_id?. (Accessed: 2024-Feb-26).

[21] DroneXL. 2023. DJI Addresses Concerns of Drone Pilots Location Being Public. https://dronexl.co/2023/03/09/dji-concerns-drone-pilots-locations-public/. (Accessed: 2024-Feb-26).

[22] Espressif Semiconductors. 2023. ESP32 Drone. https://docs.espressif.com/projects/espressif-esp-drone/en/latest/index.html. (Accessed: 2024-Feb-26).

[23] Espressif Semiconductors 2023. *ESP32WROOM32 Datasheet*. Espressif Semiconductors. Rev. 3.4.

[24] Federal Aviation Administration. 2023. UAS Remote Identification. https://www.faa.gov/uas/getting_started/remote_id. (Accessed: 2024-Feb-26).

[25] International Organization for Standardization. 2018. ISO IEC 14888-3:2018, IT Security techniques — Digital signatures with appendix — Part 3: Discrete logarithm based mechanisms. https://www.iso.org/standard/76382.html. [Accessed 2023-Nov-29].

[26] Steven D. Galbraith, Kenneth G. Paterson, and Nigel P. Smart. 2008. Pairings for cryptographers. *Discrete Applied Mathematics* 156, 16 (2008), 3113–3121. Applications of Algebra to Cryptography.

[27] Siva Ganesh Ganti. 2023. Open source code of the proof-of-concept of SNELL on the ESP32. https://github.com/ganeshteja/SNELL/. (Accessed: 2024-Feb-26).

[28] Romain Gay, Lucas Kowalczyk, and Hoeteck Wee. 2018. Tight adaptively secure broadcast encryption with short ciphertexts and keys. In *Security and Cryptography for Networks: 11th International Conference, SCN 2018*. Springer, Amalfi, Italy, 123–139.

[29] Liyao Han, Yijie Xun, Jiajia Liu, Abderrahim Benslimane, and Yanning Zhang. 2023. DP-Authentication: A novel deep learning based drone pilot authentication scheme. *Ad Hoc Networks* (2023), 103180.

[30] HeliGuy. 2023. DJI Matrice 300 RTK. https://www.heliguy.com/collections/dji-matrice-300. (Accessed: 2024-Feb-26).

[31] Hellmann. 2021. DRONAMICS and Hellmann plan pan-European transport services with cargo drones from 2022. https://www.hellmann.com/en/news/2021/dronamics-and-hellmann-plan-pan-european-transport-services-cargo-drones-2022. (Accessed: 2024-Feb-26).

[32] Tharaka Hewa, An Braeken, Madhusanka Liyanage, and Mika Ylianttila. 2022. Fog computing and blockchain-based security service architecture for 5G industrial IoT-enabled cloud manufacturing. *IEEE Transactions on Industrial Informatics* 18, 10 (2022), 7174–7185.

[33] Jianchang Lai, Yi Mu, Fuchun Guo, Willy Susilo, and Rongmao Chen. 2017. Fully privacy-preserving and revocable ID-based broadcast encryption for data access control in smart city. *Personal and Ubiquitous Computing* 21 (2017), 855–868.

[34] Jiguo Li, Wei Yao, Jinguang Han, Yichen Zhang, and Jian Shen. 2017. User collusion avoidance CP-ABE with efficient attribute revocation for cloud storage. *IEEE Systems Journal* 12, 2 (2017), 1767–1777.

[35] Lumenier. 2022. QAV-R FPV Racing Quadcopter. https://www.lumenier.com/consumer/qav-r. (Accessed: 2024-Feb-26).

[36] Mriganka Mandal. 2020. Privacy-preserving fully anonymous ciphertext policy attribute-based broadcast encryption with constant-size secret keys and fast decryption. *Journal of Information Security and Applications* 55 (2020), 102666.

[37] Cedric Mesnil. 2023. ECPy 1.2.5. https://pypi.org/project/ECPy/. (Accessed: 2024-Feb-26).

[38] Marino Miculan and Nicola Vitacolonna. 2023. Automated verification of Telegram's MTProto 2.0 in the symbolic model. *Computers & Security* 126 (2023), 103072.

[39] MIRACL. 2022. MIRACL Core. https://github.com/miracl/core. (Accessed: 2024-Feb-26).

[40] Elena Pagnin, Gunnar Gunnarsson, Pedram Talebi, Claudio Orlandi, and Andrei Sabelfeld. 2019. TOPPool: Time-aware Optimized Privacy-Preserving Ridesharing. *Proceedings on Privacy Enhancing Technologies* 2019, 4 (2019), 93–111.

[41] Anh Pham, Italo Dacosta, Bastien Jacot-Guillarmod, Kévin Huguenin, Taha Hajar, Florian Tramèr, Virgil Gligor, and J-P Hubaux. 2017. Privateride: A privacy-enhanced ride-hailing service. *Proceedings on Privacy Enhancing Technologies* 2017, 2 (2017), 38–56.

[42] Raspberry Pi. 2023. Raspberry Pi 4 Computer Model B Product Brief. https://static.raspberrypi.org/files/product-briefs/Raspberry-Pi-4-Product-Brief.pdf. (Accessed: 2024-Feb-26).

[43] Tedeschi Pietro, Sciancalepore Savio, and Di Pietro Roberto. 2023. Lightweight Privacy-Preserving Proximity Discovery for Remotely-Controlled Drones. In *Proceedings of the 39th Annual Computer Security Applications Conference* (, Austin, TX, USA,) *(ACSAC '23)*. Association for Computing Machinery, New York, NY, USA, 178–189.

[44] R. Moskowitz, S. Card, A. Wiethuechter. 2022. *UAS Operator Privacy for RemoteID Messages - draft-moskowitz-drip-operator-privacy-11*. Technical Report. IETF.

[45] Y. Sreenivasa Rao. 2017. A secure and efficient Ciphertext-Policy Attribute-Based Signcryption for Personal Health Records sharing in cloud computing. *Future Generation Computer Systems* 67 (2017), 133–151.

[46] Reuters. 2021. DHL aims to deploy longer distance drones to beat stretched supply lines. https://www.reuters.com/technology/dhl-aims-deploy-longer-distance-drones-beat-stretched-supply-lines-2021-07-19/. (Accessed: 2024-Feb-26).

[47] Doreen Riepel and Hoeteck Wee. 2022. FABEO: Fast Attribute-Based Encryption with Optimal Security. In *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security* (Los Angeles, CA, USA) *(CCS '22)*. Association for Computing Machinery, New York, NY, USA, 2491–2504.

[48] Doreen Riepel and Hoeteck Wee. 2023. FABEO: Fast Attribute-based Encryption with Optimal Security. https://github.com/DoreenRiepel/FABEO. (Accessed: 2024-Feb-26).

[49] S. Card, A. Wiethuechter, R. Moskowitz, A. Gurtov. 2022. *Drone Remote Identification Protocol (DRIP) Requirements and Terminology*. Technical Report. IETF.

[50] S. Card, A. Wiethuechter, R. Moskowitz, S. Zhao, A. Gurtov. 2022. *Drone Remote Identification Protocol (DRIP) Architecture - draft-ietf-drip-arch-24*. Technical Report. IETF.

[51] Schiller, Nico and Chlosta, Merlin and Schloegel, Moritz and Bars, Nils and Eisenhofer, Thorsten and Scharnowski, Tobias and Domke, Felix and Schönherr, Lea and Holz, Thorsten. 2023. Drone Security and the Mysterious Case of DJI's DroneID, In NDSS. *Network and Distributed System Security Symposium (NDSS)*.

[52] Hynek Schlawack. 2023. argon2-cffi 21.3.0. https://pypi.org/project/argon2-cffi/. (Accessed: 2024-Feb-26).

[53] Savio Sciancalepore and Dominik Roy George. 2022. Privacy-Preserving Trajectory Matching on Autonomous Unmanned Aerial Vehicles. In *Proceedings of the 38th Annual Computer Security Applications Conference*. 1–12.

[54] Yannick Seurin. 2012. On the Exact Security of Schnorr-Type Signatures in the Random Oracle Model. In *Advances in Cryptology – EUROCRYPT 2012*, David Pointcheval and Thomas Johansson (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 554–571.

[55] Statista. 2022. Consumer drone unit shipments worldwide from 2020 to 2030. https://www.statista.com/statistics/1234658/worldwide-consumer-drone-unit-shipments/. (Accessed: 2024-Feb-26).
[56] Pietro Tedeschi. 2023. Open source code of the proof-of-concept on the Lumenier QAV-R and ProVerif security analysis of SNELL. https://github.com/pietrotedeschi/snell/. (Accessed: 2024-Feb-26).
[57] Pietro Tedeschi, Fatima Ali Al Nuaimi, Ali Ismail Awad, and Enrico Natalizio. 2024. Privacy-Aware Remote Identification for Unmanned Aerial Vehicles: Current Solutions, Potential Threats, and Future Directions. *IEEE Transactions on Industrial Informatics* 20, 2 (2024), 1069–1080.
[58] Pietro Tedeschi, Savio Sciancalepore, and Roberto Di Pietro. 2021. ARID: Anonymous Remote IDentification of Unmanned Aerial Vehicles. In *Annual Computer Security Applications Conference* (Virtual Event, USA) *(ACSAC '21)*. Association for Computing Machinery, New York, NY, USA, 207–218.
[59] Pietro Tedeschi, Savio Sciancalepore, and Roberto Di Pietro. 2023. PPCA - Privacy-Preserving Collision Avoidance for Autonomous Unmanned Aerial Vehicles. *IEEE Transactions on Dependable and Secure Computing* 20, 2 (2023), 1541–1558.
[60] Ubuntu. 2022. Focal Fossa 20.04 LTS. https://releases.ubuntu.com/20.04/. (Accessed: 2024-Feb-26).
[61] Qian Wang, Chengzhe Lai, Gang Han, and Dong Zheng. 2023. pdRide: Privacy-Preserving Distributed Online Ride-Hailing Matching Scheme. *IEEE Transactions on Intelligent Transportation Systems* (2023).
[62] Eva Wisse, Pietro Tedeschi, Savio Sciancalepore, and Roberto Di Pietro. 2023. A^2RID-Anonymous Direct Authentication and Remote Identification of Commercial Drones. *IEEE Internet of Things Journal* (2023), 1–1.
[63] Haomiao Yang, Qixian Zhou, Mingxuan Yao, Rongxing Lu, Hongwei Li, and Xiaosong Zhang. 2019. A Practical and Compatible Cryptographic Solution to ADS-B Security. *IEEE Internet of Things Journal* 6, 2 (2019), 3322–3334.
[64] Yinghui Zhang, Robert H Deng, Shengmin Xu, Jianfei Sun, Qi Li, and Dong Zheng. 2020. Attribute-based encryption for cloud computing access control: A survey. *ACM Computing Surveys (CSUR)* 53, 4 (2020), 1–41.
[65] Jun Zhou, Zhenfu Cao, Zhan Qin, Xiaolei Dong, and Kui Ren. 2019. LPPA: Lightweight privacy-preserving authentication from efficient multi-key secure outsourced computation for location-based services in VANETs. *IEEE Transactions on Information Forensics and Security* 15 (2019), 420–434.

A NOTATION TABLE

Notation	Description
d_n	Generic Drone
$\mathcal{R}$	Generic Receiver
$(\mathsf{ssk}_n, \mathsf{spk}_n)$	Schnorr Private and Public Keys of d_n
ID_n	ID of $d_n/\mathcal{R}$
C_n	Public-key Certificate of d_n or the Generic Receiver
$(\mathsf{msk}_A, \mathsf{mpk}_A)$	USS Master Private and Public Key pair
α	Random secret nonce, aka the value of the Master Secret Key
p	Group order
$(\mathbb{G}_1, \mathbb{G}_2, \mathbb{G}_T)$	Asymmetric (Type III) prime-order bilinear groups
e	Pairing operation
g_1, g_2	Group generators for $\mathbb{G}_1$ and $\mathbb{G}_2$, respectively
$\mathcal{H}$	Hashing function
u	Generic attribute
T	Time interval
$r \in \mathbb{G}_T$	Random nonce
$\mathcal{P}$	Attribute Policy
c	Ciphertext of the CP-ABE encryption algorithm CP_FABEO.Encrypt
K_t	Ephemeral Symmetric Key
KDF	Generic Key Derivation Function
t	Current timestamp
$lat_{n,t}, lon_{n,t}, alt_{n,t}$	GNSS location, expressed as latitude, longitude, and altitude of d_n
$v_{x,n,t}, v_{y,n,t}, v_{z,n,t}$	Instantaneous speed of d_n on the three axis x, y, z at time t
$\mathcal{D}_p$	Drone position (and speed)
$\mathcal{O}_p$	Pilot location
$m_{n,t}$	RID message transmitted by d_n
C	Encrypted Pilot Location with AES.ENC and the symmetric key K_t
ec	Emergency Code
σ_m	Schnorr message signature
$\tilde{r}$	Recovered ephemeral nonce using the CP-ABE decryption algorithm CP_FABEO.Decrypt
$attr\_set_{\mathcal{R}}$	Attribute set possessed by the observer $\mathcal{R}$

Table 2: Notation and brief description.

B ACRONYMS TABLE

Acronym	Description
RID	Remote ID
FAA	Federal Aviation Administration
CP-ABE	Ciphertext-Policy Attribute-Based Encryption
WG	Working Group
IETF	Internet Engineering Task Force
drip	Drone Remote Identification Protocol
MSP	Monotone Span Programs
EC	Elliptic Curve
USS	Unmanned Service Supplier
GCS	Ground Control Station
GNSS	Global Navigation Satellite System
RF	Radio Frequency
TA	Trusted Authority
PvIR	Private Information Registry
PbIR	Public Information Registry
GPO	Generic Public Observer
PSO	Public Safety Observer
ABE	Attribute-Based Encryption
TLS	Transport Layer Security
CA	Certification Authority
KDF	Key Derivation Function
AES	Advanced Encryption Standard
SoC	System on Chip
CBOR	Coincise Binary Object Representation
DRAM	Dynamic Random Access Memory
MTU	Maximum Transmission Unit
ADS-B	Automatic Dependent Surveillance - Broadcast
VANET	Vehicular Ad Hoc Network
ICAO	International Civil Aviation Organization
FPE	Format-Preserving Encryption
MAC	Medium Access Control
RSUs	Road-Side Unit
ABS	Attribute-Based Signcryption

Table 3: Acronyms and brief description.

C *SNELL* PACKETS FORMAT ON LUMENIER QAV-R

Field	Size [B]	Description
D_{ID}	4	Drone ID.
D_{LAT}	4	Drone Latitude.
D_{LON}	4	Drone Longitude.
D_{ALT}	4	Drone Altitude.
D_{VEL}	4	Drone Speed.
D_{COG}	4	Drone Course Over Ground.
TS	4	Message Timestamp.
ES	1	Emergency Code.
ENC	[904 −2253]	SNELL encrypted ground station location C, [1 - 22] attributes.
SIG	[64,80,98]	Schnorr signature σ_m, based on used EC (`secp256k1`, `secp384r1`, `secp521r1`).

Table 4: SNELL message payload notation in our implementation on Lumenier QAV-R.

D *SNELL* PACKETS FORMAT ON ESP32

Field	Size [B]	Description
HDR	22	Data Frame Header
D_{ID}	4	Drone ID.
D_{LAT}	4	Drone Latitude.
D_{LON}	4	Drone Longitude.
D_{ALT}	4	Drone Altitude.
D_{VEL}	4	Drone Speed.
D_{COG}	4	Drone Course Over Ground.
TS	4	Message Timestamp.
ES	1	Emergency Code.
ENC	[16 - 1385]	SNELL encrypted ground station location C, [1 - 22] attributes.
SIG	64	Schnorr signature σ_m, based on `secp256k1`.

Table 5: SNELL message payload notation in our implementation on the ESP32.