Department of Mathematics and Computer Science
Security Research Group

Implementation and Performance Assessment of a Protocol for Confidential Disclosure of Pilot Location on a Constrained Device

Master Thesis report

S.G.T Ganti

Supervisors:

Dr. Savio Sciancalepore
Eindhoven University of Technology (TU/e)
Department of Mathematics and Computer Science

Dr. Pietro Tedeschi
Technology Innovation Institute
Autonomous Robotics Research Center

Dr. Habib Mostafaei
Eindhoven University of Technology (TU/e)
Department of Mathematics and Computer Science

Eindhoven, August 2023

Abstract

Unmanned Aircraft Systems (UAS), commonly known as drones, have been increasingly used in the last years, raising concerns for regulatory authorities across the globe, like the Federal Aviation Administration (FAA) and the European Union Aviation Safety Agency (EASA). To safely integrate UASs into the National Airspace System, these authorities introduced rules for the Remote Identification of unmanned aircraft (RemoteID), requiring UASs to periodically broadcast their location and identity information. While RemoteID enhances situational awareness and airspace management, it also poses a privacy risk. In particular, the transmission of ground control station (GCS) location data in plain text can lead to the compromise of the privacy and anonymity of drone pilots and their operations. This has resulted in significant opposition towards the RemoteID system, hindering its proper acceptance and implementation. This thesis aims to address some of the privacy concerns of RemoteID by developing and implementing a SelectivePilot Location Disclosure mechanism for RemoteID systems, namely SNELL. The proposed system offers selective encryption for the pilot location, focusing on protecting the privacy of the GCS location data while meeting the FAA's requirement of the message being transmitted in a single packet once every 1 to 3 seconds for RemoteID messages. By employing SNELL, the system ensures secure and fine-grained access control for the pilot's sensitive location information. The implementation and evaluation of the SNELL-based system's efficiency are performed on the ESP32 microcontroller platform, focusing on three different implementation strategies targeting security, execution time, energy consumption, and memory usage. Our results demonstrate that the system meets the FAA's RemoteID message requirements while effectively safeguarding the privacy of GCS location data.

In the first version, the "Fully Precomputed mode", all the cryptographic computations are performed on an unconstrained device, and the data is stored on the micro-controller. This results in a lower execution time and energy consumption but has a high usage of memory. The second version, "Partially Precomputed mode" has all the computations running on the micro-controller itself with some parts precomputed before the operation. This takes more time and energy than the previous mode but is more secure. In the third version, "Parallel Computed mode", the dual-core option on the microcontroller is used to speed up the computation time while also increasing the energy consumption. This mode is the most secure of all the three as there is no cryptographic data saved on the microcontroller. The maximum execution time of all three implementations is 2960ms (less than 3sec) and a maximum energy consumption is 795.5mJ, i.e., $0.33 \times 10^{-2}\%$ of the battery (7.4v 900mAh) powering such device.

Contents

List of Figures

List of Tables

List of acronyms

ABBE	Attribute Based Broadcast Encryption
ABE	Attribute Based Encryption
AES	Advanced Encryption Standard
ASD	AeroSpace and Defence Industries Association of Europe
ASTM	American Society for Testing and Materials
BLS	Barreto-Lynn-Scott
BN	Barreto-Naehrig
BVLOS	beyond visual line of sight
CAA	Civil Aviation Authorities
CBC	Cipher Block Chaining
CEN	European Committee for Standardization
CFB	Cipher Feedback
CP-ABE	Cipher-text Policy ABE
CTR	Counter
DRI	Direct Remote ID
DRAM	Data RAM
DRIP	Drone Remote ID Protocol
DSO	Digital Signal Oscilloscope
EASA	European Union Aviation Safety Agency
ECB	Electronic Code Book
ECC	Elliptic Curve Cryptography
ESP-IDF	Espressif IoT Development Framework
FAA	Federal Aviation Administration
FABEO	Fast Attribute-Based Encryption with Optimal Security
FRIA	FAA-recognized identification areas
GCS	Ground Control Station

GCM	Galois/Counter Mode
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
IBBE	Identity Based Broadcast Encryption
IDE	Integrated Development Environment
IETF	Internet Engineering Task Force
IoT	Internet of Things
IRAM	Instruction RAM
KP-ABE	Key Policy ABE
KDF	Key Derivation Function
MAC	message authentication codes
MSP	Monotone Spanning Program
MTU	Maximum Transmission Unit
OFB	Output Feedback
PBC	Pairing Based Cryptography
RAM	Random Access Memory
RNG	Random Number Generator
RID	Remote Identification
RF	Radio Frequency
Remote ID	Remote Identification
SNELL	Selective autheNticatEd piLot Location disclosure
TA	Trusted Authority
TBD	To Be Determined
UAS	Unmanned Aircraft Systems
UAV	Unmanned Areal Vehicle
USS	Unmanned Service Supplier
Wi-Fi	Wireless Fidelity

Chapter 1

Introduction

Drones and other Unmanned Aircraft Systems (UAS) have become increasingly popular and are now widely used in various industries. Drones have a number of benefits, including the capacity to assist with dangerous tasks, delivery, in-person exploration of remote areas, conflict resolution, agriculture, etc. [1]. Drones have also taken on a central role in the operations of many corporations and governmental bodies, and they have been able to break through barriers that prevented some industries from developing or from keeping up with others. Sectors like Military, Commercial, Personal, and Future Technology are adopting them globally [2]. However, with the growing number of drones in the skies, there is an ever increasing need to identify them and account their operations. To address these concerns, governments around the world have introduced various regulations, such as Remote Identification (Remote ID).

Remote ID is a system that allows a UAS to transmit essential information, including its identity, location, altitude, and take-off location, to other parties over wireless channels. Remote ID is now a mandatory requirement for most drones in major countries worldwide, with the deadline for compliance set by the Federal Aviation Administration (FAA) for September 16, 2023 in the United States. The primary goal of Remote ID is to ensure the safety and security of drone operations by promoting accountability for drone pilots and preventing the misuse of drone technologies.

In addition to its primary purpose, Remote ID can also facilitate the avoidance of collisions and conflicts between drones and other aircraft or obstacles by providing situational awareness and deconfliction tools [17]. Moreover, Remote ID can enable more advanced and complex drone operations, such as beyond visual line of sight (BVLOS), package delivery, urban air mobility, or swarm formations by providing trust and transparency.

However, like any new regulation, Remote ID also raises some issues and concerns. Some existing drones may not have the necessary hardware to support Remote ID, requiring pilots to either upgrade their drones or add additional hardware, which can be both time-consuming and expensive. Furthermore, some drone owners may be hesitant to share their drone's information with untrusted parties or authorities who may misuse it or track them down. Additionally, there have been incidents of violence against drones and drone pilots [25] [68] [31], and broadcasting sensitive data about both the drone and its pilot might increase such episodes. Malicious actors may also attempt to intercept, spoof, or jam Remote ID signals, potentially causing accidents, disruptions, or even deliberate attacks.

Till date, there are very few works in the literature that aim at preserving the privacy of pilot location in Remote ID. Hence, to address concerns about privacy and authenticity issues in Remote ID, we propose investigating, designing, deploying, and testing a system that ensures confidential disclosure of the pilot's location by building upon the existing specifications and standards. This system, namely, SNELL (acronym for Selective autheNticatEd pilot Location discLosure) could help allay concerns about pilot location privacy while still providing the benefits of Remote ID for public safety and advanced drone operations.

The rest of the report is organized as follows: Chapter 2 provides background information on

the relevant regulations by the Civil Aviation Authorities (CAA)s, as well as available specifications and standards, and cryptographic technologies that are relevant to the work. Chapter 3 explores the scenario and adversary model, as well as use cases, to extract the requirements and features necessary for the project. In Chapter 4, a comparison table is presented that compares similar works to find gaps in the research. The chapter also discusses the main research question and the supporting sub-research questions.

Chapter 5 provides the implemented SNELL protocol structure explaining the cryptographic processes involved. Chapter 6 discusses the requirements and reasoning for the hardware selection for this thesis project. The packet structure along with the software and libraries used in the project are also present in this chapter.

In Chapter 7, the results from the performance evaluation of the various implementation strategies are presented along with relevant discussion. This includes a comparison of the execution times and energy consumption of the implementation modes. Finally, Chapter 8 summarises the results of the thesis and highlights future research opportunities.

Chapter 2

Preliminaries

This chapter discusses the existing mandatory and proposed regulations and specifications by the Civil Aviation Authorities and other groups' works related to Remote ID. Later, the cryptographic technologies used in the implementation are introduced and discussed.

2.1 Civil Aviation Authorities

The US-based FAA has issued a final rule (January 15, 2021) [30] on Remote ID which requires most drones operating in the US airspace to have remote identification capabilities. According to this, there are 3 ways to meet the Remote ID requirements (see Fig. 2.1):

1. Operating a **standard Remote ID drone** (built-in Remote ID capabilities).
2. Operating a drone with a Remote ID **broadcast module**.
3. Operate without Remote ID requirement.

The first way is to use new drones that are manufactured with built-in Remote ID capability in accordance with the Remote ID requirements. The second talks about existing drones that are retrofitted with a broadcast module. In the final one, pilots can fly their drones without any need for Remote ID. This can be done in special FAA-recognized identification areas (FRIA) that are sponsored by community-based organizations or educational institutions. While the second and third methods are restricted to Line-Of-Sight (LOS), the first method has none. In the first method (standard Remote ID drone), the FAA talks about broadcasts containing the pilot or Ground Control Station (GCS) location and elevation while in the second method (broadcast module) this is changed to the take-off location and elevation. Also, Standard Remote ID drones are required to send Emergency status. The rule mentions the use of Wi-Fi and Bluetooth technologies for the broadcasts. Other CAA across the globe also created similar rules for UAS Remote Identification (RID) [36]. Japanese Civil Aviation Bureau has released a document on the requirements for remote ID devices and applications[21]. The European Union Aviation Safety Agency (EASA) has published Commission Delegated Regulation [7] and Commission Implementing Regulation [6]. CAAs currently issue performance-based regulations that do not require specific methodologies but instead mention industry-consensus technical standards as viable means of compliance. In general, the FAA and EASA proposed two main modes of the RID:

1. Network-based RID: UAS transmits data via the internet. available globally.
2. Broadcast RID: UAS transmits data locally one-way via Bluetooth or Wi-Fi.

The Network-based RID messages are sent by the Unmanned Areal Vehicle (UAV) via the Internet to a Unmanned Service Supplier (USS) that processes these messages for further usage. FAA mentions in its final rule [30] that the implementation and use of Network-based RID mode is

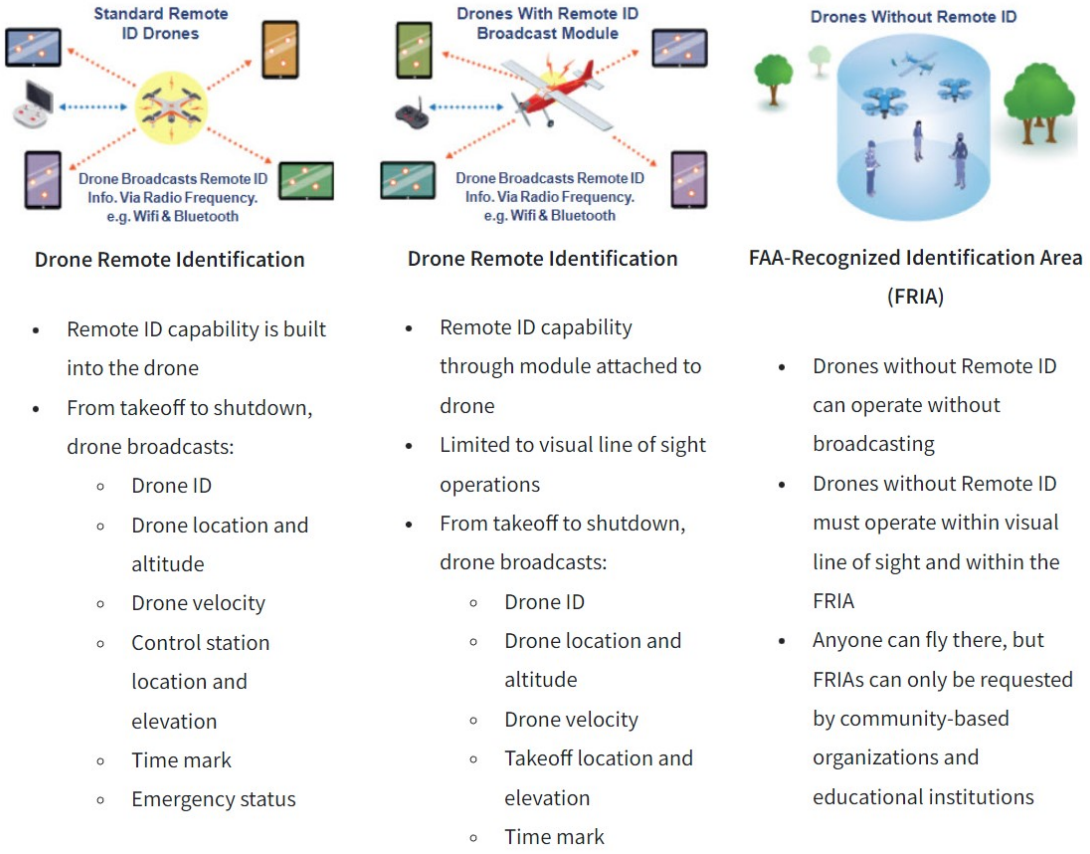


Figure 2.1: The 3 ways Drone Pilots can meet Remote ID rule from FAA final rule [30]

eliminated due to various drawbacks and detriments. On the other hand, EASA still has the option to use Network based RID mode. Further, this mode is made manadatory to use inside special zones called U-spaces [22]. The Broadcast RID messages can be broadcasted over Bluetooth (BT) 4.0 or 5.0, Wi-Fi beacon (802.11), or Wi-Fi Neighbourhood aware Networking (Wi-Fi NAN). On the receiving end of the broadcast, the specification expects a user with a smartphone that is capable of receiving the BT and Wi-Fi signals. From practical tests, it is observed that Wi-Fi based transmissions gave a higher range value while also providing larger packet sizes.

2.2 Standards and Specifications

2.2.1 ASTM specification

Based on the rules of the Civil Aviation Authorities (CAAs), ASTM International (formerly American Society for Testing and Materials (ASTM)) has created a work item that resulted in the standard "Standard Specification for Remote ID and Tracking" [10]. This standard, also known as "F3411," defines the performance minimums for Remote ID in compliance with the regulations. It specifies the type of data and the format to be broadcasted, covering various message types and formats that carry information about the entire drone operation and transmission methods. The message types involved in both modes of transmission (Network-based and Broadcast) are listed in Fig. 2.2. Based on the type of message, the specification gives several aspects such as: broadcast frequency, mandatory and optional fields, public and private information, encryption and authentication, etc.. The relevant information for the project is mentioned in the later chapters. On the issue of privacy, the specification describes some methods like obfuscation of pilot location but the

Msg Type	Message Name	Purpose
0x0	Basic ID Message	Provides ID for UA, characterizes the type of ID, and identifies the type of UA
0x1	Location/Vector Message	Provides location, altitude, direction, and speed of UA
0x2	Authentication Message[A]	Provides authentication data for the UA
0x3	Self-ID Message[A]	Message that can be used by Operators to identify themselves and the purpose of an operation
0x4	System Message[A]	Includes Remote Pilot location and multiple aircraft information (group) if applicable, and additional system information
0x5	Operator ID[A]	Provides Operator ID
0xF	Message Pack[A]	A payload mechanism for combining the messages above into a single message pack. Used with Bluetooth Extended Advertising, Wi-Fi Neighbor Awareness Network, and Wi-Fi Beacon

Figure 2.2: Message Types mentioned in the F3411-22a specification [10]

methods to perform this are not discussed.

2.2.2 ASD-STAN Standardization

The Aerospace and Defence Industries Association of Europe - Standardization is an industrial non-profit association that develops and maintains European standards for the European aerospace and defence industry. ASD-STAN acts as the technical body of European Committee for Standardization (CEN) for Aerospace and is responsible for developing a set of standards for unmanned aircraft systems (UAS) under the "D05/WG08 UAS" Working Group. These standards aim to ensure compliance with regulatory requirements and a high and uniform level of safety for UAS pilots. The prEN 4709-002 Direct Remote ID (DRI) standard developed by ASD-STAN under the D05/WG08 UAS Working Group addresses the requirements of the UAS/direct remote ID (add-on or onboard). The European UAS Digital Remote ID Technical Standard discussed in the document provides a common framework for the identification and tracking of UAS in Europe and includes tables outlining the various communication technologies and identification messages used in the standard. A white paper published by ASD-STAN on DRI [9] mentions the inclusion of ASTM F3411-19 in the standard and gives a detailed explanation of the compatibility with the ASTM standard for Remote ID.

2.2.3 IETF Drone Remote ID Protocol (DRIP) working group

In response to the FAA's rule on RID, the Internet Engineering Task Force (IETF) assembled a group called the Drone Remote ID Protocol (DRIP) working group to further define the requirements, create policies and architectures to support the RID as the FAA's rule does not specify on the methodology for this.

The DRIP working group's charter [81] mentions that its goal is to specify how RID can be made trustworthy and available in both Internet and local-only connected scenarios, especially in emergency situations. It further mentions that the specifications produced by the group will need to balance public safety authorities' need to know trustworthy information with UAS pilots' and other involved parties' privacy. The group is working on the following.

- Technical Requirements for applying IETF protocols to the RID
- Architecture that addresses the technical requirements by attempting to reuse existing IETF standardized protocols or architectures.
- Designing new protocols or extending existing ones to fit the UAS environment and standardizing them.

A document for each of the categories mentioned above is released. RFC9153 [19] defines the terminology and requirements for the solutions produced by the DRIP Working Group. The main types of requirements that are mentioned in the document are:

1. General
2. Identifier

3. Privacy
4. Registries

Of all the categories mentioned, all the Privacy requirements and some of the requirements from the rest of the categories are relevant to the current project. The Privacy requirements are discussed here.

- PRIV-1 Confidential Handling: MUST enable confidential handling of information assigned as private by the user.
- PRIV-2 Encrypted Transport: MUST enable selective strong encryption of private data and support configurable disabling of encryption
- PRIV-3 Encrypted Storage: SHOULD facilitate selective strong encryption of private data at rest
- PRIV-4 Public/Private Designation: SHOULD facilitate designation of which information is public and private.
- PRIV-5 Pseudonymous Rendezvous: MAY enable mutual discovery of and communication among the UAS operators without revealing pilot identity or location.

The rationale provided in the document mentions that these requirements are only for DRIP and not for either F3411[10] or UAS RID in any specific jurisdiction.

2.3 Encryption

Encryption is a fundamental cryptographic technique used to protect data from unauthorized access during transmission or storage. It involves transforming plaintext (readable data) into ciphertext (encrypted data) using cryptographic algorithms and keys.

2.3.1 Symmetric Encryption

Symmetric encryption employs a single shared secret key for both the encryption and decryption processes. This same key is used by both the sender (encryptor) and the recipient (decryptor) to protect and access the information. The process involves applying cryptographic algorithms that utilize substitution and permutation techniques to convert plaintext into ciphertext. One of the most widely used symmetric encryption algorithms is the Advanced Encryption Standard (AES), renowned for its high security and efficiency. AES operates on fixed-size blocks of data (128, 192, or 256 bits) and employs corresponding key sizes.

While symmetric encryption is efficient and fast, it does face challenges, particularly in key distribution. This is especially pertinent in scenarios involving multiple communicating parties. It's important to note that, in some cases, the keys in symmetric encryption can also be pre-shared, simplifying the key distribution process. However, as the number of users increases, distributing and managing secret keys securely becomes a critical concern. To address this, hybrid encryption systems leverage both symmetric and asymmetric encryption techniques. In such systems, symmetric encryption secures the actual data, while asymmetric encryption is employed to protect and securely distribute the symmetric keys. This combination offers an effective solution to the challenges posed by symmetric encryption alone.

2.3.2 Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES), or Advanced Encryption Standard, is a widely used symmetric encryption algorithm that ensures the confidentiality and security of data. It was selected by the National Institute of Standards and Technology (NIST) in 2001 to replace the ageing Data Encryption Standard (DES) as a more secure and efficient encryption algorithm [53] .

Encryption Process

AES operates on fixed-size blocks of 128 bits and supports key lengths of 128, 192, or 256 bits. The encryption process involves multiple rounds of mathematical transformations, including byte substitution using a fixed table (S-box), row shifting, and column mixing. The number of rounds varies with the key length: 10 rounds for 128-bit keys, 12 rounds for 192-bit keys, and 14 rounds for 256-bit keys.

Security Level

The security level of AES is determined by the key length. AES-128 offers a security level of 128 bits, AES-192 provides 192 bits, and AES-256 provides 256 bits of security. These key lengths are considered computationally infeasible to break through brute-force attacks. In our implementation, we opt for the highest security level of 256 bits by using AES-256.

AES Modes

AES supports different modes of operation [23], each offering specific advantages for particular use cases. The Electronic Code Book (ECB) mode performs independent block encryption, but it is vulnerable to identical plaintext blocks producing identical ciphertext. The Cipher Block Chaining (CBC) mode introduces feedback by making each block's encryption dependent on the previous ciphertext block, making it suitable for securing large data. The Cipher Feedback (CFB) mode operates on smaller units and allows encryption of arbitrary-length messages. The Output Feedback (OFB) mode generates a key stream independent of the plaintext, making it suitable for stream ciphers. The Counter (CTR) converts AES into stream ciphers, enabling parallel encryption/decryption and making it well-suited for high-performance scenarios. The Galois/Counter Mode (GCM) mode combines AES encryption with authentication, providing both confidentiality and data integrity. GCM operates in a way similar to CTR mode but also includes authentication tags for verifying data authenticity. Each mode is chosen based on desired security properties and application requirements.

As we mention in the later discussions, we only have to encrypt the data of one block size (16 bytes), so, the choice of the mode becomes less crucial in terms of security, as long as the key used is strong. The main concern with using a single block of plaintext is that there is no opportunity for the chaining effect (in CBC mode) or the unique counter value (in CTR mode) to come into play, which is where these modes derive their strength when encrypting multiple blocks of data. Hence for our implementation, we use the ECB mode.

2.3.3 Asymmetric Encryption (Public-key Encryption)

Asymmetric encryption, also known as public-key cryptography, uses a pair of mathematically related keys: a public key and a private key. The public key is made available to everyone and can be used by other parties to encrypt data destined for the owner of the corresponding private key. The private key is kept secret and can be used to decrypt data encrypted with the associated public key.

Asymmetric encryption schemes provide essential security features such as digital signatures and key exchange protocols. Digital signatures allow a sender to sign a message using their private key, providing authenticity and non-repudiation. Recipients can verify the signature using the sender's public key, ensuring the message's integrity and origin.

In the context of drone remote identification systems, both symmetric and asymmetric encryption play vital roles. Symmetric encryption ensures the confidentiality of sensitive data transmitted between drones and ground stations, while asymmetric encryption enables secure key exchange and message authentication.

In the subsequent sections, we will delve into advanced cryptographic techniques, including Elliptic Curve Cryptography (ECC) and Pairing Based Cryptography (PBC), which offer unique advantages and are relevant to the scope of this thesis.

2.4 Elliptic Curve Cryptography (ECC)

Elliptic Curve Cryptography (ECC) is an asymmetric cryptographic technique that leverages the mathematics of elliptic curves to provide robust security with shorter key lengths compared to traditional asymmetric algorithms like RSA. ECC has gained significant attention due to its efficiency and effectiveness, making it particularly well-suited for resource-constrained environments, such as drone communication systems.

The foundation of ECC lies in the mathematical properties of elliptic curves, which are defined by an equation of the form: $y^2 = x^3 + ax + b$. The curve's points and arithmetic operations are defined over a finite field, creating a cyclic group. The difficulty of solving the Elliptic Curve Discrete Logarithm Problem (ECDLP) [32, 34] forms the basis of ECC's security.

In ECC, each participant in the communication process possesses a pair of keys: a private key and a corresponding public key. The public key is derived from the private key and can be freely distributed. However, it is computationally infeasible to determine the private key from the public key alone, providing strong security.

Key generation, encryption, and decryption in ECC involve arithmetic operations on points over the elliptic curve. The efficiency of these operations makes ECC particularly attractive for resource-constrained devices like drones, as it allows for secure communication without a significant computational burden.

ECC offers the same level of security as traditional asymmetric algorithms like RSA but with much shorter key lengths. For example, a 256-bit ECC key is equivalent in security to a 3072-bit RSA key [12, 13], and 128-bit AES key (symmetric) [14]. The smaller key sizes lead to reduced computational overhead and lower memory requirements, making ECC an ideal choice for applications where resource efficiency is critical.

2.5 Pairing Based Cryptography

Pairing Based Cryptography (PBC) is an advanced cryptographic paradigm that exploits the mathematical concept of pairings, particularly the Weil [49] and Tate pairings [60], to enable new cryptographic functionalities. PBC is well-suited for applications that require efficient and secure solutions for identity-based encryption, secure multi-party computation, and attribute-based encryption.

A pairing is a bilinear map that maps elements from two distinct cyclic groups [38] on elliptic curves, denoted as $G1$ and $G2$, to elements in a third group GT. This mapping is represented as in Eq. 2.1.

$$e : G1 \times G2 \rightarrow GT. \tag{2.1}$$

where e is the pairing function.

Based on the groups $G1$ and $G2$ there are three types of pairings described in the literature:

1. $G1 = G2$
2. $G1 \neq G2$ but there is an efficiently computable homomorphism $\phi : G2 \rightarrow G1$
3. $G1 \neq G2$ and there are no efficiently computable homomorphisms between $G1$ anf $G2$.

When the groups $G1$ and $G2$ are the same, the pairing is called symmetric. These mappings need to fulfill the bilinearity and non-degeneracy properties [48] to be pairing groups. These properties make them useful in constructing cryptographic protocols with enhanced security features.

The bilinearity property of pairings allows for interesting interactions between the elements in $G1$ and $G2$ when mapped to GT. Specifically, for elements P and Q in $G1$ and $G2$, respectively, and scalar values a and b, the following equation Eq. 2.2 holds:

$$e(aP, bQ) = e(P, Q)^{ab}. \tag{2.2}$$

Another essential property of pairings is non-degeneracy mentioned in Eq. 2.3. Non-degeneracy ensures that the pairing function e is non-trivial and can distinguish between different elements. In other words, if P and Q are non-identity elements in $G1$ and $G2$, respectively, then $e(P, Q)$ is a non-trivial element in GT, and:

$$e(P, Q) \neq 1 \tag{2.3}$$

One of the significant applications of PBC is Identity-Based Encryption (IBE) [66]. In traditional public-key encryption schemes, a user's public key is represented by an arbitrary string. However, in IBE, a user's public key can be any unique identifier, such as an email address or a username. The private key is generated based on this identifier and a master secret held by a trusted authority. The PBC-based IBE scheme like the work in [18], offers a more user-friendly approach to encryption, simplifying key management and distribution.

Attribute-Based Encryption (ABE) [62] is yet another application of PBC. In ABE, access policies are associated with encrypted data, and access to the data is granted based on specific attributes or credentials possessed by the user. PBC-based ABE schemes provide fine-grained access control, making them valuable in scenarios where access restrictions must be tightly controlled.

In the context of drone remote identification, Pairing Based Cryptography can play a crucial role in enabling secure and efficient communication between drones and ground stations. By leveraging the unique properties of pairings, PBC can enhance the security and privacy of sensitive data exchanged during drone operations.

Security level

The security of PBC relies on the hardness of the computational problems on discrete logarithms in bilinear groups. [63] mentions that
" The security of pairing-friendly curves is evaluated by the hardness of the following discrete logarithm problems:

- The elliptic curve discrete logarithm problem (ECDLP) in $G1$ and $G2$
- The finite field discrete logarithm problem (FFDLP) in GT

"

Some of the popular Barreto-Lynn-Scott (BLS)-curves and Barreto-Naehrig (BN)-curves are shown in Tab. 2.1 [63, 11].

Curve Type	Security Level (Bits)	$\|G1\|$ (Bits)	$\|G2\|$ (Bits)	$\|GT\|$ (Bits)
BN254	100	512	1024	3072
BN462	128	928	1856	5568
BLS12-381	128	768	1536	4608
BLS12-461	128	928	1856	5568
BLS48-581	256	1168	2336	7008

Table 2.1: Security Levels and Bit Sizes of BN and BLS Curves in Pairing-Based Cryptography [63, 11]

While Pairing Based Cryptography introduces additional computational overhead compared to traditional cryptographic techniques, its ability to enable advanced cryptographic functionalities makes it a powerful tool for addressing various security challenges in modern communication systems.

2.6 Attribute Based Encryption

Attribute-based encryption (Attribute Based Encryption (ABE)) [62] is a type of encryption that allows fine-grained access control of encrypted data using authorization policies. It is a generalization of public-key encryption where the cipher text and user's secret key are determined by

attributes such as their email address, the country in which they live, and so on. In such a system, a cipher text can only be decrypted if the user key's set of attributes fulfill the access control policy associated with the cipher text.

ABE is classified into two main types: Cipher-text Policy ABE and Key Policy ABE. In Cipher-text Policy ABE, users possess sets of attributes, and cipher texts are associated with access control policies based on specific attribute sets required for decryption. On the other hand, Key Policy ABE takes the opposite approach, embedding policies within user keys while cipher texts are labeled with attributes. While Cipher-text Policy ABE is commonly discussed in the context of open scenarios where users can define precise access policies, the choice between CP-ABE and KP-ABE depends on the specific use case and requirements.

In a traditional one-to-one encryption scheme usually, there is only one entity that performs decryption using an associated private key. In a multi-entity scenario, this means that the data must be encrypted once for every entity (to be able to decrypt with its private key) present which creates a lot of overhead in terms of computation, storage, and transmission. ABE helps in such cases where a piece of data is only encrypted once and based on the attributes of the user, it can be decrypted.

Another advantage of ABE is the revocation aspect of access control. Traditional encryption does provide support to deal with cases where the private keys are compromised except for the generation and redistribution of new keys. Whereas, ABE has the option to revoke the access to users with certain attributes and thereby minimize the damage done. ABE also enables opportunistic decryption where the broadcaster and the receivers do not need to know each other and do not need an active two-way communication link between them to decrypt the message.

In a CP-ABE system, there is one sender and many receivers. Along with these, there is a Trusted Authority (TA) which maintains the parameters and keys required to run the system.

2.6.1 Fast Attribute-Based Encryption with Optimal Security (FABEO)

Fast Attribute-Based Encryption with Optimal Security (FABEO)[59] is a fast attribute-based encryption scheme that achieves optimal, adaptive security with multiple challenge ciphertexts. It uses asymmetric prime-order Bilinear groups and supports efficient hashing to G1. FABEO combines several techniques, including smaller ciphertext/key sizes, fast decryption via randomness reuse, and adaptive security without efficiency penalties. It subsumes other state-of-the-art ABE schemes on all parameters of practical interest, with smaller ciphertexts, faster encryption, and fewer pairings for decryption. FABEO achieves close to 128-bit security when instantiated over the popular BLS12-381 curve. FABEO adopts Type-3 bilinear pairing groups. In this work, we use the FABEO CP-ABE scheme provided in [59]. There are four stages:

Phase 1: **Setup (CP_FABEO.Setup(λ))**

- The system parameters are initialized at a TA. Output cryptographic parameters $G := (p, G_1, G_2, G_T, e, g_1, g_2)$, where p is a prime, G_1, G_2, and G_T are asymmetric bilinear groups of prime order p, $e : G_1 \times G_2 \to G_T$ is the bilinear pairing operation with generators g_1 and g_2 for G_1 and G_2, respectively.
- The TA generates the master key pair (public-private), which is used in the later stages. Output the master public key $mpk := (G, H, e(g_1, g_2)^{\alpha})$ and the master secret key $msk := \alpha$, where $\alpha \leftarrow Z_p$ is a nonce generated uniformly and independently from Z_p, and $H : |U| + 1 \to G_1$ is a collision-resistant hashing function and U is the universe of attributes.

Phase 2: **Key generation (CP_FABEO.KeyGen($mpk, msk, S \subseteq U$))**

- The TA assigns attributes to receivers based on their access rights
- The TA uses the master secret key to generate a unique private key for each user based on their attributes.

(a) Choose a random $r \leftarrow Z_p$.
(b) Compute $sk_3 := g_2^r$ and $sk_1 := g_1^\alpha \cdot H(|U|+1)^r$.
(c) For each attribute $u \in S$, compute $sk_{2,u} := H(u)^r$,
(d) Output the secret key $sk_S := (sk_1, \{sk_{2,u}\}_{u \in S}, sk_3)$.

- The user's private key contains a set of attributes that the user possesses

Phase 3: **Encryption (CP_FABEO.Encrypt(**$mpk, (M, \pi)$**))**

- The sender uses the master public key and encrypts the data based on an access policy that specifies a set of attributes. the Monotone Spanning Program (MSP) matrix $M \in \mathrm{Z}_p^{n_1 \times n_2}$ is of the order $n_1 \times n_2$. $\pi : [n1] \rightarrow U$ is a function mapping each row of M to an attribute in the Access Policy.
  (a) Choose a random $s_1 \leftarrow Z_p, v \leftarrow \mathrm{Z}_p^{n_2-1}, s' \leftarrow \mathrm{Z}_p^\tau$, where τ is the maximum numbers times an attribute is used in M.
  (b) Compute $d := e(g_1, g_2)^{\alpha s_1}, ct_1 := g_2^{s_1}, ct_{2,j} := g_2^{s'[j]}$, for $j \in \tau$.
  (c) For each row $i \in [n_1]$, compute $ct_{3,i} := H(|U|+1)^{M_i(s_1||v)^\top} \cdot H(\pi(i))^{s'[\rho(i)]}$, where $\rho(i) := |\{z|\pi(z) = \pi(i), z \leq i\}|$ denotes the $\rho(i)^t h$ occurence of the attribute $\pi(i)$
  (d) Output the ciphertext $ct := (ct_1, \{ct_{2,j}\}_{j \in [\tau]}, ct_3, d)$.
- The encryption process generates a ciphertext that can only be decrypted by users possessing the attributes specified in the access policy.

Phase 4: **Decryption (CP_FABEO.Decrypt(**$mpk, (M, \pi), S, ct, sk$)textbf))

- The receiver attempts to decrypt the message using their private key sk.
- If the private key contains the attributes S that satisfy the access policy (M, π) used for encryption, the decryption is successful and the original plaintext message is obtained.
  (a) If S satisfies (M, π), there exist constants $\{\gamma_i\}_{i \in I}$ such that $\sum_{i \in I} \gamma_i M_i = (1, 0, ..., 0)$.
  (b) Compute $d := e(sk_1, ct_1) \cdot \frac{\prod_{j \in [\tau]} e(\prod_{i \in I, \rho(i)=j} (sk_{2,\pi(i)})^{\gamma_i}, ct_{2,j})}{e(\prod_{i \in I} (ct_{3,i})^{\gamma_i}, sk_3)}$.

2.7 Integrity and Authenticity

A message that is broadcast on the radio space is susceptible to tampering. A malicious entity could intercept the message packet and then introduce some false data or distort the existing data. Without additional methods, it is not possible for a receiver to know if a message's integrity is preserved. Similarly, a malicious entity could be posing as the sender and broadcast the messages. Thus it is also required by the receiver that the sender of the message is who they are supposed to be. This is referred to as authenticity. This is typically achieved using message authentication techniques like message authentication codes (MAC), authenticated encryption (AE), or digital signatures.

2.7.1 Digital signatures

Digital signatures play a vital role in ensuring the authenticity and integrity of messages in modern cryptography. They are a mathematical mechanism that employs both symmetric and asymmetric encryption techniques to achieve these goals. A digital signature involves the use of a public/private key pair, similar to asymmetric encryption. However, unlike encryption, where the sender uses the recipient's public key to encrypt the message, in digital signatures, the sender uses their private key to sign the message. This results in a unique digital signature that is attached to the message.

The process begins with the sender creating a hash of the message. This hash is then encrypted using the sender's private key, forming the digital signature. The recipient can verify the signature by using the sender's public key to decrypt the signature and obtain the hash. The recipient then

generates a hash of the received message and compares it with the decrypted hash from the signature. If the two hashes match, it indicates that the message is authentic and has not been tampered with during transmission.

Digital signatures provide strong security guarantees. They ensure that the message originated from the known sender, as only the sender possesses the private key required to produce the valid signature. Additionally, they confirm that the message has not been altered in transit, as even a minor modification of the message would result in a completely different hash value. Overall, digital signatures are a crucial tool in establishing trust and security in digital communications.

2.7.2 Schnorr signatures

Schnorr signatures stand out as a prominent digital signature scheme that offers various advantages over alternatives like RSA or DSA [24]. In the context of digital signatures, a signer using Schnorr signatures creates a unique signature value by leveraging their secret key and the message being signed. This value is then combined with the signer's public key to generate the final signature. Schnorr signatures excel in efficiency, requiring fewer computations and shorter keys, which translates to more compact signatures [52]. Furthermore, they exhibit provable security within the random oracle model, meaning their security can be mathematically demonstrated under specific assumptions [3].

An appealing feature of Schnorr signatures is their support for key aggregation. This property allows multiple parties to collaboratively generate a joint signature without exposing their individual secret keys. Such capability makes Schnorr signatures particularly well-suited for deployment in multi-party and distributed systems [4]. The Schnorr signature process is characterized by two main steps: signing and verifying. Prior to either of these steps, the sender (signer) generates an Elliptic Curve Public-key pair. While the public key is shared openly, the private key remains confidential and is exclusively used by the sender.

Key Generation:

1. Choose a prime p and a generator g of the group Z_p^*. where, $Z_p^* = \{a \in Z_p : \gcd(a, p) = 1\}$
2. Choose a secret key $x \in Z_p^*$.
3. Compute the public key $y := g^x$.

In the signing step, the sender uses their private key and various cryptographic functions to compute the signature parts. These parts are then combined with the message and transmitted to the recipient. Specifically, the sender first generates a random number, called the nonce, and computes a random point on the elliptic curve. Then, the sender computes a hash of the message and the random point to get a random value, called the challenge. Finally, the sender uses their private key and the challenge to compute the signature value. The signature is then sent along with the message to the recipient.

Sign:

1. Choose a random nonce $k \in Z_p^*$.
2. Compute $r := g^k \mod p$; $e := H(m||r)$, where H is a cryptographic hash function.; $s := k + xe \mod p$.
3. Output the signature $\sigma := (r, s)$.

In the verification step, the recipient uses the sender's public key and the parts of the received message to compute a verification value. The verification value is computed by raising the generator of the elliptic curve to the signature value, and then multiplying the result by the sender's public key raised to the challenge value. The recipient then compares this computed value with the received signature value. If the two values are the same, the signature is considered valid and the message is assumed to have been sent by the sender. Otherwise, the signature is invalid and

the message may have been tampered with or sent by an unauthorized party.

Verify:

1. Compute $e := H(m||r)$; $v := g^s y^{-e} \mod p$;
2. If $v = r$, output 1 (valid signature); otherwise, output 0 (invalid signature).

Overall, the Schnorr signature scheme provides an efficient and secure way for the sender to sign a message and for the recipient to verify the signature using the sender's public key. The work in [65] mentions that a 4t-bit Schnorr key provides t-bit level of security. Hence, a 512-bit (64 Bytes) key provides 128-bit security.

Chapter 3

Scenario and Adversary Model

This chapter describes the reference scenario of the Drone Remote ID where the System message with the pilot location is broadcasted. Later, we discuss the adversarial models along with some of the use cases where the proposed system might be used. Finally, the requirements are extracted from all the previous sections.

3.1 Scenario

The entities in the reference scenario are the UAV (drone) and the observer(s). A USS is also needed but is not an active part of the scenario at runtime. The UAV is expected to transmit the location of the pilot or the GCS once every 1 to 3 seconds (depending on the message type mentioned in [10]) and an observer, if present, can decode the RID message received. However, from the point of view of the operations executed on the drone, the presence of an observer is not mandatory.

The drone / UAV must be compliant with the FAA regulation of Remote ID, which means, it is either a standard RID drone or it is equipped with a RID broadcast module. Thus, it uses for example the Wi-Fi (2.4 GHz /5 GHz) radio technology to transmit Remote ID messages. In most cases, this technology is also used for communication between the UAV and the GCS. Additionally, for the system message in the RID, the UAV needs the Global Navigation Satellite System (GNSS) position, typically the Global Positioning System (GPS) data of the GCS. Depending on the technology at the GCS, there are two possible cases :

1. GCS has the technology to provide the UAV with its GPS location
2. the GPS location of the UAV's take-off place is used

In the first case, a live location is sent from the GCS to the UAV and this is transmitted by the UAV via Wi-Fi or Bluetooth. In the second case, where the GCS does not have the capability to acquire the location or for autonomous operations, the UAV just transmits its take-off location as the location of the GCS.

As there are multiple entities involved, for this, there needs to be some authority that takes care of the management and control of the active operations of a given area. A USS performs the job of a Trusted Authority (TA) to communicate with the UAV or the GCS and the observers for these purposes. It maintains archives of drone operations and registries of the information required by the UAV and the observers. The communication between the UAV and the observers with the USS takes place before the drone operation and does not occur during the operation.

The role of the observer is only to passively receive the messages in this scenario. The observer could be a civilian with a simple receiver like a smartphone. Such observers are called "General Public Observers" But there could also be some safety organizations or aviation authorities that also monitor the aviation space and receive the RID messages sent by the UAVs. These are

collectively called the "Public Safety Observers". All the observers are assumed to have a capable RF receiver that can receive and decode the RID messages. It is also assumed that the observers have acquired the necessary information required to verify the received RID messages, from the USS before the scenario. Some of these observers could require a way to communicate with the pilot or the UAV during emergencies which could be arranged via the USS so that there is no need for direct contact. However, under normal circumstances, there is no communication between the UAV and the USS regarding the current system. The scenario is presented in the picture below in Fig. 3.1.

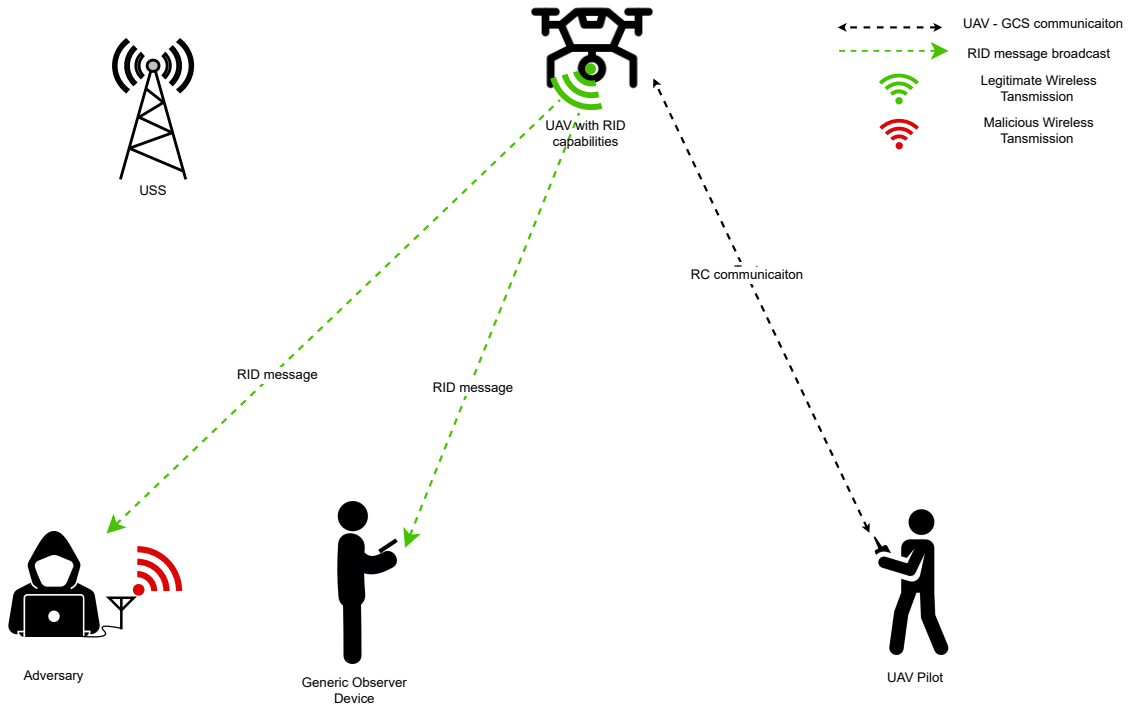


Figure 3.1: Reference scenario of Remote ID broadcast

In the scenario, the UAV is the unidirectional broadcaster and observers are stateless users as they do not hold any previous information from the broadcast that is useful for the next message.

3.2 Adversary model

For the scenario of this work, it is assumed that the UAV and its pilot (and hence the GCS) are legitimate and do not violate the laws during the operation. Thus, the adversary only belongs to the observer category. It can be observed from the scenario that the adversary's actions are confined to the current UAV operation and there is no long-term effect of it from just accessing the GCS location alone. However, there could be potential threats when combined with other types of RID messages.

3.2.1 Possible Objectives

The adversary could aim to achieve one or several of the following objectives :

- Compromise privacy and safety of drone operation ← unauthorized access to pilot location data
- Criminalize the operation ← rouge impersonation

- Breach of trust between the observers and UAV by making it unable to access RID information ← RF flooding/masquerading, corrupting cryptographic material exchange
- Jeopardize multiple operations ← attacks on USS
- Behavioral analysis (social engineering) ← in conjunction with other information about the pilot / UAV for other attacks

Based on these objectives, the adversary can be classified to have both passive and active capabilities, where it can keep sniffing the packets sent by the UAV or also try to transmit rogue messages to meddle with the operation.

3.3 Use-cases

The advantages of using ABE and Digital Signatures in RID messages include selective disclosure, Tamper resistance, and protection against spoofing and replay attacks. These features create some unique use cases for the system. Each use case is explained in the following subsections.

3.3.1 UAV Operations at Country Borders

Drone operations frequently extend to the borders between countries, encompassing a range of activities such as aerial surveys, border patrols, and environmental monitoring. However, in these cross-border scenarios, a significant concern arises regarding the sharing of pilot locations with observers from multiple countries. The exposure of sensitive location data to all observers on both sides of the border can raise issues of privacy and even national security.

The current work addresses this challenge through the application of Attribute-Based Encryption (ABE). By leveraging ABE encryption, pilots can exercise precise control over the accessibility of their location data. This ensures that the pilot's location is only disclosed to observers within their own country, thus safeguarding their privacy and addressing the potential security risks associated with indiscriminate data sharing.

Furthermore, the utilization of ABE encryption also supports international collaboration at border regions. pilots can participate in cross-border initiatives, contribute to joint missions, and share valuable data without compromising the security of their location information.

3.3.2 Pilot Guidance in Safety-Critical Operations

In situations where pilots need to be tracked and monitored for enhanced assistance in safety-critical operations, the public sharing of their locations can pose significant risks. To mitigate these risks, the pilot's location data can be made accessible only to the command-control center responsible for providing essential guidance and oversight during the operation.

Consider scenarios such as disaster response, hazardous site inspections, or medical assistance operations where real-time communication with pilots is imperative. In such cases, the command-control center plays a pivotal role in ensuring the safety and success of the operation. Selective disclosure ensures that sensitive location data remains confidential and secure, safeguarding both the pilot and the integrity of the mission.

Additionally, the work also facilitates the method for pilots and UAVs of the same deployment (group) to be aware of each other's locations without the need for extra secure communication infrastructure.

3.3.3 Search and Rescue Operations

In dire circumstances, such as when a pilot is lost or stranded in areas with no Internet or cell coverage, the use of drones can be a lifeline. By employing ABE encryption, the pilot can securely communicate their exact location to rescuers without exposing this critical information to potential threats. This ensures that sensitive location data remains confidential and can only be accessed

by authorized personnel, expediting the search and rescue process while minimizing the risk of exploitation.

Drones also play a vital role in search and rescue operations, providing a swift and efficient means of locating individuals in emergency situations. Involving volunteers in search operations for missing persons is a common strategy. However, protecting the privacy and safety of these volunteers is of utmost importance. Protecting the location data, allows volunteers to participate without disclosing their personal whereabouts to unauthorized entities. This approach enhances the effectiveness of crowdsourced search efforts while preserving the privacy and security of those involved.

Ensuring secure and private communication between pilots, drones, and rescuers, contributes to more efficient and effective operations, ultimately safeguarding both individuals in distress and those involved in the rescue process.

3.3.4 Privacy in Public Events

Drones are finding increasingly diverse applications in fields such as event coverage, journalism, and reporting. In scenarios involving public events or potentially hazardous situations, such as covering protests or riots, it is of paramount importance to ensure the pilot's location remains confidential. Publicly revealing the pilot's location could not only compromise the safety of the entire operation but also expose them to potential threats or interference. By selective encryption of the pilot's location data, only authorized entities, such as the event organizers or approved stakeholders, can access the decrypted information. This approach strikes a balance between the need for transparency in event coverage and the protection of the pilot's privacy.

3.4 Requirements and Features

Based on the previous chapters and sections, the requirements for the current work can be extracted. This includes the requirements imposed by the standards, scenarios, use cases, and any other additions. Some of the topics mentioned below are more of a feature than a requirement. Hence, to differentiate between the features and requirements of the work, they are mentioned in the headings. A feature is represented by **F** and a requirement by **R**, both followed by a number for identification purposes. In the later chapters, we focus only on the requirements.

3.4.1 Requirements From the Standards and Regulations

As mentioned in sections **2.1** and **2.2**, the FAA and the ASTM standard introduced some requirements:

1. **Transmission once every 1 to 3 seconds** [R2]
   The ASTM standard describes the System Message (Type 0X04) as equivalent to the static message, as the pilot location does not change as fast as the UAV's location. Thus, the UAV is required to transmit the System Message at least once every 3 seconds.

   This is translated to the current work as all the computations must be finished and the packet is delivered within 3 seconds.

2. **Entire message in a single Wi-Fi packet** [R3]
   The System Message that is described in the ASTM F3411-22a is 25 bytes long. This implies that the message is transmitted in a single packet. We intend to follow this requirement in this work too. As the current work suggests the use of encryption and also Digital signatures, it is obvious that the size of the message increases. Thus, it should fit into a single Wi-Fi packet of with an MTU of 2312 bytes and does not exceed this limit.

3. **Toggle-able Encryption** [F1]
   The regulations require that the System Message is not encrypted in some jurisdictions based

on the local laws or during some special cases mid-operation. In such situations, the encryption must be turned off and the message must be transmitted in plain text. But this does not put any restriction on the authentication part. So, the message is always authenticated regardless of the scenario. This is a compulsory feature that must be implemented rather than a requirement.

4. **No persistent Internet connection** [R4]
   The standards suggest that during the operation, any eligible observer must be able to receive and process the information without the need for the Internet.

3.4.2 Requirements From the scenario and use-cases

1. **Controlled Access** [R1]
   Most of the scenarios presented are based on the concept that only a selected group of observers are able to decrypt the messages to protect the privacy of the pilot.

2. **Encryption of non-continuous data and need for extra bytes**
   In the system message, the pilot Altitude field is not directly next to the Latitude and Longitude fields. And depending on the encryption scheme used, the cipher text could definitely be larger than the plain text. This also means that there is a requirement to have additional bytes at the end of the message, or the message fields must be restructured accordingly.

3. **Offloading Computations** [F2]
   As the encryption and authentication processes are to be performed by the UAV, which is a device limited in terms of both computation and energy consumption, the work should first ensure that the algorithms can be implemented in the constrained device. It should also facilitate offloading the computations to the GCS or having an alternate method of pre-computations and storage. This is a feature that must be available to fulfill the previous requirement [R2].

4. **Dynamic change of policies/attributes during operation** [R5]
   In special situations, the UAV might need to change the policies or attributes used for encryption, and the work must be able to support this. This can be done in one of two ways

   - Selecting different policies by the drone
   - Creating new policies

   The work must be able to accommodate at least the first type.

5. **Opportunistic Decryption** [R6]
   During the operation, there is no two-way communication between the UAV and the observer. This means that the observer decrypts the messages in an opportunistic way. And from the requirement [R4] it is clear that the decryption works with the information acquired from the USS before the operation.

6. **Authenticity of the message** [R7]
   As the message is broadcasted, there are chances that the message data is tampered with and then retransmitted by an adversary. To prevent this, the message must be authenticated.

Chapter 4

Literature Study and Research questions

Based on the requirements from the previous chapter, the literature study is performed and some of the important papers are discussed in the following sections. Then a comparison is presented where the gaps in the research are found and summarised in a table. Later we discuss the research questions for the thesis phase.

4.1 Literature Study

The keywords used to find the related works and the literature study are:

- Broadcast Encryption
- Selective disclosure/access
- ABE in Constrained devices
- Privacy in broadcast
- Message authenticity
- Location privacy

The literature sources are searched for in the Scopus database and Science Direct by using an incremental combination of the keywords. The full text of the source is accessed via TU/e's Library search and the standards are accessed from their own websites.

4.1.1 Overview of closely related works

A generalized version of our work falls under the category of ***privacy (and authenticity) in broadcast communications***. So we find related literature in academia and the industry that worked on solving this problem in various relevant fields. We see these problems in the fields of Wireless Sensor Networks (WSN), Ad hoc networks like Mobile ad hoc Networks (MANETs), WANETs (Wireless ad hoc networks), and Vehicular ad hoc networks (VANETS), pay-TV, etc.

Broadcast systems have always been present to share information with a huge audience. There are many popular forms of broadcast communication such as radio stations and Digital Video Broadcasting (DVB). Usually, these are intended to be accessible to anyone. A special mention is the Ham Radio where licensed users can tune to a frequency and talk to any other person on the same frequency via broadcast. The laws mandate that the operators (pilots) are not allowed to encrypt the data. This open and unencrypted form of communication might be disadvantageous

in some cases as the broadcaster has no knowledge of who is receiving the data. For example, commercial broadcasters want to restrict access to their content to a group of premium users or some private users want to selectively disclose their data.

On the other hand, the receiver cannot always be certain about the broadcaster and the authenticity and integrity of a message that is received, which is not verifiable in an open broadcast system (without extra measures). For example in Automatic Dependent Surveillance-Broadcast (ADS-B) which is used by air crafts to transmit their location periodically to ground station observers, there are no mechanisms that prove the message authenticity. This makes it prone to spoofing attacks.

In order to mitigate these problems, works emerged focusing on the aspects of the first problem of selected access through Broadcast Encryption, which evolved from using group keys, where a subset of the receivers have a key that can be used to decrypt the broadcast message. Using a group key means that a new key is to be distributed to the group every time a receiver exits the group.

To overcome this issue, several papers came up with the idea of Identity Based Encryption where the identity of the user is used to generate an access policy that determines who can decrypt the message. In [51], Mu et al. propose two identity-based schemes for authenticated broadcasting and distributed message authentication that support multiple broadcasters to dynamically determine a group of receivers and broadcast to them. Delerablée [26] describes an Identity Based Broadcast Encryption (IBBE) system with constant size cipher texts and private keys with no restriction on the total number of possible users in the system during setup. The public keys are of linear size and were shorter than the then-existing works. Lv et al.,[44] come up with a traitor tracing mechanism based on the IBBE with no trusted agents. Yao et al. [78] proposed a forward-secure Hierarchical Identity Based Encryption scheme based on the bilinear Diffie-Hellman assumption in the random oracle model which allows users to join dynamically.

Some papers used Attribute-Based Encryption, a more generalized version of IBE where several attributes (as opposed to having only Identity) are used to define policies. Sun and Hu[69] put forth an Attribute Based Broadcast Encryption (ABBE) system which uses the orthogonality property of composite-order bilinear groups and the attribute vector to achieve constant-size ciphertext. Phuong et al.[56] further reduced the size of the ciphertext and the decryption key in ABBE with direct revocation ability which is required in real-time broadcasting applications like PayTV. Chen et al. in [20] put more focus on making the attributes "dynamic" by using a fading function and allowing users to update each attribute separately. Li et al.[41] encompasses all the features from the previous works and incorporates a mediated attribute-based encryption to achieve outsourced storage and outsourced decryption to reduce storage and computational overheads.

On the problem of broadcast authentication, several works are done that adopt various techniques to tackle the issue [80] [37] [43] [76].

There are also works that focus on solving both problems simultaneously by either using an encryption scheme and an authentication scheme (one after the other) or using singcryption schemes which do both signing and encryption at a smaller computational cost and data overhead.

The next important aspect is to understand the feasibility of running the encryption algorithm on a constrained device. Although running traditional encryption and authentication algorithms on constrained embedded devices is common, it is still fairly new to run Attribute-Based Encryption algorithms. The authors of [54] talk about running various CP-ABE and KP-ABE schemes on an Espressif Micro-controller and compare various ABE schemes' computation costs with and without hardware acceleration for encryption and decryption. Their study shows that it is feasible to run ABE with a relatively low number of attributes and that decryption is more computationally intensive than encryption. The authors in [71] also performed a similar study but on a slightly powerful device which is a single-board computer that runs Linux. The authors in [46] evaluate the performance of ABE in secure over-the-air (OTA) software updates in automotive embedded platforms.

The work by Tedeschi et al. [72] examines the new Federal Aviation Administration (FAA) standard called remote identification, which aims to enhance accountability for Unmanned Aerial Vehicles (UAVs). However, this standard requires UAV pilots to broadcast sensitive information

like identity and location, raising security and privacy concerns. The review systematically explores vulnerabilities in the remote ID system that can be exploited by attackers to compromise safety and accountability. It also evaluates existing solutions to address privacy issues associated with remote ID. The paper concludes by highlighting challenges that need industry and academic attention and suggests future research directions to enhance the security and privacy of UAVs in this evolving landscape.

The Works by Wisse et al. [74], and Tedeschi et al. [73] provide solutions and frameworks to achieve privacy in Remote ID systems by using pseudonyms and signcryption techniques.

4.1.2 Similar works

The previous section gave an overview of the context of the various groups under which the research is done. Now, in this section, we discuss similar works and how our work compares to them.

Ruan et al. [61] combined a broadcast authentication scheme with a broadcast encryption scheme to create a privacy-preserving location service. They use the full public key broadcast encryption scheme (FBE) [27] which uses a public key scheme with an option for revocation. And for authentication, they use a simple one-time signature technique that enables receivers to authenticate broadcast messages from any sender [47]. The work is simulated on ns-2 simulator and the results are mentioned in the paper. There is no explicit mention of the hardware used but it can be safely assumed that it is an unconstrained device. The authors also do not talk about the size of the message packet. Hence, this work does not satisfy all the requirements. He et al.[35] use Hierarchical Identity Based Broad Encryption (HIBBE) [42] scheme in UAV networks along with mutual authentication using identity-based signcryption (IBSC) [15]. They further add pseudonyms for receiver privacy. Though this work is done for UAV networks, the evaluation is done in NS3 and not implemented on a UAV. Thus there is no mention of the impact of the system on the computation power of the UAV. The paper mentions that the drone is "equipped with wireless communication modules and high-speed processors", which implies that there is no constraint on the computational resources. Finally, they show that their system incurs only a few milliseconds of overhead.

Prado et al. [57] argue that the then-existing geocasting algorithms focus only on either privacy or reliability but not both. They propose a direction-based dissemination geocasting algorithm that uses public key encryption, pseudonyms, and authentication to achieve privacy and security and use a transmission-coverage algorithm that is used in mobile sensor networks to reduce communication overhead. The proposed algorithm makes sure that the message does not go outside a certain radius and avoids duplicate transmissions. They evaluated their work using the NS3 simulator with a VANET infrastructure. Though they discuss the computational costs, they only talk about the time taken by a laptop running the simulation. This is not an accurate indicator as the real nodes on VANET would be constrained.

Yang et al. [77] put forth a "lightweight and practical cryptographic solution" for ADS-B which uses FFX (Format-preserving, Feistel-based encryption, with multiple implementation variances), a Format Preserving Encryption (FPE) algorithm accepted by NIST [16] and created their own version of timed efficient stream loss-tolerant authentication protocol (TESLA) [55] called Adaptive TESLA that exploits the one-way keychain with retro-active key issuing. Their work uses different messages for the data and the keys and takes around 30ms for the encryption. This work does not provide controlled access over the decryption and thus also doesn't have the option for dynamic change of control which comes with it. Though the messages comply with the size requirement of ADS-B, the authors propose to use multiple messages which violates our requirement to have all the data in a single packet.

Adragana et al. [8] proposed a distinctive approach to address the challenge of protecting location privacy, which is becoming increasingly crucial due to the proliferation of mobile communication devices and location-based services. Their work centers around an obfuscation-based strategy, where the goal is to manipulate location data to ensure the privacy of users' physical positions. To achieve this, the authors introduced a variety of obfuscation operators that can be used in isolation or in combination to alter the accuracy and precision of location information.

This approach primarily operates on a theoretical level, focusing on the conceptual foundations of location privacy protection through obfuscation. However, the work lacks a comprehensive exploration of the practical implementation aspects, such as computational resource utilization or communication overhead. As a result, while their approach introduces a novel perspective on location privacy, it may not address the full spectrum of challenges associated with real-world deployment and scalability.

In [45], Mandal introduces a Fully Anonymous Ciphertext Policy Attribute Based Broadcast Encryption for preserving the privacy of the access policies and the recipients of a broadcast. This work only satisfies the requirements of controlled access and low computation time. The rest of the requirements are not satisfied. Rezazadeh et al. [58] discuss the need for a lightweight authentication in vehicle-to-vehicle (V2V) communication systems and propose a protocol that only adds 149 bytes of cryptographic overhead and combine this with beacon encryption using Hash-based Message Authentication Code (HMAC) and Advanced Encryption Standard (AES) respectively to achieve anonymity and unlinkability. Their work mentions the use of user-inaccessible cryptographic materials stored in vehicle Tamper proof Devices (TPDs). Except for controlled access and partially the dynamic change of control, the work satisfies all other requirements and comes very close to what we are going to do.

The emergence of convoy driving in the autonomous vehicle domain introduces new challenges to location privacy. The work by Xin et al. [79] addresses these concerns by proposing a dynamic mixed zone establishment scheme. The scheme leverages the closed group nature of convoys to broadcast mix zone establishment requests, ensuring privacy both within and outside the convoy. To prevent attacks, vehicles can use multiple pseudonyms, and the scheme includes methods for tracing offending vehicles' real identities. This approach offers enhanced security and lower overhead compared to traditional vehicular network-based solutions. The work uses the Elliptic Curve Digital signature algorithm and encryption scheme for their system. The evaluation is performed on an unconstrained device which takes less than 3 milliseconds for all different communication messages. Though they present a discussion on storage memory usage, there is no mention of the communication overhead.

Another interesting work is by Lai et al., [40] where they brought a solution by building upon a previous work [39], for smart cities to broadcast information to legitimate users while preserving the privacy of the data and the recipients as well as those with revoked access. Though this paper looked promising from a security point-of-view by offering concrete security proofs an analyses, it was not implemented or simulated in any way. The available data is not enough to know if the work would fulfill the requirements. It is also clear that the work does not support offline decryption or authentication. A draft by Moskowitz et al. [50] attempts to solve a similar problem to ours on pilot location privacy in RID messages. They propose to use an Encryption In Place algorithm for the 10 bytes of pilot latitude, longitude, and altitude. There is no mention of authentication or a way for multiple users to receive and decrypt the message without the internet.

4.2 Comparison and Gaps

In section 3.4, we have listed the possible requirements and features. Now, we use those to compare the similar works to see if any of them fulfill all the requirements. The list of requirements is as follows:

- Controlled Access [R1]
- Low Computation time [R2]
- Message size less than WiFi MTU and in a single packet [R3]
- Offline Decryption [R4]
- Dynamic policy changes [R5]

- Opportunistic Decryption [R6]
- Message Authenticity [R7]

SNo.	Ref No.	[R1]	[R2]	[R3]	[R4]	[R5]	[R6]	[R7]
1.	[61]	◑ (public keys)	○	-	●	◑ (revocation)	●	●
2.	[35]	◑ (ID-based)	○	●	●	○	○	●
3.	[57]	◑ (public keys)	-	○	●	○	●	●
4.	[77]	○	●	○	●	○	○	○
5.	[8]	○	-	-	○	○	○	○
6.	[45]	●	●	○	○	○	○	○
7.	[58]	○	●	●	●	◑ (revocation)	●	●
8.	[79]	○	●	-	○	◑ (updates)	●	●
9.	[40]	●	-	-	○	◑ (revocation)	●	○
10.	[50]	○	-	●	●	○	○	○

●: complies; ○: does not comply; ◑: partially complies; - : no information

Table 4.1: Comparison of similar works based on our requirements

In the Tab. 4.1, the full circle (●) indicates that the work complies with the requirement and the empty circle (○) means that it does not comply. The half-filled circle (◑) is used to show that it partially complies and the reason is mentioned in the brackets of the same cell. And if there is not enough information then we use a dash (-). We see that none of the similar works meet all the requirements that are proposed in this study. Hence, based on the gaps in these works, we formulate our current work, and the research questions are presented in the next section.

4.3 Research Questions

From all the literature studies and comparisons, we conclude that no other works provide a solution for all the requirements mentioned earlier. So, we propose to perform research towards the most optimal solution that satisfies all the requirements. For this we formulate the following main research question and its sub-questions as follows:

Main question:

What is the most efficient way to achieve selective disclosure of pilot location in Remote ID messages on constrained UAVs?

sub questions:

RQ1 What is the effect of our system on a normal drone operation in terms of computation and energy consumption?

RQ2 What is the impact of the number of attributes (expressibility of the policies) used in the policy on energy consumption and computation costs?

RQ3 What is the optimal number of attributes that can be used without significantly affecting a drone operation time(due to additional Energy consumption and Computational costs)?

RQ4 How much effect does the use of precomputed ABE(AES) keys have on the operation? (Find the optimal way)

RQ5 What is the impact of changing the policies during the drone operation when:

- Adopting a new policy from the existing list
- creating a new policy

RQ6 What is the trade-off of authentication and energy in our system?

Chapter 5

SNELL Protocol Structure and Implementation Procedure

This chapter presents the Selective autheNticatEd piLot Location disclosure (SNELL) protocol designed for achieving a robust and secure drone Remote Identification system. The protocol encompasses two fundamental phases: the Registration Phase and the Online Phase. These are discussed in detail in the following sections.

5.1 Registration Phase

In the Registration phase, the UAV and the GCS communicate with a USS, which acts as the TA. During this phase, all the cryptographic materials (keys and initialization parameters) required for the correct functioning of the system are exchanged between the authorities and the UAV and its GCS. Next, the USS distributes the applicable keys to the requested observers. The flowchart shown in Fig. 5.1 summarizes the process. The UAV and the observer communicate with the USS only in this phase. Tab. A.1 contains the notation used in this chapter along with a brief description.

The operations in this phase are as follows:

- The USS performs the CP_FABEO.Setup( ) (see sec.2.6.1) to generate the master public key mpk_A and master secret key msk_A Cipher-text Policy ABE (CP-ABE), as in Eq. 5.1:

$$\begin{aligned}(msk_A, mpk_A) &= CP\_FABEO.Setup() \\ msk_A &:= \alpha \\ mpk_A &:= (G, H, e(g_1, g_2)^{\alpha})\end{aligned} \tag{5.1}$$

  where $g1, g2$ are the curve generator points and α is a random nonce in Z_p.

- When a UAV establishes a secure communication connection with the USS by providing the required identification, both the entities exchange the public keys with each other, i.e., the UAV shares the public key for its Schnorr signature scheme Spk and the USS shares the master public key for the CP-ABE mpk_A. During this process, both entities store the exchanged information for further use.

- When an observer sends a request along with its ID, the USS performs the CP_FABEO.KeyGen() (see 2.6.1) operation to generate a secret key for the CP-ABE, specific to that observer based on their attributes $u = \{ u1, u2, u3, ... \}$ given in Eq. 5.2.

$$sk_s = CP\_FABEO.Keygen(mpk_A, msk_A, u) \tag{5.2}$$

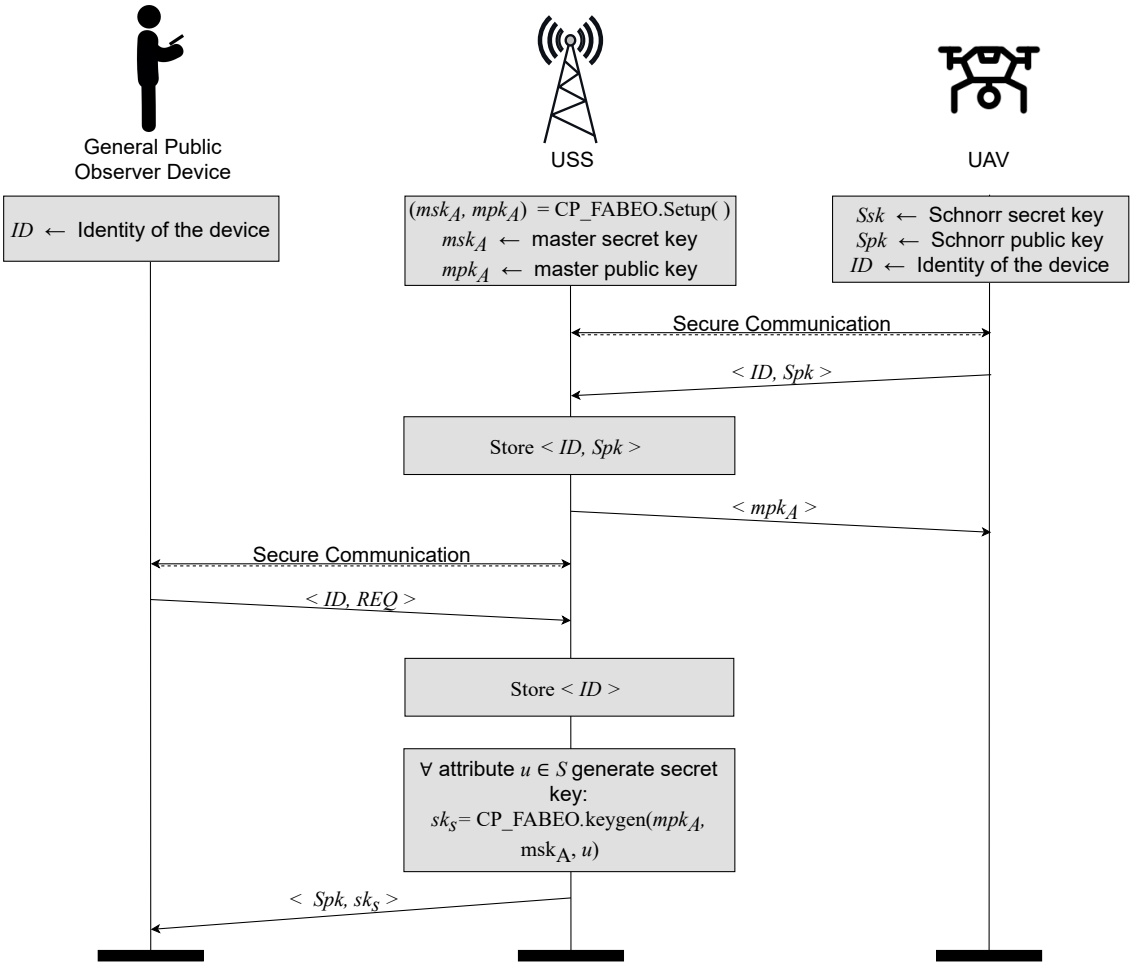


Figure 5.1: Sequence Diagram of the Registration Phase of the SNELL protocol.

Then, the USS shares with the observer, the public key for Schnorr Signature scheme Spk previously obtained from the UAV and the CP-ABE secret key of that specific observer sk_s. The information is stored locally at the observer for future use.

5.2 Online Phase

During the Online phase, the UAV broadcasts the RID message every 3 seconds as mentioned in the requirements (see 3.4). The message contains the pilot location part, which is encrypted in two steps followed by an authentication step. The flowchart in Fig. 5.2 gives an overview of all the steps involved in the protocol. Fig. 5.2 shows that the operations are grouped into three sets on both the UAV and the pilot, representing the reference parts of the protocol. CP-ABE allows for fine-grained access control, ensuring that only observers with the appropriate attributes can decrypt specific parts of the message. Thus, only eligible receivers (observers) are then able to decrypt the message, i.e., the observers that satisfy the policy used for encryption by the UAV. AES, on the other hand, provides robust encryption for protecting the GCS location data. The Schnorr digital signature makes the message resistant to replay attacks.

Firstly,the UAV generates a nonce r in G_T and then encrypts it using CP-ABE, with a desired policy P, to produce a ciphertext c in Eq. 5.3.

$$c = CP\_FABEO.Encrypt(mpk_A, P, r) \tag{5.3}$$

Next, it (UAV) generates a symmetric key K from the nonce using a Key Derivation Function (KDF), and uses this key to encrypt the retrieved pilot location data $L_{GCS} = [lat, lon, alt]$ using AES, in equations Eq. 5.4, Eq. 5.5:

$$K_t = KDF(r) \tag{5.4}$$

$$C = AES.Encrypt(L_{GCS}, K_t) \tag{5.5}$$

In the next step, the UAV adds the encrypted data to the message which has the UAV ID, location data $L_{UAV} = [lat, long, alt]$, UTC timestamp t, and the emergency code ec. The encrypted nonce c is attached at the end to create the final message, expressed in Eq. 5.6

$$m_t = [ID||L_{UAV,t}||C||c||t||ec] \tag{5.6}$$

Finally, the UAV generates the Schnorr signature σ for the entire data in the message, using the UAV's secret key SsK given in Eq. 5.7. Finally, the data along with the Schnorr signature is put into a Wi-Fi packet and sent for transmission. The UAV goes through the whole process to transmit every single message.

$$\sigma_{m,t} = Schnorr.Sign(m_t, Ssk) \tag{5.7}$$

On the observer side, the process is reversed with slight changes. The received message is parsed and separated into the respective components. Firstly, the observer verifies the digital signature against the message. If this is successful, then the observer decrypts the ABE ciphertext c (given that it possesses the required attributes for the policy specified by the UAV) to retrieve the nonce r. It then uses the KDF with this nonce to obtain the symmetric key K. Finally, the key is used to decrypt the pilot location data using AES, expressed in the equations Eq. 5.8 to Eq. 5.11 given below.

$$Schnorr.Verfiy(m_t, \sigma_{m,t}, Spk) == 1 \tag{5.8}$$

$$r = CP\_FABEO.Decrypt(mpk_A, P, c, sk_s, attr\_set) \tag{5.9}$$

$$K_t = KDF(r) \tag{5.10}$$

$$L_{GCS} = AES.Decrypt(C, K_t) \tag{5.11}$$

In conclusion, the protocol structure presented in this chapter offers a comprehensive and secure solution to preserve the privacy and confidentiality of the location of the pilot in messages, while still fulfilling the rules set by FAA. The Registration Phase facilitates the establishment of a way to transmit the messages, through the secure exchange of required cryptographic keys. The Online Phase enables UAVs to broadcast the GCS location securely, and authorized observers can efficiently decrypt the messages.

The UAV side of the Online phase is the main focus of this work and the rest of the system (registration phase and observer side of Online phase) is implemented only to the minimum so as to ensure the correct execution of the operations and help verify and evaluate the main implementation.

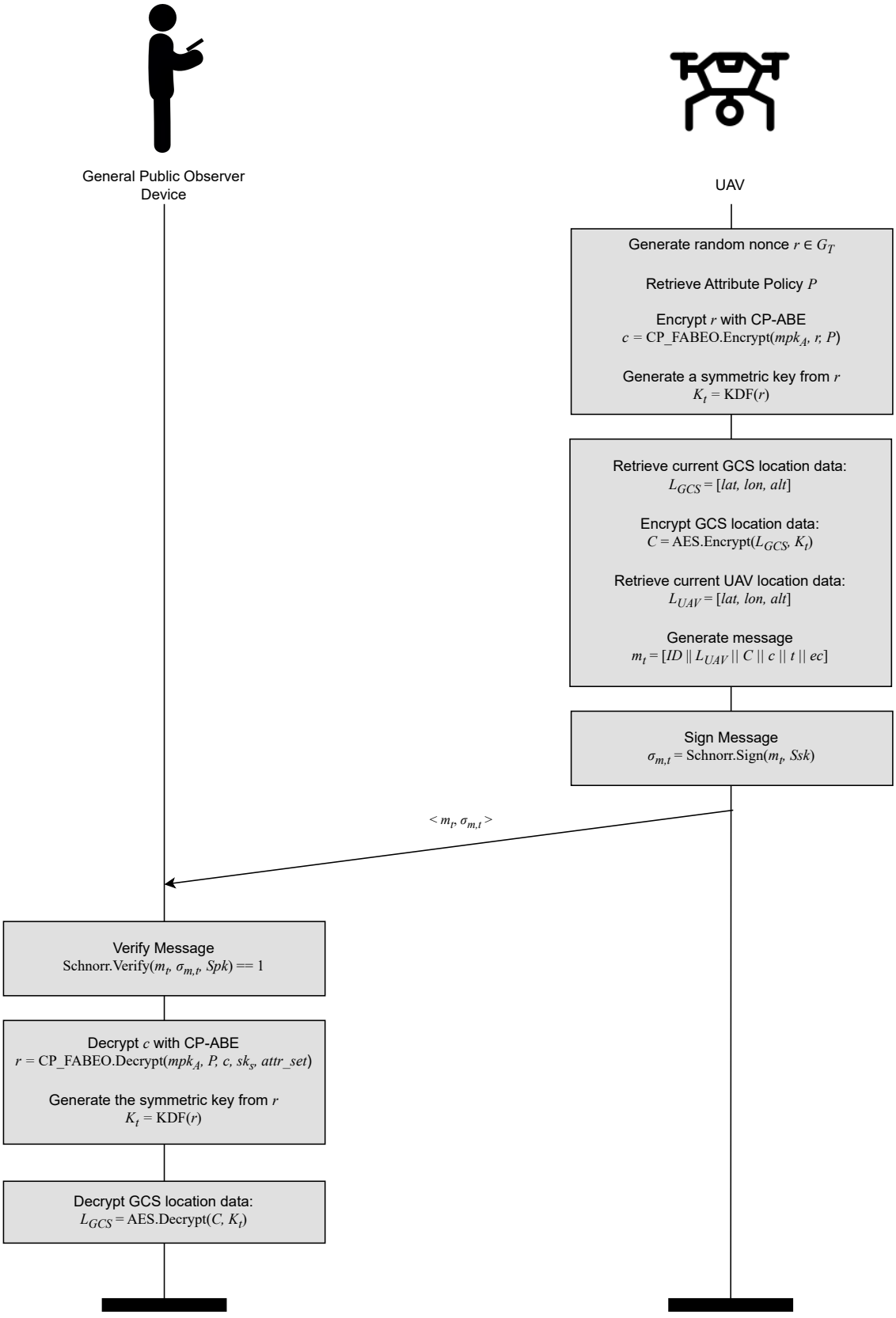


Figure 5.2: Sequence diagram of the Online Phase of the SNELL protocol.

Chapter 6

Implementation

This chapter focuses on the design choices and practical implementation of the cryptographic components in the SNELL protocol. We delve into the details of implementing the Schnorr signature scheme, AES encryption, and FABEO CP-ABE. Additionally, we explore various optimizations to enhance the performance and efficiency of our system. By presenting the implementation details and strategies, we aim to provide a comprehensive understanding of how these cryptographic techniques and optimizations are integrated into the SNELL protocol. Through careful implementation and optimization, we strive to achieve a robust, secure, and efficient system that meets the requirements defined in section 3.4.

6.1 Device Selection

6.1.1 Requirements and Criteria

When selecting a suitable embedded microcontroller or microprocessor for a project, it is crucial to define the requirements clearly to narrow down the search and ultimately choose the most optimal device. In this particular case, the project necessitates a device capable of transmitting data over Wi-Fi, which implies the need for an onboard Wi-Fi transceiver. Additionally, the device should support cryptographic operations, specifically AES, Digital Signatures, ECC, and PBC.

The inclusion of a Wi-Fi transceiver is essential for the transmission of SNELL-based messages. This feature also allows the device to exchange data with the GCS or connect to the internet, opening up possibilities for data exchange over networks. The Wi-Fi transceiver should support the desired Wi-Fi standard(s) such as 802.11a/b/g/n/ac/ax to ensure compatibility with various receivers and provide adequate data transfer rates. It should also provide support for the transmission and reception of custom-built data packets which contain the messages.

There are some embedded devices that have a microcontroller or microprocessor without Wi-Fi capabilities but have an additional dedicated Wi-Fi transceiver. Though this kind of setup works, it consumes more energy to run two separate chips. So, it would be ideal to have a device that has both the computation and the radio.

In terms of cryptographic operations, the device should be capable of performing AES encryption and decryption. AES is widely used and offers a high level of security, making it suitable for protecting sensitive data during transmission or storage. The GCS location information is going to be encrypted using AES. Additionally, it is preferable that the device should support key lengths of various sizes to accommodate different security requirements.

Digital signatures play a crucial role in ensuring data integrity, authenticity, and non-repudiation. The selected device should be capable of generating and verifying digital signatures. As we plan to implement the Schnorr Signature scheme, which requires Elliptic curves, the device must have support for it.

ECC is another important class of cryptographic techniques known for its efficiency and strong security. The device should support ECC operations for the digital signature scheme adopted.

ECC offers shorter key lengths compared to traditional cryptographic algorithms while maintaining a similar level of security, making it suitable for resource-constrained devices. However, there might be a signification computational costs involved to perform Elliptic Curve operations.

Furthermore, the project requires support for Pairing-Based Cryptography (PBC), which is needed to implement the CP-ABE scheme chosen. The selected device should include the necessary computational capabilities to handle PBC efficiently and have readily available support of Big Number libraries to implement the selected scheme.

In summary, the desired embedded microcontroller or microprocessor for this project should meet the following criteria:

- Onboard Wi-Fi transceiver supporting the required Wi-Fi standards.
- Capabilities for AES encryption and decryption with key lengths of 128, 192, and 256 bits.
- Capabilities for ECC-based operations, for implementing Schnorr Digital Signatures.
- Computational capabilities to handle Pairing-Based Cryptography (PBC) efficiently.
- Good supporting libraries to implement all of the above.

Considering these requirements helps narrow down the search and allow for the selection of an embedded microcontroller or microprocessor that best fits the needs of the project.

6.1.2 Espressif Semiconductor's ESP32

One viable option that meets the specified requirements is the ESP32 microcontroller developed by Espressif Semiconductors [29]. The ESP32 is a popular and versatile microcontroller widely used in Internet of Things (IoT) applications due to its comprehensive feature set and robust performance. The following data is provided in the Datasheet of the ESP32 microcontroller [29] The ESP32 microcontroller includes an integrated Wi-Fi transceiver, making it suitable for wireless data transmission over Wi-Fi networks. It supports various Wi-Fi standards, such as 802.11b/g/n, providing compatibility with a wide range of Wi-Fi networks. With its onboard Wi-Fi capabilities, the ESP32 enables seamless connectivity and data exchange without the need for an additional Wi-Fi transceiver, reducing energy consumption and simplifying the hardware setup. Further, the ESP32 supports the transmission and reception of custom-built MAC layer packets which is essential to the project.

In terms of cryptographic operations, the ESP32 offers hardware acceleration for AES encryption and decryption. It supports key lengths of 128, 192, and 256 bits, allowing flexibility in choosing the appropriate level of security for data encryption. The AES hardware acceleration enhances the overall performance of cryptographic operations, ensuring efficient and secure data transmission.

Moreover, the ESP32 includes an integrated cryptographic co-processor that supports various cryptographic algorithms. This feature enables an efficient implementation of ECC-based operations, such as the Schnorr digital signature scheme mentioned in the requirements.

The ESP32 microcontroller also features a built-in hardware-based Random Number Generator (RNG), which is essential for implementing cryptographic systems. Cryptographic algorithms heavily rely on high-quality random numbers for key generation, nonce generation, and other security-sensitive operations. The RNG on the ESP32 provides a reliable source of random numbers, ensuring the cryptographic operations performed on the device are secure and resistant to attacks that rely on predictable or weak key material. The availability of a robust and hardware-based RNG on the ESP32 adds an extra layer of security to the overall cryptographic system implemented on the device.

Additionally, the ESP32 offers sufficient computational capabilities to handle Pairing-Based Cryptography (PBC). While it does not have dedicated hardware acceleration for PBC, the ESP32's dual-core Xtensa LX6 processor with up to 240 MHz clock frequency provides ample processing power for performing PBC operations efficiently. Furthermore, the ESP32 ecosystem

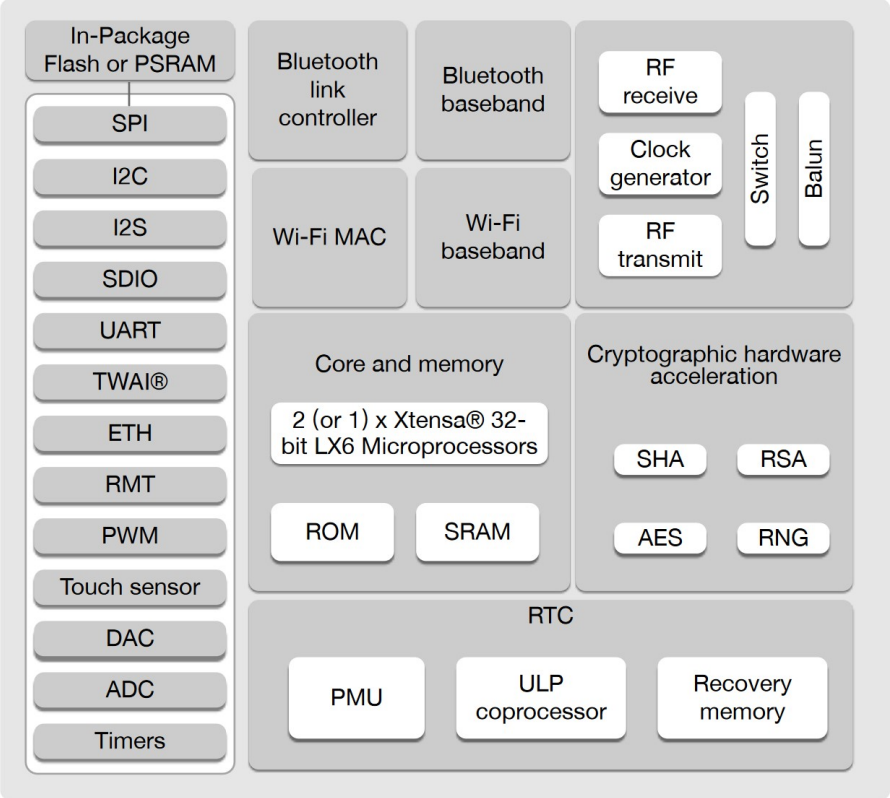


Figure 6.1: Functional Block Diagram of ESP32

provides libraries and resources that can be utilized to implement the required PBC functionality effectively.

The functional block diagram of the ESP32 microcontroller, shown in Fig. 6.1, provides a comprehensive overview of its features and capabilities. This diagram showcases the integrated Wi-Fi transceiver, cryptographic hardware acceleration for AES, hardware-based Random Number Generator (RNG), and other key components.

Espressif Semiconductors provides an extensive software development kit (SDK) for the ESP32, along with comprehensive documentation and support. The SDK includes libraries for Wi-Fi connectivity, AES encryption, ECC operations, and other essential functionalities, simplifying the implementation of the project requirements. The availability of reliable and well-maintained libraries contributes to the development process and reduces the implementation effort.

Considering its onboard Wi-Fi transceiver, hardware acceleration for AES encryption, support for ECC-based operations, and adequate computational capabilities for handling PBC, the ESP32 microcontroller from Espressif Semiconductors appears to be a suitable choice for the project requirements. Its robust feature set, comprehensive SDK, and extensive community support make it an attractive option for developing Wi-Fi-enabled devices with cryptographic capabilities.

6.2 Packet Structure

In order to ensure effective communication and interoperability between RID systems, it is essential to establish a well-defined packet structure for the messages. The packet structure serves as the foundation for transmitting and receiving RID-related information between UAV and observers. This section focuses on the design and definition of the SNELL based message packet structure, which encompasses the organization, format, and specific fields required to encapsulate relevant

data. By establishing a standardized packet structure, stakeholders in the unmanned aviation industry can facilitate seamless implementation based on SNELL, enable efficient data exchange, and promote the reliable identification and tracking of UAV in airspace systems.

Considering that remote identification data should be transmitted through Wi-Fi packets, it is important to take into account the size limitations imposed by the Maximum Transmission Unit (MTU) of the ESP32 microcontroller. Through analysis of the ESP32 SDK code and conducting experiments, it has been determined that the MTU for ESP32 is 1500 bytes, which includes both the Data Frame headers and the Frame Check Sequence (FCS). Further investigation reveals that the mandatory header occupies 22 bytes, and an additional 4 bytes are reserved for the FCS at the end of the packet. Consequently, this leaves us with a total payload space of 1474 bytes within each Wi-Fi packet for our message. Adhering to this limitation, it is crucial to design the message structure with efficiency and optimization in mind to make the most effective use of the available space and ensure successful transmission and reception of the RID-related data. We also have to consider the Schnorr signature scheme, a widely recognized digital signature algorithm known for its simplicity and robustness. With the Schnorr signature scheme, the signer utilizes their private key to generate a digital signature, while the validity of the signature can be verified by anyone possessing the corresponding public key. In this scheme, the signature consists of two components: the R and the s values (see section 2.7.1). These values are derived from the signer's private key, the message being signed, and other relevant parameters. Typically, the Schnorr signature occupies 64 bytes of data, with each component represented by 32 bytes. This signature must be at the end of the message.

Adding a timestamp in the packet serves the purpose of enhancing authentication and preventing replay attacks. By including a 4-byte timestamp in UTC format, we introduce a time-based element that enables the recipient to validate the freshness of the data. This helps verify the authenticity of the packet, ensuring it is recent and has not been reused from a previous session.

Following the design outlined in the SNELL paper, the remaining sections of the packet adhere to a similar structure based on the rules by the FAA [30]. These sections include essential components such as a 4-byte field for the UAV's unique ID (UID), a 20-byte section for the drone data, and a 1-byte emergency code. The UID serves as a distinct identifier for each UAV, facilitating proper identification and tracking within the system. The 20-byte drone data section accommodates relevant information pertaining to the UAV, such as its position, altitude, and velocity. The 1-byte emergency code allows for the indication of emergency situations, enabling timely response and appropriate actions.

Next, the packet encompasses the encrypted GCS data and the CP-ABE cryptography materials. The GCS data, encrypted using AES encryption, contains information about the location of the ground control station. The CP-ABE cryptography materials comprise the components to obtain the key for decrypting the GCS data, allowing for secure and fine-grained access control.

By following a similar structure to that proposed in the SNELL paper, we ensure consistency and compatibility with existing standards and practices, promoting interoperability and effective communication within the remote identification system. This structured packet design allows for the seamless integration of cryptographic operations, laying the foundation for a secure and privacy-preserving remote identification framework.

The overall packet structure is presented in Fig. 6.2 below. The name of each data field and the size in bytes is mentioned. Here, the size of the CP-ABE data is not fixed and varies depending on the number of attributes in the policy and also the implementation. However, from the MTU and the size of the other fields, we can calculate that the maximum size of this can be 1365 bytes.

Data Field	Data Frame Header	UID	UAV Data	encrypted GCS Data	CPABE Data	Time Stamp	Emergency code	Schnorr sign	FCS
Length (bytes)	22	4	20	16	[0-1365]	4	1	64	4

Figure 6.2: SNELL packet structure

6.3 Software and Libraries

The ESP32 microcontroller supports various programming languages such as Arduino, Micropython, and Lua, along with versatile Integrated Development Environment (IDE) like Arduino IDE, ESP-IDF, and PlatformIO. From a cybersecurity perspective, selecting Arduino as the software tool for programming the ESP32 offers several advantages. Arduino's extensive library ecosystem and the active developer community ensure access to secure and vetted code libraries, minimizing the risk of introducing vulnerabilities. The Arduino IDE provides a controlled environment for code development, compilation, and uploading, reducing the likelihood of introducing unintended security flaws during the development process. Additionally, Arduino's commitment to documentation, tutorials, and support forums allows developers to enhance their cybersecurity knowledge and implement secure coding practices. By adopting device-agnostic coding practices, we design code that is not tightly coupled to the microcontroller, enabling easier evaluation of potential security vulnerabilities and ensuring the ability to apply security patches and updates across different platforms. Considering these factors, Arduino emerges as a suitable choice for programming the ESP32, enabling the development of a secure and resilient remote identification system. The various steps of the SNELL-based system require various cryptographic functions. They can be summarized as:

1. Random Number Generation
2. Big number Modulo Arithmetic
3. Symmetric Encryption
4. Elliptic Curve Operations
5. Pairing Based Operations
6. Hashing functions (and hashing to curve point)
7. Key Derivation Functions

For the best results, the hardware-based Random Number Generator (RNG) of the ESP32 chip is used as the source of the randomness. Thus the libraries that are going to be used must be able to perform the other operations.

Almost all of the cryptographic libraries available support Big number Modulo Arithmetic. Thus, the libraries required for ECC and PBC satisfy the requirements. A general cryptographic library should encompass the rest of the requirements of Symmetric encryption, Hashing, and Key Derivation Functions.

Mbed-TLS

Mbed-TLS was chosen for its extensive support for cryptographic operations, including big number modulo arithmetic, Elliptic Curve operations, and hashing functions (SHA). These functionalities were essential for the implementation of the Schnorr algorithm, which forms a crucial component of the SNELL protocol. Mbed-TLS is known for its reliability, actively maintained codebase, and comprehensive documentation, making it an attractive choice for embedded systems like ESP32. Additionally, a comparative study on ECC libraries for embedded devices [67] provided positive insights into the capabilities of Mbed-TLS. The AES encryption implementation offered by Mbed-TLS is used for the symmetric encryption part of the system to encrypt the GCS location.

MIRACL Core

In the quest for an efficient solution for Pairing Based Operations, MIRACL Core [64], a specialized library for elliptic curve cryptography, was considered. While working with MIRACL Core for the CP-ABE implementation, it became apparent that MIRACL Core also provided suitable support

for the Schnorr algorithm. This realization led to the acknowledgement that MIRACL Core could have been utilized for both the Schnorr algorithm and the CP-ABE scheme, providing a more consistent and streamlined approach.

To justify the selection of MIRACL Core for the CP-ABE scheme, the justification provided in the thesis in [75] was adopted. The work highlighted the suitability of MIRACL Core for pairing-based operations and its applicability to advanced cryptographic schemes. Given the necessity for pairing operations in the CP-ABE scheme and the positive evaluation of MIRACL Core's capabilities, it was deemed the best option for achieving the desired functionality and efficiency in the implementation. Further, MIRACL Core also has support for two Key Derivation Functions of which KDF2 [5] is used in our implementation.

By combining Mbed-TLS and MIRACL Core, the implementation was able to address all cryptographic requirements of SNELL, including the generation of secure random numbers, big number modulo arithmetic, symmetric encryption, elliptic curve operations, pairing-based operations, hashing functions, and key derivation functions. This comprehensive cryptographic foundation ensured the security and integrity of the SNELL RID system, allowing for selective disclosure of the pilot location to authorized observers.

Wireless Message Transmission

In this project, the ESP32 Wi-Fi transmission function **esp_wifi_80211_tx( )** provided by the Espressif SDK [70] plays a crucial role in transmitting the message. Utilizing the ESP32's built-in WiFi capabilities, this function allows custom Wi-Fi frames to be sent, enabling the implementation of the custom SNELL protocol. Once prepared, the encrypted and signed message is formatted into the appropriate Wi-Fi frame structure by adding the Data Frame headers and sent using the **esp_wifi_80211_tx( )** function.

6.4 Our Work

Our work only uses publicly available and open-source software and libraries. The code for the implementation and testing of our work along with other codes used during the project are made available publicly under GNU Public License on our GitHub repository [33].

Chapter 7

Performance Evaluation and Results

The SNELL RID system has been successfully implemented and tested on the ESP32 DevKit V1 board which comes with the ESP32-WROOM-32 module [28] that contains the ESP32 microcontroller along with 4MB of Flash memory. This chapter presents a comprehensive evaluation of the system's performance, focusing on execution times and energy consumption. Each component and cryptographic operation within the system has been thoroughly evaluated to assess its efficiency and effectiveness. The results obtained from these evaluations are documented and analyzed to gain insights into the system's overall performance.

7.1 Implementation Strategies

Before discussing the details of the CP-ABE implementation, it is essential to calculate the size of the CP-ABE cryptographic materials. In the previous chapter, the section on RID packet structure (Sec. 6.1.2) talks about the various fields in the Wireless Fidelity (Wi-Fi) packet and only gives a range for the CP-ABE cryptographic materials. Here, the size of this part of the message is calculated. The implemented CP-ABE Encryption generates the following as the output: $C := (C_0, C_1, C_2, \{C_{3,i}\}_{i=1}^{n})$ (see 2.6.1)

- one point on $G_T : C_0$ (Cp)
- two points on $G_2 : C_1, C_2$ (g_s0, h_s1)
- an array of n points on $G_1 : \{C_{3,i}\}_{i=1}^{n}$ (ct)

In addition to the data fields mentioned above, the Policy used for encryption also needs to be included in the message. However, the size of these data fields varies depending on the pairing curve used. The Tab. 2.1 provides the sizes of the points in each of the fields G_1, G_2, and G_T for some well-known Pairing Friendly curves.

In the MIRACL Core library, points in G_1 and G_2 are represented as ECP and ECP2 points, respectively, while G_T points are represented as FP12 points [64]. The library offers a compression option where only the first coordinate of the ECP and ECP2 points is stored, and the second coordinate of the point can be calculated when needed. This feature significantly impacts the system's efficiency and helps reduce the message size during transmission.

Considering the compression option where only half the original size of the point (ECP/ECP2) plus an extra byte is required, the size of the CP-ABE cryptographic materials is given by Eq. 7.1

$$2 \cdot (\frac{size(G_2)}{2} + 1) + size(G_T) + n \cdot (\frac{size(G_1)}{2} + 1) + size(Policy) \tag{7.1}$$

Tab. C.1 presents the calculated packet sizes, including the communication overhead, for different Pairing Friendly Curves (PFCs) with varying numbers of attributes in the Policy, excluding the size of the Policy itself. Upon analyzing the table, it becomes evident that only curves BN254 and BLS12-381 offer policies of a size that can be reasonably expressed in a real scenario. Between these two options, BN254 is selected for implementation and evaluation, primarily because it offers comparable security to BLS12-381 while being faster in terms of computational performance [75]. In appendix H, the comparison of execution times of the implementation for both curves is presented. From Fig.H.1 and Fig.H.2, it is clear that the execution times of BN254 are lower than those of BLS12-381. Therefore, BN254 proves to be the more efficient and practical choice for the SNELL RID system operating on the ESP32 microcontroller.

A policy of decent size approaches approximately a hundred bytes of data (which can be further reduced using shorthand notations). From the table, it is evident that the packet size with $n = 23$ attributes is 1408 Bytes, leaving 92 Bytes for the policy. This is fixed as the limit for the number of attributes possible in the Policy.

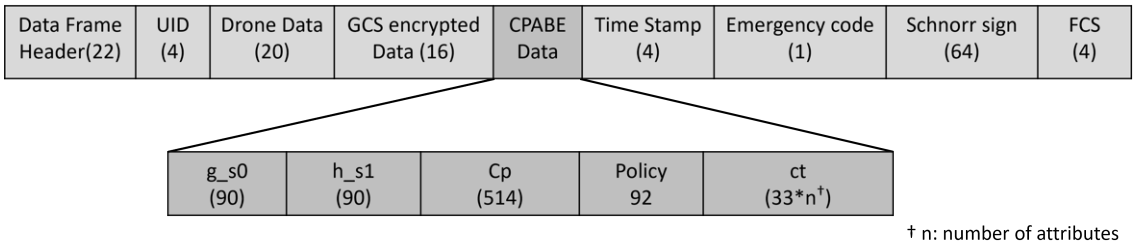


Figure 7.1: SNELL packet structure with CP-ABE data.

The implementation of the SNELL RID system on the ESP32 microcontroller follows a bottom-up approach, gradually increasing the onboard computation. This approach leads to the development of three versions of the system:

Fully Precomputed Mode

In this version, all the CP-ABE cryptographic materials and the corresponding AES keys are precomputed on an unconstrained device and stored in a file on the microcontroller. During the Online phase, the microcontroller accesses the precomputed data from the file and utilizes the AES key to encrypt the GCS location data. The encrypted data, along with the CP-ABE cryptographic materials, is then assembled into the message packet. The microcontroller generates the Schnorr signature of the packet and finally, the packet is transmitted.

The Fully Precomputed mode can be useful in scenarios where the cryptographic computations and key generation are time-consuming or resource-intensive. By precomputing all the cryptographic materials on an unconstrained device, the microcontroller can offload these computations to a more powerful system. This can result in faster execution times on the microcontroller during the Online phase, as it only needs to access the precomputed data and perform the Schnorr signature generation, which is relatively less resource-intensive.

However, this strategy comes with certain trade-offs. One of the main trade-offs is the increased complexity and management overhead of storing and accessing the precomputed data on the microcontroller. Additionally, the fully precomputed approach may require more storage space on the microcontroller to store the precomputed cryptographic materials, which could be a limitation for devices with limited memory.

Furthermore, since the precomputed data is stored on the microcontroller, there could be potential security risks if the device is compromised. If an attacker gains access to the precomputed data, they might be able to recover sensitive cryptographic materials and compromise the system's security.

Overall, the Fully Precomputed mode is a trade-off between offloading resource-intensive computations to an unconstrained device to improve performance and the increased complexity and potential security risks associated with managing precomputed data on the microcontroller.

Partially Precomputed Mode

In this version, all the cryptographic computations are performed on the microcontroller itself except for the computation of the MSP (Monotone Spanning Program). All the key steps in this version along with their average computation times are given in F.

To optimize execution times and accommodate the maximum number of attributes in the policy (limited by the MTU of the Wi-Fi packet, 1500 Bytes), the random nonce r on G_T is precomputed and stored on the microcontroller. During the Online phase, this precomputed nonce is used, speeding up the execution time. The details are discussed in the next section.

While the "Partially Precomputed" mode still incurs some overhead in terms of storing and accessing the precomputed data on the microcontroller, it is significantly reduced compared to the "Fully Precomputed" version. By focusing solely on precomputing the random nonce, the memory requirements are considerably lower, making it a more feasible option for constrained microcontroller platforms.

Moreover, from a security perspective, this mode introduces a valuable advantage. Storing and utilizing only the random nonce on G_T removes the need to handle and store sensitive cryptographic materials and AES keys on the microcontroller. As a result, the exposure of critical security information is minimized, reducing potential security risks associated with unauthorized access to cryptographic materials. In the event of a device compromise, an attacker would only gain access to the random nonces, which hold no intrinsic value in themselves. This provides an added layer of protection for the SNELL RID system and the privacy of drone pilots.

The "Partially Precomputed" mode not only demonstrates improved efficiency in execution times but also underscores its role in enhancing the security and integrity of the SNELL RID system.

Parallel Computed

The "Parallel Computed" version presents a compelling approach by harnessing the dual-core architecture of the ESP32 microcontroller to significantly accelerate cryptographic computations. By dividing the computational task between the two cores, this mode efficiently utilizes the available processing resources, resulting in faster execution times. This approach capitalizes on the specific hardware capabilities of the ESP32, making it a targeted solution rather than a device-agnostic one. The division of the computational task between the two cores of the ESP32 is shown in Fig. E.1.

The adoption of the parallelized strategy is particularly beneficial in scenarios where faster cryptographic computations are of utmost importance, such as real-time or time-sensitive applications. By simultaneously utilizing both cores, the computational tasks can be executed in a more rapid manner, resulting in reduced processing time for generating the cryptographic materials. This offers a notable advantage for quicker responses and minimizing latency.

However, while the parallelized version offers significant performance improvements, it does come with trade-offs, notably an increase in power consumption. The simultaneous operation of both cores for computational tasks requires higher power utilization, which can have implications for UAVs which are primarily battery-powered devices. Thus, when deploying the parallelized mode, careful consideration should be given to the energy requirements and overall power efficiency of the system.

Comparing the "Parallel Computed" version with the previous modes, it demonstrates faster cryptographic computations than the "Partially Precomputed" mode, outperforming that strategy in terms of time efficiency. On the other hand, the Fully Precomputed mode provides enhanced security by completely removing the need to store sensitive cryptographic materials on the microcontroller.

The trade-off between performance and security could be crucial in selecting the most suitable mode for a specific application. The decision should be based on the specific requirements, constraints, and use-case scenarios to strike a balance between faster cryptographic computations and the preservation of data security and privacy.

7.1.1 Schnorr Signature

The Schnorr Signature scheme is tested with various elliptic curves: SECP256k1, SECP256r1, SECP384r1, and SECP521r1, assessed for their suitability for the scheme and the SNELL RID system. For each curve, the size of the signature is calculated and presented in the Tab. D.1. The execution times and energy consumption for these are evaluated and discussed in the following sections. In all subsequent parts of the evaluation, the Schnorr Signature implementation uses SECP256k1, resulting in a signature size of 64 bytes.

7.2 Execution Time of the SNELL Protocol

7.2.1 Execution time

We define the execution time of the protocol as the total time required from the start to the finish of the SNELL RID message. This includes the data/file access (if needed) (FA), CP-ABE encryption (ABE_Enc), key derivation (KD), AES encryption (AES_Enc), Schnorr Signature generation ($Sign$), packet assembly ($Assy$) and packet transmission (Txn). Depending on the implementation mode, some of these might not be present.

The evaluation of the SNELL RID system's execution times was conducted on the ESP32 microcontroller for all three implemented modes: "Fully Precomputed," "Partially Precomputed," and "Parallel Computed." 100 tests are conducted for each number of attributes (n = 1 to 23) in all three modes.

In the Fully precomputed mode, accessing the precomputed data from the file storage during run time incurs some overhead. This overhead, coupled with the cryptographic operations and the Schnorr signature generation, collectively contributes to the overall execution time given in Eq. 7.2.

$$Execution\_time(Fully\_Precomuted) = FA + AES\_Enc + Assy + Sign + Txn \quad (7.2)$$

Fig. 7.3a shows the total execution time for the different number of attributes possible in the Policy in the Fully precomputed mode. There is a clear increasing trend in the execution times as the number of attributes increases. Some of the average times are given in Tab. 7.1. The total computation time ranges from 293.9ms at n=2 to 307.8ms at n=23.

n	FA+AES_Enc+Assy (ms)	Sign (ms)	Total (ms)
2	12	281.9	293.9
6	14.8	281.9	296.7
10	15.9	282	297.90
13	19.5	282.2	301.7
17	20.9	282.2	303.1
23	25.4	282.4	307.80

Table 7.1: Average times for some number of attributes in the policy in Fully Precomputed Mode.

The "Partially precomputed" mode has more computations taking place on the ESP32 microcontroller. The execution time for this mode is given by Eq. 7.3.

$$\begin{aligned} Execution\_time(Partially\_Precomputed) =& FA + ABE\_Enc_{P.P} + KD + AES\_Enc \\ &+ Assy + Sign + Txn \\ ABE\_Enc_{P.P} =& RNG + g\_s0h\_s1\_Comp + CP\_Comp + ct\_Comp \end{aligned} \tag{7.3}$$

where, $ABE\_Enc_{P.P}$ is the execution time of CP-ABE encryption with rnonce precomputed. This includes the execution time for random number generation of n+1 random numbers in Z_p (RNG), along with the other cryptographic computations which are part of the FABEO encryption algorithm - $g\_s0h\_s1\_Comp$, CP_Comp, and ct_Comp. In this mode, the CP-ABE Encryption time adds to the execution time from the previous mode. However, there is a reduction in the file access time as only the random nonce r which is 384 Bytes long, needs to be retrieved instead of the entire CP-ABE cryptographic data along with the AES key, which ranges from 579 Bytes to 1305 Bytes. The computation time for the CP-ABE Encryption is the largest part of the execution time in this mode and it increases with the increase in the number of attributes which is reflected in the Fig. 7.3b. Tab. 7.2 provides the average times of each computation part involved for some number of attributes in the Policy for the "Partially precomputed" mode. Here, at n=2, the computation time equals 913.5ms and increases till 2961.8ms at n=23. The significant increase in the time is attributed to the onboard computation of the CP-ABE and cryptographic materials.

n	FA (ms)	CP-ABE.Encrypt (ms)	KD+AEC_Enc (ms)	Assy (ms)	Sign (ms)	Txn (ms)	Total (ms)
2	31.1	565.9	0.3	16.4	291.7	0.2	913.8
6	31.1	925.3	0.3	31.8	291.8	0.2	1298.0
10	31.1	1309.5	0.3	47.2	291.7	0.2	1707.0
13	31.1	1573.6	0.3	58.8	291.7	0.2	1989.5
17	31.1	1905.5	0.3	74.2	291.8	0.2	2346.2
23	31.1	2484.3	0.3	97.3	291.8	0.2	2962.1

Table 7.2: Average times for various number of attributes in the policy in Partially Precomputed Mode.

In the "Parallel Computed" mode, all the computations are done on the ESP32 microcontroller itself (except the MSP computation), utilizing the available two cores. Due to data dependencies, the Packet Assembly, Schnorr Signature generation, and Packet transmission are done on a single core and at the end of the process. An example of the process for n = 12 attributes is given in the timing diagram in Fig. 7.2. As additional tasks are running on Core 1, the total execution time becomes the time taken by the tasks running on Core 1. Thus, the execution time can be estimated as given in Eq. 7.4. The execution time in this mode also includes the execution time for the generation of the ephemeral random nonce ($rnonce\_Gen$). Fig. 7.3c shows the results. It is evident that there is a significant improvement in the computation time in this mode. For n=2 it is 778.79ms which is around 135ms lower than that of Partially Precomputed mode. This improvement becomes more evident with the increase in the number of attributes in the policy. At n=23, there is a very high difference of almost 1075ms.

It is worth noting that the observed trend does not hold for n=1 and n=2. Surprisingly, the execution time for n=1 is longer than that for n=2. After a thorough examination of the implementation, it is determined that the behavior is influenced by the WiFi processes running concurrently on the ESP32. The WiFi drivers, operating with the highest priority in the background, inadvertently impact our system's performance. This is confirmed when the implementation is evaluated with WiFi drivers disabled, resulting in similar execution times for n=1 and n=2. Further analysis revealed that this effect reduces with an increase in the number of attributes.

$$\begin{aligned} Execution\_time(Parallel\_Computed) =& ABE\_Enc_{P.C} + KD + AES\_Enc + Sync \\ &+ Assy + Sign + Txn \\ ABE\_Enc_{P.C} =& rnonce\_Gen + 0.5 \cdot ct\_Comp \end{aligned} \tag{7.4}$$

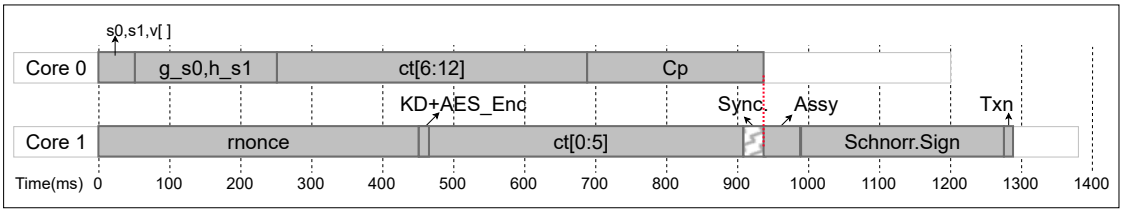
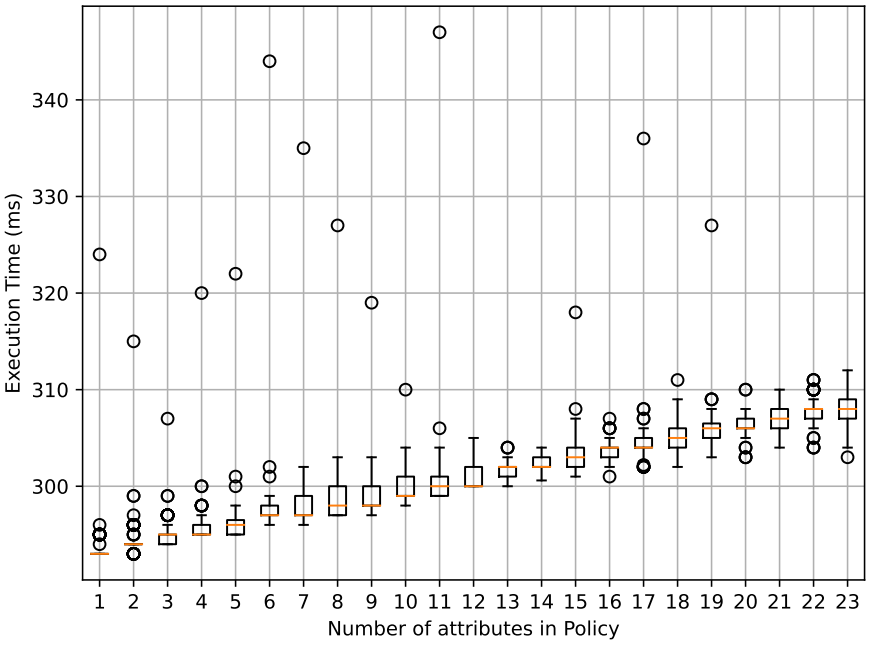


Figure 7.2: Gantt chart for n = 12 in Parallel computed mode.

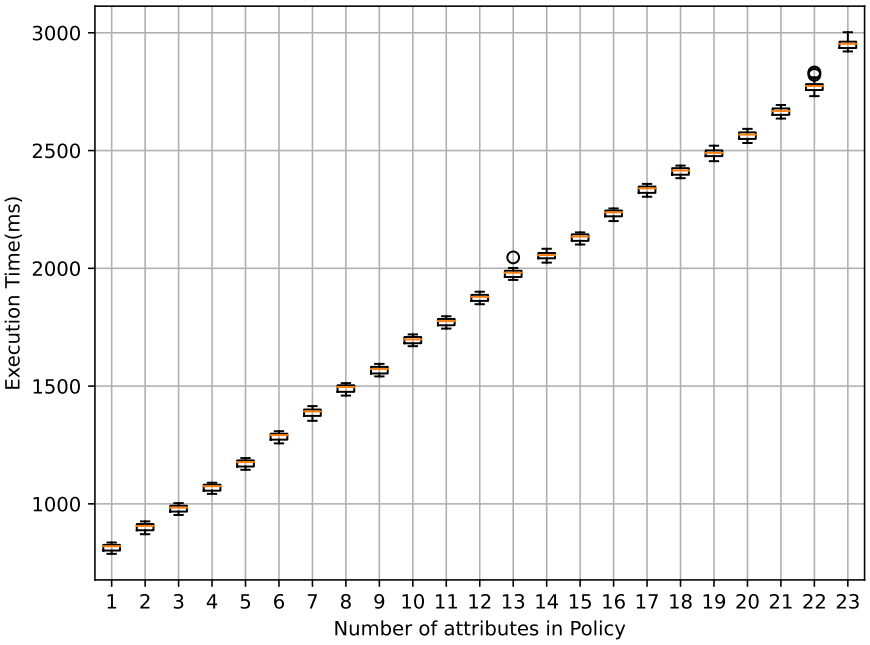
n	Core 0 (ms)	Core 1 (ms)	Sync (ms)	Assy (ms)	Sign (ms)	Txn (ms)	Total (ms)
2	484.3	478.58	1.31	8.1	282.04	0.17	778.79
6	705.45	626.66	0.03	23.51	283.89	0.17	987.34
10	1100.99	626.96	0.4	38.92	283.89	0.17	1195.3
13	1372.62	626.71	0.03	50.48	283.92	0.17	1373.39
17	1167.23	1216.84	0.03	65.89	283.9	0.17	1575.43
23	1475.18	1505.86	0.39	89.01	283.86	0.17	1887.89

Table 7.3: Average times for some number of attributes in the policy in Parallel Computed Mode.

The average execution times for different numbers of attributes in the policy for all three modes are depicted in the graph in Fig.7.3d. It can be observed that all the execution times in all three modes stay below the 3-second mark satisfying the time requirement (see 3.4). The average times of the operations taking place on the two cores for different attributes in this mode are presented in Tab. 7.3.

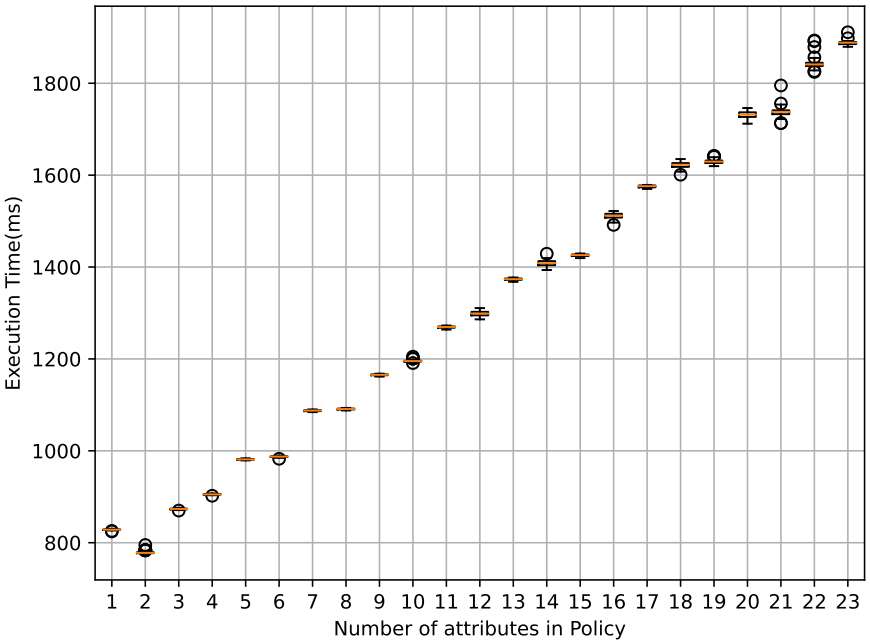


(a) Fully Precomputed version.

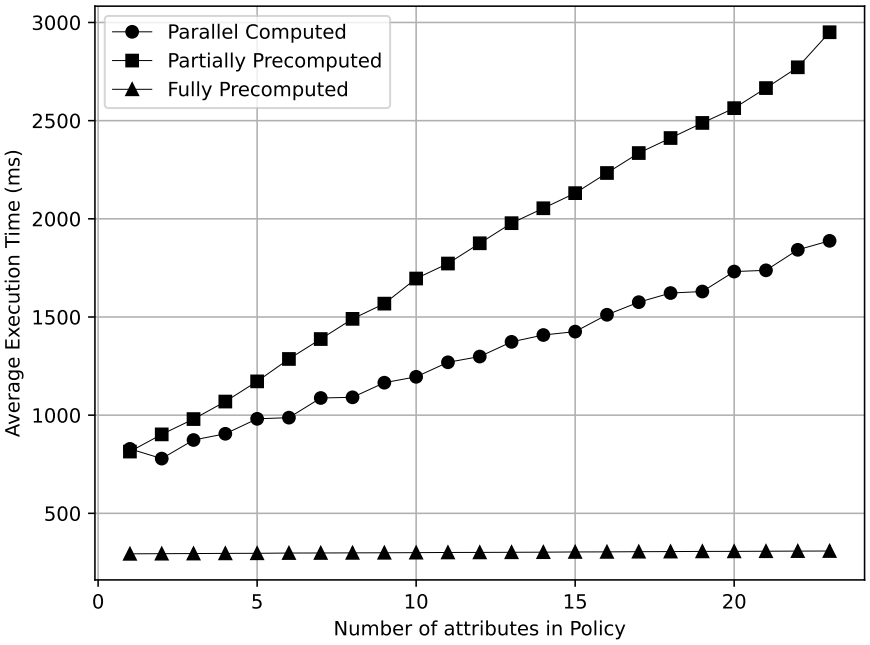


(b) Partially Precomputed version.

Figure 7.3: Evaluation of Execution Times.



(c) Parallel compute mode.



(d) Average Execution Times for different versions and attribute counts.

Figure 7.3: Evaluation of Execution Times.

7.3 Energy Consumption of the SNELL protocol

The evaluation of energy consumption for the SNELL RID system is an essential aspect to understand the system's impact on the battery level when implemented on a UAV. This section presents the evaluation setup followed by the results obtained for the three modes.

Then the electrical energy consumed by a load can be calculated by Eq. 7.5.

$$\begin{aligned} E[mJ] &= V \cdot \int_0^T i(t)dt \\ &= V \cdot i \cdot (t_2 - t_1) \end{aligned} \tag{7.5}$$

where E is consumed Electric Energy in Joules, V is Voltage across the load in Volts, i is current drawn the load in amps and t is execution time in seconds.

To assess the energy consumption accurately, we first need to find the amount of current passing through the microcontroller and the voltage across it. Along with these, the time duration of execution also needs to be measured. The Voltage supplied by a battery remains constant and is known to be 7.4V. A Digital Storage Oscilloscope (KEYSIGHT Technologies DSO-X 3014T) has been used for the other two measurements. The current cannot be directly measured with the oscilloscope. So, a resistor of known value is connected in series with the ESP32 microcontroller, and the voltage across this is measured. The current is then calculated using Ohm's Law.

The circuit is shown in Fig. 7.4 (left). A resistor of 10Ω is connected in series to the microcontroller. The oscilloscope probes are connected in parallel across the resistor (shown by the voltmeter symbol). From this, the final formula to calculate the Energy is given by Eq. 7.6:

$$E(J) = 7.4 \cdot (V/10) \cdot \Delta t \tag{7.6}$$

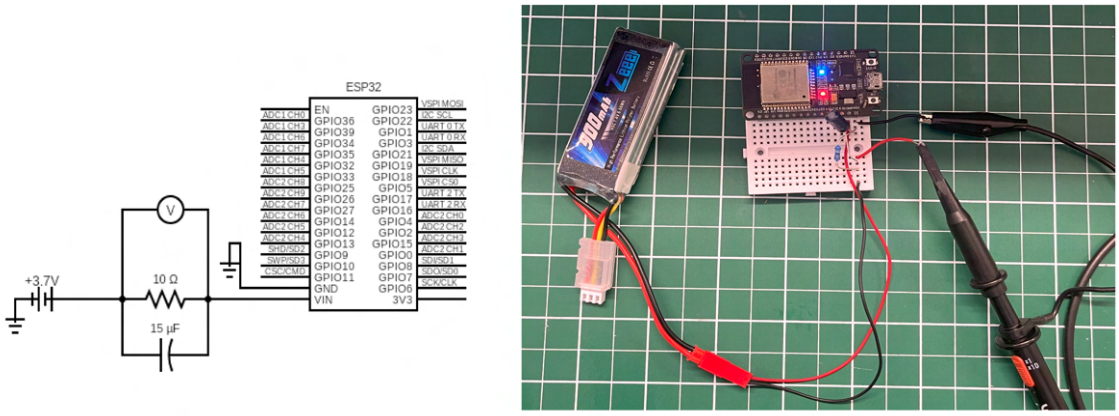


Figure 7.4: Evaluation Circuit, theoretical (left) and actual (right).

The actual evaluation circuit is shown in Fig. 7.4 (right) and as an example, the oscilloscope output for n = 13 for all the three modes is given in Fig. G.1. The Tab. 7.4 give the average values of energy consumption for various number of attributes in the Policy for the three modes. It can be observed that the "Fully precomputed" mode takes the least time and current as there are no heavy computations running on the microcontroller. The "Partially Precomputed" mode is the next in current consumption while it takes longer time for the computations. Finally, the " Parallel computed" mode draws the highest amount of current among all the modes but the computation time is lesser than the previous mode.

The first three graphs in Fig.7.5 show the energy consumption in all three modes individually and the last graph (Fig.7.5d) shows the average energy consumption in all three modes. In this

n	Fully Precomputed (mJ)	Partially Precomputed (mJ)	Parallel Computed (mJ)
2	75.37 (0.00031%)	237.16 (0.00099%)	385.38 (0.00161%)
6	76.27 (0.00032%)	337.27 (0.00141%)	462.96 (0.00193%)
10	76.84 (0.00032%)	444.75 (0.00186%)	539.74 (0.00225%)
13	77.24 (0.00032%)	518.43 (0.00216%)	605.58 (0.00253%)
17	78.36 (0.00033%)	612.47 (0.00256%)	680.4 (0.00284%)
23	79.95 (0.00033%)	775.2 (0.00323%)	795.53 (0.00332%)

Table 7.4: Average Energy Consumption (along with the percentage charge of the battery) for various numbers of attributes in the three modes.

graph, it is evident that as the number of attributes increases, the energy consumed in both "Partially Precomputed" mode and " Parallel Computed" mode gets closer and closer. This is because the energy consumption depends on both the amount of current drawn by the microcontroller and the time for computation. As this trend continues, after some point, the " Parallel Computed" mode would consume less energy than the "Partially Precomputed" mode, making it the best choice of the two on the ESP32 microcontroller, in terms of time and energy along with security.

Assessing the impact of energy consumption on battery life is crucial in practical UAV applications. By converting the energy consumption values to battery percentages, we gain a better understanding of how the SNELL system affects the overall UAV operation. For our evaluation, we use a 900mAh, 7.4V battery. The total electrical energy present in the battery is given by Eq.7.7:

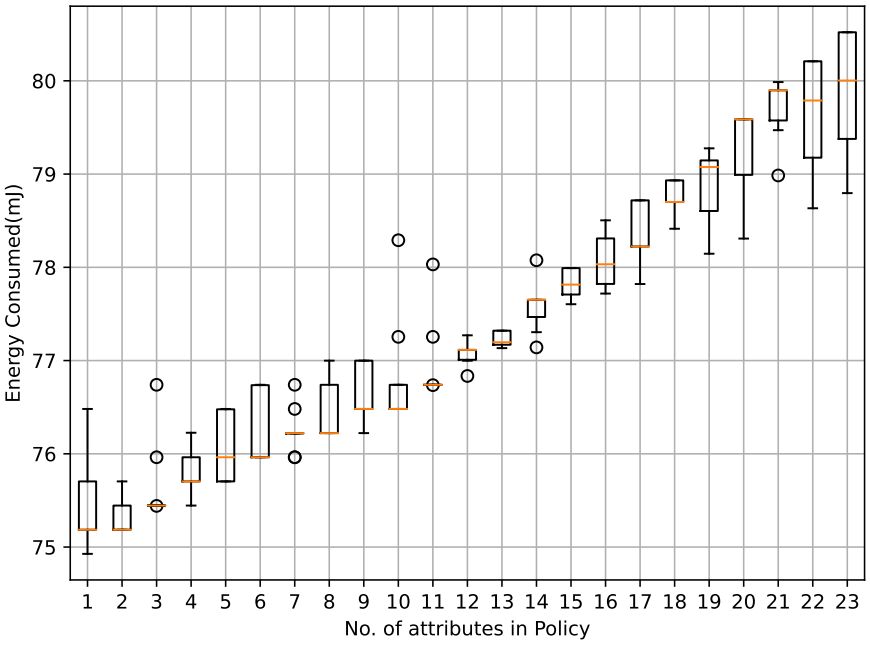
$$\begin{aligned} E &= q \cdot V \cdot 3600[J] \\ &= 0.9 \cdot 7.4 \cdot 3600J \\ &= 239760J \end{aligned} \tag{7.7}$$

We can now calculate what percentage of this total charge is consumed to generate every message.

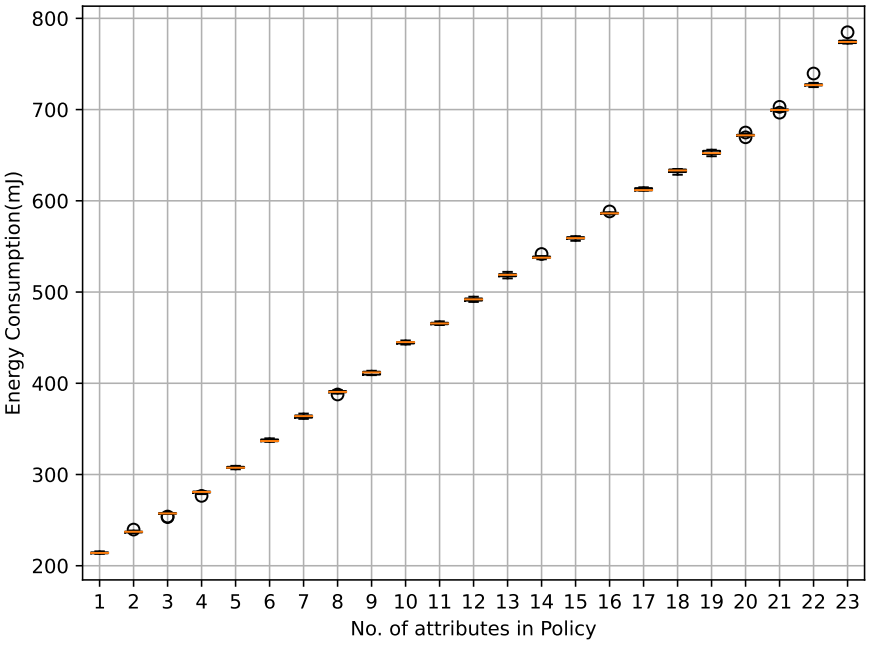
The percentages mentioned in brackets in Tab. 7.4 demonstrate that the energy consumption in all three modes remains incredibly low, with the highest percentage being 0.00332%. This reflects the minimal impact on the battery life of the UAV when using the SNELL system. The energy efficiency of the "Fully Precomputed" mode is particularly noteworthy, as it consumes the least amount of energy, just around 0.0003%. To bring these values into perspective, For one hour of drone operation, with the messages being transmitted every 3 seconds, the Fully precomputed mode consumes a maximum of 0.396%, Partially Precomputed mode consumes 3.88% and Parallel Computed version consumes 3.98%.

7.3.1 Schnorr Signature

The microcontroller draws less current during the computation of the Schnorr signature, compared to the CP-ABE encryption. This is visible in the DSO outputs for Partially Precomputed and Parallel Computed versions in Fig.G.1. Towards the end of the execution process, there is a drop in the signal amplitude which when measured for the time interval coincides exactly with the start and execution of the Schnorr sign algorithm. On average the Schnorr signing takes 283ms and draws 65.74mA of current. From this, the energy consumed to generate the Schnorr signature for a message can be calculated to be around 68.8mJ. This amounts to 86-91.2 percent in the Fully Precomputed version, 8.87-29 percent in the Partially Precomputed, and 8.6-17.8 percent in the Parallel Computed version.

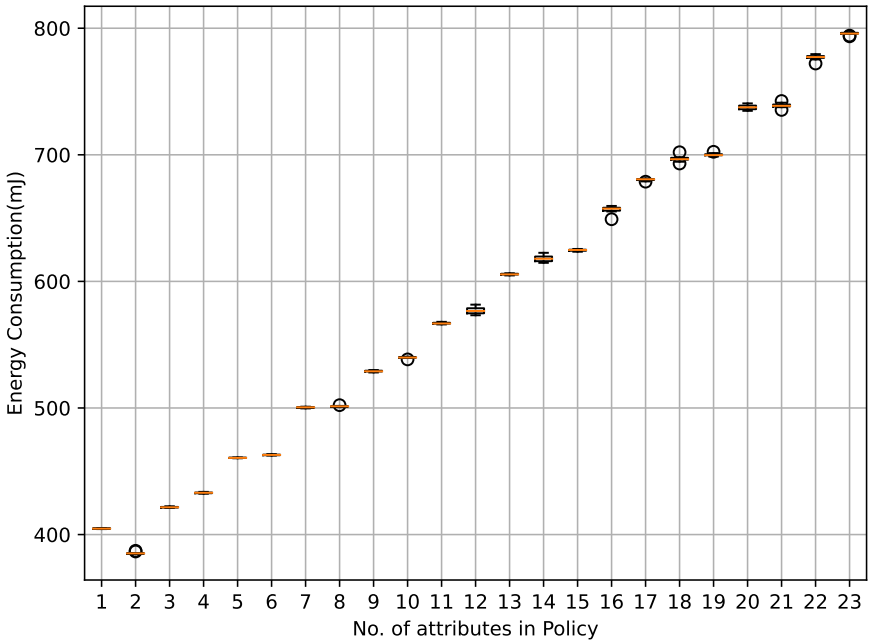


(a) Fully Precomputed version.

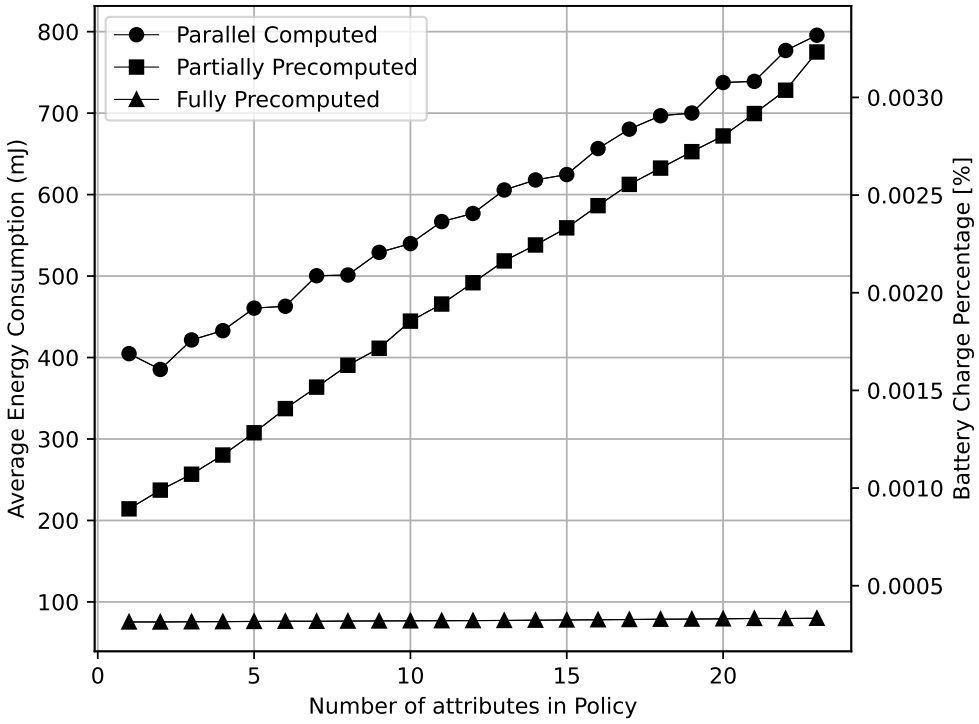


(b) Partially Precomputed version.

Figure 7.5: Evaluation of Energy Consumption.



(c) Parallel Computed mode.



(d) Average Energy Consumption (also expressed as percentage of battery charge) in different versions and attribute counts.

Figure 7.5: Evaluation of Energy Consumption.

7.4 RAM Usage Analysis of the SNELL Protocol

In this section, we present an analysis of the Random Access Memory (RAM) usage for different configurations of our implementation, aiming to understand the memory consumption patterns in relation to various number of attributes and implementation modes. The RAM usage plays a crucial role in evaluating the efficiency and suitability of specific modes to the constraints of the system, and we provide a comprehensive overview of the observed RAM consumption for different scenarios, enabling us to draw valuable insights into memory management and optimization.

Arduino IDE does not have the capability to calculate static RAM sizes. However, the Espressif Semiconductors' Espressif IoT Development Framework (ESP-IDF) provides the tools: "size", "size-components" and, "size-files", which provide a detailed breakdown of the estimated size of the static Data RAM (DRAM), Instruction RAM (IRAM), and Flash from the compiled files. So, the project is compiled using ESP-IDF with Arduino as a component and the results from the "size" tool for all three modes with various number of attributes are presented in the Table7.5 below.

	Fully Precomputed			Partially Precomputed			Parallel Computed		
n	DRAM (KB)	IRAM (KB)	Flash (KB)	DRAM (KB)	IRAM (KB)	Flash (KB)	DRAM (KB)	IRAM (KB)	Flash (KB)
2	29.72	85.99	671.27	33.29	85.99	683.36	32.78	86.32	650.02
6	29.72	85.99	671.27	34.79	85.99	683.37	34.26	86.32	650.03
10	29.72	85.99	671.27	36.39	85.99	683.39	35.88	86.32	650.05
13	29.72	85.99	671.27	37.69	85.99	683.4	37.17	86.32	650.06
17	29.72	85.99	671.27	39.49	85.99	683.38	38.97	86.32	650.04
23	29.72	85.99	671.27	42.43	85.99	683.39	41.91	86.32	650.06

Table 7.5: Memory Usage in the three modes for various number of attributes

It is evident from the analysis that the usage of IRAM remains constant across different numbers of attributes. This consistency arises from the implementation structure, where only the policy changes while the core code remains constant. In contrast, the Flash values display fluctuations that lack a discernible pattern. These variations could be attributed to several factors such as memory alignment and compiler optimizations.

However, the most notable trend is observed in DRAM usage. As shown in the graph depicted in Fig. 7.6, the DRAM Usage varies distinctly among the three modes (Fully Precomputed, Partially Precomputed, and Parallel Computed) as the number of attributes (n) in the policy increases from 1 to 23. In the Fully Precomputed mode, the DRAM consumption remains relatively stable despite the variations in n. This is due to the fact that although file access is present, no CP-ABE encryption is performed, resulting in lower overall DRAM usage. Conversely, the Partially Precomputed and Parallel Computed modes exhibit linear growth in DRAM consumption as the number of attributes in the policy increases. This is particularly noticeable in the Partially Precomputed mode, which records the highest DRAM usage among the three modes. The inclusion of file access could be a contributing factor in this mode, as file operations may require additional memory.

The total available RAM on ESP32 is 520KB out of which 328KB is allocated for DRAM, 128KB for IRAM and, 64KB for cache. The development board used contains 4MB of Flash memory. Out of the 328KB of DRAM, only 160KB is available for static DRAM allocation and the rest 160KB can only be allocated at runtime as heap, leaving 8KB of memory reserved for ROM functions. From this information we calculate that, the maximum static DRAM consumed by the system (at n=23) in the three modes is:

- Fully Precomputed: 18.5%
- Partially Precomputed: 26.52%
- Parallel Computed: 26.199%

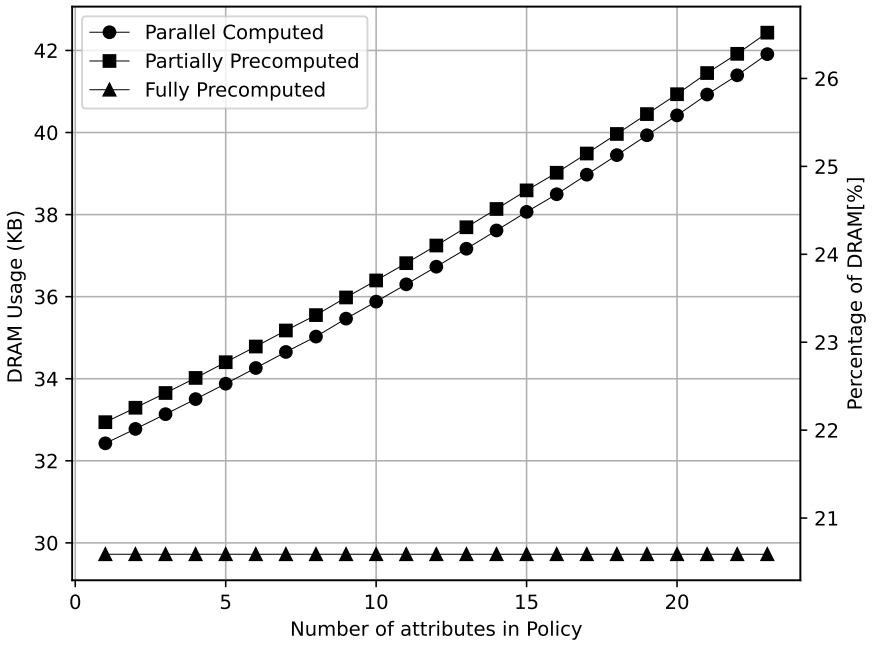


Figure 7.6: DRAM Usage in all three modes for various number of attributes in the policy

7.5 Evaluation Summary

Parameter	Ranking (Highest to Lowest)
Storage	F.P >P.P >P.C
Execution Time	P.P >P.C >F.P
Energy Consumption	P.C >P.P >F.P
RAM Usage	P.C >P.P >F.P
Security Risks	F.P >P.P >P.C

F.P = Fully Precomputed; P.P = Partially Precomputed; P.C = Parallel Computed

Table 7.6: Evaluated Parameters ranked for all the three modes based on the results obtained from the previous sections

Tab. 7.6 presents a comprehensive summary of the evaluated parameters for n=1 to n=23 attributes in the policy. In the Fully Precomputed mode, there is a high demand for memory due to the storage of cryptographic materials, offering advantages such as low computation time and energy consumption. However, this mode compromises on security due to the risk of exposing sensitive materials. On the other hand, the Partially Precomputed mode reduces memory usage but entails increased computation time, energy consumption, and RAM usage due to increased onboard computations. This mode offers moderate security, as it minimizes the storage of sensitive materials. The Parallel Computed mode balances memory and computation, demonstrating high security while slightly increasing energy consumption through the utilization of both cores of the microcontroller. The RAM usage in this mode is nearly comparable but slightly less than that of the Partially Precomputed mode. The choice of mode should consider specific requirements and constraints on hardware and software support to optimize performance and security in the SNELL protocol implementation.

Chapter 8

Conclusion

8.1 Discussion and Conclusion

This report delved into the realm of privacy-preserving solutions for safeguarding the location information of drone pilots within remote identification systems. The focal point was the SNELL protocol, which was not only thoroughly analyzed but also practically implemented and evaluated on the resource-constrained ESP32 microcontroller. The evaluation encompassed diverse protocol configurations, scrutinizing their computation time, energy consumption, and memory utilization. The report disclosed that selecting an optimal configuration involves a delicate balance between computational resources, battery power, storage memory, and potential security risks. Notably, the Fully-Precomputed configuration surfaced as a viable option for scenarios with limited resources (computation and/or battery life) but ample storage capacity. In contrast, the Parallel Computed configuration, unique to the ESP32 microcontroller, demonstrated superior security features with no storage reliance, albeit at the expense of increased energy consumption. Moreover, the evaluation unveiled the correlation between the number of attributes in a policy and both execution time and energy consumption, suggesting a trend where the Parallel Computed version's energy consumption is less than that of the Partially Precomputed version as the number of attributes in the policy increases.

The evaluation provides answers to all the sub-research questions presented in the section 4.3. The sub-research questions are mentioned again here:

1. **What is the effect of our system on a normal drone operation in terms of computation and energy consumption?**

   The evaluation talks about these metrics and also provides an example of energy consumption in various configurations as the battery charge percentage.

2. **What is the impact of the number of attributes (expressibility of the policies) used in the policy on energy consumption and computation costs?**

   As the number of attributes in the policy increases, the number of cryptographic computations required also increases proportionally. This results in a linear trend between the number of attributes in the policy and the execution time/energy consumption.

3. **What is the optimal number of attributes that can be used without significantly affecting a drone operation time(due to additional Energy consumption and Computational costs)?**

   From our evaluation, even having the maximum number of attributes (n=23) in the policy consumes less than one Joule of energy per message in the Parallel Computed mode. In a

normal drone operation, this is less than one percent of the energy consumed per second (power). Hence, it is safe to say that the system can be run optimally even at the maximum number of attributes in the policy.

4. **How much effect does the use of precomputed ABE(AES) keys have on the operation?**

   The Fully Precomputed version requires the least amount of execution time and electrical energy. Hence, the precomputation of the ABE(AES) keys has a very high impact on the system.

5. **What is the impact of changing the policies during the drone operation when:**

   - **Adopting a new policy from the existing list**
   - **creating a new policy**

   From the evaluations, it is evident that the only effect of changing the policy is only determined by the number of attributes in the new policy. Thus the execution time and energy consumption vary proportionally to the difference in the number of attributes in the new policy.

6. **What is the trade-off of authentication and energy in our system?**

   The percentage of energy consumption of the Schnorr signature generation is mentioned in Sec.7.3.1. In the Fully Precomputed, as this is the only time-consuming cryptographic process that is calculated onboard, it goes up to 90 percent of the total energy. In the Partially precomputed and the Parallel Computed versions, the amount of energy consumed by Schnorr signature generation is less than 10 percent of the total energy consumed.

Our system satisfies all the proposed requirements. The total execution time is less than 3 seconds and fits into one single packet of data. The SNELL protocol provides selective access and authenticity. The setup phase of the protocol enables opportunistic access and offline decryption. Finally, the implementation also allows for dynamic change of policies during operation. A final comparison of our work to the others used in Ch.4 is presented in Tab. 8.1.

SNo.	Ref No.	[R1]	[R2]	[R3]	[R4]	[R5]	[R6]	[R7]
1.	[61]	◑ (public keys)	○	-	●	◑ (revocation)	●	●
2.	[35]	◑ (ID-based)	○	●	●	○	○	●
3.	[57]	◑ (public keys)	-	○	●	○	●	●
4.	[77]	○	●	○	●	○	●	●
5.	[8]	○	-	-	○	○	○	○
6.	[45]	●	●	○	○	○	○	○
7.	[58]	○	●	●	●	◑ (revocation)	●	●
8.	[79]	○	●	-	○	◑ (updates)	●	●
9.	[40]	●	-	-	○	◑ (revocation)	●	○
10.	[50]	○	-	●	●	○	○	○
11.	Our system	●	●	●	●	●	●	●

●: complies; ○: does not comply; ◑: partially complies; - : no information

Table 8.1: Comparison of similar works based on our requirements

8.2 Future Work

While this report has shed light on various aspects of the SNELL protocol and its implications for privacy-preserving drone operations, there are several intriguing avenues for future research that can build upon the findings presented.

One potential direction is the investigation of adaptive policy management strategies that dynamically adjust the number of attributes in ABE policies based on contextual factors such as location, time, or operational requirements. This could potentially optimize the balance between energy consumption and computational costs, leading to more efficient and responsive remote identification systems. Furthermore, exploring ways to integrate renewable energy sources or energy-efficient hardware components could contribute to extending the operational lifetime of drones, particularly in scenarios where battery power is a limiting factor.

Additionally, as the landscape of drone technology evolves, addressing the integration of more advanced cryptographic techniques could enhance the overall security posture of remote identification systems. Exploring post-quantum cryptography methods and their compatibility with resource-constrained devices like the ESP32 microcontroller would be particularly relevant, given the increasing focus on quantum-resistant encryption methods.

Moreover, expanding the evaluation framework to encompass real-world deployment scenarios and operational considerations could provide insights into the practical applicability of the SNELL protocol. This includes assessing the protocol's performance in complex operational environments and potential challenges arising from interactions with other communication protocols or networks.

Bibliography

[1] 35 Important Pros & Cons Of Drones. https://environmental-conscience.com/drones-pros-cons/. Accessed: April 2023. 1

[2] Future of Drones: Applications & Uses of Drone Technology in 2021. https://www.businessinsider.com/drone-technology-uses-applications. Accessed: April 2023. 1

[3] Schnorr Digital Signature. https://www.geeksforgeeks.org/schnorr-digital-signature/. 12

[4] What are Schnorr Signatures? https://www.bitstamp.net/learn/blockchain/what-are-schnorr-signatures/. 12

[5] IEEE Standard Specifications for Public-Key Cryptography - Amendment 1: Additional Techniques. *IEEE Std 1363a-2004 (Amendment to IEEE Std 1363-2000)*, pages 1–167, 2004. 34

[6] Commission Implementing Regulation (EU) 2019/947 of 24 May 2019 on the rules and procedures for the operation of unmanned aircraft (Text with EEA relevance.), May 2019. 3

[7] Commission Delegated Regulation (EU) 2019/945 of 12 March 2019 on unmanned aircraft systems and on third-country operators of unmanned aircraft systems, Mar 2021. 3

[8] Claudio Ardagna, Marco Cremonini, Sabrina Vimercati, and Pierangela Samarati. An obfuscation-based approach for protecting location privacy. *Dependable and Secure Computing, IEEE Transactions on*, 8:13 – 27, 03 2011. 21, 23, 51

[9] ASD-Stan. Direct Remote ID : Introduction To The European UAS Digital Remote ID Technical Standard. *Direct remote ID - ASD-stan*, 2021. 5

[10] ASTM International. Standard Specification for Remote ID and Rracking. Technical Report ASTM Standard F3411-22, ASTM International, West Conshohocken, PA, 2022. 4, 5, 6, 14

[11] Razvan Barbulescu and Sylvain Duquesne. Updating Key Size Estimations for Pairings. *Journal of Cryptology*, 32(4):1298–1336, jan 2018. 9

[12] Elaine Barker. Recommendation for key management:Part 1 – General. Technical report, may 2020. 8

[13] Elaine Barker and Allen Roginsky. Transitioning the use of cryptographic algorithms and key lengths. Technical report, mar 2019. 8

[14] Elaine B. Barker and Quynh H. Dang. Recommendation for Key Management Part 3: Application-Specific Key Management Guidance. Technical report, jan 2015. 8

[15] Paulo S. L. M. Barreto, Benoît Libert, Noel McCullagh, and Jean-Jacques Quisquater. Efficient and provably-secure identity-based signatures and signcryption from bilinear maps. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3788 LNCS:515 – 532, 2005. 21

[16] Mihir Bellare, Phillip Rogaway, and Terence Spies. The FFX Mode of Operation for Format-Preserving Encryption. Technical Report vol. 20, NIST, Gaithersburg, MD, USA, 2010. 21

[17] Max Bergström. Direct Remote Id based UAS Collision Avoidance System. Master's thesis, KTH, Lättkonstruktioner, marina system, flyg- och rymdteknik, rörelsemekanik, 2022. 1

[18] Dan Boneh and Matt Franklin. Identity-Based Encryption from the Weil Pairing. In Joe Kilian, editor, *Advances in Cryptology — CRYPTO 2001*, pages 213–229, Berlin, Heidelberg, 2001. Springer Berlin Heidelberg. 9

[19] S. Card (Ed.), A. Wiethuechter, R. Moskowitz, and A. Gurtov. Drone Remote Identification Protocol (DRIP) Requirements and Terminology. RFC 9153 (Informational), February 2022. 5

[20] Nanxi Chen, Mario Gerla, Dijiang Huang, and Xiaoyan Hong. Secure, selective group broadcast in vehicular networks using dynamic attribute based encryption. 2010. 20

[21] Infrastructure Transport Civil Aviation Bureau, Ministry of Land and Japan Tourism. Requirements for Remote ID Devices and Applications. Technical report, 2022. 3

[22] European Commission. U-space Rolling Plan for ICT standardisation. *Rolling Plan for ICT standardisation*, May 2023. 4

[23] Information Technology Laboratory Computer Security Division. Current modes - block cipher techniques: CSRC. 7

[24] Wikipedia contributors. Schnorr signature — Wikipedia, The Free Encyclopedia. 2023. [Online; accessed 12-April-2023]. 12

[25] Bruce Crumley. Town official shoots drone. *DroneDJ*, Aug 2021. 1

[26] Cécile Delerablée. Identity-based broadcast encryption with constant size ciphertexts and private keys. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4833 LNCS:193 – 215, 2007. 20

[27] Yevgeniy Dodis and Nelly Fazio. Public key trace and revoke scheme secure against adaptive chosen ciphertext attack. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2567:100 – 115, 2003. 21

[28] Espressif Semiconductors. *ESP32WROOM32 Datasheet*, 2 2023. Rev. 3.4. 35

[29] Espressif Systems. *ESP32 series Datasheet*, 2023. v4.3. 30

[30] FAA. Final Rule on Remote Identification of Unmanned Aircraft, Dec 2020. 3, 4, 32

[31] Dave Faherty. Burke County Man's drone shot down by gunfire, deputies investigating, Jul 2022. 1

[32] Steven D. Galbraith and Pierrick Gaudry. Recent progress on the elliptic curve discrete logarithm problem. *Designs, Codes and Cryptography*, 78(1):51–72, nov 2015. 8

[33] S.G.T Ganti. Implementation for SNELL protocol on ESP32 microcontroller. `https://github.com/ganeshteja/SNELL.git`, 2023. GitHub repository. 34

[34] Darrel Hankerson and Alfred Menezes. *Elliptic Curve Discrete Logarithm Problem*, pages 397–400. Springer US, Boston, MA, 2011. 8

[35] S. He, Q. Wu, J. Liu, W. Hu, B. Qin, and Y.-N. Li. Secure communications in unmanned aerial vehicle network. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 10701 LNCS:601–620, 2017. 21, 23, 51

[36] Zihe He and Tian Tan. Survey on Worldwide Implementation of Remote Identification and Discussion on Drone Identification in China. In *2021 IEEE 3rd International Conference on Civil Aviation Safety and Information Technology (ICCASIT)*, pages 252–258, 2021. 3

[37] G. Huang, R. Wang, and Y. Xu. Broadcast authentication protocol scheme based on layer-cluster in WSN. *Nanjing Hangkong Hangtian Daxue Xuebao/Journal of Nanjing University of Aeronautics and Astronautics*, 42(1):72–76, 2010. 20

[38] Bruce Ikenaga. Cyclic Groups. *Millersville University of Pennsylvania*, Jan 2019. 8

[39] Jianchang Lai, Yi Mu, Fuchun Guo, Willy Susilo, and Rongmao Chen. Anonymous identity-based broadcast encryption with revocation for file sharing. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9723:223 – 239, 2016. 22

[40] Jianchang Lai, Yi Mu, Fuchun Guo, Willy Susilo, and Rongmao Chen. Fully privacy-preserving and revocable ID-based broadcast encryption for data access control in smart city. *Personal and Ubiquitous Computing*, 21(5):855 – 868, 2017. 22, 23, 51

[41] Xuejun Li, Yawen Yuan, and Chunhua Jin. An attribute-based broadcast encryption scheme suitable for the broadcasting network;. *Jisuanji Yanjiu yu Fazhan/Computer Research and Development*, 55(7):1409 – 1420, 2018. 20

[42] Weiran Liu, Jianwei Liu, Qianhong Wu, Bo Qin, and Yan Li. Practical chosen-ciphertext secure Hierarchical Identity-Based Broadcast Encryption. *International Journal of Information Security*, 15(1):35 – 50, 2016. 21

[43] Y. Liu, J. Li, and M. Guizani. PKC based broadcast authentication using signature amortization for WSNs. *IEEE Transactions on Wireless Communications*, 11(6):2106–2115, 2012. 20

[44] Xixiang Lv and Bo Yang. Traitor tracing using identity based public-key cryptography. *Chinese Journal of Electronics*, 15(4):687 – 691, 2006. 20

[45] M. Mandal. 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. 22, 23, 51

[46] Michele La Manna, Luigi Treccozzi, Pericle Perazzo, Sergio Saponara, and Gianluca Dini. Performance evaluation of attribute-based encryption in automotive embedded platform for secure software over-the-air update. *Sensors (Switzerland)*, 21(2):1 – 14, 2021. 20

[47] Bacem Mbarek, Aref Meddeb, Wafa Ben Jaballah, and Mohamed Mosbah. An Efficient Broadcast Authentication Scheme in Wireless Sensor Networks. *Procedia Computer Science*, 109:553–559, 2017. 8th International Conference on Ambient Systems, Networks and Technologies, ANT-2017 and the 7th International Conference on Sustainable Energy Information Technology, SEIT 2017, 16-19 May 2017, Madeira, Portugal. 21

[48] Alfred Menezes. An Introduction to Pairing-Based Cryptography. In *An Introduction to Pairing-Based Cryptography*, 2005. 8

[49] Victor S. Miller. The Weil Pairing, and Its Efficient Calculation. *Journal of Cryptology*, 17(4):235–261, aug 2004. 8

[50] Robert Moskowitz, Stuart W. Card, and Adam Wiethuechter. UAS Operator Privacy for RemoteID Messages. Internet-Draft draft-moskowitz-drip-operator-privacy-11, Internet Engineering Task Force, December 2022. Work in Progress. 22, 23, 51

[51] Yi Mu, Willy Susilo, Yan-Xia Lin, and Chun Ruan. Identity-based authenticated broadcast encryption and distributed authenticated encryption. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 3321:169 – 181, 2004. 20

[52] Gregory Neven, Nigel Smart, and Bogdan Warinschi. Hash function requirements for Schnorr signatures. *Journal of Mathematical Cryptology*, 3, 06 2009. 12

[53] National Institute of Standards and Technology. Advanced encryption standard (AES). Technical report, 2023. 6

[54] Pericle Perazzo, Francesca Righetti, Michele La Manna, and Carlo Vallati. Performance evaluation of Attribute-Based Encryption on constrained IoT devices. *Computer Communications*, 170:151 – 163, 2021. 20

[55] A. Perrig, D. Song, R. Canetti, J. D. Tygar, and B. Briscoe. Timed Efficient Stream Loss-Tolerant Authentication (TESLA): Multicast Source Authentication Transform Introduction. RFC 4082 (Informational), June 2005. 21

[56] Tran Viet Xuan Phuong, Guomin Yang, Willy Susilo, and Xiaofeng Chen. Attribute based broadcast encryption with short ciphertext and decryption key. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9327:252 – 269, 2015. 20

[57] A. Prado, S. Ruj, M. Stojmenovic, and A. Nayak. Applying transmission-coverage algorithms for secure geocasting in VANETs. *International Journal of Computational Science and Engineering*, 16(1):17–26, 2018. 21, 23, 51

[58] M.A. Rezazadeh Baee, L. Simpson, X. Boyen, E. Foo, and J. Pieprzyk. ALI: Anonymous Lightweight Inter-Vehicle Broadcast Authentication with Encryption. *IEEE Transactions on Dependable and Secure Computing*, 2022. 22, 23, 51

[59] Doreen Riepel and Hoeteck Wee. FABEO: Fast Attribute-Based Encryption with Optimal Security. Cryptology ePrint Archive, Paper 2022/1415, 2022. `https://eprint.iacr.org/2022/1415`. 10

[60] Damien Robert. The geometric interpretation of the Tate pairing and its applications. Cryptology ePrint Archive, Paper 2023/177, 2023. `https://eprint.iacr.org/2023/177`. 8

[61] X. Ruan, B. Yu, J. Xu, and L. Yang. A secure privacy-preserving hierarchical location service for mobile ad hoc networks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 4864 LNCS:760–771, 2007. 21, 23, 51

[62] Amit Sahai and Brent Waters. Fuzzy identity-based encryption. volume 3494, page 457 – 473, 2005. 9

[63] Yumi Sakemi, Tetsutaro Kobayashi, Tsunekazu Saito, and Riad S. Wahby. Pairing-Friendly Curves. Internet-Draft draft-irtf-cfrg-pairing-friendly-curves-11, Internet Engineering Task Force, November 2022. Work in Progress. 9

[64] Michael Scott. The MIRACL Core Library. *Github - The MIRACL Core Cryptographic Library*, Jul 2021. 33, 35

[65] Yannick Seurin. On the Exact Security of Schnorr-Type Signatures in the Random Oracle Model. Cryptology ePrint Archive, Paper 2012/029, 2012. `https://eprint.iacr.org/2012/029`. 13

[66] Adi Shamir. Identity-Based Cryptosystems and Signature Schemes. In George Robert Blakley and David Chaum, editors, *Advances in Cryptology*, pages 47–53, Berlin, Heidelberg, 1985. Springer Berlin Heidelberg. 9

[67] Tjerand Silde. Comparative Study of ECC Libraries for Embedded Devices. Technical report, Norwegian University of Science and Technology, 2019. 33

[68] Ishveena Singh. Florida man shoots Down Sheriff's office drone investigating possible burglary, Jul 2021. 1

[69] Jin Sun and Yupu Hu. Fully secure attribute-based broadcast encryption. *Xi'an Dianzi Keji Daxue Xuebao/Journal of Xidian University*, 39(4):23–28+154, 2012. 20

[70] Espressif Systems. ESP32 SDK Arduino, 2023. 34

[71] Mohammad Bany Taha, Chamseddine Talhi, and Hakima Ould-Slimane. Performance evaluation of CP-ABE schemes under constrained devices. volume 155, page 425 – 432, 2019. 20

[72] Pietro Tedeschi, Fatima Ali Al Nuaimi, Ali Ismail Awad, and Enrico Natalizio. Privacy-Aware Remote Identification for Unmanned Aerial Vehicles: Current Solutions, Potential Threats, and Future Directions. *IEEE Transactions on Industrial Informatics*, pages 1–12, 2023. 20

[73] Pietro Tedeschi, Savio Sciancalepore, and Roberto Di Pietro. ARID: Anonymous Remote IDentification of Unmanned Aerial Vehicles. page 207 – 218, 2021. Cited by: 10; All Open Access, Bronze Open Access. 21

[74] E. Wisse, P. Tedeschi, S. Sciancalepore, and R.D. Pietro. A2RID -Anonymous Direct Authentication and Remote Identification of Commercial Drones. *IEEE Internet of Things Journal*, pages 1–1, 2023. 21

[75] Eva M.C Wisse. A2rid: An Anonymous RemoteID-compliant group signature scheme for commercial drones. Master's thesis, Eindhoven University of Technology, 2022. 34, 36

[76] H. Yang, H. Li, and X.S. Shen. ADS-B Broadcast Authentication. *Wireless Networks (United Kingdom)*, pages 61–84, 2023. 20

[77] H. Yang, Q. Zhou, M. Yao, R. Lu, H. Li, and X. Zhang. A practical and compatible cryptographic solution to ADS-B security. *IEEE Internet of Things Journal*, 6(2):3322–3334, 2019. 21, 23, 51

[78] Danfeng Yao, Yevgeniy Dodis, Nelly Fazio, and Anna Lysyanskaya. Id-based encryption for complex hierarchies with applications to forward security and broadcast encryption. page 354 – 363, 2004. 20

[79] Xin Ye, Jin Zhou, Yuedi Li, Mingsheng Cao, Dajiang Chen, and Zhiguang Qin. A location privacy protection scheme for convoy driving in autonomous driving era. *Peer-to-Peer Networking and Applications*, 14, 05 2021. 22, 23, 51

[80] W. Yu, L. Yang, and S. Wang. New Lattice-Based Broadcast Authentication Protocol for Wireless Sensor Networks. *Security and Communication Networks*, 2022, 2022. 20

[81] Éric Vyncke. Drone Remote ID Protocol charter, 2021. 5

Appendix A

Notation

Notation	Description
ID	Identity of the device (UAV \observer)
(msk_A, mpk_A)	ABE Master key pair at USS
(Ssk, Spk)	Schnorr key pair at UAV
p	Group Order
$(G1, G2, GT)$	Type 3 (Asymmetric) bilinear groups of prime order
e	Pairing operation
$g1, g2$	Group generators for G1 and G2
H	Hashing function to G1
u	Generic attribute
α	Random secret nonce in Zp used as the Master secret key
$r \in GT$	Random nonce
P	Attribute Policy
c	Cipher text from CP_FABEO.Encrypt
t	Current timestamp
K_t	Ephemeral symmetric key
$[lat, lon, alt]$	Latitude, Longitude and Altitude
L_{GCS}	Location data of the GCS including the latitude, longitude and altitude
L_{UAV}	Location data of the UAV including the latitude, longitude and altitude
C	Encrypted $L_{GCS} obtained from AES.Encrypt$
m_t	RID message packet
ec	Emergency Code
$\sigma_{m,t}$	Schnorr signature
r	recovered random nonce
attr_set	Attribute set of the observer

Table A.1: Notation used and their description

Appendix B

Curve Parameters

B.1 BN254

Paarameter	Value
P	0x2523648240000001BA344D8000000086121000000000013A700000000000013
a	0x00
b	0x02
G	(0x2523648240000001BA344D8000000086121000000000013A700000000000012, 0x0000000000000000000000000000000000000000000000000000000000000001)
n	0x2523648240000001BA344D8000000007FF9F800000000010A10000000000000D
h	0x01

Table B.1: BN254 curve parameters

B.2 BLS12-381

Paarameter	Value
P	0x1a0111ea397fe69a4b1ba7b6434bacd764774b84f38512bf6730d2a0f6b0f6241eabfffeb153ffffb9 feffffffffaaab
a	0x00
b	0x04
G	(0x17F1D3A73197D7942695638C4FA9AC0FC3688C4F9774B905A14E3A3F171BAC 586C55E83FF97A1AEFFB3AF00ADB22C6BB, 0x08B3F481E3AAA0F1A09E30ED741D8AE4FCF5E095D5D00AF600DB18CB2C04B3ED D03CC744A2888AE40CAA232946C5E7E1)
n	0x73EDA753299D7D483339D80809A1D80553BDA402FFFE5BFEFFFFFFFF00000001
h	0x396C8C005555E1568C00AAAB0000AAAB

Table B.2: BLS12381 curve parameters

Appendix C

Packet sizes

Attributes	BN254	BN462	BLS12-381	BLS12-461	BLS48-581
n	n*33+514+135	n*59+930+135	n*49+770+135	n*59+930+135	n*74+1170+135
1	682	1124	954	1124	1379
2	715	1183	1003	1183	1453
3	748	1242	1052	1242	1527
4	781	1301	1101	1301	1601
5	814	1360	1150	1360	1675
6	847	1419	1199	1419	1749
7	880	1478	1248	1478	1823
8	913	1537	1297	1537	1897
9	946	1596	1346	1596	1971
10	979	1655	1395	1655	2045
11	1012	1714	1444	1714	2119
12	1045	1773	1493	1773	2193
13	1078	1832	1542	1832	2267
14	1111	1891	1591	1891	2341
15	1144	1950	1640	1950	2415
16	1177	2009	1689	2009	2489
17	1210	2068	1738	2068	2563
18	1243	2127	1787	2127	2637
19	1276	2186	1836	2186	2711
20	1309	2245	1885	2245	2785
21	1342	2304	1934	2304	2859
22	1375	2363	1983	2363	2933
23	1408	2422	2032	2422	3007
24	1441	2481	2081	2481	3081
25	1474	2540	2130	2540	3155
26	1507	2599	2179	2599	3229
27	1540	2658	2228	2658	3303
28	1573	2717	2277	2717	3377
29	1606	2776	2326	2776	3451
30	1639	2835	2375	2835	3525

Table C.1: Packet size for various Pairing Friendly Curves based on number of Attributes in Policy

The Tab. C.1 provides the sizes of the packet for various number of attributes with the CP-ABE encryption implemented using different curves. The size of the policy is not included in the calculation.

The cells in Red color have sizes above the MTU of 1500 Bytes and hence are not possible to be used in the implementation. The cells in Yellow color have sizes below 1500 Bytes but are too big to accommodate the policy in the message. So, these are also not used. The cells in Green color have enough space to include a policy of reasonable length for the given number of attributes.

Appendix D

Schnorr Signature Sizes

Curve	message size
SECP256k1	64
SECP256r1	64
SECP384r1	80
SECP521r1	84

Table D.1: Schnorr Signature sizes for various Elliptic Curves

Appendix E

Parallel Computed Version Specifics

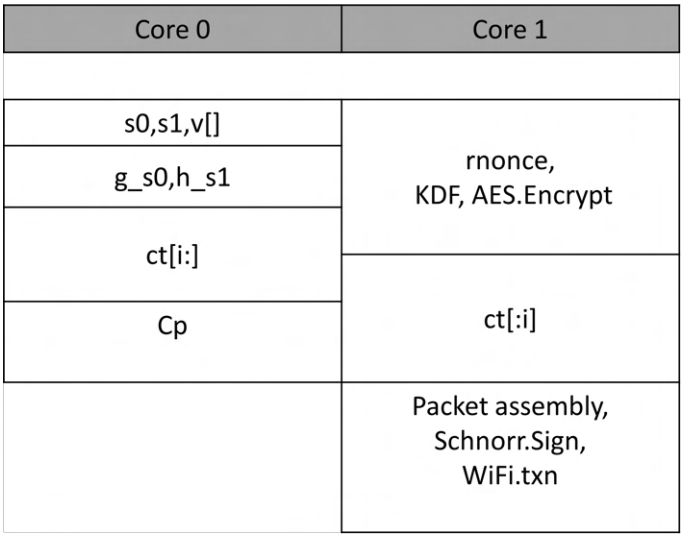


Figure E.1: Division of computation tasks between the two cores on ESP32

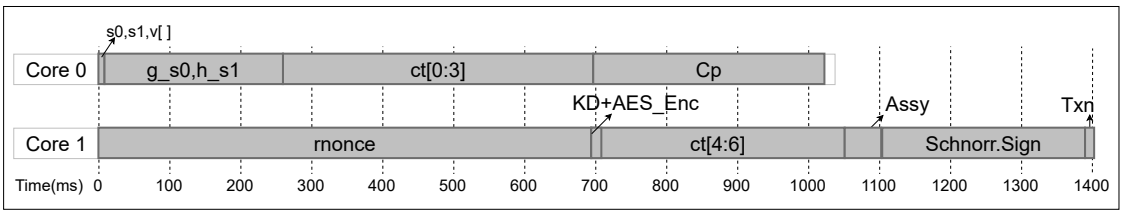


Figure E.2: Timing Diagram for Parallel Computed mode when using BLS12-381

Appendix F

SNELL Implementation

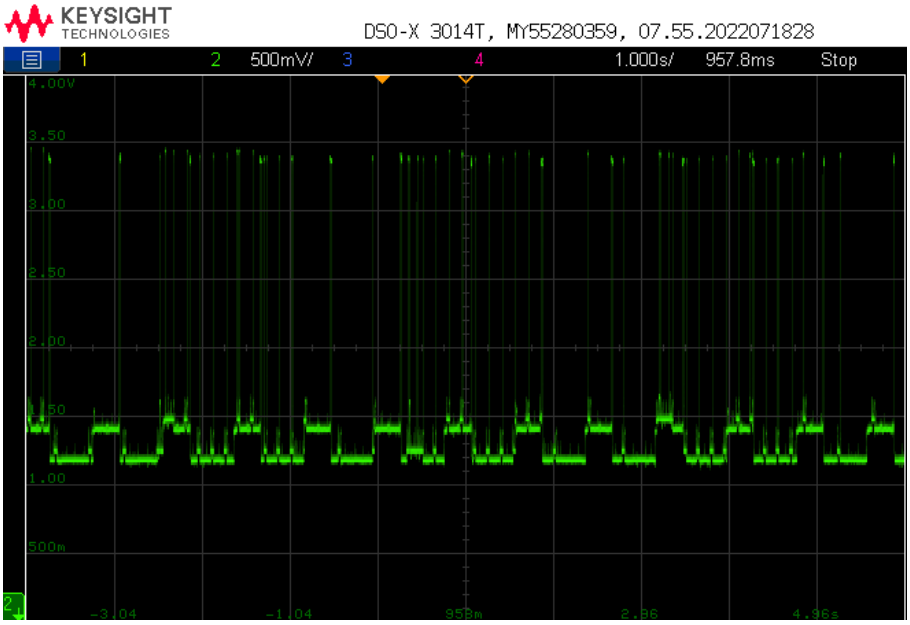
Key computation steps in SNELL RID Implementation with their average computation times:

- **rnonce generation** - 450ms
- **CP-ABE Encryption of rnonce**
    1. **s0,s1,v[] generation** - 4ms - 55ms
    2. **g_s0,h_s1 computation** - 190ms
    3. **Cp computation** - 240ms
    4. **ct[] computation** - 70ms - 2070ms
- **Symmteric Key derivation** - 650µs
- **AES encryption** - 800µs
- **Packet assembly** - 4ms - 100ms
- **Schnorr Signature Generation** - 282-292ms
- **Packet Transmission** - 10ms

Total time - 1251ms - 3407ms

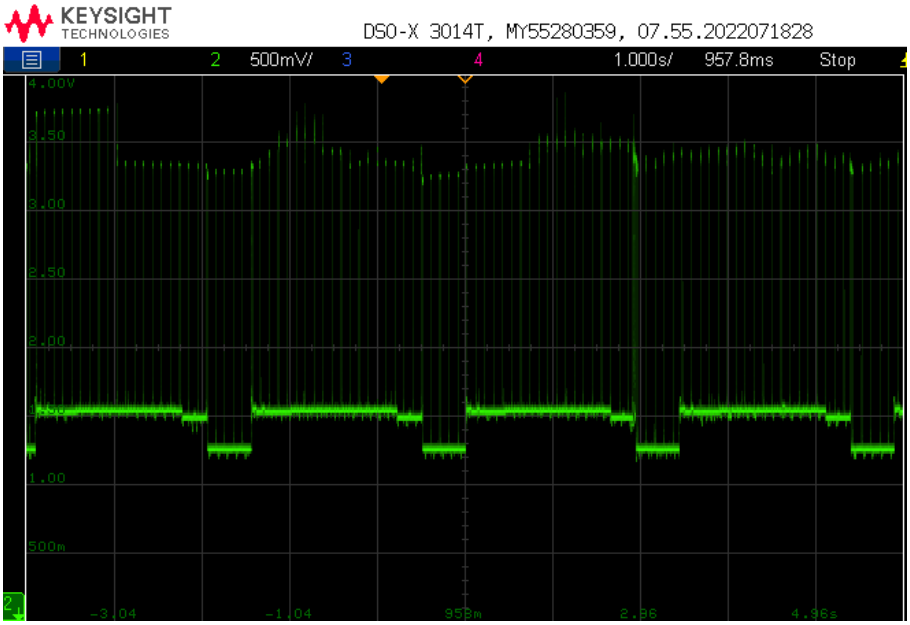
Appendix G

Oscilloscope Outputs

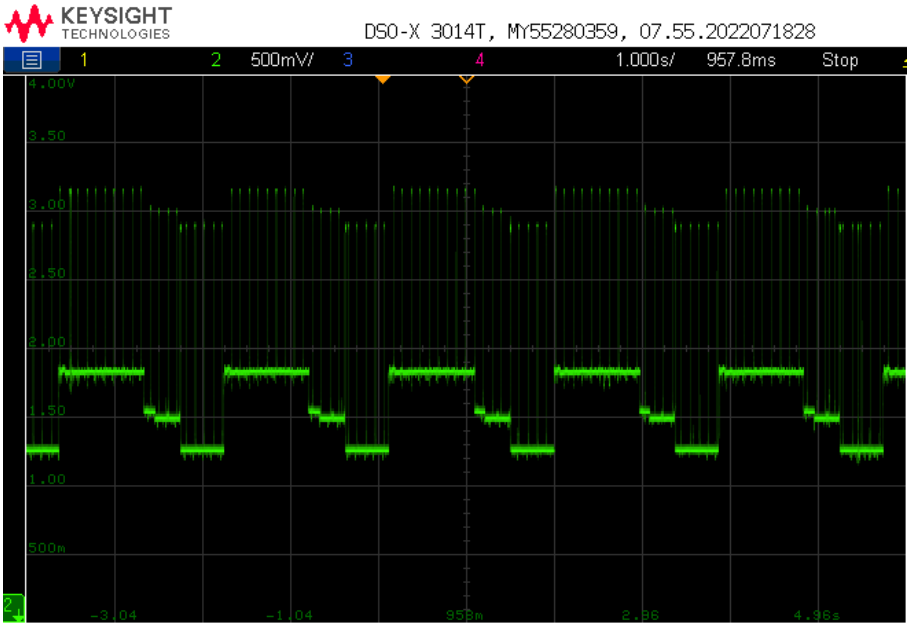


(a) Fully Precomputed mode

Figure G.1: Oscilloscope Outputs for current consumption in the three modes



(b) Partially Precomputed mode



(c) Parallel Computed mode

Figure G.1: Oscilloscope Outputs for current consumption in the three modes

Appendix H

Execution times for BN254 and BLS12-381 curves

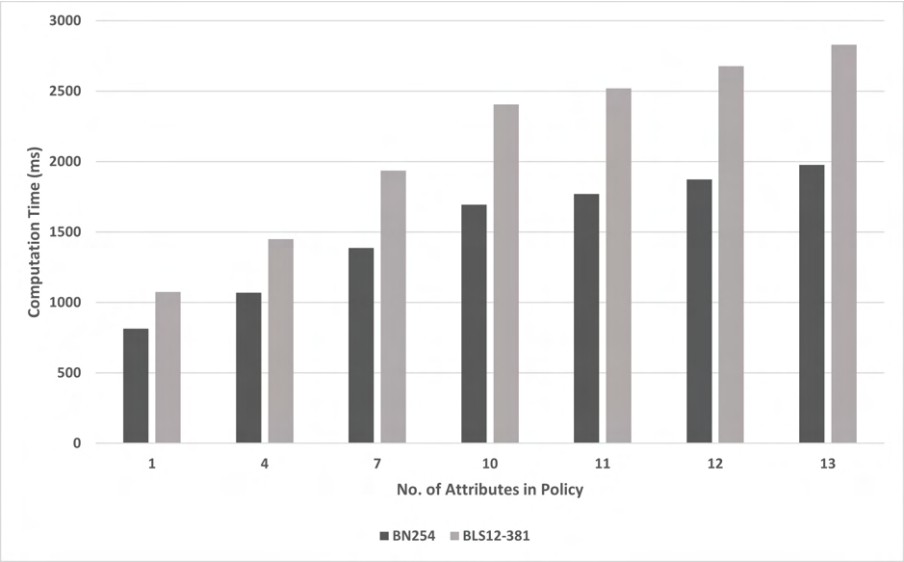


Figure H.1: Partially Precompted version

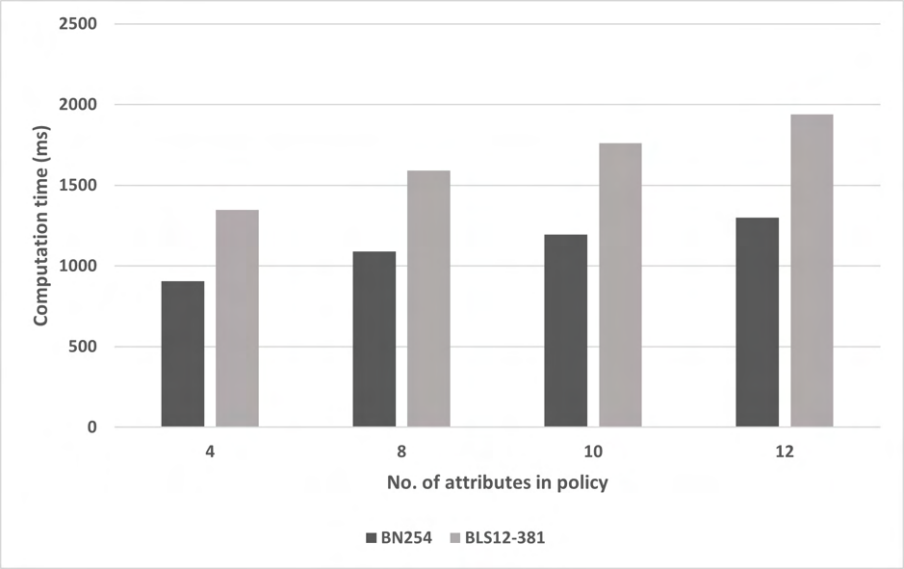


Figure H.2: Parallel Computed version