University of Sheffield

Privacy-aware Secure Authentication and Handover Protocols for 5G-enabled Mobile Communication

Rabiah Othman Alnashwan

Supervisors: Dr. Prosanta Gope & Dr. Benjamin Dowling

This dissertation is submitted for the degree of Doctor of Philosophy

in the

Department of Computer Science

February, 2025

Declaration

All sentences or passages quoted in this document from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure.

Name: _____

Signature: _____

Date: _____

Abstract

The evolution of mobile communication has facilitated technological advancements that enable seamless global connectivity. With the advent of 5G technology, wireless communication has taken a significant leap forward, promising unparalleled speed, capacity, and connectivity. As we enter this era of advanced communication, we also need to consider its implications for security and privacy. The integration of 5G technology brings new opportunities and challenges, making it essential to thoroughly examine the security and privacy frameworks that support this advanced network. Compared to the previous mobile communication generations, 5G offers a more robust security infrastructure by strengthening two key protocols: Authentication and Key Agreement (AKA) and Handover (HO). Although 5G-AKA significantly improves security measures, it is worth noting that the current protocols lack support for several essential security and privacy properties, such as forward secrecy, forward privacy, and unlinkability. Thus, a critical need remains to address these gaps to ensure comprehensive protection in 5G networks.

In response to the issues in respect of security and privacy, this thesis proposes three novel AKA and HO schemes. The three proposed schemes have different security and privacy goals that support improved security and privacy features compared to the conventional 5G-AKA and HO protocols currently utilized and other existing solutions. In particular, we examine challenges associated with integrating ultra-dense small cell networks (SCNs) into the 5G infrastructure. This exploration led us to investigate the concept of region-based handovers and to propose, to the best of our knowledge, the first scheme that provides privacy-preserving, secure inter-region-based AKA and HO scheme. This scheme provides secure authentication for roaming users with an efficient and seamless handover process. To enhance security and privacy measures further, we undertake an additional investigation into fortifying resilience against key compromise impersonation attacks. This involves proposing a novel, secure, privacy-preserving Universal Handover scheme (UniHand) tailored for SCNs within the 5G mobile communication framework. Finally, in pursuit of seamless compatibility with 5G networks, we introduce an improved iteration of the 5G-AKA and HO protocols. Referred to as Pretty Good User Privacy (PGUP), this novel symmetry-based scheme aims to mitigate security and privacy vulnerabilities inherent in the existing 5G-AKA and HO protocols while maintaining high compatibility with the 5G infrastructure.

Acknowledgements

Foremost, I thank God Almighty for guiding me through this journey and blessing me with the strength, patience and perseverance to complete this thesis.

I extend my gratitude to my supervisors, Dr. Prosanta Gope and Dr. Benjamin Dowling, whose unwavering support, profound expertise, and constant encouragement have been the foundation of this journey. I also sincerely thank my viva examiners, Prof. John A. Clark and Dr. Attila Altay Yavuz, for their valuable insights and constructive feedback, which have significantly contributed to the refinement of this thesis. I would also like to thank my friends within the Security of Advanced Systems Group at the University of Sheffield for their support, friendship, and for making this journey both enjoyable and enriching.

My sincere appreciation also extends to Imam Mohammad Ibn Saud Islamic University and the Saudi Arabian Cultural Bureau in London for supporting my PhD studies.

To my beloved family and friends, words cannot fully express my gratitude. Your love and constant encouragement have been a source of unwavering strength throughout this journey. To my parents, your belief in me has been my greatest motivation, and I am forever grateful for your endless support.

Finally, to my husband, my heart is filled with gratitude for you. You have been my biggest source of comfort throughout this demanding journey. Your endless patience and boundless understanding have been the pillars that supported me during the toughest times.

As I close this chapter of my life, I carry with me not just a thesis, but a heart full of gratitude for everyone who has been part of this transformative journey.

Contents

List of Figures

List of Tables

Notation Used

The next list describes several acronyms that will be later used within the body of the document

gNB Next Generation Node B

HgNB Home gNodeB

HN Home Network

UE User Equipment

AE Authenticated Encryption

AKA Authentication and Key Agreement

AMF Access and Mobility Management Function

AV Transformed Authentication Vector

$EUFCMA$ Existential Unforgeability under the Chosen Message Attack

$HEAV$ 5G Home Environment Authentication Vector

HO Handover

KCI Key Compromise Impersonation

KDF Key Derivation Function

KEF Key-escrow Freeness

KEM Key Encapsulation Mechanism

MA Mutual Authentication

MAC Message Authentication Code

PFP Perfect Forward Privacy

PFS Perfect Forward Secrecy

PKW Puncturable Key Wrapping

$PPRF$ Puncturable Pseudorandom Functions

PPT Probabilistic Polynomial Time

$SanSig$ sanitizable signature

SCN Small Cell Network

SRM Secure Revocation Management

$SUCI$ Subscription Concealed Identifier

$SUPI$ Subscription Permanent Identifier

UHO Universal Handover

$USIM$ Universal Subscriber Identity Module

f_1, f_2, f_1^* Message Authentication Functions

f_3, f_4, f_5, f_5^* Key Derivation Functions.

Chapter 1

Introduction

The proliferation of connected devices is on a continuous upward trajectory, with forecasts predicting a staggering 50 billion connected devices worldwide by 2030 [43]. In response to this exponential growth, mobile networks have evolved significantly, progressing from the first generation (1G) to the early stages of the development of the sixth generation (6G). Each subsequent network generation has provided an evolved security landscape, with the fifth generation (5G) representing a notable improvement, with substantial infrastructural changes, compared to its predecessors. Unlike the incremental changes seen in the first four generations, 5G introduces stringent requirements and enhancements to the security architecture alongside novel network services. The International Telecommunication Union Radiocommunication Sector (ITU-R) has outlined key performance requirements for 5G, which include seamless mobility, high data rates, availability, connectivity, and spectrum efficiency [2].

The 5G mobile communication networks present a heterogeneous architecture, accommodating existing and emerging access technologies. These include advancements in Wireless Local Area Network (WLAN) technologies, Long-Term Evolution (LTE), and the New Radio (NR) interface [51]. Moreover, 5G integrates diverse network technologies, such as cloud computing, Software-Defined Networking (SDN) [68], Network Function Virtualization (NFV) [41], and ultra-dense Small Cell Networks (SCN), into a unified, programmable software-centric infrastructure.

Ultra-dense small cell technology is critical in fulfilling various 5G requirements, including high network density, capacity, and spectrum efficiency [9]. Small cells, encompassing a variety of types such as macro, micro, pico, and femtocells, refer to low-powered cellular radio access nodes. The distinctions among these cells lie in their supported coverage area, signal strength, service capacity, and transmission power. The primary goal of SCNs is to increase the density of wireless cells/nodes and reduce the coverage area to approximately 10 to 100 metres. This approach enhances network capacity, brings the network closer to connected users, and provides better throughput service with low-powered transmission, thereby boosting spectrum efficiency. However, to leverage the benefits of SCNs, users frequently have to switch between cells, a process known as Handover (HO). The HO procedure seamlessly transfers a mobile user's wireless responsibility from one cell operated by a base station (BS) to another cell/BS.

Although leveraging enabling technologies within the 5G environment offers undeniable

advantages, they are also vulnerable to various security and privacy challenges. The interplay among the technologies within a 5G framework can significantly impact overall security and privacy. This is also the case with the deployment of SCNs, which has introduced additional security and privacy considerations due to the increased frequency of HO procedures resulting from the dense arrangement of small cells [40]. As the number of small cells proliferates, so does the frequency of HO procedures, consequently increasing network latency. To address this, new mechanisms are introduced to mitigate latency and improve security in HO procedures.

The first and most crucial step when users want to connect to the network and receive services is to secure the communication link between customers and the network while prioritizing a strong degree of privacy for the users. Authenticating communication between network participants and their service providers is paramount for network security. Each party involved must verify the legitimacy of the other end throughout the communication process. An authenticated communication network is typically established through an Authentication and Key Agreement (AKA) protocol. However, designing a secure AKA for 5G presents unique challenges, particularly with integrating SCNs, which involve frequent handovers and stringent latency requirements. For 5G security, the 3rd Generation Partnership Project (3GPP) defines two primary AKA protocols: 5G-AKA as specified in Technical Specification (TS) [4], and the Extensible Authentication Protocol (EAP-AKA') detailed in TS [10]. Although both protocols share similarities, they differ in their message flow. Thus, in this work, we focus solely on 5G-AKA for our analysis and evaluation.

The 5G-AKA protocol represents a notable advancement compared to its predecessors but nonetheless fails to achieve some crucial security features. Consequently, a significant amount of research effort has been directed towards enhancing and analysing 5G-AKA, such as [33, 26, 15]. These investigations have identified numerous weaknesses in the current 5G-AKA protocol. For instance, an identity replay attack, as outlined by Fouque et al. [39], which targets several AKA protocols, also poses a threat to the 5G-AKA protocol [26]. Moreover, [24] identified a logical vulnerability that compromises the confidentiality of the sequence number due to the use of exclusive or (XOR) and the absence of randomness. Another attack, known as the confusion attack, was documented by Cremers and Wild [33], and leads to identity misbinding and enables impersonation attacks. One of the most significant vulnerabilities identified is that the current 5G-AKA protocol does not support perfect forward security (PFS) or perfect forward privacy (PFP). PFS ensures past session keys remain secure if long-term secrets are compromised, while PFP aims to maintain user anonymity even under such compromises. This deficiency exposes both data confidentiality and user privacy to potential compromise. Furthermore, the existence of linkability vulnerabilities within 5G-AKA raises additional privacy concerns [67], particularly those relating to active attackers seeking to undermine user anonymity. These vulnerabilities open the door to linkability attacks, which are serious privacy breaches whereby an adversary can correlate different actions or pieces of information to the same user or entity, even in systems designed anonymously. In the context of 5G networks, an attacker could exploit these vulnerabilities to track user movements or usage patterns.

Despite all the previous works conducted in this domain, none of the existing protocols in the literature fulfils the comprehensive security requirements outlined for 5G networks (see Sections 3.4.1 and 3.5 in Chapter 3). Thus, a notable gap exists in the research land-

scape, particularly about developing a protocol that effectively achieves the necessary security properties for 5G networks. These properties include mutual authentication, user anonymity, user unlinkability, PFS, and PFP. Furthermore, such a protocol should also consider SCN environmental issues and ensure seamless adaptability to 5G infrastructure.

The remainder of this chapter is organized as follows: Section 1.1 outlines the aims and objectives of this thesis, Section 1.2 presents the thesis contributions, Section 1.3 provides an overview of the thesis structure, and Section 1.4 lists the key publications resulting from this research. Finally, Section 1.5 outlines related collaborative works completed during this PhD research.

1.1 Aims and Objectives

In this thesis, our primary objective is to enhance the security and privacy aspects of 5G networks, explicitly focusing on AKA and HO schemes. Through developing and analysing novel protocols, we aim to address existing vulnerabilities and improve the security and privacy of these protocols and the network overall. In order to accomplish this objective, we have identified current open questions, which include:

RQ$_1$ Is it possible to design an authenticated key exchange and handover scheme that ensures security and preserves user privacy, all while supporting a ***region-based*** handover with seamless connectivity for roaming users?

RQ$_2$ Is it possible to design an authenticated key exchange and handover scheme to safeguard user privacy, ensure security, and incorporate a *Universal handover* mechanism with ***Key Compromise Impersonation resilience*** and ***Key-Escrow-Free*** properties?

RQ$_3$ Is it possible to design a *symmetric-key-based* authenticated key exchange and handover scheme that aligns with existing 5G infrastructure while providing ***perfect forward secrecy, forward privacy, and unlinkability***, and maintaining computational efficiency for resource-constrained user equipment?

1.2 Thesis Contributions

Following the research conducted to address our research questions, this thesis provides the following **key contributions**:

1. A *novel privacy-preserving inter-region* authentication and handover scheme for roaming users in 5G network. This scheme leverages asymmetric-key-based authenticated key exchange and handover protocols that preserve user privacy and network security while providing a seamless region-based handover mechanism and effective membership revocation management.

2. A *novel privacy-preserving Universal Handover scheme* (UniHand) that achieves seamless user mobility and required security and privacy properties, including KCI resilience and KEF properties for roaming users in 5G.

3. A *new variant of puncturable key wrapping*, denoted as PKW^+, aimed at achieving PFS, PFP, and unlinkability within a symmetric base setting. This cryptographic primitive is explicitly tailored to align with the specific requirements and nature of the 5G infrastructure.

4. A *novel symmetric-key-based authenticated key exchange and handover scheme* (PGUP) for 5G networks. This scheme aligns with existing 5G infrastructure while achieving PFS, PFP, and unlinkability. PGUP incorporates the new variant of puncturable key wrapping (PKW^+), enhancing security within the constraints of symmetric cryptography.

1.3 Thesis Overview

The overall thesis is arranged according to the framework illustrated in Figure 1.1, and aim to address the following:

- In **Chapter 2**, entitled "Preliminaries", we introduce the fundamental concepts, notations, and cryptographic primitives essential for understanding the subsequent chapters. The chapter establishes the theoretical foundation upon which the thesis builds.

- In **Chapter 3**, entitled "Background and Related Work", we present a comprehensive analysis of the existing literature related to the secure AKA and HO protocols in 5G networks.

- In **Chapter 4**, entitled "Privacy-Aware Secure Region-based Handover for 5G", we introduce our first major contribution: a novel privacy-aware secure region-based handover protocol. We detail the protocol design, security analysis, and performance evaluation, demonstrating its advantages over existing solutions.

- In **Chapter 5**, entitled "UniHand: Privacy-Preserving Universal Handover for 5G", we build on the previous chapter in presenting an enhanced protocol that extends privacy preservation to universal handover scenarios. This chapter elaborates on the design principles, security proofs, and efficiency considerations of the protocol.

- In **Chapter 6**, entitled "PGUP: Pretty Good User Privacy for 5G", we present our third significant contribution: We present a forward-secure user privacy authentication and handover protocol, providing a detailed description and rigorous security proofs and discussing its practical implications.

- In **Chapter 7**, entitled "Conclusion and Future Work," we summarize the key findings and contributions of the thesis. We discuss the implications of our work and its limitations and propose directions for future research in the field of secure and privacy-preserving handover protocols for 5G networks.

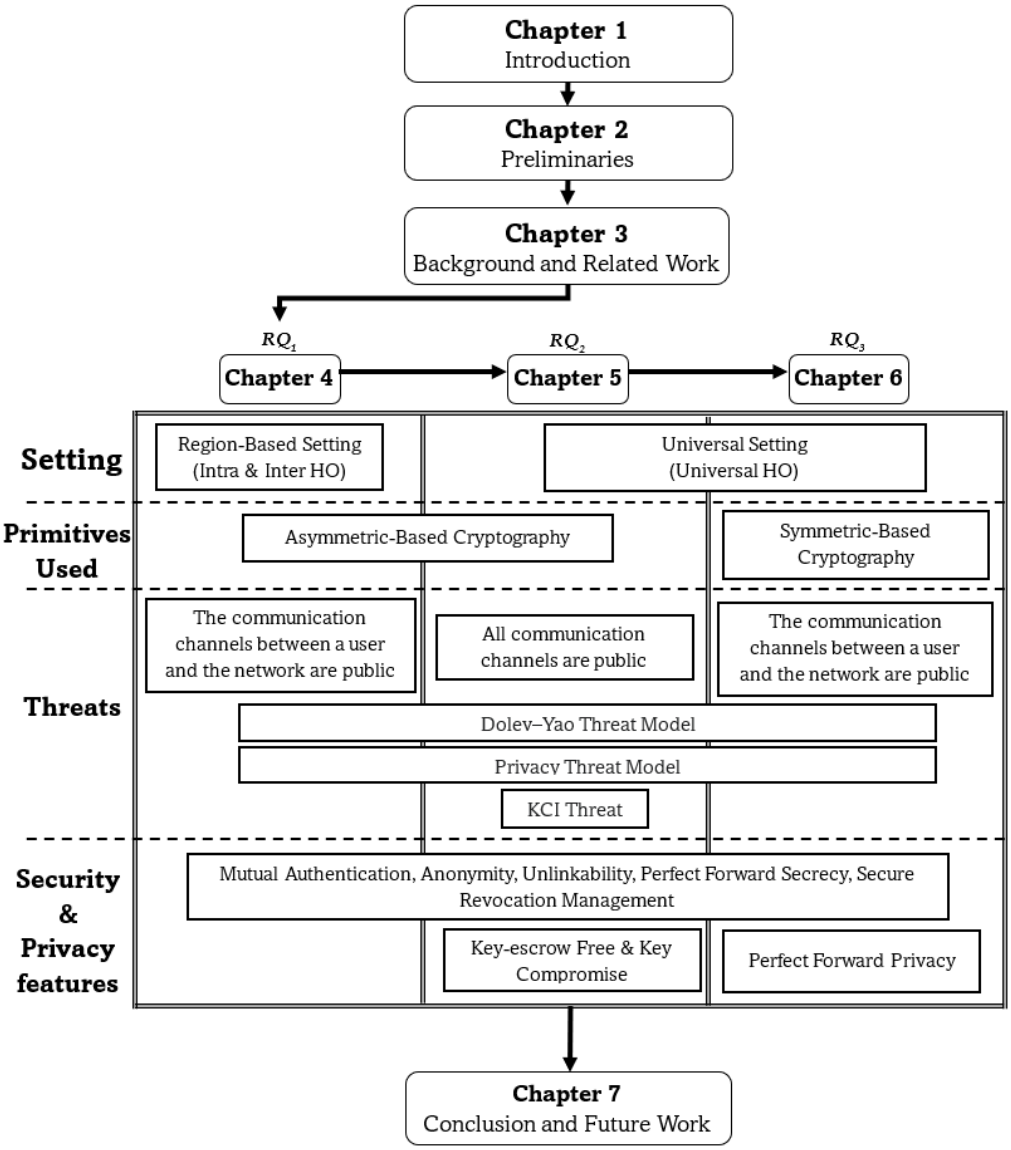

Figure 1.1: *Thesis structure overview*

1.4 Key Publications

The research findings discussed in the key chapters, which directly tackle the thesis research questions, have been published in the following peer-reviewed conferences and journals:

- R. Alnashwan, P. Gope and B. Dowling, "Privacy-Aware Secure Region-Based Handover for Small Cell Networks in 5G-Enabled Mobile Communication," in IEEE Transactions on Information Forensics and Security, vol. 18, pp. 1898-1913, 2023, doi: 10.1109/TIFS.2023.3256703. *This work is presented in Chapter 4.*
- R. Alnashwan, P. Gope, and B. Dowling, "UniHand: Privacy-preserving Universal Handover for Small-Cell Networks in 5G-enabled Mobile Communication with KCI Resilience," presented at the 2024 IEEE 37th Computer Security Foundations Symposium (CSF), IEEE Computer Society, Apr. 2024, pp. 96–111. doi: 10.1109/CSF61375.2024.00007. *This work is presented in Chapter 5.*
- R. Alnashwan, P. Gope, and B. Dowling, "PGUP: Pretty Good User Privacy for 5G-enabled Secure Mobile Communication Protocols," This work has been accepted in Privacy Enhancing Technologies Symposium (PETS 2025). *This work is presented in Chapter 6.*

1.5 Additional Collaborative Works

The following publications represent collaborative works completed during the course of this PhD research, but are not directly included in the main body of this thesis:

- R. Alnashwan, P. Gope, et al., "Strong Privacy-Preserving Universally Composable AKA Protocol with Seamless Handover Support for Mobile Virtual Network Operator," This work was accepted in ACM Conference on Computer and Communications Security (CCS'24).
- R. Alnashwan, B. Dowling, and Bhagya Wimalasiri, "Path Privacy and Handovers: Preventing Insider Traceability Attacks During Secure Handovers." This work has been accepted in IEEE Computer Security Foundations Symposium (CSF'25)

Chapter 2

Preliminaries

This chapter presents the fundamental concepts and security frameworks essential for understanding the cryptographic methods and security analyses employed throughout this thesis. We begin by introducing the notation and terminology that will be consistently employed in our discussions. The chapter then provides an overview of essential cryptographic primitives and advanced schemes that form the building blocks of our proposed security solutions for 5G networks. We also present the security framework that will be used to analyse and prove the security of our protocols.

2.1 Further Notation

This thesis uses specific notation throughout. Different typefaces distinguish various elements: participants in security games (like adversary $\mathcal{A}$ and challenger $\mathcal{C}$) are represented one way, while adversarial queries (such as **Corrupt** and **Reveal**) have their own style. Protocol algorithms are formatted distinctly (e.g. $\mathsf{AE}, \mathsf{KDF}$).

In our notation, we denote the set of all λ-bit long integers as $\{0,1\}^\lambda$, while $\{0,1\}^*$ represents the set of all bit strings of arbitrary length. We define a set X as empty if and only if $X = \emptyset$. The operation of uniformly sampling an element x from a set X, without replacement, is denoted by $(x \xleftarrow{\$} X)$. For a probabilistic algorithm A, we express the output b resulting from A's execution on input x and random coins as $b \xleftarrow{\$} A(x)$. Throughout our work, we employ $\perp$ as a generic symbol to indicate failure or invalid output from probabilistic algorithms. Additionally, we adopt the convention of using 1^λ to denote the *security parameter*. This parameter dictates the security level of cryptographic schemes and is used as input for key generation and other probabilistic algorithms.

2.2 Cryptographic Primitives

This section introduces and formalises the security of the underlying cryptographic primitives that constitute the building blocks for this thesis. We begin with fundamental concepts and progress to more specific schemes, each playing a crucial role in our proposed schemes for secure and efficient network authentication in 5G networks.

2.2.1 Key Derivation Function

Key Derivation Functions (KDFs) are cryptographic primitives designed to derive one or more secret keys from a master secret (such as a password, passphrase, or shared secret key) and other optional inputs [8]. KDFs are crucial in various cryptographic protocols and applications, including password-based encryption, key exchange protocols, and secure key storage. They serve to transform potentially weak or non-uniform input-keying material into cryptographically strong and uniformly distributed output keys. Common KDF constructions include HKDF, PBKDF2, and scrypt, each offering different security properties and performance characteristics. In our proposed schemes in Chapters 4, 5 and 6 of this thesis, we employ KDFs to securely derive keys for various cryptographic operations, ensuring that even if the master secret is weak, the derived keys maintain high entropy and security.

Definition 1 (Key Derivation Function) *A key derivation function* (KDF) *is a deterministic algorithm, taking four inputs: a random seed r, a length l, salt s and context c (where s and c are optional), and returning a key k of L bits $k = \mathsf{KDF}(r, l, s, c)$.*

The security of a KDF is captured through a game between an adversary and a challenger. First, the challenger generates a random value and salt, which are shared with the adversary. The adversary is allowed to make queries to obtain derived keys for chosen inputs. Next, the adversary is given a challenge key, which is either a real derived key or a random value. The adversary can then continue making queries, but not on the challenge input. Finally, the adversary outputs a guess to determine whether the challenge key was real or random. The KDF is considered secure if no efficient adversary can distinguish between the real and random key with a probability significantly better than chance.

Definition 2 (KDF Security) *Let KDF be a key derivation function. For an adversary $\mathcal{A}$, we capture the* KDF *security of* KDF *via the game played between challenger $\mathcal{C}$, and PPT $\mathcal{A}$ as follows:*

1. *$\mathcal{C}$ generates (r, α), where r is a random value sampled from distribution α. Next, $\mathcal{C}$ generates a uniformly random salt s, and returns (α, s) to $\mathcal{A}$.*
2. *$\mathcal{A}$ queries arbitrary (c_i, l_i) (where $c \notin (c_1.....c_i)$) to $\mathcal{C}$, who returns $k_i = \mathsf{KDF}(r, l_i, s, c_i)$.*
3. *Next, $\mathcal{A}$ chooses l and $c \notin (c_1.....c_t)$. $\mathcal{C}$ samples a bit $b \xleftarrow{\$} \{0,1\}$, and computes $k_0 = \mathsf{KDF}(r, l, s, c)$ and $k_1 \xleftarrow{\$} \{0,1\}^l$. $\mathcal{C}$ returns k_b to $\mathcal{A}$.*
4. *$\mathcal{A}$ outputs a guess bit b'.*

We say that KDF *is secure if for all PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathrm{KDF}}_{\mathsf{KDF}}(\mathcal{A})$ is negligible, where:*

$$\mathbf{Adv}^{\mathrm{KDF}}_{\mathsf{KDF}}(\mathcal{A}) = 2 \cdot |\mathbf{Pr}[(b = b')] - \frac{1}{2}|$$

2.2.2 Message Authentication Codes

Message Authentication Codes (MAC) is a cryptographic primitive that provides data integrity and authentication [16]. MACs allow the verification of a message's authenticity and

integrity using a shared secret key, ensuring that the message has not been tampered with and comes from the claimed sender. Common MAC algorithms include HMAC, CMAC, and GMAC, each with distinct security properties and use cases. MACs are widely used in network protocols, secure communication systems, and data storage solutions to detect unauthorized modifications and verify the source of messages. We employ MACs to ensure data integrity and authenticity in our proposed scheme in Chapter 6, leveraging their efficiency and strong security guarantees in symmetric key settings.

Definition 3 (Message Authentication Code) *A Message Authentication Code (MAC) consists of three algorithms* $\mathsf{MAC} : \{\mathsf{KGen}, \mathsf{Tag}, \mathsf{Verify}\}$ *associated with three sets: the secret-key space* $\mathcal{K}$*, the message space* $\mathcal{M}$*, and the tag space* $\mathcal{T}$*.*

- $k \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$*: a probabilistic key generation algorithm that takes as input a security parameter* 1^λ *and outputs a secret key* $k \in \mathcal{K}$*, where* $\mathcal{K}$ *is the key space.*
- $\tau \leftarrow \mathsf{Tag}(k, m)$*: a tagging algorithm that takes as input a secret key* $k \in \mathcal{K}$ *and a message* $m \in \mathcal{M}$*, and outputs a tag* $\tau \in \mathcal{T}$*.*
- $\{0, 1\} \leftarrow \mathsf{Verify}(k, m, \tau)$*: a verification algorithm that takes as input a secret key* $k \in \mathcal{K}$*, a message* $m \in \mathcal{M}$*, and a tag* $\tau \in \mathcal{T}$*, and outputs 1 if the tag is valid for the message under the given key, and 0 otherwise.*

MACs Correctness requires that for all $k \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$, all $m \in \mathcal{M}$, and $\tau \leftarrow \mathsf{Tag}(k, m)$, we have $\mathsf{Verify}(k, m, \tau) = 1$.

The security of a MAC is captured through the existential unforgeability under chosen message attacks (EUFCMA). The security game involves an adversary interacting with a tagging oracle, which provides valid tags for messages chosen by the adversary. The adversary wins the game if they can produce a valid (message, tag) pair that was not previously queried. This ensures that even if an attacker has access to many valid MAC tags, they cannot generate a valid tag for a new message.

Definition 4 (MAC Security) *Let* $\mathsf{MAC} = (\mathsf{KGen}, \mathsf{Tag}, \mathsf{Verify})$ *be a message authentication code scheme. For an adversary* $\mathcal{A}$*, we define the advantage of* $\mathcal{A}$ *in breaking the unforgeability of* MAC *as follows:*

$$\mathbf{Adv}^{EUFCMA}_{\mathsf{MAC}}(\mathcal{A}) = \Pr[k \xleftarrow{\$} \mathsf{KGen} : \mathcal{A}^{\mathsf{Tag}_k(\cdot)} \text{ forges}]$$

Where:

- $\mathsf{Tag}_k(\cdot)$ *is the tagging oracle that takes a message* M *as input and returns a tag* $\tau = \mathsf{Tag}(k, M)$*.*
- *The adversary is allowed to adaptively query the tagging oracle with messages of their choice.*
- *The adversary eventually outputs a candidate forgery* (M^*, τ^*)*.*
- *The adversary wins if:*

– $\mathsf{Verify}_k(M^*, \tau^*) = 1$ *(i.e., valid forgery).*

– M^* *was never queried to the tagging oracle before.*

We say that MAC *is secure if for all PPT adversaries* $\mathcal{A}$, $\mathbf{Adv}^{EUFCMA}_{\mathsf{MAC}}(\mathcal{A})$ *is negligible in the security parameter.*

2.2.3 Authenticated Encryption

Authenticated Encryption (AE) is a cryptographic scheme that simultaneously ensures data confidentiality, integrity, and authenticity [17]. It combines encryption for secrecy with authentication to verify the message source and detect tampering. This dual functionality gives AE more security for many applications, protecting against various attacks. AE employs various composition methods, each with distinct security and performance characteristics. Encrypt-then-MAC (EtM) encrypts the plaintext before computing a MAC on the ciphertext, offering strong security by detecting tampering before decryption. Encrypt-and-MAC (E&M) performs these operations independently, providing weaker security due to potential information leakage from the MAC of the plaintext and the lack of binding between the ciphertext and the MAC. MAC-then-Encrypt (MtE) computes a MAC on the plaintext before encrypting both, but can be complex to implement securely. Integrated AE schemes, such as GCM, CCM, and OCB [58], combine encryption and authentication in a single step, balancing robust security with computational efficiency. Therefore, we adopt the integrated AE to secure communications in Chapters 4, 5 due to the provided security guarantees, and widespread acceptance in modern cryptographic protocols.

Definition 5 (Authenticated Encryption) *The* AE *consists of three algorithms* $\mathsf{AE} : \{\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *associated with two sets: the secret-key space* $\mathcal{K}$*, and the message space* $\mathcal{M}$*.*

- $k \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$*: a probabilistic key generation algorithm that takes as input a security parameter* 1^λ *and outputs a secret key* $k \in \mathcal{K}$*, where* $\mathcal{K}$ *is the key space.*
- $c \xleftarrow{\$} \mathsf{Enc}(k, m)$*: a probabilistic encryption algorithm takes an input of a secret key* $k \in \mathcal{K}$*, and a message* $m \in \mathcal{M}$ *and outputs a ciphertext* $c \in \{0,1\}^*$*.*
- $m/\perp \leftarrow \mathsf{Dec}(k, c)$*: a decryption algorithm takes an input of a secret key* $k \in \mathcal{K}$ *and a ciphertext* $c \in \{0,1\}^*$*, and outputs either a message* $m \in \mathcal{M}$ *or* $\perp$ *for failure.*

AE Correctness: The correctness of an AE scheme requires that for all keys $k \in \mathcal{K}$ and all messages $m \in \mathcal{M}$, we have $\mathsf{Dec}(k, \mathsf{Enc}(k, m)) = m$.

For the security of AE we follow standard cryptographic frameworks established in the literature, particularly those formalized by Bellare and Rogaway in their foundational work on authenticated encryption and security notions for symmetric encryption schemes [17]. These definitions capture both confidentiality and authenticity, which are the two primary security properties of authenticated encryption. The confidentiality notion is modeled using IND-CPA security, ensuring that an adversary cannot distinguish between encryptions of different messages. The authenticity notion is captured using INT-CTXT security, preventing adversaries from forging valid ciphertexts.

Definition 6 (Authenticated Encryption Security) *Let* $\mathsf{AE} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be an authenticated encryption scheme. Let* $\ell(\cdot)$ *be a length function of* AE*, which takes the length of a message and returns the ciphertext length. The security of* AE *is captured through the following two games, where an adversary* $\mathcal{A}$ *interacts with a challenger:*

1. ***Confidentiality (IND-CPA):*** *The advantage of* $\mathcal{A}$ *in breaking the indistinguishability under chosen plaintext attack (IND-CPA) is defined as:*

$$\mathbf{Adv}^{\textit{IND-CPA}}_{\mathsf{AE}}(\mathcal{A}) = |\Pr[k \xleftarrow{\$} \mathsf{KGen}(1^\lambda), b \xleftarrow{\$} \{0,1\} : \mathcal{A}^{\mathsf{Enc}_k(\cdot)}(M_0, M_1) = b] - \frac{1}{2}|$$

   where:

   - $\mathsf{Enc}_k(M)$ *is the encryption oracle that takes message* M *as input and returns a ciphertext* $C \xleftarrow{\$} \mathsf{Enc}(k, M)$*.*
   - $\mathcal{A}$ *submits two messages* $M_0, M_1 \in \mathcal{M}$ *of equal length to the challenger.*
   - *The challenger samples a bit* $b \xleftarrow{\$} \{0,1\}$ *and returns the ciphertext* $C^* = \mathsf{Enc}_k(M_b)$*.*
   - $\mathcal{A}$ *continues making encryption queries and eventually outputs a guess bit* b'*.* $\mathcal{A}$ *succeeds if* $b = b'$ *.*

2. ***Authenticity (INT-CTXT):*** *The advantage of* $\mathcal{A}$ *in breaking the integrity of ciphertexts (INT-CTXT) is defined as:*

$$\mathbf{Adv}^{\textit{INT-CTXT}}_{\mathsf{AE}}(\mathcal{A}) = \Pr[k \xleftarrow{\$} \mathsf{KGen}(1^\lambda) : \mathcal{A}^{\mathsf{Enc}_k(\cdot)} \textit{ forges}]$$

   where:

   - $\mathsf{Enc}_k(\cdot)$ *is the encryption oracle that, given a message* M*, returns a ciphertext* $C \xleftarrow{\$} \mathsf{Enc}(k, M)$*.*
   - $\$(\cdot)$ *is an oracle that, on input* M*, returns a random string of length* $\ell(|M|)$*.*
   - $\mathcal{A}$ *queries the encryption oracle adaptively with messages of their choice.*
   - *Eventually,* $\mathcal{A}$ *outputs a candidate forgery* C^**.*
   - $\mathcal{A}$ *succeeds (i.e. forges) if* $\mathsf{Dec}(k, C^*) \neq \perp$ *and* C^* *was not previously output by the encryption oracle.*

We say that AE *is secure if for all PPT adversaries* $\mathcal{A}$*, both* $\mathbf{Adv}^{\textit{IND-CPA}}_{\mathsf{AE}}(\mathcal{A})$ *and* $\mathbf{Adv}^{\textit{INT-CTXT}}_{\mathsf{AE}}(\mathcal{A})$ *are negligible in the security parameter.*

While AE provides a robust solution for ensuring both confidentiality and integrity of data, modern cryptographic applications often require additional flexibility. This need led to the development of Authenticated Encryption with Associated Data (AEAD), an extension of the AE concept. AEAD builds upon the foundations of AE but introduces a key feature that addresses a common requirement in real-world scenarios: the ability to include additional, non-confidential data in the authentication process. This evolution in authenticated encryption schemes offers enhanced versatility without compromising the security guarantees provided by traditional AE.

2.2.4 Authenticated Encryption with Associated Data

Authenticated Encryption with Associated Data (AEAD) is a cryptographic scheme that ensures a message's confidentiality, integrity, and authenticity while also authenticating additional, non-confidential associated data. It extends traditional Authenticated Encryption by allowing context-specific information to be bound to the ciphertext without being encrypted. Precisely, we follow Rogaway's nonce-based AEAD scheme [57]. Nonce-based AEAD is a deterministic authentication encryption scheme that takes as input a secret key, a unique nonce, plaintext to be encrypted, and associated data to be authenticated but not encrypted. It produces a ciphertext and an authentication tag as output. Using a nonce makes each encryption unique, even for identical plaintexts. It is particularly useful in message encryption scenarios with message authentication, such as in network protocols or encrypted file systems. AEAD's robust security properties make it a cornerstone of modern cryptographic protocols, and its applications are central to secure communication within the proposed scheme in Chapter 6.

Definition 7 (Authenticated Encryption with Associated Data) *The AEAD consists of three algorithms* $AEAD : \{\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec}\}$ *associated with four sets: the secret-key space* $\mathcal{SK}$*, the nonce space* $\mathcal{N}$*, the associated data space* $\mathcal{AD}$ *and the message space* $\mathcal{M}$*.*

- $sk \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$*: a probabilistic key generation algorithm that takes as input a security parameter* 1^λ *and outputs a secret key* $sk \in \mathcal{SK}$*, where* $\mathcal{SK}$ *is the key space.*
- $c \leftarrow \mathsf{Enc}(sk, n, ad, m)$*: a deterministic encryption algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a nonce* $n \in \mathcal{N}$*, an associated data* $ad \in \mathcal{AD}$*, and a message* $m \in \mathcal{M}$ *and outputs a ciphertext* c*.*
- $m/\perp \leftarrow \mathsf{Dec}(sk, n, ad, c)$*: a deterministic decryption algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a nonce* $n \in \mathcal{N}$*, an associated data* $ad \in \mathcal{AD}$*, and a ciphertext* $c \in \{0,1\}^*$*, and outputs either a message* $m \in \mathcal{M}$ *or* $\perp$ *for failure.*

AEAD Correctness: The correctness of an AEAD scheme requires that for all secret keys $sk \in \mathcal{SK}$, all nonces $n \in \mathcal{N}$, all associated data $ad \in \mathcal{AD}$, and all messages $m \in \mathcal{M}$, we have $\mathsf{Dec}(sk, n, ad, \mathsf{Enc}(sk, n, ad, m)) = m$

Definition 8 (AEAD Security) *Let* $\mathsf{AEAD} = (\mathsf{KGen}, \mathsf{Enc}, \mathsf{Dec})$ *be an authenticated encryption with associated data scheme. Let* $\ell(\cdot)$ *be a length function of* AEAD*. The security of* AEAD *is captured through the following two security games, where an adversary* $\mathcal{A}$ *interacts with a challenger:*

1. ***Confidentiality (IND-CPA):*** *The advantage of* $\mathcal{A}$ *in breaking the indistinguishability under chosen plaintext attack (IND-CPA) is defined as:*

$$\mathbf{Adv}^{IND\text{-}CPA}_{\mathsf{AEAD}}(\mathcal{A}) = \left| \Pr[sk \xleftarrow{\$} \mathsf{KGen}, b \xleftarrow{\$} \{0,1\} : \mathcal{A}^{\mathsf{Enc}_{sk}(\cdot,\cdot,\cdot)}(n, ad, M_0, M_1) = b] - \frac{1}{2} \right|$$

where:

- $\mathcal{A}$ *is given access to an encryption oracle* $\mathsf{Enc}_{sk}(\cdot,\cdot,\cdot)$*.*

- *$\mathcal{A}$ submits a nonce n, associated data ad, and two messages $M_0, M_1 \in \mathcal{M}$ of equal length to the challenger.*
- *The challenger samples a secret bit $b \xleftarrow{\$} \{0,1\}$ and returns the ciphertext $C^* = \mathsf{Enc}_{sk}(n, ab, M_b)$.*
- *$\mathcal{A}$ continues making encryption queries and eventually outputs a guess bit b'. $\mathcal{A}$ succeeds if $b = b'$.*

2. ***Authenticity (INT-CTXT):*** *The advantage of $\mathcal{A}$ in breaking the integrity of ciphertexts (INT-CTXT) is defined as:*

$$\mathbf{Adv}^{\textit{INT-CTXT}}_{\mathsf{AEAD}}(\mathcal{A}) = \Pr[sk \xleftarrow{\$} \mathsf{KGen} : \mathcal{A}^{\mathsf{Enc}_{sk}(\cdot,\cdot,\cdot),\mathsf{Dec}_{sk}(\cdot,\cdot,\cdot)} \textit{ forges}]$$

where:

- *$\mathsf{Enc}_{sk}(n, ad, M)$ is the encryption oracle that takes nonce n, associated data ad, and message M as input.*
- *$\mathsf{Dec}_{sk}(n, ad, C)$ is the decryption oracle that takes n, ad, and ciphertext C as input.*
- *$\mathcal{A}$ succeeds if it outputs a tuple (n, ad, C) such that:*
  - *$\mathsf{Dec}_{sk}(n, ad, C) \neq \perp$ (i.e., the forgery is accepted as valid).*
  - *(n, ad, C) was not previously output by the encryption oracle (i.e., $\mathsf{Enc}_{sk}(n^*, ad^*, M) \to C^*$ was never queried for $n = n^*$, $ad = ad^*$, and $C = C^*$).*

We say that AEAD *is secure if for all PPT adversaries $\mathcal{A}$, both $\mathbf{Adv}^{\textit{IND-CPA}}_{\mathsf{AEAD}}(\mathcal{A})$ and $\mathbf{Adv}^{\textit{INT-CTXT}}_{\mathsf{AEAD}}(\mathcal{A})$ are negligible in the security parameter.*

2.2.5 Diffie-Hellman Key Exchange

The Diffie-Hellman key exchange is a cryptographic primitive that allows two parties to securely share a common secret key over an insecure communication channel [23]. This key can then be used for secure communication using symmetric encryption algorithms.

This process involves a cyclic group G of order q with generator g, where two communicating parties selects a private key, computes a corresponding public key using the generator, and exchanges these public keys. The shared secret is then computed by each party using their private key and the other party's public key, resulting in a secure symmetric encryption. The security of this primitive fundamentally relies on the Computational Diffie-Hellman (CDH) assumption, which states that given g^a and g^b, it's computationally infeasible to calculate g^{ab}, based on the difficulty of the discrete logarithm problem. Building upon this, the Decisional Diffie-Hellman (DDH) assumption provides a stronger security notion. DDH assumes that distinguishing the shared secret g^{ab} from a random group element is computationally infeasible, offering a more robust foundation for advanced cryptographic protocols. While CDH forms the basic security premise of Diffie-Hellman, DDH extends this to enable stronger security proofs and more sophisticated cryptographic constructions, thus providing greater assurance for complex protocols built on top of the Diffie-Hellman key exchange. We utilised the DDH assumption in our thesis, specifically in **Chapters 4 and 5**, as it serves as a stronger notion compared to CDH and provides key indistinguishability, which is crucial for ensuring the security of our proposed protocols.

2.2.5.1 Computational Diffie-Hellman Assumption

The Computational Diffie-Hellman (CDH) problem is a fundamental concept in modern cryptography. It centers on the computational difficulty of deriving a specific group element from partial information. The CDH assumption states that given a cyclic group G of order q with generator g, and given g^a and g^b for unknown random integers a and b, it is computationally infeasible to compute g^{ab} without knowing either a or b.

Definition 9 *Let $G \in \mathbb{Z}$ be a cyclic group family, with a generator g and a prime order q. We say that the CDH assumption holds if no PPT $\mathcal{A}$ can compute g^{ab}, with non-negligible advantage $\mathbf{Adv}^{G,g,q}_{\mathsf{CDH}}$, where $a, b \xleftarrow{\$} \mathbb{Z}_q$:*

$$\mathbf{Adv}^{G,g,q}_{\mathsf{CDH}}(\mathcal{A}) = Pr[\mathcal{A}(g, g^a, g^b) = g^{ab}]$$

2.2.5.2 Decisional Diffie-Hellman Assumption

The Decisional Diffie-Hellman (DDH) problem [22] is a key concept in modern cryptography. It focuses on the computational challenge of distinguishing between certain distributions in a cyclic group. The DDH assumption states that it is computationally challenging to determine if given group elements are related in a specific way or are randomly generated. This assumption is crucial for many security protocols, especially in public-key cryptography and key exchange, enabling secure communication and data protection.

Definition 10 (Decisional Diffie-Hellman) *Let $G \in \mathbb{Z}$ be a cyclic group family, with a generator g and a prime order q. We say that the DDH assumption holds if no PPT $\mathcal{A}$ can distinguish (g^a, g^b, g^{ab}) from (g^a, g^b, g^c), with non-negligible advantage $\mathbf{Adv}^{G,g,q}_{\mathsf{DDH}}$, where $a, b, c \xleftarrow{\$} \mathbb{Z}_q$:*

$$\mathbf{Adv}^{G,g,q}_{\mathsf{DDH}}(\mathcal{A}) = |Pr[\mathcal{A}(g^a, g^b, g^{ab} = 1] - Pr[\mathcal{A}(g^a, g^b, g^c = 1]|$$

2.2.6 Key Encapsulation Mechanism

A Key Encapsulation Mechanism (KEM) is a public-key cryptographic primitive designed to securely transmit symmetric keys. KEM play a crucial role in 5G security protocols, particularly within the Elliptic Curve Integrated Encryption Scheme (ECIES) employed in 5G specifications (TS 33.501). In the context of the 5G Authentication and Key Agreement (5G-AKA) protocol, which is a central focus of this thesis, the KEM component of ECIES is especially relevant for concealing users' identities.

Definition 11 (Key Encapsulation Mechanism) *A Key encapsulation mechanism consists of three algorithms $\mathsf{KEM} : \{\mathsf{KGen}, \mathsf{Encaps}, \mathsf{Decaps}\}$ associated with a key space $\mathcal{K}$.*

- $(sk, pk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$*: a probabilistic key generation algorithm that takes as input a security parameter 1^λ and outputs public and secret keys $pk, sk \in \mathcal{K}$.*
- $(k_s, C) \xleftarrow{\$} \mathsf{Encaps}(pk)$*: a probabilistic encapsulation algorithm that takes as input a public key $pk \in \mathcal{K}$ and outputs a symmetric key $k_s, \in \mathcal{K}$ and an encapsulation (key or ciphertext) C.*

- $(k_s) \leftarrow \mathsf{Decaps}(sk, C)$*: a deterministic decapsulation algorithm that takes as input a secret key* $sk \in \mathcal{K}$ *and an encapsulation* C*, and outputs either a symmetric key* $k_s \in \mathcal{K}$ *or* $\perp$ *to indicate failure.*

KEM Correctness requires that for all $(sk, pk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$ and all $(k_s, C) \xleftarrow{\$} \mathsf{Encaps}(pk)$, we have $\mathsf{Decaps}(sk, C) = k_s$.

Definition 12 (KEM Security) *The security of a KEM under the IND-CPA is defined as follows. For an adversary* $\mathcal{A}$*, we define the advantage of* $\mathcal{A}$ *in breaking the IND-CPA security of KEM as:*

$$\mathbf{Adv}^{\textit{IND-CPA}}_{\mathsf{KEM}}(\mathcal{A}) = |2 \cdot \Pr[\mathit{Exp}^{\textit{IND-CPA}}_{\mathsf{KEM}}(\mathcal{A}) = 1] - 1|$$

$\mathbf{Exp}^{\mathbf{IND\text{-}CPA}}_{\mathsf{KEM}}(\mathcal{A})$

1: $(sk, pk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$
2: $(k_s, C^*) \xleftarrow{\$} \mathsf{Encaps}(pk)$
3: $k'_s \xleftarrow{\$} \mathcal{K}$
4: $b \xleftarrow{\$} \{0, 1\}$
5: $b' \leftarrow \mathcal{A}(pk, C^*, k_b)$
6: **return** $(b' = b)$

We say that KEM is IND-CPA secure if for all PPT adversaries $\mathcal{A}$, $\mathbf{Adv}^{\textit{IND-CPA}}_{\mathsf{KEM}}(\mathcal{A})$ *is negligible in the security parameter* λ*.*

2.2.7 Puncturable Pseudorandom Functions

Puncturable Pseudorandom Functions (PPRFs), introduced by Sahai and Waters [61], extend standard Pseudorandom Functions (PRFs) by allowing selective restriction of the function's capability. While a PRF, given a secret key, generates outputs indistinguishable from random values, a PPRF permits the key holder to "puncture" the function at specific inputs. After puncturing, the function remains operational for all inputs except the punctured ones, without revealing information about the function's output on those points. This selective restriction property enables PPRFs to serve as fundamental building blocks in various cryptographic constructions, including obfuscation techniques, and forward-secure encryption protocols. In the context of this thesis, PPRFs play a crucial role in the development of Puncturable Key Wrapping schemes (Section 2.2.10) and in PKW^+ introduced Chapter 6.

Definition 13 (Puncturable Pseudorandom Function) *A puncturable pseudorandom function* PPRF *consists of three algorithms* $\mathsf{PPRF} : \mathsf{Setup}, \mathsf{Eval}, \mathsf{Punc}$ *associated with three sets: the key space* $\mathcal{K}$*, the input space* $\mathcal{X}$*, and the output space* $\mathcal{Y}$*.*

- $k \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$*: a probabilistic algorithm that takes as input a security parameter* 1^λ *and outputs a key* $k \in \mathcal{K}$*.*

- $y/\perp \leftarrow \mathsf{Eval}(k, x)$*: an algorithm that takes as input a key* $k \in \mathcal{K}$ *and an input* $x \in \mathcal{X}$*, and outputs a value* $y \in \mathcal{Y}$*.or* $\perp$ *to indicate failure*
- $k_s \leftarrow \mathsf{Punc}(k, x)$*: a deterministic algorithm that takes as input a key* $k \in \mathcal{K}$ *and an element* $x \in \mathcal{X}$*, and outputs a punctured key* $k_s \in \mathcal{K}$*.*

PPRF correctness *requires that for all* $k \in \mathcal{K}$ *and every* $x, y \in \mathcal{Y}$*:*

1. $\mathbf{Pr}[\mathsf{Eval}(k_0, x)] \neq \perp \| \mathsf{Setup}(1^\lambda) \xrightarrow{\$} k_0] = 1$
2. *If* $k_s \leftarrow \mathsf{Punc}(k, x)$ *and* $y \neq x$*, then* $\mathsf{Eval}(k, y) = \mathsf{Eval}(k_s, y)$*.*
3. *If* $k_s \leftarrow \mathsf{Punc}(k, x)$*, then* $\mathsf{Eval}(k_s; x) = \perp$

Definition 14 (PPRF Security) *A Puncturable Pseudorandom Functions* $\mathsf{PPRF} = (\mathsf{Setup}, \mathsf{Eval}, \mathsf{Punc})$ *satisfies forward pseudorandomness if for all PPT adversaries* $\mathcal{A}$*, the advantage of* $\mathcal{A}$ *in the fpr experiment is negligible:*

$$\mathbf{Adv}^{fpr}_{\mathsf{PPRF}}(\mathcal{A}) = |\Pr[Exp^{fpr}_{\mathsf{PPRF}}(\mathcal{A}) = 1] - \frac{1}{2}|$$

$\underline{\mathbf{Exp}^{fpr}_{\mathsf{PPRF}}(\mathcal{A})}$
1: $b \xleftarrow{\$} \{0,1\}$; $u \leftarrow 0$; $\mathbf{T}[\cdot,\cdot] \leftarrow \perp$
2: $b^* \leftarrow \mathcal{A}^{\text{New,Corr,Ro\$-Eval,Punc}}()$
3: **return** $b^* = b$

$\underline{\mathsf{New}()}$
1: $u++$; $k_u \xleftarrow{\$} \mathsf{Setup}(1^\lambda)$
2: $\mathcal{C}_u, \mathcal{E}_u, \mathcal{P}_u \leftarrow \emptyset$; $corr_u \leftarrow$ false

$\underline{\mathsf{Punc}(i, x)}$
1: $k_i \leftarrow \mathsf{Punc}(k_i, x)$
2: $\mathcal{P}_i \leftarrow \mathcal{P}_i \cup x$

$\underline{\mathbf{Corrupt(i)}}$
1: **if** $\mathcal{C}_i \nsubseteq \mathcal{P}_i$ **then**
2: **return** $\perp$
3: **end if**
4: $corr_i \leftarrow$ true
5: **return** k_i

$\underline{\mathbf{Ro}\text{–}Eval\$(i, x)}$
1: **if** $x \in \mathcal{E}_i$ or $corr_i =$ true **then**
2: **return** $\perp$
3: **end if**
4: $y_1 \leftarrow \mathsf{Eval}(k_i, x)$
5: **if** $y_1 = \perp$ **then return** $\perp$
6: **end if**
7: **if** $\mathbf{T}[i, x] = \perp$ **then**
8: $\mathbf{T}[i, x] \xleftarrow{\$} \mathcal{Y}$
9: **end if**
10: $y_0 \leftarrow \mathbf{T}[i, x]$
11: $\mathcal{C}_i \leftarrow \mathcal{C}_i \cup x$
12: **return** y_b

Figure 2.1: *PPRF security*

where the FPR game $\mathbf{G}^{fpr}_{\mathsf{PPRF}}(\mathcal{A})$*, shown in Figure 2.1* $Ro - \mathsf{Eval}$*, allows the adversary to have oracle access (*$Ro\$ - \mathsf{Eval}$*) to either the real function evaluated with a hidden key or a randomly sampled function. Forward security is ensured by allowing access to a puncturing oracle (*Punc*) and a corruption oracle (**Corrupt**), through which the adversary can obtain secret keys that have been modified (punctured) at all challenge points.*

To ensure consistency in the puncturing operation, regardless of the order in which inputs are punctured, we define invariant puncturing as follows:

Definition 15 (Invariant PPRF) *A PPRF is invariant to puncturing if for all keys* $k \in \mathcal{K}$ *and all elements* $x_0, x_1 \in \mathcal{X}$*,* $x_0 \neq x_1$ *it holds that*

$$\mathsf{Punc}(\mathsf{Punc}(k, x_0), x_1) = \mathsf{Punc}(\mathsf{Punc}(k, x_1), x_0)$$

2.2.8 Sanitizable Signatures

Sanitizable Signatures (SanSig) [12] are a signature scheme where signing capabilities can be delegated to another party: the so-called sanitizer. The sanitizer can modify parts of a signed message (i.e., perform admissible modifications), generating another valid signature over the modified message without the original signer's assistance. SanSig requires a pair of efficient deterministic algorithms ADM and MOD. MOD maps any message m to a modified message $m' = \mathsf{MOD}(m)$, while $\mathsf{ADM}(\mathsf{MOD}) \rightarrow \{0,1\}$ indicates whether the modification is admissible and matches ADM, with $\mathsf{ADM}(\mathsf{MOD}) = 1$ if the modification is admissible.

Definition 16 (Sanitizable Signatures) *The* SanSig *consists of six algorithms* $\mathsf{SanSig} = \{\mathsf{KGen}, \mathsf{Sign}, \mathsf{Sanit}, \mathsf{Verify}, \mathsf{Proof}, \mathsf{Judge}\}$*:*

- ***Key Generation:*** *is a pair of key generation algorithms for the signer and the sanitizer respectively:*

$$(sk_{sig}, pk_{sig}) \xleftarrow{\$} \mathsf{KGen}_{sig}(1^\lambda),$$

$$(sk_{san}, pk_{san}) \xleftarrow{\$} \mathsf{KGen}_{san}(1^\lambda).$$

- ***Signing:*** *takes as input a message $m \in \{0,1\}^*$, a signer private key sk_{sig}, sanitizer public key pk_{san} and the modifiable message segments (*ADM*).* Sign *either outputs a signature σ, or $\perp$ if failed:*

$$\sigma \leftarrow \mathsf{Sign}(m, sk_{sig}, pk_{san}, \mathsf{ADM}).$$

- ***Sanitizing:*** *takes as input an original message m, a modification of the original message* MOD*, a signature σ, signer public key pk_{sig} and sanitizer private key sk_{san}.* Sanit *outputs either a modified message m^* and a signature σ^*, or $\perp$ if failed:*

$$(m^*, \sigma^*) \leftarrow \mathsf{Sanit}(m, \mathsf{MOD}, \sigma, sk_{san}, pk_{sig}).$$

- ***Verification:*** *takes as input a message m, a signature σ and the public keys of the signer pk_{sig} and sanitizer pk_{san}.* Verify *outputs a bit $b \in \{0,1\}$, where $b = 1$ if σ verifies message m under pk_{san} and pk_{sig}, and $b = 0$ otherwise:*

$$b \leftarrow \mathsf{Verify}(m, \sigma, pk_{sig}, pk_{san})$$

- ***Proof:*** *takes as input a message m, signer private key sk_{sig}, sanitizer public key pk_{san} and a set of signature-message pairs (m_i, σ_i) where $i = 1, \ldots, q$.* Proof *outputs a string $\pi \in \{0,1\}^*$*

$$\pi \leftarrow \mathsf{Proof}(m, \sigma, (M_1, \sigma_1), ..., (m_q, \sigma_q), sk_{sig}, pk_{san})$$

- ***Judge:*** *takes as input a message m, a signature σ, a proof π and public keys of both signer pk_{sig} and sanitizer pk_{sig}.* Judge *outputs a $flag \in Sign, San$ indicating which party generated the signature:*

$$flag \leftarrow \mathsf{Judge}(m, \sigma, pk_{san}, pk_{sig}, \pi)$$

Correctness of Sanitizable Signatures In SanSig, the correctness of the signing, sanitizing, and proof algorithms must be maintained. This implies that all messages signed or sanitized by honest parties are accepted, and honest parties can generate a valid proof to assist the judge algorithm in accurately identifying the correct signer. For a security parameter $n \in \mathbb{N}$, where $(sk_{sig}, pk_{sig}) \xleftarrow{\$} KGen_{sig}(1^n)$, $(sk_{san}, pk_{san}) \xleftarrow{\$} KGen_{san}(1^n)$ and $m \in \{0,1\}^*$:

- **Signing Correctness**: for any $\mathsf{ADM} \in \mathbb{N} \times 2^N$, and $\sigma \leftarrow Sign(m, sk_{sig}, pk_{san}, \mathsf{ADM})$, then it follows that:
$$\mathsf{Verify}(m, \sigma, pk_{sig}, pk_{san}) = 1.$$
- **Sanitization Correctness**: for any σ with $\mathsf{Verify}(m, \sigma, pk_{sig}, pk_{san}) = 1$, $\mathsf{MOD} \subseteq N \times \{0,1\}^l$ matching ADM from σ and any pair $(m^*, \sigma^*) \leftarrow Sanit(m, \mathsf{MOD}, \sigma, pk_{sig}, sk_{san})$, then it follows that:
$$\mathsf{Verify}(m^*, \sigma^*, pk_{sig}, pk_{san}) = 1.$$
- **Proof Correctness**: for any signature σ, MOD matching ADM from σ, and any $(m^*, \sigma^*) \leftarrow Sanit(m, \mathsf{MOD}, \sigma, sk_{san}, pk_{sig})$ with $\mathsf{Verify}(m^*, \sigma^*, pk_{sig}, pk_{san}) = 1$, and any $m_1, \cdots, m_q$ and $\mathsf{ADM}_1, \cdots, \mathsf{ADM}_q$ with $\sigma_i \leftarrow Sign(m_i, sk_{sig}, pk_{san}, \mathsf{ADM}_i)$ and $(m_i, \sigma_i) = (m, \sigma)$ for some i any $\pi \leftarrow Proof(m^*, \sigma^*, (m_1, \sigma_1), ..., (m_q, \sigma_q), sk_{sig}, pk_{san})$, then it follows that:
$$\mathsf{Judge}(m^*, \sigma^*, pk_{san}, pk_{sig}, \pi) = flag.$$

Sanitizable Signatures security: According to [12], SanSig must satisfy signature unforgeability along with several additional security properties: immutability, privacy, transparency, and accountability. Each of these properties plays a crucial role in maintaining the integrity and security of the SanSig scheme. While here we provide an overview of these key properties, it's important to note that comprehensive security proofs for each are extensively discussed by Brzuska et al. [27]. Among these properties, unforgeability is particularly important for many of the security proofs developed later in this thesis. Therefore, we present the formal unforgeability experiment (Experiment $\mathrm{Unforgeability}_{\Pi}^{\mathsf{SanSig}}(n)$ - (Algorithm 1)) in this section, as it forms the foundation for subsequent security analyses.

1. **Unforgeability:** The unforgeability of SanSig was initially defined by [12] and [27], based on the conventional notion of unforgeability in regular signature schemes. In SanSig, only the signer and sanitizers are capable of generating valid signatures. Consequently, no one can generate a valid pair (σ^*, m^*) without the secret keys of either the signer or the sanitizer, such that $\mathsf{Verify}(\sigma^*, m^*, pk_{san}, pk_{sig}) = 1$.

   Definition 17 (Sanitizable Signature Unforgeability) *A Sanitizable signature scheme is unforgeable, if no PPT algorithm $\mathcal{A}$ can forge a signature with probability greater than ε, such that the advantage of $\mathcal{A}$ winning the following experiment is negligible:*

$$\mathbf{Adv}_{\mathsf{SanSig}}^{EUFCMA}(\mathcal{A}) = Pr[Exp_{\mathsf{SanSig},\mathcal{A}}^{EUFCMA}(n) = 1] \leq \varepsilon$$

Algorithm 1 $\mathbf{Exp}^{EUFCMA}_{\mathsf{SanSig}}(n)$

$(sk_{sig}, pk_{sig}) \xleftarrow{\$} \mathsf{KGen}_{sig}(1^n)$
$(sk_{san}, pk_{san}) \xleftarrow{\$} \mathsf{KGen}_{san}(1^n)$
$(m^*, \sigma^*) \xleftarrow{\$} \mathcal{A}^{\mathsf{Sign}(\cdot,\cdot,sk_{sig},\cdot),\mathsf{Sanit}(\cdot,\cdot,\cdot,sk_{san},pk_{sig},\cdot)}(pk_{sig}, pk_{san})$
  Let $(m_i, \mathsf{ADM}_i, pk_{san,i})$ and σ_i for $i = 1, 2, \ldots, q$
  denote the queries and answers to and from algorithm Sign,
  and $(m'_j, \mathsf{MOD}_j, \sigma'_j)$ and (m''_j, σ''_j) for $j = q+1, \ldots, r$
  denote the queries and answers to and from algorithm Sanit.
if $\mathsf{Verify}(m^*, \sigma^*, pk_{sig}, pk_{san}) = 1$ **and then**
    for all $i = 1, 2, \ldots, q$ we have $(pk_{san}, m^*) \neq (pk_{san,i}, m_i)$ **and**
    for all $j = q+1, \ldots, r$ we have $(pk_{sig}, m^*) \neq (pk_{sig,j}, m''_j)$
  return 1
else
  return 0
end if

2. **Immutability:** Sanitizers are restricted to modifying only the parts of the message that have been explicitly permitted by the signer. If a sanitizer attempts to modify other parts of the message, they will be unable to generate a valid signature for the altered message.

3. **Accountability:** This ensures a non-repudiation property, where neither the signer nor the sanitizer can deny authorship of the signatures. In the event of a dispute, the Judge algorithm is employed to resolve it. There are two types of accountability:

   (a) Sanitizer Accountability: This ensures that a malicious sanitizer cannot manipulate the Judge algorithm to falsely accuse the signer of signing a message.

   (b) Signer Accountability: This ensures that a malicious signer cannot manipulate the Judge algorithm to falsely accuse the sanitizer of signing a message.

4. **Privacy:** The full original message cannot be recovered from sanitized messages. Thus, an adversary cannot retrieve the original information of the modifiable parts from the sanitized message. For example, if the user's Identity (UID) is updated, the older version should not be recoverable.

5. **Transparency:** Signatures created by the signer or the sanitizer should be indistinguishable to others. This ensures that it is difficult to determine whether a message has been signed or sanitized, thereby enhancing privacy and security.

2.2.8.1 Construction and Relevance to This Thesis

Sanitizable signatures can be constructed using conventional/standard cryptographic primitives, with various approaches sharing a common fundamental structure. This structure typically consists of an inner part covering the fixed (non-sanitizable) elements and an outer part encompassing the entire message, including the sanitizable portions. The seminal work

by Ateniese et al.[12] introduced this concept, using chameleon hashes combined with a standard digital signature scheme. Subsequent research has explored different implementations of this inner-outer structure. For instance, Brzuska et al. [28] investigated unlinkability properties while maintaining this basic framework. Fleischhacker et al. [38] presented constructions using RSA-based signatures, and Lai et al. [48] developed efficient sanitizable signatures without random oracles. Despite their varying approaches, all these constructions adhere to the principle of an immutable inner signature and a flexible outer signature.

In this thesis, we employ sanitizable signatures as a conceptual framework, focusing on their general properties and applications rather than a specific construction. This approach allows for a broader exploration of their potential in network security and authentication while aligning with methods presented in the literature [12, 28, 38, 48]. Our theoretical discussions and analyses are applicable to the broader concept of sanitizable signatures, independent of the underlying cryptographic primitives used. However, we adopt a two-layer structure based on conventional digital signature schemes for implementation purposes, specifically using RSA signatures. This structure maintains the consistent framework of an inner signature for immutable parts and an outer signature encompassing the entire message, including sanitizable components. This approach balances security, efficiency, and flexibility in our specific use case while utilizing well-established cryptographic primitives to achieve the desired sanitizable properties. A comprehensive explanation of our implementation, including specific algorithms and processes, is provided in Appendix A.1.

The use of SanSig is integral to this thesis, with their significance demonstrated in Chapters 4 and 5. SanSig addresses crucial requirements in our proposed secure and flexible network authentication schemes. By providing a customizable signature mechanism, it enables the core network to delegate authentication responsibility to distributed entities, such as the gNodeB (gNB). This decentralization reduces computational and communication overhead on the core network while maintaining robust end-to-end security. In contrast to conventional signatures, the flexibility offered by SanSig is essential for the proposed scheme's distributed authentication processes. This characteristic is particularly valuable in the context of 5G networks, where it enhances the efficiency of authentication and key agreement protocols and facilitates seamless and secure handover processes between heterogeneous network components, all while preserving the stringent security requirements of next-generation mobile networks.

2.2.9 Accumulators

Accumulators, introduced by Benaloh and de Mare in 1993 [21], are cryptographic schemes that compress multiple inputs into a single, compact string while preserving proofs for each input. The key property of an accumulator is its ability to generate short membership proofs, enabling efficient verification of whether an element belongs to the represented set without revealing the entire set contents.

The original accumulators proposed by Benaloh and de Mare, referred to as static accumulators, were based on the strong RSA assumption. These accumulators allowed for the addition of new elements and the regeneration of the accumulator with every addition. A significant advancement came in 2002 when Camenisch and Lysyanskaya [30] introduced dynamic accumulators. These accumulators addressed the limitation of static accumulators by allowing for efficient addition and deletion of elements without needing to regenerate the en-

tire accumulator. The key innovation lay in the computational efficiency of these operations, which remained independent of the accumulator's size.

In 2007, Li, Li, and Xue [50] proposed universal accumulators, further extending the capabilities of dynamic accumulators. Universal accumulators introduced the ability to generate efficient non-membership proofs in addition to membership proofs. This innovation significantly broadened the applicability of accumulators in cryptographic protocols, allowing for more complex set operations and queries. This thesis uses the universal accumulator for our proposed schemes to manage user revocation lists. Performing non-membership queries is especially beneficial in our context, as it allows us to manage and verify network access rights efficiently.

Definition 18 (Universal Accumulators) *The Universal Accumulators (*ACC*), consists of a family of algorithms,* $\mathsf{ACC} = \{\mathsf{KGen}, \mathsf{Gen}, \mathsf{Update}, \mathsf{NonWitCreate}, \mathsf{Verify}\}$*. These algorithms form the foundation of the accumulator's functionality. However, for the purposes of this thesis, we focus primarily on non-membership within the revocation list. Consequently, we will not delve into the details of membership witness creation and verification steps, as they are not central to our proposed schemes.*

- ***Key Generation:*** *A probabilistic key generation algorithm that takes as input a security parameter* 1^λ *and outputs a secret key* (sk)*.*

$$(sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$$

- ***Accumulator Generation:*** *This algorithm takes a secret key* sk*, and a set* X *of values to accumulate (the Revocation list), which is initially empty* $(X = \emptyset)$*.*

$$(\ddot{\mathsf{C}}) \xleftarrow{\$} \mathsf{Gen}(sk, X)$$

- ***Accumulator Update:*** *This algorithm takes a secret key (*sk*), the original accumulator* $\ddot{\mathsf{C}}$ *and the new value to be accumulated* x^**. It returns the updated accumulator* $\ddot{\mathsf{C}}^*$*.*

$$(\ddot{\mathsf{C}}^*) \leftarrow \mathsf{Update}(sk, \ddot{\mathsf{C}}, x^*)$$

- ***Non-membership Witness Generation:*** *For any element* x^* *that is not in* X*. There is an algorithm that takes a secret key (*sk*), the original accumulator* $\ddot{\mathsf{C}}$*,* X *and* x^**. to generate a non-membership witness* c_x *for* x^**.*

$$(c_x) \xleftarrow{\$} \mathsf{NonWitCreate}(sk, \ddot{\mathsf{C}}, X, x^*)$$

- ***Non-membership Witness Verification:*** *To verify that* $x \notin X$ *the verification, the algorithm takes the original accumulator* $\ddot{\mathsf{C}}$*,* c_x *and* x*. It outputs a bit* $b \in \{0, 1\}$*, where* $b = 1$ *if witness* c_x *hold, hence the value* $x \notin X$ *and not in the accumulator* $\ddot{\mathsf{C}}$*, and* $b = 0$ *otherwise.*

$$b \leftarrow \mathsf{Verify}(\ddot{\mathsf{C}}, c_x, X)$$

- ***Non-membership Witness Update*** *if a new element x^* has been added to the accumulator $\ddot{\mathsf{C}}^*$, this algorithm is responsible for updating the non-membership witness. It takes the original accumulator $\ddot{\mathsf{C}}$, the updated accumulator $\ddot{\mathsf{C}}^*$, x^*, x and the original non-membership witness c_x. It outputs the new non-membership witness c_x^*.*

$$(c_x^*) \xleftarrow{\$} \mathsf{NonWitUpdate}(\ddot{\mathsf{C}}, \ddot{\mathsf{C}}^*, x^*, x, c_x)$$

ACC ***Correctness:*** The correctness of an ACC scheme requires that for all non-membership $c_x \xleftarrow{\$} \mathsf{NonWitCreate}(sk, \ddot{\mathsf{C}}, X, x^*)$ where $sk \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$, $\ddot{\mathsf{C}} \xleftarrow{\$} \mathsf{Gen}(sk, X)$ and $X \neq \phi$, we have $\mathsf{Verify}(\ddot{\mathsf{C}}, c_x, x) = 1$

2.2.9.1 Construction and Relevance to This Thesis

Accumulator-based approaches offer advantages over conventional revocation lists, providing constant-size proofs and enabling efficient updates, independent of the number of revoked users. This facilitates rapid and scalable verification processes in large-scale distributed systems like 5G networks. Three main approaches can be considered when managing revocation. The first involves using membership proofs for revoked users, maintaining a list of revoked credentials. However, this method could potentially allow revoked users to manipulate their membership proof unless complex security measures are implemented. The second approach uses membership proofs for non-revoked users, but this introduces challenges in managing and updating the list of valid users, especially when new joins outnumber revocations. The third and most efficient approach employs non-membership proofs via a universal dynamic accumulator. This method represents revoked users through an accumulator, with users providing non-membership proofs to demonstrate their non-revoked status. This approach offers enhanced scalability, remaining efficient even as the revocation list grows. As 5G networks scale to billions of devices, the ability to quickly prove a device is not on the revocation list becomes essential for seamless, secure authentication. This approach not only enhances security but also supports the ultra-reliable low-latency communication (URLLC) requirements of 5G.

Therefore, this thesis employs the universal accumulator proposed by Li, Li, and Xue [50], leveraging its specific properties for applications in network security and authentication. Our focus is primarily on the non-membership proofs, a feature crucial for the efficient management of revocation lists in 5G networks. We present a simplified implementation of this accumulator in Appendix A.2. This adaptation of the universal accumulator forms an integral component of our research, particularly in Chapters 4 and 5. It addresses key requirements for secure and efficient network authentication, especially in the context of revocation list management in 5G.

2.2.10 Puncturable Key Wrapping

Puncturable Key-Wrapping (PKW) is a cryptographic scheme that combines the functionalities of key wrapping with the concept of puncturable pseudorandom functions (PRFs), introduced by Backenda et al. [13]. In a PKW scheme, a master key can be used to securely wrap (encrypt) other keys. The provided 'puncturing' process enables fine-grained control over key usage, enhancing security in scenarios where key exposure is a concern. PKW's

unique properties make it particularly useful in forward-secure encryption schemes, secure messaging systems, and other applications requiring dynamic key management in a symmetric key setting.

Definition 19 (Puncturable Key Wrapping) *The PKW consists of a tuple of four algorithms PKW:*$\{\mathsf{KGen}, \mathsf{Wrap}, \mathsf{Unwrap}, \mathsf{Punc}\}$ *associated with four sets: the secret-key space* $\mathcal{SK}$*, the tag space* $\mathcal{T}$*, the header space* $\mathcal{H}$ *and the wrap-key space* $\mathcal{K}$*.*

- $sk \xleftarrow{\$} \mathsf{KGen}()$*: a probabilistic algorithm that takes no inputs and outputs a secret key* $sk \in \mathcal{SK}$*.*
- $C/\perp \leftarrow \mathsf{Wrap}(sk, T, H, K)$*: a deterministic wrapping algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a tag* $T \in \mathcal{T}$*, a header* $H \in \mathcal{H}$*, and a key* $K \in \mathcal{K}$ *and outputs either a ciphertext* $C \in \{0,1\}^*$ *or* $\perp$ *for failure.*
- $K/\perp \leftarrow \mathsf{Unwrap}(sk, T, H, C)$*: a deterministic unwrapping algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a tag* $T \in \mathcal{T}$*, a header* $H \in \mathcal{H}$*, and a ciphertext* $C \in \{0,1\}^*$ *and outputs either a key* $K \in \mathcal{K}$ *or* $\perp$ *for failure.*
- $sk' \leftarrow \mathsf{Punc}(sk, T)$*: a deterministic puncturing algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *and a tag* $T \in \mathcal{T}$ *and returns an updated secret key* $sk' \in \mathcal{SK}$*.*

Correctness of PKW scheme requires that a wrapped key can be retrieved from its wrapping ciphertext, except in one specific situation. This exception occurs when the secret key has been "punctured" (modified in a specific way) using the same tag that was employed during the wrapping process. It is important to note that the wrapped key should still be recoverable even if the secret key has been punctured using different tags. To express this idea in formal terms, we state that for all $T \in \mathcal{T}, H \in \mathcal{H}, K \in \mathcal{K}$, where $\overline{T}_1, \overline{T}_2 \in \mathcal{T}^*$ and $T \notin \overline{T}_1 and T \notin \overline{T}_1$,

$$\mathbf{Pr}[\mathsf{Unwrap}(sk_{\setminus\overline{T}_1}, T, H, \mathsf{Wrap}(sk_{\setminus\overline{T}_2}, T, H, K)) = K | sk \xleftarrow{\$} \mathsf{KGen}()] = 1$$

The notation $sk_{\setminus(T_1,T_2,...,T_n)} = \mathsf{Punc}(...(\mathsf{Punc}(\mathsf{Punc}(sk, T_1), T_2), ...), T_n)$ is a compact way to represent a secret key that results from sequentially puncturing sk on $T_1, ..., T_n \in \mathcal{T}$.

Definition 20 (PKW puncture invariance) *A puncturable key-wrapping scheme* $\mathsf{PKW} = (\mathsf{KGen}, \mathsf{Wrap}, \mathsf{Unwrap}, \mathsf{Punc})$ *is puncture invariant if for all keys* $sk \in \mathcal{SK}$ *and all tags* $T_0, T_1 \in \mathcal{T}$ *it holds that:*

$$\mathsf{Punc}(\mathsf{Punc}(sk, T_0), T_1) = \mathsf{Punc}(\mathsf{Punc}(sk, T_1), T_0)$$

Additionally, the authors [13] have provided a consistent definition for the PKW scheme, inspired by the definition of consistent puncturable signature schemes provided by Bellare et al. [20]. In a consistent PKW scheme, the output of the Wrap algorithm depends solely on the tag, header, and wrap-key input. The (puncturing) state of the secret key does not affect the output, except in cases where the output is $\perp$ due to puncturing. This property ensures a predictable behaviour of the wrapping process, regardless of the secret key's puncture history, unless the key has been punctured for the specific tag being used.

Definition 21 (PKW consistency) *A puncturable key wrapping scheme* $\mathsf{PKW} = (\mathsf{KGen}, \mathsf{Wrap}, \mathsf{Unwrap}, \mathsf{Punc})$ *is consistent if for all keys* $K \in \mathcal{K}$*, all headers* $H \in \mathcal{H}$*, all tags* $(T_1, \ldots, T_n) \in \mathcal{T}^*$ *and all* $T \in \mathcal{T} \setminus T_1, \ldots, T_n$ *it holds that:*

$$\Pr\left[\mathsf{Wrap}(sk, T, H, K) = \mathsf{Wrap}(sk_{\setminus(T_1,\ldots,T_n)}, T, H, K) \mid sk \xleftarrow{\$} \mathsf{KGen}()\right] = 1$$

Puncture invariance and consistency collectively ensure that a PKW scheme exhibits a form of indifference to the order of puncturing operations. These properties allow for flexible puncture and wrapping sequences reordering without impacting the scheme's subsequent behaviour.

2.2.10.1 Construction and Relevance to This Thesis

PKW scheme, introduced by Günther et al., can be constructed using standard cryptographic primitives, including a Puncturable Pseudorandom Function (PPRF) and AEAD. This approach allows for efficient key management and forward security in symmetric-key settings.

However, the direct implementation of PKW in mobile network protocols, particularly in 5G environments, presents challenges. The primary issue lies in the potential for key-reuse across different cryptographic operations within the same protocol. This reuse can lead to security vulnerabilities, as the compromise of one key could potentially affect multiple parts of the protocol [55].

To address these limitations, this thesis introduces a new variant called Puncturable Key Wrapping$^+$ (PKW^+). PKW^+ is designed specifically to overcome the challenges of applying PKW in 5G protocols. It introduces additional key separation mechanisms and a dedicated key derivation function, which help mitigate the risks associated with key-reuse. The details of PKW^+ and its specific applications in 5G security are discussed in depth in Chapter 6.

2.3 Security Framework

This section introduces formal definitions of the security objectives that robust authentication and key agreement (AKA) and Handover (HO) protocols aim to accomplish. To formalise the security properties of the proposed schemes in this thesis, we follow Bellare-Rogaway [19] key exchange models. These models essentially capture the security of a key exchange protocol as a game played between a probabilistic polynomial-time (PPT) adversary $\mathcal{A}$ and a challenger $\mathcal{C}$. The adversary wins the game if it either causes a winning event (i.e. breaking authentication (MA) or anonymity) or terminates and guesses a challenge bit b (i.e. breaking key indistinguishability). We utilise the Khan et al. framework [45] to capture notions of *user unlinkability* (Unlink), and eCK framework [49] to capture key indistinguishability (KIND).

The security experiment includes a Clean predicate (**clean**), which defines specific limitations on the adversary's actions. This predicate is protocol-specific and plays a crucial role in determining the adversary's capabilities and restrictions. The exact definition and implications of the Clean predicate will be elaborated in the relevant chapters where specific protocols are analysed.

2.3.1 Execution Environment

Here, we describe the shared execution environment of all security games. Our analysis uses three distinct games that assess different properties of a key exchange protocol: MA, KIND and Unlink. In our games, the challenger $\mathcal{C}$ maintains a single Core Network (CN), running a number of instances of the key exchange protocol Π, and a set of (up to) n_P users $\mathrm{UE}_1, \ldots, \mathrm{UE}_{n_P}$ (representing users communicating with the CN), and systems $\mathrm{gNB}_1, \ldots, \mathrm{gNB}_{n_P}$ (representing base station (gNBs/HgNBs) communicating with the CN), each potentially running up to n_S executions of protocol Π. We abuse notation and use π_i^s to refer to the s-th session owned by party i, and also as the state maintained by that session. We introduce the state maintained by each session:

- $id \in \{1, ..., n_P\}$: Index of the session owner.
- $\rho \in \{\mathrm{UE}, \mathrm{gNB}, \mathrm{CN}\}$: Role of the session.
- $s \in \{1, ..., n_S\}$: Index of the session.
- $sid \in \{\{0,1\}^*, \perp\}$: Session identifier, initialised as $\perp$.
- $pid \in \{1, ..., n_P, \perp\}$: Partner UE identifier ($\perp$ if $\rho = \mathrm{UE}$).
- $gid \in \{1, ..., n_P, \perp\}$: Partner gNB's identifier.
- $msg_s \in \{\{0,1\}^*, \perp\}$: Messages sent by the session.
- $msg_r \in \{\{0,1\}^*, \perp\}$: Messages received by the session.
- $k_i \in \{\{0,1\}^\lambda, \perp\}$: Long-term CN/UE symmetric key.
- $k \in \{\{0,1\}^\lambda, \perp\}$: Established session key.
- $\alpha \in \{\texttt{in-progress}, \texttt{accepted}, \perp\}$: Session status.
- $it \in \{\{0,1\}^*, \perp\}$: Secret internal state of the session.

After initialisation, $\mathcal{A}$ can interact with $\mathcal{C}$ via adversary queries. We capture a network adversary capable of injecting, modifying, dropping, delaying or deleting messages at will via **Send** queries. Our models allow $\mathcal{A}$ to initialise UE and gNB sessions owned by particular parties. Finally, $\mathcal{A}$ can leak the long-term secrets of sessions via **Corrupt** queries, session keys via **Reveal** and the internal state of sessions via **StateReveal** queries, as described below.

Adversary Queries. Here, we define queries that represent the behaviours of the adversary $\mathcal{A}$ during the execution of the experiments. Note that not all queries are available to the adversary in the same game:

- **Create**(i, s, ρ): allows $\mathcal{A}$ to initialize new UE and gNB sessions π_i^s such that $\pi_i^s.id = i$, $\pi_i^s.\rho = \rho$.
- **Send**$(m, i, s, \rho) \leftarrow m'$: allows $\mathcal{A}$ to send message m to a session π_i^s where $\pi_i^s.\rho = \rho$. π_i^s processes the message and potentially outputs a message m'.

- **CorruptLTK**$(i) \rightarrow k_i$: allows $\mathcal{A}$ to leak the shared long-term key k_i of UE_i.
- **CorruptASK**$(i, \rho) \rightarrow (sk)$: allows $\mathcal{A}$ to leak the long-term asymmetric keys of a party, where $\rho \in \{\text{CN}, \text{gNB}, \text{UE}\}$ (for instance, **CorruptASK**$(\text{CN}, 0)$ or **CorruptASK**(gNB, i)). $\mathcal{C}$ checks if $\mathcal{A}$ previously corrupted these secrets, returning $\perp$ if so, otherwise the $\mathcal{C}$ returns sk_i^{ρ}.
- **StateReveal**$(i, s, \rho) \rightarrow \pi_i^s$: allows $\mathcal{A}$ to reveal the internal state of π_i^s where $\pi_i^s.\rho = \rho$.
- **Reveal**$(i, s, \rho) \rightarrow k$: allows $\mathcal{A}$ to reveal the secret session key k computed during session π_i^s where $\pi_i^s.\rho = \rho$.
- **Test**$(i, s, \rho) \rightarrow k_b$ (**Only used in the** KIND **security experiment**): allows $\mathcal{A}$ to play the KIND security game. When $\mathcal{C}$ receives a **Test**(i, s, ρ) query, if **Test** has already been issued, $\pi_i^s.\alpha = \texttt{accepted}$, or π_i^s is not **clean**, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ sets $k_0 \leftarrow \pi_i^s.k$, and $k_1 \xleftarrow{\$} \{0,1\}^{\lambda}$, and returns k_b to $\mathcal{A}$ (where b was sampled by $\mathcal{C}$ at the beginning of the experiment).
- **Test**$(s, i, s', i') \rightarrow m$ (**Only used in the** Unlink **security experiment**): allows $\mathcal{A}$ to play the Unlink security game. When $\mathcal{C}$ receives a **Test**(s, i, s', i') query, initialises a new session π_b, where $(\pi_0 = \pi_i^s)$ or $(\pi_1 = \pi_{i'}^{s'})$, b was sampled by $\mathcal{C}$, and both π_i^s and $\pi_{i'}^{s'}$ are **clean**. **Test** query is only allowed to be issued by $\mathcal{A}$ if no session $\pi.\alpha \neq \texttt{in-progress}$ such that $\pi.id = i$. $\mathcal{C}$ will respond to any **Send**(m, i, s, UE) or **Send**(m, i', s', UE) queries with $\perp$ until $\pi_b.\alpha \neq \texttt{in-progress}$.
- **SendTest**$(m) \rightarrow (m')$ (**Only used in the** Unlink **security experiment**): allows $\mathcal{A}$ to send a message m to π_b after issuing **Test**. $\mathcal{C}$ returns a $\perp$ if $\pi_b.\alpha \neq \texttt{in-progress}$.

2.3.1.1 Matching Conversations

To capture what secrets the adversary is allowed to compromise without trivially breaking the security of our scheme, we need to define how sessions are *partnered*, and whether those sessions are *clean*. Partnering ensures that we can trace important sessions to other corruptions $\mathcal{A}$ has made, and cleanness predicates determine which secrets $\mathcal{A}$ were not allowed to compromise. Matching conversations are typically used in the BR model [18], and the eCK-PFS model relaxes this notion to *origin sessions*. However, these partnering methods inadequately address our setting, where the gNB essentially acts as a proxy, re-encrypting messages between the UE and the CN. Thus, two problems occur: we need to capture the messages that UE authenticates to the CN, and we also need to capture the fact that the gNB sends messages to two parties, neither of which exactly match gNB's transcript. Our solution is two-fold: we use *matching sessions (identifiers)* to capture the messages authenticated between the UE and the CN, and we introduce *matching subsets* to capture the subset of messages authenticated between the gNB and the CN and UE respectively.

Definition 22 (Matching Subset) *Let $S \subseteq T$ denote that all strings s in the set S are substrings of T. A session π_i^s has a matching subset with another session π_j^t, if $\pi_j^t.msg_r \neq \perp$, $\pi_i^s.\rho \neq \pi_j^t.\rho$, and if $\pi_j^t.\rho = \text{gNB}$ $\pi_i^s.msg_r \subseteq \pi_j^t.msg_s$ and $\pi_i^s.msg_s \subseteq \pi_j^t.msg_r$, and if $\pi_i^s.\rho = \text{gNB}$, $\pi_j^t.msg_r \subseteq \pi_i^s.msg_s$ and $\pi_j^t.msg_s \subseteq \pi_i^s.msg_r$.*

Next, we introduce the notion of *matching sessions*, where the session identifier *sid* of both sessions are either equal (or where one is a prefix string of the other).

Definition 23 (Matching Sessions) *Let $S \subset T$ denote that a string S is a (potentially equal) prefix of a string T. A session π_i^s is a matching session of π_j^t, if $\pi_j^t.sid \neq \perp \neq \pi_i^s.sid$, $\pi_i^s.\rho \neq \pi_j^t.\rho$ and $\pi_j^t.sid \subset \pi_i^s.sid$ or $\pi_i^s.sid \subset \pi_j^t.sid$.*

2.3.2 Mutual Authentication

Mutual Authentication (MA) in cryptographic protocols ensures both parties can verify each other's identity. The MA security model is a game where an adversary attempts to make a clean session accept without a genuine matching partner. Here we describe the overall goal of $\mathcal{A}$ in the MA security game. The experiment $\mathbf{Exp}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. At the beginning of the experiment, $\mathcal{C}$ generates long-term asymmetric keys for the CN and each user UE_i and each gNB gNB_i (where $i \in [n_P]$) and long-term symmetric keys for each user UE_i, and then interacts with $\mathcal{A}$ via protocol-specific queries. These queries are to be specified during the security analysis of the proposed schemes. $\mathcal{A}$ wins (and $\mathbf{Exp}^{\mathsf{MA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if the adversary has caused a **clean** session to accept (and set $\pi_i^s.\alpha \leftarrow \texttt{accepted}$) and there either exists no matching subset session π_j^t, or no matching session π.

In the mutual authentication game, $\mathcal{A}$'s goal is to cause a session π_i^s to accept without a matching session (i.e. no CN session that outputs the messages received by π_i^s) or matching subset (i.e. no gNB session has output those messages). We say that a protocol Π is MA-secure, if there exist no PPT algorithms $\mathcal{A}$ that can win the MA security game against a clean session with a non-negligible advantage. We formalise this notion below.

Definition 24 (Mutual Authentication Security) *Let Π be a key exchange protocol, and $n_P, n_S \in \mathbb{N}$. For a given cleanness predicate* **clean***, and a PPT algorithm $\mathcal{A}$, we define the advantage of $\mathcal{A}$ in the mutual authentication* MA *game to be:*

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = |\Pr[\mathbf{Exp}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}|.$$

We say that Π is MA*-secure if, for all PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is negligible in security parameter λ.*

2.3.3 Key Indistinguishability

Key Indistinguishability (KIND) in cryptographic protocols ensures that session keys are computationally indistinguishable from random keys. In this security game, an adversary interacts with the system and tries to distinguish between a real session key and a random key. The protocol is considered secure if no PPT adversary can make this distinction better than random guessing. Here, we describe the overall goal of $\mathcal{A}$ in the key indistinguishability KIND security game. The experiment $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. At the beginning of the experiment, $\mathcal{C}$ generates long-term symmetric keys for the CN and each user UE_i and each gNB gNB_i (where $i \in [n_P]$), samples a random bit $b \xleftarrow{\$} \{0,1\}$ and then interacts with $\mathcal{A}$ via protocol-specific queries. These queries are to be

specified during the security analysis of the proposed schemes. At some point, $\mathcal{A}$ issues a $\mathbf{Test}(i, s, \rho)$ query and either receives $\pi_i^s.k$ or a random key from the same distribution (based on b). $\mathcal{A}$ eventually terminates, outputs a guess b' and wins (and $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if $b' = b$.

We say that a protocol Π is KIND-secure, if there exist no PPT algorithms $\mathcal{A}$ that can win the KIND security game with non-negligible advantage, formalising this notion below.

Definition 25 (Key Indistinguishability) *Let Π be a key exchange protocol, and $n_P, n_S \in \mathbb{N}$. For a given cleanness predicate* **clean***, and a PPT algorithm $\mathcal{A}$, we define the advantage of $\mathcal{A}$ in the key indistinguishability* KIND *game to be:*

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = |\Pr[\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}|.$$

We say that Π is KIND*-secure if, for all PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is negligible in the parameter λ.*

2.3.4 Unlinkability

User Unlinkability (Unlink) in cryptographic protocols ensures that an attacker cannot determine whether different communication sessions belong to the same user. In this security game, an adversary interacts with the system and tries to link user sessions. The system is considered secure if no PPT adversary can distinguish between randomly chosen user sessions better than guessing. Here we describe the overall goal of $\mathcal{A}$ in the Unlink security game. The experiment $\mathbf{Exp}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger $\mathcal{C}$ and an adversary $\mathcal{A}$. At the beginning of the experiment, $\mathcal{C}$ generates long-term symmetric keys for the CN and each user UE_i (where $i \in [n_P]$), samples a random bit $b \xleftarrow{\$} \{0,1\}$ and then interacts with $\mathcal{A}$ via protocol-specific queries. These queries are to be specified during the security analysis of the proposed schemes. $\mathcal{A}$ wins (and $\mathbf{Exp}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if $\mathcal{A}$ terminates and outputs a guess bit b' such that $b' = b$.

We say that a protocol Π is Unlink-secure if there exist no PPT algorithms $\mathcal{A}$ that can win the Unlink security game with a non-negligible advantage, which we formalise below.

Definition 26 (Unlinkability) *Let Π be a key exchange protocol, and $n_P, n_S \in \mathbb{N}$. For a given cleanness predicate* **clean***, and a PPT algorithm $\mathcal{A}$, we define the advantage of $\mathcal{A}$ in the unlinkability* Unlink *game to be:*

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = |\Pr[\mathbf{Exp}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) = 1] - \frac{1}{2}|.$$

We say that Π is Unlink*-secure if, for all PPT $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is negligible in the parameter λ.*

Chapter 3

Background and Related Work

5G represents the latest generation of mobile networks, marking a significant leap forward compared to its predecessors. Unlike the incremental advancements seen in the first four generations, 5G cannot be viewed as a simple evolution of 4G. The requirements for 5G are significantly more stringent, featuring extensive enhancements in security and overall network architecture. As a heterogeneous network, 5G integrates a mix of existing and new access technologies, which include advancements in WLAN technologies, LTE, and the new 5G NR interface. In addition, 5G introduces a unified, programmable, software-centric infrastructure by incorporating cutting-edge network technologies, such as cloud computing, ultra-dense Small Cell Network (SCN), Software-Defined Networking (SDN), and Network Function Virtualization (NFV). This integration creates a cohesive, software-driven system and infrastructure. Although all enabling technologies used in 5G are crucial for meeting its stringent requirements, numerous security and privacy issues must be resolved before these technologies can be fully leveraged to maximize their benefits to the 5G mobile network.

This thesis focuses specifically on the security and privacy issues related to SCN technologies within 5G networks. 5G employs SCNs to optimize radio spectrum utilization by increasing the number of base stations (BS) and reducing their coverage areas. However, this approach introduces challenges. The proliferation of BSs in 5G significantly raises the frequency of the handover processes necessary to maintain user mobility between cells. This increased frequency can lead to higher latency, as secure links must be established for each transmission, which impacts the overall speed of the 5G network. While efforts to reduce latency in handovers are essential for 5G performance, they must be balanced with the need to maintain stringent security properties and avoid potential vulnerabilities. However, the perceived conflict between these goals is often underestimated. This oversight is especially problematic in specific scenarios, such as SCNs or when dealing with resource-constrained devices, such as smartphones. This requires careful consideration of factors that include computational efficiency, key management strategies, and privacy-preserving techniques. By addressing these aspects, it is possible to develop solutions that provide robust security guarantees without compromising the low latency requirements that are crucial for 5G performance.

Substantial research has been conducted in developing Authentication and Key Agreement (AKA) and Handover (HO) protocols for 5G, in order to analyse and identify security and privacy weaknesses. Peltonen, Sasse, and Basin [56] offer a comprehensive formal security

analysis of the conventional 5G-HO protocol. Similarly, Basin et al. [15] and Cremers and Dehnel-Wild [33] contribute thorough formal security analyses for the standard 5G-AKA protocol. These efforts reveal underspecified security requirements and unveil vulnerabilities in the current versions of 5G-AKA and HO protocols, such as linkability attacks, identity confusion attacks, and issues related to PFS and confidentiality attacks on sequence numbers [4].

The remainder of this chapter is organized as follows: Section 3.1 provides an overview of the evolution of cellular network generations from lG to 5G. Details about the 5G network architecture, security, and service requirements can be found in Section 3.2. Section 3.3 describes the conventional 5G-AKA and HO protocols standardized by the 3GPP group. Section 3.4 delves into the security and privacy vulnerabilities of the current 5G-AKA and HO protocols and proposes desirable security and privacy properties to address these vulnerabilities. Section 3.5 concludes by highlighting key security and privacy challenges, which identify research gaps in the domain of 5G security.

3.1 Evolution of Mobile Networks

In recent decades, mobile networks have undergone significant advancements, evolving from 1G to the initial stages of 6G. Corresponding developments in the security landscape have accompanied each progression in network generation.

The first generation (lG) of mobile networks, introduced in the 1980s, enabled voice calls over analogue signals within a single country but not internationally. According to [44], the primary systems were the Advanced Mobile Phone Service in the USA, the Total Access Communication System in Europe, and Nordic Mobile Telephone in Eastern Europe, Nordic countries, and Russia. Despite its functionality, lG was vulnerable to interception due to the lack of encryption, allowing easy eavesdropping and credential theft using simple frequency scanners.

Following the first generation of analogue communication signals, the second generation (2G) of mobile networks, introduced around 1991, brought digital communication, enabling data transmission such as SMS alongside voice services. The Global System for Mobile Communications (GSM) became the main standard for 2G networks, mainly using the 900 MHz and 1800 MHz frequency bands, although other bands were used in some areas. GSM was widely adopted because it used the radio spectrum efficiently and allowed for international roaming. Despite improvements, 2G faced security issues such as spamming and fake BS attacks. To address these issues, 2G implemented security measures such as one-way authentication, user credential protection, and SIM cards for user verification. To further enhance 2G capabilities, bridging technologies such as 2.5G (General Packet Radio Service) and 2.75G (Enhanced Data rates for GSM Evolution) were developed, providing improved data transmission speeds and better overall performance [53].

Building on previous generations, 3G, launched by NTT DoCoMo in 2001, enhanced services with data-driven applications over the Internet [53]. Initially offering low bandwidth (144 Kbps mobile, up to 2 Mbps fixed), 3G improved over time to support video calls, mobile internet, TV, and MMS. It addressed vulnerabilities like fake base stations and short keys by implementing mutual authentication (EAP-AKA), increasing key lengths, and using multiple encryption techniques [6]. Despite advancements, 3G still faced issues such as viruses,

Table 3.1: *Evolution of security landscapes [46]*

Network	Application	Security Mechanism	Security Challenges
1G	Voice	No explicit security and privacy measures	Physical, eavesdropping and call interception
2G	Voice and Text	One-way authentication, anonymity and encryption-based protection	Fake base stations, radio link security and spamming
3G	Voice and Internet	Adopted the 2G security, secure network access, introduced AKA protocol and two-way authentication	Virus, spyware, malicious applications, IP traffic security vulnerabilities, encryption keys and roaming security.
4G	Voice, video, and Internet	Trust mechanisms, encryption keys, access security, integrity protection and new encryption (EPS-AKA).	Increased IP traffic threats (DoS attacks), APT (Advanced Persistent Threats), data integrity.
5G	Voice, video, Internet, and massive IoT	5G-AKA, EAP-AKA', SUPI concealment, RAN security, improved key hierarchy.	User linkability attacks, sequence number deconcealment, fake base station attacks.

malicious apps, IP traffic security vulnerabilities, and encryption key management.

After 3G networks, there was a demand for more advanced and secure systems with higher data rates, leading to the introduction of 4G. This generation was characterized by an all-IP end-to-end architecture, using the IP higher-layer protocol as a transmission medium. The development of 4G not only focused on service enhancements but also on increasing security. To achieve superior security compared to previous generations, 4G implemented new cryptographic algorithms, such as the EPS Integrity Algorithm (EIA) and EPS Encryption Algorithm (EEA), and increased key lengths to 256 bits, up from the 128-bit keys used in 3G [42]. According to the 4G technical specification (TS 33.401), 3GPP is mandated using the EPS-AKA mechanism [6]. Table 3.1 summarizes the evolution of security across the five generations. The next sections will present a comprehensive discussion of the fifth generation.

3.2 5G Architecture

The fifth generation (5G) network has been designed to meet stringent requirements and support enabling technologies while also ensuring backward compatibility with previous generations. The key components of the 5G system include the User Equipment (UE), Next Generation Radio Access Network (NG-RAN), and 5G Core Network (5GC), as shown in Figure 3.1.

- **User Equipment (UE)** The UE comprises the Mobile Equipment (ME) and the Universal Subscriber Identity Module (USIM). This includes devices such as smartphones, tablets, Internet of Things (IoT) devices. These devices connect to the 5G network, facilitating various applications, from basic communication to advanced data services.
- **Next Generation Access Network (NG-RAN)** The NG-RAN represents the advanced radio access technology specific to 5G. Key components include :
  - gNodeB (gNB): The gNodeB, or next-generation Node B, is the base station that establishes the radio link between the UE and the 5GC. It is responsible for trans-

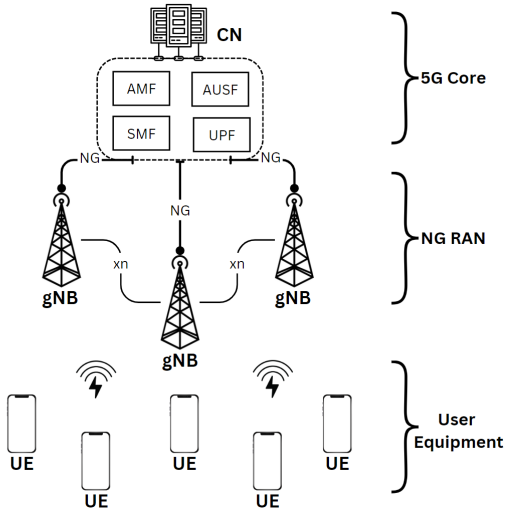

Figure 3.1: *Overview of the 5G system*

mitting and receiving signals to and from the UE and performing essential radio resource management tasks. The gNB can be further segmented into:

- Central Unit (CU): Manages non-real-time operations such as mobility management and higher-layer protocol processing.
- Distributed Unit (DU): Handles real-time operations such as media access control and radio link control.
- Radio Unit (RU): Performs the physical layer processing and radio signals' actual transmission and reception.

- **5G Core Network (5GC)** The 5GC is the backbone of the 5G architecture and is designed around a service-based architecture (SBA). This architecture allows network functions (NFs) to interact with each other through standardized, web-based protocols (such as HTTP/2 and REST APIs), enhancing flexibility, scalability, and efficiency. Key elements of the 5GC include:
  - Access and Mobility Management Function (AMF): Manages user authentication, registration, connection, and mobility.
  - Session Management Function (SMF): Oversees session establishment, modification, and termination, as well as IP address allocation and Quality of Service (QoS) enforcement.
  - User Plane Function (UPF): Routes and forwards user data packets between the UE and external data networks, ensuring efficient data transmission and traffic management.
  - Authentication Server Function (AUSF): Ensures secure access by authenticating users.
  - Unified Data Management (UDM): Manages subscriber data and profiles as a centralized database.

- Network Slice Selection Function (NSSF): Helps select the appropriate network slice to meet specific service requirements.
- Policy Control Function (PCF): Implements policies related to QoS, access control, and charging rules.
- Network Exposure Function (NEF): Provides APIs to expose network capabilities to third-party applications, fostering innovation and integration.
- Application Function (AF): Supports application-specific requirements and interacts with the core network to provide enhanced services.
- Security Anchor Function (SEAF): Acts as an intermediary for authentication processes between UEs and the core network, enhancing security during user access.
- Authentication credential Repository and Processing Function (ARPF): Stores authentication credentials and performs cryptographic computations for authentication procedures.
- Subscription Identifier De-concealing Function (SIDF): Decrypts concealed user identities to retrieve permanent identifiers, supporting user privacy and security.

By integrating these advanced components and adhering to a flexible architectural framework, the 5G network delivers unparalleled performance. It enables various innovative applications and services while maintaining seamless compatibility with legacy systems.

3.2.1 Security Architecture

The 5G security architecture, as delineated in 3GPP's Technical Specification version 16.3 published in August 2020 [4], presents a multi-layered approach to network security. As illustrated in Figure 3.2, this architecture is structured into three distinct and securely isolated strata. The Application stratum primarily involves service providers and encompasses services requiring robust security measures, such as mobile payments, often independent of network providers. The Home stratum demands stringent security protocols due to its critical role in managing the UDM, AUSF, and USIM, which store sensitive user information, including the SUPI and long-term cryptographic keys. The Serving stratum provides security services comparable to the Home stratum but focuses on core network functions, including the Network Repository Function (NRF), AMF, and NEF. The Transport stratum, while requiring lower security sensitivity compared to the aforementioned strata, is responsible for gNodeB functions, certain user functions, and select core functionalities. This layered security paradigm in 5G networks facilitates a granular implementation of security measures, tailoring protection levels to each layer's specific requirements and sensitivities within the 5G network architecture. The security architecture shown in Figure 3.2 also illustrates the security domains, which include:

- Network access security (I) consists of a set of security mechanisms that protect users accessing the network, regardless of the access type (3GPP or non-3GPP). This domain includes two key components: the Non-Access Stratum (NAS) and Access Stratum (AS). NAS, operating in the Home/Serving Stratum, secures mobility and session management between ME and SN. AS, functioning in the Transport Stratum within the 5G NR, ensures security at the radio interface level.

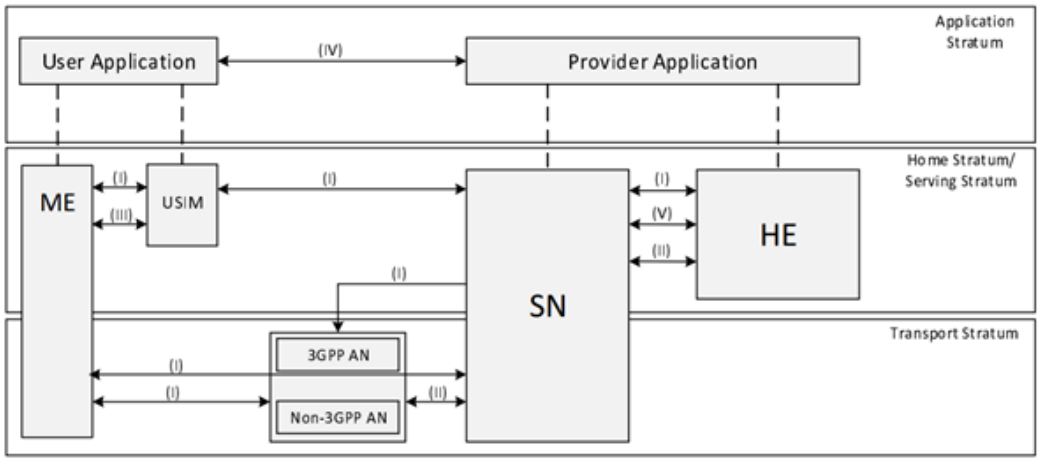

Figure 3.2: *Overview of 5G security architecture [4]*

- Network domain security (II) consists of security mechanisms providing secure communication between network nodes.
- User domain security (III) consists of a set of security mechanisms that provide secure access between users and their equipment.
- Application domain security (IV) consists of a set of security mechanisms providing secure communication between the user and provider domains.
- SBA domain security (V) consists of a set of security mechanisms that provide a secure communication channel between SBA functionalities within other domain networks, such as a serving domain network.
- Visibility and configurability of security (VI) consists of a set of mechanisms that inform users about the state of the security feature (i.e., whether it is running or not).

3.2.2 5G Service Requirements

To meet the growing demand for enhanced services, 5G aims to fulfil a set of stringent requirements identified by various companies and organizations. For instance, the ITU-R has outlined key performance metrics for 5G, including seamless mobility, high data rates, high availability, massive connectivity, and strong spectrum efficiency. Figure 3.3 illustrates these proposed 5G requirements, which can be viewed from both user and system/management perspectives. 5G targets a performance improvement of at least 10 times compared to 4G. For example, 4G specifies that latency should be under 10 ms, and 5G aims to reduce this to less than 1 ms. In addition, 5G networks are expected to support up to 1 million connected devices per km^2, a substantial increase from the 1,000 devices per km^2 supported by 4G. Figure 3.4 highlights some of the key differences between 4G and 5G requirements.

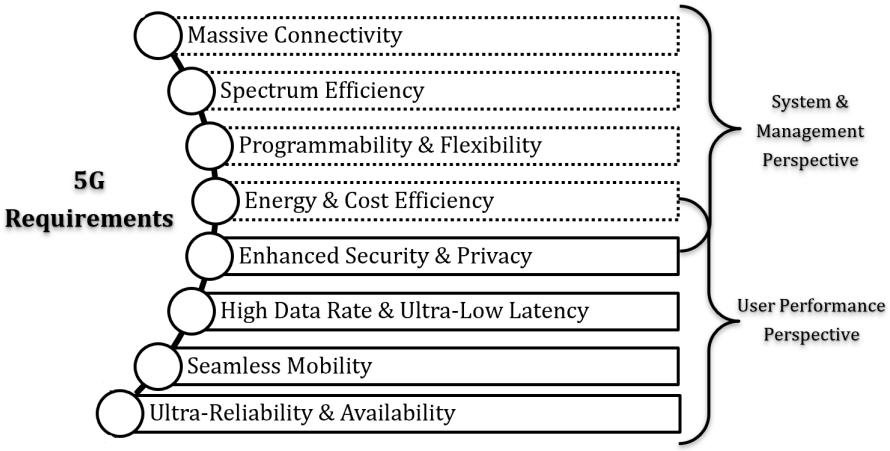

Figure 3.3: *5G requirements [51]*

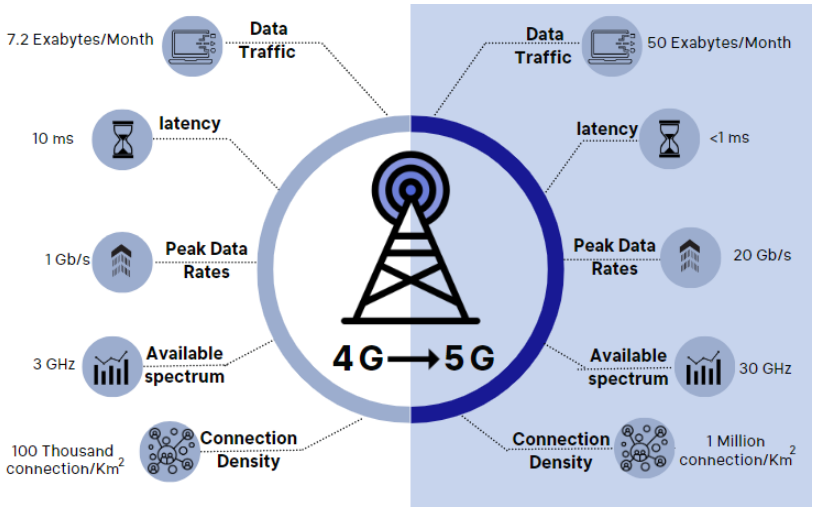

Figure 3.4: *Comparison of 4G and 5G requirements [1]*

3.3 5G Authentication and Handover Protocols

This section provides an overview of two fundamental security protocols of the 5G network: the 5G Authentication and Key Agreement (5G-AKA) protocol and the Handover (5G-HO) protocol. These protocols ensure secure, private, and seamless communication in 5G systems, balancing robust security with high-performance requirements.

The 5G protocols involve three major entities. The *User Equipment* (UE) is equipped with a USIM, SUPI, a long-term key (k), and a sequence number (SQN). The *Core Network* (CN) consists of various network services (SMF, AMF, AUSF, and UPF) and is responsible for maintaining UE information and handling the authentication process. The *base station* (gNB) provides network services for users through the assistance of the CN. When a UE joins the network and enters the coverage area of a gNB, mutual authentication occurs between the UE and gNB with the help of the CN. Communication between the UE and the gNB or CN occurs via insecure air channels during this process. In contrast, the CN and gNB communicate over authenticated channels, specifically the N2 interface, as outlined in TS 33.501 [4]. For abstraction, our description of the 5G protocols combines all core network functions into a single abstract entity, assuming secure internal communications within the 5GC. This approach allows us to focus on critical security aspects while eliminating the internal 5GC messages.

3.3.1 5G Authentication and Key Agreement

The 3GPP defines two main AKA protocols for 5G authentication: 5G-AKA [4] and the Extensible Authentication Protocol (EAP-AKA') [10]. Both protocols utilize the most recent version of the 3GPP for secure authentication and key agreement, sharing the same underlying 3GPP-AKA infrastructure and specifications with almost identical cryptographic constructions. Given these substantial similarities, this work focuses primarily on 5G-AKA for consistency.

The 5G-AKA utilises seven different symmetric key algorithms, denoted $(f_1, ..., f_5, f_1^*, f_5^*)$, and each with specific functionality. For example, 5G-AKA uses f_1, f_2 and f_1^* for message authentication, while f_3, f_4 and f_5 is used for key derivation. Table 3.2 presents how these cryptographic algorithms are used and the notations employed in 5G-AKA.

In 5G-AKA, the challenge's freshness combines an optional *resynchronization* phase, which is performed if the SQN gets out of sync between the UE and CN. The 5G-AKA protocol is illustrated in Figure 3.5.

3.3.1.1 Challenge-Response Phase

Step 1: This phase starts when a UE sends an authentication request using the SUCI, generated using ECIES, or the Globally Unique Temporary User Equipment Identity (GUTI) to CN.

Step 2: Upon receiving the authentication request, the CN de-conceals the SUCI and retrieves the SUPI. Then, the CN computes the expected response xRES* and the authentication challenge using the following: a random nonce R, authentication token (AUTN), challenge expected response (HxRES), and K_{SEAF}, and sends it to the gNB. The AUTN verifies the freshness of the challenge, where it combines the Message Authentication

Table 3.2: *5G-AKA notations.*

Notation	Description
R	Random Challenge (128 bit)
SQN	Sequence Number (48 bit)
gNB_{name}	gNB Name
K	Long-term key
MAC	$f_1(K, SQN \| R)$
AK	$f_5(K, R)$
RES	$f_2(K, R \| gNB_{name})$
AUTN	(CONC, MAC)
HxRES*/HRES*	$SHA_{256}(R \| xRES* / RES*)$
K_{SEAF}	KDF$(K, R \|$ gNB$_{name} \| SQN)$

Code (MAC), the nonce R, with the corresponding sequence number SQN of the UE. The HxRES* is only sent to the gNB, which is the hashed value of the actual RES*.

Step 3: Upon receiving the authentication challenge, the gNB stores the HxRES*, as well the K_{SEAF}, then forwards the challenge to the UE.

Step 4: Next, the UE verifies the authenticity and freshness of the challenge by:

1. Extracting the xSQN (the incremented value of SQN) and the MAC from AUTN.
2. Verifying the correctness of the MAC and responding with "Mac-failure" if errors occur.
3. Verifying the freshness of the SQN (i.e. ensuring SQN $\leq$ xSQN.)

If all verifications hold, the UE computes a challenge-response RES* and derives K_{SEAF} to secure the channel used with the gNB, then sends RES* to the gNB.

Step 5: Upon receiving the RES*, the gNB computes the hash value of RES* (HRES*) and compares it with the HxRES* received from the CN. If the HRES* equals HxRES*, the authentication will succeed. Subsequently, the gNB forwards RES to the CN containing the SUCI or SUPI and the gNB name.

Step 6: Upon receiving the RES* from the gNB, the CN checks if the RES equals xRES*, hence confirming the success of the authentication procedure, then notifying the gNB of the decision.

3.3.1.2 Resynchronization Phase

When a synchronization failure occurs (Sync. failure), the UE responds with a "Sync-failure, AUTS" message. This message enables the CN to re-synchronize with the UE by updating its SQN_{CN} to match the UE's SQN_{UE}, as described in [TS 33.102, Sec. 6.3.5,6.3.3]. To

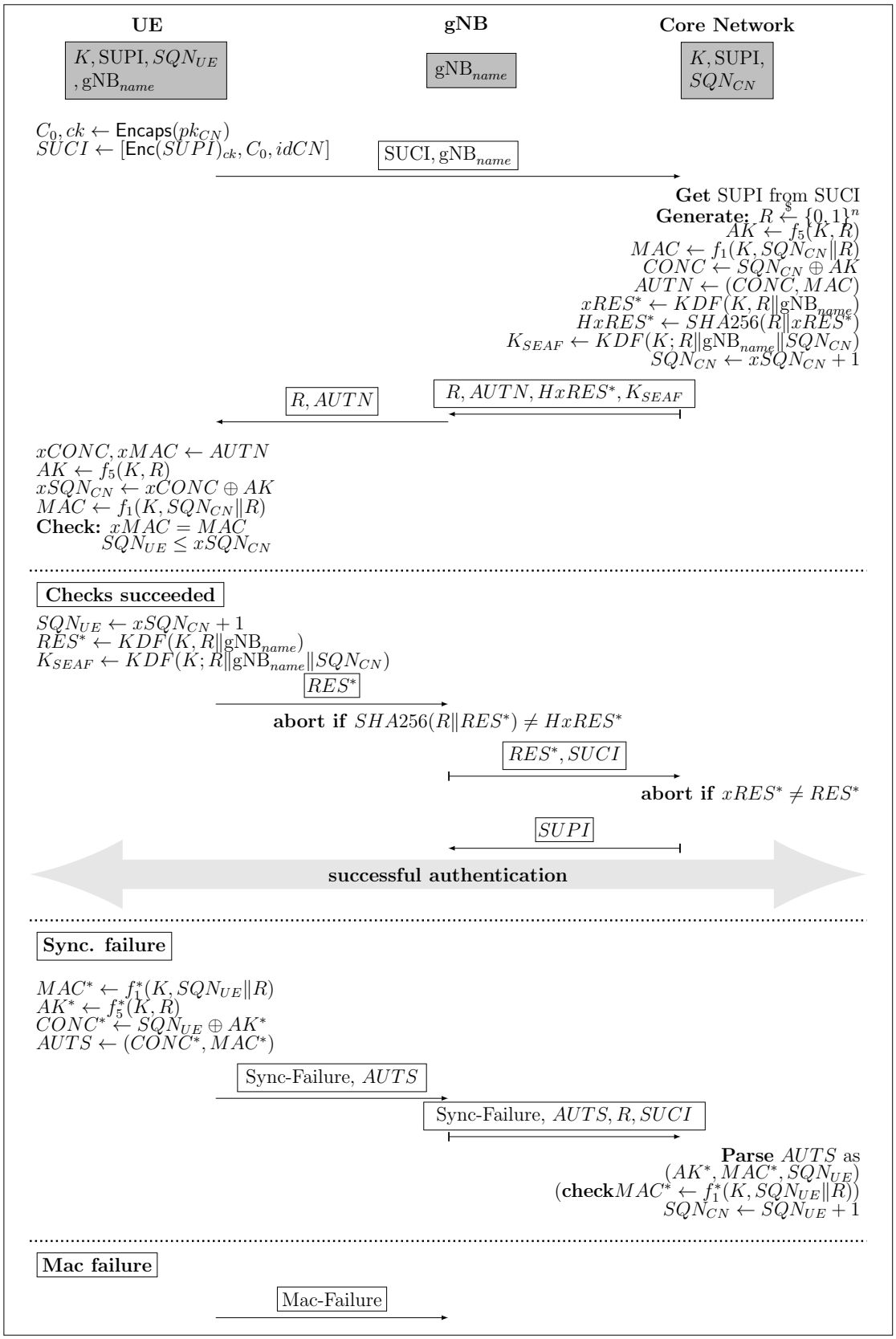

Figure 3.5: *5G-AKA protocol*

protect SQN_{UE} from eavesdropping, SQN_{UE} is concealed by XORing it with a private value: $CONC^* = SQN_{UE} \oplus AK^*$, where $AK^* = f5^*(K, R)$. The AUTS message is structured as $AUTS = CONC^*, MAC^*$, where $MAC^* = f1^*(K, SQN_{UE} \| R)$.

3.3.2 5G Handover

In cellular networks, handovers are crucial for maintaining seamless connectivity during ongoing calls and data sessions as users move between different network areas. This procedure involves transferring a mobile user's connection from one base station to another. This section focuses on intra-system HO within the 5G ecosystem, specifically examining the process of a device transitioning from a source gNB (SgNB) to a target gNB (TgNB), both of which adhere to the 5G standard. Intra-system handovers typically occur when a user moves between the coverage areas of two 5G-compliant gNBs. While inter-system handovers (transitions between gNBs where only one implements the 5G standard) also exist, our discussion will concentrate exclusively on intra-system handovers to align with our focus on 5G infrastructure.

During an intra-5G HO, the SgNB can initiate one of two protocol variations: the Xn-based HO or the N2-based HO. In an Xn-based HO, communication between the two gNBs occurs directly over the Xn interface, reducing the number of messages routed through the core network. Conversely, an N2-based HO lacks direct connectivity between the two gNBs. Instead, messages are relayed through the 5GC using the N2 interface, which connects the gNBs to the CN [56]. Although these protocols differ, they are both considered viable options in the 5G specification, with no preference indicated. We focus on the Xn-based HO, which optimizes the process by enabling direct communication between the SgNB and TgNB. This method minimizes the number of messages that need to be routed through the core network.

In the 5G handover process, four key network components play vital roles: the UE, the SgNB, the TgNB, and the CN. This procedure starts with using pre-established keying parameters obtained during the 5G-AKA protocol. These parameters encompass:

1. k_{SEAF}, a shared key between the UE, gNB and the CN, generated during 5G-AKA (see Figure 3.5);
2. k_{AMF}, derived from k_{SEAF}, where $k_{AMF} = \mathsf{KDF}(k_{SEAF}, SUPI)$;
3. k_{gNB}, a session key generated jointly by the UE and SgNB, where $k_{\text{gNB}} = (k_{AMF},$ NAS COUNT), with NASCOUNT) being a 32-bit security counter that increments with each NAS message to ensure freshness and prevent replay attacks; and
4. Next Hop (NH), an intermediary key, and its associated counter Next Hop Chaining Counter (NCC), derived collaboratively by the UE and SgNB.

As outlined in Figure 3.6, the 5G-HO protocol operates as described. (for a comprehensive understanding, please refer to [TS 23.502] and [TS 38.300].):

Step 1 ($\boldsymbol{m_1}$) : The handover process begins when the SgNB initiates a handover based on UE measurement reports. The SgNB sends a Handover Request message "HO Req" directly to the TgNB over the Xn interface, including the current NCC, k_{gNB}, TgNB_{ID}, CRNTI and PDUSessionID.

Step 2 (m_2) : Upon receiving this request, the TgNB performs admission control to check resource availability. If resources are available, the TgNB prepares to receive the UE and sends a Handover response message "HO Resp" back to the SgNB.

Step 3 (m_3) : Next, the SgNB sends an RRC Reconfiguration message encrypted using session key (k_{gNB}) to the UE, instructing it to connect to the TgNB.

Step 4 (m_4) : Upon reception of the message, the UE synchronizes with the TgNB and sends an RRC Reconfiguration Complete message encrypted using newly generated (k^*_{gNB}) to confirm the connection.

Step 5 (m_5) : Following this, the TgNB initiates the path switch procedure by sending a Handover configuration message to the CN. This request includes the NCC received from the SgNB.

Step 6 (m_6) : Upon receiving the message, the CN plays a crucial role in updating the security context. It computes the new $NH = KDF(K_{AMF}, NH)$ and increments the NCC by 1. The CN then sends a configuration response back to the TgNB, including the new NH key and the incremented NCC.

Step 7 (m_7) : The TgNB sends a UE Context Release message to the SgNB over the Xn interface, signalling the completion of the handover process. The SgNB then releases the resources associated with the UE.

Step 8 (m_8) : Finally, the CN informs the UE of a successful handover and registration.

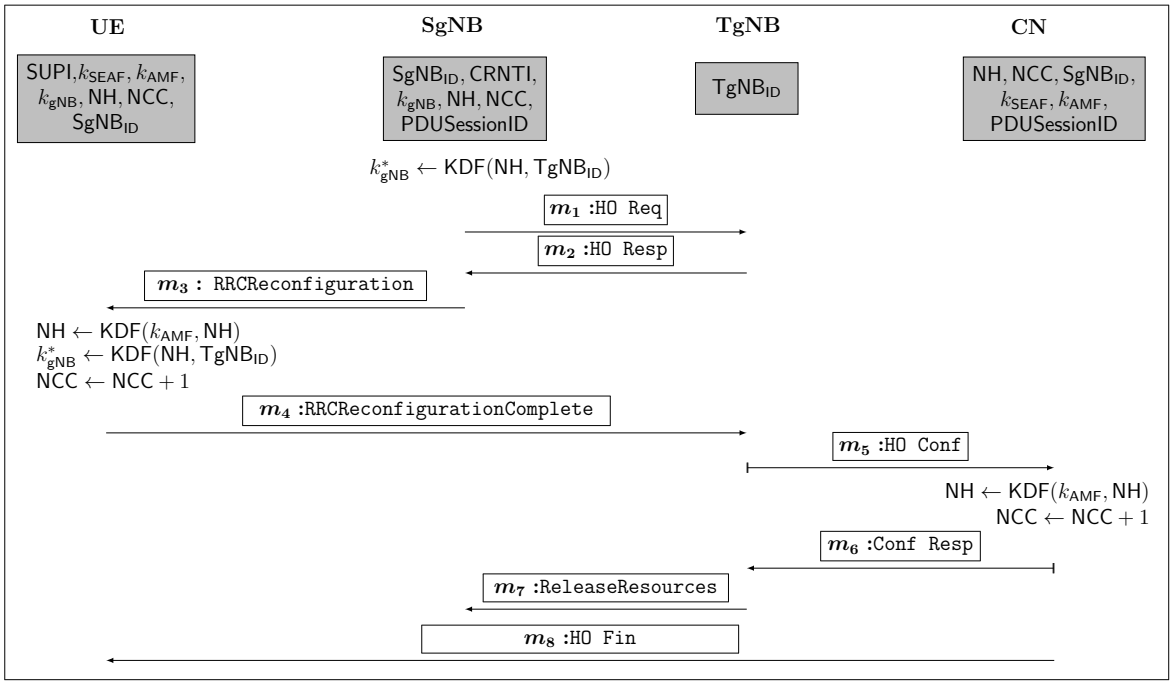

Figure 3.6: *5G*-HO *protocol.*

3.4 Security and Privacy Vulnerabilities in 5G-AKA and HO Protocols

Substantial research has been done on 5G authentication and handover protocols to analyse and identify security and privacy weaknesses in the conventional 5G protocols (i.e. 5G-AKA and 5G-HO). Notably, Basin et al. [15] and Cremers et al. [33] conducted comprehensive security evaluations of the 5G AKA protocol using the Tamarin verifier tool, identifying several critical vulnerabilities. Their analysis uncovered several critical vulnerabilities. For example, a key finding revealed that certain essential authentication properties are compromised before the key confirmation phase, a step not explicitly mandated by the 5G standard. This oversight results in insufficient authentication guarantees, particularly in scenarios lacking key confirmation. These studies also highlighted that several security objectives are only achievable under additional assumptions not explicitly specified in the standard. This reliance on implicit, undocumented assumptions crucial for maintaining security exposes the protocol to potential vulnerabilities. Furthermore, the researchers uncovered a significant privacy attack enabling active attackers to trace subscribers by eavesdropping on the radio access link. This attack facilitates the identification of whether different services are being accessed by the same subscriber, thereby breaching subscriber privacy. This discovery underscores several known security and privacy issues within the 5G-AKA protocol, such as the lack of Perfect Forward Secrecy (PFS) and vulnerability to linkability attacks, where an attacker can correlate multiple sessions to a single subscriber. Moreover, Singla et al. [62] have identified a critical security vulnerability in current 5G authentication processes, leading to fake base station attacks. Their research highlights the significant threat posed by fake base stations, which deceive UE into connecting, resulting in severe security issues such as eavesdropping, man-in-the-middle (MITM), and denial-of-service (DoS) attacks. These fake base stations exploit weaknesses in the network's authentication and bootstrapping mechanisms, tricking UEs into establishing connections and enabling attackers to intercept and potentially alter communications. To counter these threats, the authors propose enhanced mutual authentication based on a hierarchical identity-based signature scheme (Schnorr-HIBS). This approach aims to enhance verification of network credentials and secure bootstrapping procedures, aiming to improve the security and resilience of 5G networks against fake base station attacks.

Moreover, Peltonen et al. [56] conducted a thorough formal security analysis of the conventional 5G-HO scheme utilizing the Tamarin prover. Their analysis reveals that PFS is not consistently maintained, posing a significant risk to overall security and user privacy. The increased frequency of handovers compounds this issue due to the smaller cell sizes and coverage areas inherent to the SCN technology in 5G. Furthermore, they identify that 5G-HO key management is susceptible to desynchronization attacks. Suppose an attacker manages to compromise a legitimate gNB, which is plausible given current 5G network vulnerabilities. In that case, they can manipulate the NCC value, causing desynchronization of the gNB and disrupting the key agreement process, thereby further undermining network security.

3.4.1 Desirable Security and Privacy Requirements

Critical security and privacy vulnerabilities have been identified in 5G-AKA and HO protocols. These include compromised authentication guarantees, susceptibility to fake base station attacks, and the lack of PFS. It is essential to establish robust and comprehensive

security requirements to overcome these vulnerabilities. To address these risks and enhance the resilience of 5G protocols, we have identified a set of key security and privacy properties. These properties are designed to address weaknesses identified in the current 5G protocols, ensuring stronger authentication, privacy protection, and overall network security. By adhering to these requirements, we aim to offer solutions that align with the stringent requirements of 5G while overcoming the limitations of existing protocols.

- ***Mutual Authentication (MA):*** MA is the process by which both the network components and the UE verify each other identities before establishing a secure communication channel. This bidirectional verification process is essential for preventing unauthorized access, guarding against impersonation attacks, and mitigating MitM threats.
- ***User Anonymity (UA):*** Privacy is a critical security requirement in 5G networks, necessitating user identities never being transmitted in plaintext. However, encryption alone is insufficient for preserving user privacy, as honest but curious network components can still expose user identities. To address this, schemes should replace users' long-term identities with encrypted temporary identifiers, ensuring user anonymity is maintained throughout communication.
- ***Perfect Forward Secrecy (PFS):*** PFS plays a critical role in 5G by safeguarding the integrity of session keys previously generated, even in the event of long-term secret compromises. This security feature is especially significant in 5G networks, given the frequent handovers resulting from the increased deployment of cells. With numerous session keys generated during each handover, it becomes imperative for 5G protocols to ensure that any compromise does not jeopardize the security of past session keys.
- ***User Unlinkability*** (Unlink)***:*** Unlink is a crucial privacy feature in 5G networks that extends the concept of basic user anonymity. It ensures that a user's actions or transactions within the network cannot be traced back to the same user across different interactions or sessions. In other words, even if an adversary observes multiple transactions or activities conducted by a user, he/she cannot establish a connection or link between these activities to identify the user behind them.
- ***Perfect Forward Privacy (PFP):*** PFP extends the concept of PFS to the realm of user privacy. While PFS focuses on protecting past session keys, PFP aims to maintain user anonymity and unlinkability even if long-term secret keys are compromised in the future. By doing so, even when the adversary compromises the long-term secret key, he/she will be unable to break the anonymity or link users' temporary identities, which the current 5G-AKA protocol is vulnerable to.
- ***Key Escrow-Free (KEF):*** The key escrow problem, whereby a trusted entity maintains copies of all users' secret keys, has been widely debated among researchers. Although key escrow can be essential in some domains, it raises significant concerns regarding the trustworthiness of the third party controlling the escrow. In today's information security landscape, it is challenging to trust a single entity fully, as it may exploit this trust to reveal encrypted information within the network. In addition, if the escrow entity is compromised, it creates a single point of failure, jeopardizing the entire network. Furthermore, the concept of key escrow contradicts the principles of

end-to-end encryption and the stringent security requirements of 5G networks. Therefore, entrusting a single party with the authority to generate or maintain users' secret keys is not recommended. Providing a KEF property enhances security and protects users in the event of a compromised third party.

- ***Key Compromise Impersonation resilience (KCI):*** KCI resilience refers to a security property that prevents attackers from using a compromised key to impersonate other entities within the network to the compromised party. In a KCI attack, the adversary exploits the compromised key to deceive the user into believing they are securely communicating with a legitimate party, thereby enabling the attacker to manipulate future communications. Ensuring KCI resistance in 5G networks prevents attackers from leveraging compromised keys to undermine network integrity. This protection ensures that the security and authenticity of the communicating parties remain intact, even if a long-term secret key is compromised, thereby safeguarding against impersonation and maintaining trust within the network.

- ***Secure Revocation Management (SRM):*** Given the rapidly growing number of users in the 5G network, implementing an effective subscription/revocation management mechanism is essential. SRM is a critical process that efficiently revokes access privileges for compromised or unauthorized users, enhancing security by isolating threats. Additionally, SRM optimizes network performance by allocating resources exclusively to valid users. This capability is crucial for addressing evolving security challenges while maintaining the scalability and reliability of 5G services as the user base expands.

3.4.2 Proposed Enhancements for 5G AKA and HO Protocols

In response to the previously identified security and privacy vulnerabilities and the underspecified requirements in the current 5G-AKA and HO protocols [4], many researchers have proposed solutions to address these vulnerabilities. Among the proposed solutions are ReHand [37], RUSH [71], LSHA [69], CPPHA [31], [26] and more. These solutions aim to enhance the security and privacy of AKA and HO protocols and can be categorized into two groups based on their underlying cryptographic foundations: *symmetric-based* and *asymmetric-based* schemes. Table 3.3 provides a comprehensive comparison of these proposed solutions according to the security and privacy requirements discussed earlier in Section 3.4.1.

Symmetric-based solutions The endeavour to improve the 5G-AKA protocol has led to developing several symmetric-key-based solutions, aiming to address privacy concerns while maintaining compatibility and efficiency with the existing infrastructure. A notable contribution in this domain is the AKA' protocol proposed by Wang et al. [67]. This solution stands out for addressing the linkability attack while requiring only minor modifications to the current 5G-AKA protocol. This minimal alteration ensures high compatibility with existing 5G systems, making AKA' an attractive option for practical implementation. However, despite its compatibility, AKA' falls short in achieving all security requirements discussed in 3.4.1, such as PFP, PFS and KCI. Similarly, Braeken [26] introduced a symmetric-based AKA protocol designed to enhance the existing 5G-AKA framework and overcome identity replay attacks discovered in 5G-AKA. However, similar to AKA', Braeken's protocol also fails to

meet PFP and PFS and is vulnerable to linkability attacks. While it supports in-session unlinkability in case of successful authentication, user identities can be linked if authentication fails due to the reused GUTI [4].

In the realm of 5G-HO protocols, Fan et al. [37] introduced ReHand (Region-based Handover), a novel scheme designed to address security and privacy concerns in SCNs. ReHand aims to provide user anonymity and fast revocation while maintaining secure handover procedures. A key feature of this protocol is its support for expedited authentication. However, this feature applies only to users roaming within a single region (denoted by ‡ in Table 3.3). Specifically, it facilitates authentication between Home gNodeBs (HgNBs) that are under the same gNodeB (gNB). This process utilizes a shared secret group key. ReHand requires re-executing the initial authentication phase for inter-region handovers, whereby users transmit pseudo-IDs to the Authentication Center (AuC), which then updates and returns these identifiers. However, a significant weakness lies in its susceptibility to undetected desynchronization in the pseudo-ID update process, potentially compromising user privacy and authentication integrity. Moreover, although ReHand incorporates a membership revocation mechanism based on Nyberg's one-way accumulator [54], this choice introduces its own set of limitations. While functional, Nyberg's accumulator exhibits lower efficiency than alternative accumulator designs. Furthermore, its static nature necessitates that the AuC regenerate and redistribute the entire revocation list to all regions whenever a user is added or removed, a process that significantly impacts the overall efficiency and scalability of the protocol.

Yan and Ma [69] propose LSHA, a symmetric-based handover and authentication protocol that exploits neighbouring base handover (denoted by ‡) in the 5G network. LSHA's security framework revolves around each gNB possessing a unique secret key and establishing session keys with its neighbouring gNB, all orchestrated by the AMF. This key infrastructure forms the foundation for securing handover procedures. Although LSHA demonstrates robust defences against DoS and desynchronization attacks, it falls short in providing PFP, unlinkability and PFS. The protocol only supports partial (denoted by ◎) PFS during handovers, a limitation stemming from its reliance on the 5G-AKA protocol specified in the 3GPP standard. It also lacks a comprehensive discussion or formal proof of how user anonymity is achieved and maintained throughout the protocol's operations (denoted by N/P).

Asymmetric-based solutions In contrast to symmetric-key approaches, researchers

Table 3.3: *Comparison of features in state-of-the-art protocols*

Type	Scheme	MA	UA	PFS	PFP	Unlink	KCI	KEF	SRM	UHO	PM
Symmetric	**5G** - [4]	✓	✓	✗	✗	✗	✗	✗	✗	✗	CF
	AKA' - [67]	✓	✓	✗	✗	✓	✗	✗	✗	✗	Tamarin
	ReHand - [37]	✓	✓	✗	✗	✓	✗	✗	✓	‡	CF
	LSHA - [69]	✓	N/P	◎	✗	✗	✗	✗	✗	‡	BAN+Scyther
	Protocol of- [26]	✓	✓	✗	✗	✗	✗	✗	✗	✗	RUBIN
Asymmetric	**AKA$^+$** - [47]	✓	✓	✗	✗	✓	✗	✗	✗	✗	Bana-Comon
	AAKA - [70]	✓	✓	✗	✗	✓	✗	✓	✗	✗	CF
	RUSH - [71]	✓	✓	◎	✗	✗	✗	✓	✗	✓	BAN+AVISPA
	CPPHA - [31]	✓	✓	✗	✗	✓	✗	✗	✗	✓	BAN+Scyther

MA:Mutual Authentication, **UA**:User Anonymity, **PFS**: Perfect Forward Secrecy, **PFP**: Perfect Forward Privacy, **Unlink**: Unlinkability, **KCI**: Key compromise impersonation resilience, **KEF**: Key-escrow Free , **SRM**: Secure Revocation Management, **UHO**: Universal HO, ◎: Partial, **N/P**: no information provided ‡: Neighbour/Region-based HO, **PM**: Proof method, **CF**: Computational Formal security analysis

have also explored asymmetric-key solutions, leveraging public-key cryptography to address 5G-AKA and handover security challenges. Acknowledging the paramount importance of security and user privacy within the 5G landscape, research has focused on tracking users' digital footprints within cellular networks. Notable contributions in this field include the AAKA protocol proposed by Yu et al. [70], which employs advanced cryptographic primitives such as the DDH, zero-knowledge proofs, Boneh-Boyen signatures, Keyed-Verification Anonymous Credentials, and ElGamal encryption. Similarly, Koutsos [47] proposes the AKA$^+$ solution, which enhances the privacy features of 5G-AKA while adhering to its design and efficiency constraints, effectively mitigating IMSI catcher attacks and achieving user unlinkability. Despite the advancements in authentication privacy, both AAKA and AKA$^+$ fail to provide PFS or PFP, and their scope remains limited to authentication, leaving security and privacy during handover operations unaddressed.

Recognizing this gap, several researchers have focused on enhancing privacy and security during handover processes in 5G networks. For instance, Zhang et al. [71] propose RUSH, a universal handover authentication protocol that preserves user anonymity and ensures KEF. However, it is important to note that RUSH achieves only partial PFS (denoted by ◎) due to its dependency on the conventional 5G-AKA protocol, which is known to fail in providing PFS. The RUSH protocol provides universal handover in 5G HetNets through blockchains and utilises chameleon hashes to achieve user anonymity. However, their analysis omits the implications of blockchain use, potentially overlooking related security and privacy issues. Additionally, RUSH does not address the linkability issue inherent in chameleon hashes, which, despite allowing collision computation, retain a consistent structure that could be exploited to link transactions or communications. Consequently, we argue that RUSH falls short of achieving unlinkability.

Similarly, Cao et al. [31] propose CPPHA, an asymmetric-based scheme utilizing SDN in 5G. CPPHA's authentication handover module (AHM) predicts user movements and predistributes user information to anticipated gNBs. Although this approach aims for efficient handovers, it raises significant privacy concerns. Despite claims of anonymity and unlinkability, CPPHA's privacy protections are weakened by the widespread distribution of user identities across network entities. This method only offers weak anonymity against external eavesdroppers and introduces a critical vulnerability: if a single base station is compromised, the privacy of all users could be jeopardized.

Security analysis method While comparing the security and privacy properties of related works is crucial, examining the methods these works employ to substantiate their claims is equally important. Table 3.3 illustrate security analysis techniques utilized in the related works. The diversity of methods observed, ranging from computational formal proofs to symbolic automated tools, reflects the multifaceted nature of cryptographic protocol analysis. By discussing these approaches, we can gain insight into the strengths and limitations of each method in demonstrating specific security properties, setting the stage for our own methodological choices in subsequent chapters.

Computational Formal Security Analysis, which is used in the conventional 5G protocol, ReHand, and AAKA, offers unparalleled strength in security assurances. This method provides guarantees rooted in well-established computational hardness assumptions, effectively mirroring real-world attack scenarios. Its power lies in its ability to uncover subtle vulnerabilities that other methods might overlook while delivering the concrete security bounds

essential for practical implementation decisions. Importantly, computational analysis can capture complex attack scenarios such as side-channel attacks, timing attacks, and differential cryptanalysis, often beyond the scope of simpler models.

In contrast, Symbolic Automated Analysis Tools (Tamarin [52], Scyther [34], AVISPA [11]), used by AKA' [67], LSHA [69], RUSH [71], and CPPHA [31], offer advantages in automation and efficiency. However, their reliance on abstract models can lead to oversimplification, potentially missing critical attacks. For instance, symbolic models often assume perfect cryptography, which may overlook vulnerabilities arising from specific implementations of cryptographic primitives. They may also fail to capture attacks that exploit the algebraic properties of certain operations, such as XOR, modular exponentiation or key/ciphertext length hiding, which are crucial in securing many cryptographic protocols. In addition, symbolic models struggle to represent the probabilistic aspects of security, potentially missing probabilistic attacks or failing to provide accurate security bounds.

Similarly, BAN-logic [64] and its derivatives, such as the RUBIN logic used in Braeken's protocol [26], although useful for reasoning about authentication, fall short in comprehensively capturing the full spectrum of security properties crucial in modern cryptographic protocols. Many critiques of BAN logic highlight its incompleteness and poor semantics, making it inadequate for detecting certain obvious security issues [5]. These limitations are particularly problematic when addressing advanced security properties such as PFS and resistance to KCI, which are crucial in the 5G environment.

The Bana-Comon indistinguishability [14] used in AKA$^+$ [47], although innovative in bridging symbolic and computational approaches, lacks the established track record and comprehensive coverage of traditional computational methods. While promising, it may not yet fully capture the nuanced security requirements of complex 5G protocols, particularly in scenarios involving advanced threat models or novel cryptographic constructions.

Given the critical nature of 5G security and the paramount importance of robust, real-world applicable security guarantees, this thesis employs computational formal pen-and-paper security analysis throughout. This approach provides the most fine-grained, rigorous, and practically relevant security assessments, ensuring that our protocols can withstand the sophisticated threats faced by modern network infrastructures.

3.5 Discussion

This chapter provided an overview of the 5G architecture, focusing on vulnerabilities in two key protocols: AKA and HO. Research by Peltonen et al. [56] and Basin et al. [15] has highlighted critical security gaps such as linkability attacks and the absence of PFS. Building on this, we identified key security and privacy properties necessary to meet 5G's requirements and overcome some of the vulnerabilities in the current 5G protocols.

Our review of related research led to a comparative analysis, presented in Table 3.3, which evaluates state-of-the-art protocols alongside current 5G protocols based on the security and privacy properties discussed in Section 3.4.1. This comparison reveals that current protocols face challenges in addressing all properties comprehensively, particularly in ensuring secure inter-region handovers in SCNs. Notable gaps exist in PFS, PFP, KCI resilience, and support for universal handover. These identified gaps and challenges motivate the following research questions:

RQ$_1$ 5G currently supports only intra-region handovers and lacks secure inter-region scenarios. This limitation necessitates a new authentication and handover scheme. The ideal solution would achieve desired security properties while supporting seamless user mobility both within and between regions. Such a scheme is crucial for the effective deployment of SCNs in 5G environments. This leads us to our primary research question: *Is it possible to design an authenticated key exchange and handover scheme that ensures security and preserves user privacy, all while supporting a* ***region-based*** *handover with seamless connectivity for roaming users?* This research question will be addressed in **Chapter 4**.

RQ$_2$ Another significant challenge facing 5G-AKA and HO protocols is the lack of KCI resilience and KEF properties. This deficiency presents a critical security vulnerability within the 5G network, particularly in scenarios involving active attackers who can impersonate an honest party to a compromised entity. Incorporating KCI resilience and KEF properties is crucial for enhancing security, ensuring fairness in key generation, and mitigating the risk of protocol failure due to a single key compromise. This brings us to our second research question: *Is it possible to design an authenticated key exchange and handover scheme to safeguard user privacy, ensure security, and incorporate a Universal handover mechanism with* ***KCI resilience*** *and* ***KEF*** *properties?* This question will be thoroughly examined in **Chapter 5**.

RQ$_3$ Finally, while securing and preserving privacy in AKA and HO protocols within 5G and mobile communication systems is essential, it is crucial that any advancements over the current protocols supported by the 3GPP group align with the primitives used within the existing infrastructure. Given that 5G predominantly operates in a symmetric-based environment, solutions based on symmetric cryptography are preferred over asymmetric ones. This preference facilitates a smoother transition to the enhanced proposed version without disrupting the user experience. This consideration leads us to our third research question: *Is it possible to design a* symmetric-key-based *authenticated key exchange and handover scheme that aligns with existing 5G infrastructure while providing* ***PFS, PFP, and unlinkability****, and maintaining computational efficiency for resource-constrained user equipment?* This question will be addressed in **Chapter 6**.

Chapter 4

Privacy-Aware Secure Region-based Handover for 5G

The 5G mobile communication network provides seamless connectivity between users and service providers, aiming to meet several stringent requirements, such as seamless mobility and massive connectivity. Despite the numerous benefits 5G offers, it also brings about significant security and privacy concerns. For instance, integrating small cell networks (SCN) into 5G enhances the quality of services (QoS) by bringing the network closer to connected users. However, this also significantly increases the number of handover procedures (HO), which can impact the security, latency, and efficiency of the network. Therefore, it is crucial to design a scheme that supports seamless handovers through a secure authentication process. In the realm of 5G networks, region-based handover schemes have emerged as a promising solution to address these challenges. However, research in this area remains limited, with only one existing study proposing a region-based handover scheme [37], which focuses solely on intra-region handovers. This gap in the literature underscores the need for more comprehensive approaches that address both intra and inter-region scenarios. Region-based Handovers play a crucial role in maintaining session continuity, ensuring uninterrupted service for users as they move within and between regions. Additionally, region-based handovers substantially reduce the burden on the core network by eliminating the need for new authentication procedures for the same customer during movements, thereby optimizing resource utilization. These benefits contribute significantly to the overall efficiency and performance of 5G networks, particularly in densely deployed small-cell environments.

In this chapter, we propose a secure region-based handover scheme that ensures seamless connectivity for SCNs in 5G. Our scheme uniquely leverages asymmetric-key-based authenticated key exchange and handover protocols to maintain user privacy and network security while providing an effective region-based handover mechanism and efficient membership revocation management. We introduce three privacy-preserving protocols to address different communication scenarios: an initial authentication protocol, an intra-region handover protocol, and an inter-region handover protocol. This comprehensive approach sets our work apart by addressing both intra and inter-region handovers securely and efficiently.

Our proposed scheme employs two primary cryptographic schemes: sanitizable signatures (SanSig) [12] and universal dynamic accumulators [50]. Both schemes are detailed in Chapter 2, in Sections 2.2.8 and 2.2.9, respectively. SanSig enables partial message modification with-

out invalidating the signature, contributing significantly to user privacy during handovers. Universal dynamic accumulators, on the other hand, facilitate efficient user revocation management and scalable membership verification. These primitives play crucial roles in our protocol's design, supporting our objectives of enhanced privacy and efficient user management in 5G SCNs. Section 4.2 elaborates on the specific application of these primitives within our protocol and how they contribute to our comprehensive security and privacy framework.

Motivation and Contributions

Despite extensive research in this domain, existing protocols have not fully achieved all 5G requirements for a fast, secure, privacy-preserving, and reliable HO authentication scheme. A significant gap in the literature is the lack of secure inter-region HO scenarios in 5G networks. Inter-region handover in 5G is essential for maintaining seamless connectivity as users move between different cells. It ensures uninterrupted service for applications such as video calls, online gaming, and real-time data streaming, where continuity is critical. In addition, inter-region handover plays a role in optimizing network resource utilization and supporting seamless roaming between different geographic areas, contributing to the overall quality of experience for 5G users.

Therefore, there is a need for an authentication and HO scheme that achieves the desired security properties in 5G networks, as explained in Section 3.4.1, and achieves seamless user mobility in and between regions to facilitate SCNs in 5G. Additionally, managing users in the network is an essential feature for any handover scheme due to the continuous change in the number of users. While user revocation is crucial, only Rehand [37] provided revocation management using Nyberg's static accumulator. However, dynamic accumulators offer improved scalability and flexibility compared to static accumulators, particularly in the rapidly changing environment of 5G networks.

This chapter proposes a region-based HO scheme for SCN. This is the first to achieve secure, privacy-preserving inter-region HO for roaming users with effective revocation management in 5G without any additional infrastructural support such as blockchain. The major contributions of this chapter can be summarized as follows:

- A concrete solution for SCN roaming environments in 5G that provides a secure HO scheme supporting seamless user mobility in and between regions. To the best of our knowledge, this is the *first* solution to achieve secure, privacy-preserving inter-region 5G HO for roaming users;
- An effective user membership revocation scheme for many users in 5G using dynamic universal accumulators [50]. We specifically use a dynamic non-membership accumulator, dynamic to accommodate for joining/revoking users, and non-membership to leverage the number of legitimate joining users exceeding the number of revoked users;
- A rigorous formal security analysis of our proposed scheme shows that our scheme achieves mutual authentication, user unlinkability and secure key exchange.
- A comparative study of the proposed scheme with closely related existing schemes shows that our scheme is secure and computationally efficient.

These contributions aim to address important security and efficiency concerns in 5G handovers, with a particular focus on inter-region scenarios. The proposed approach offers potential improvements in user management, security and privacy in 5G environment.

Chapter Organisation

The remainder of the chapter is organised as follows:

- Section 4.1 introduces the system and threat models, providing the foundation for understanding the security context of our work.
- Section 4.2 presents the proposed secure inter-region HO authentication scheme, with a detailed description of the proposed protocols, including registration, initial authentication, and intra-region and inter-region HO protocols. This section forms the core of our contribution.
- Section 4.3 provides a formal security analysis of our proposed protocols, demonstrating the robustness of our approach.
- Section 4.4 presents security and performance evaluation and comparison of the proposed scheme with other related schemes, highlighting the security and efficiency of our solution.
- Finally, Section 4.5 concludes the chapter, summarizing key findings and suggesting future research directions.

We provide the notation used in this chapter in Table 4.1.

4.1 System and Threat Model

This section provides the foundational framework of our proposed scheme. We begin by detailing the system model, which outlines the key components, their interactions, and the overall architecture of our proposed solution. Subsequently, we present the threat model, which delineates the potential threats and attack vectors that our scheme is designed to counter.

4.1.1 System Model

The proposed system model aligns with the architecture of SCNs in 5G [3], comprising four principal components: the core network (CN), the 5G radio base station (gNB), Home gNB (HgNB), and User Equipment (UE). In standard SCN implementations, the HgNB Management System (HeMS) is responsible for gNB/HgNB configuration according to operator policies. However, this model integrates HeMS functionality into the CN, centralizing the configuration responsibilities for all system entities, including HgNB, gNB, and UE. Consequently, the CN generates the necessary certificates, secret keys, and public keys for these components.

The architecture positions gNB and HgNB as intermediaries facilitating user connectivity to the core network. Each gNB administers a cluster of HgNBs, establishing a *Region* under

Table 4.1: *Notation and cryptographic functions*

Symbol	Description
Notation :	
CN	Core network
gNB & HgNB	Base station and home base station
UE	User equipment
EID	gNB ID
ZUID	Zone user ID
HID	HgNB ID
RUID	Region user ID
RID	Region ID
ω_U	non-membership witness
pid, T_{ID}	User pseudo IDs
T_U	User subscription validity period
RL_v RL_{new}	Revocation list
k_i	Long-term key
k_s	Session key
$\mathsf{AE.Enc}\{k_i, m\}$	Authenticated encryption
$\mathsf{AE.Dec}\{k_i, m\}$	Authenticated decryption
$\mathbb{C}_H$	HgNB certificate
$\mathbb{C}_U$	UE certificate
pk_{sig}^{CN}, sk_{sig}^{CN}	CN public and secret signing keys
pk_{san}^{gNB}, sk_{san}^{gNB}	gNB public and secret sanitising keys
pk_{san}^{HgNB}, sk_{san}^{HgNB}	HgNB public and secret sanitising keys
ACK	Acknowledgement
σ	SanSig Signature
CryptographicFunctions :	
AE.Enc/AE.Dec	Authenticated encryption/decryption
KDF	Key derivation function
SanSig.Sign	Sanitizable signature (sign)
SanSig.Sanit	Sanitizable signature (sanitise)
DDH	Directional Diffie-Hellman

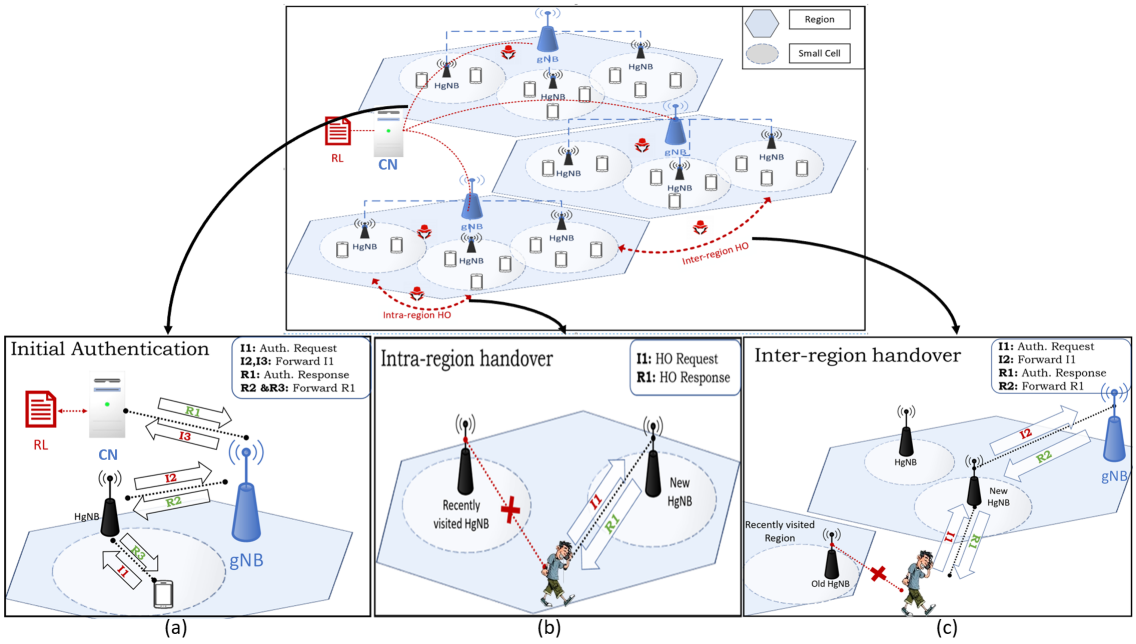

Figure 4.1: *System model [hexagon shape → a region managed by one gNB, oval shape → small cell emulating SCN.]*

gNB jurisdiction, as depicted in Figure 4.1. This hierarchical structure assigns inter-region HO management to gNBs, while HgNBs oversee intra-region HOs and key exchange processes. In accordance with the 5G specification defined in 3GPP TS 33.501, this model follows the standardised security framework, where we assume secure communication channels between network entities (CN, gNB, and HgNB). These channels provide confidentiality, integrity, and replay protection, preventing unauthorised access and data manipulation.

Notably, the standard 5G handover protocol [56] necessitates CN (encompassing all 5G core entities) involvement in each handover protocol. This requirement increases transmission overhead and handover latency, potentially impacting network performance and user experience. The integration of SCNs into 5G networks further exacerbates these challenges, as the increased number of handovers in dense small cell deployments amplifies the impact of these challenges.

To address these limitations, the proposed scheme introduces a regionalized structure. Each region consists of a gNB and its associated HgNBs. For intra-region roaming, user verification is conducted by a designated HgNB utilizing the public key of the region's gNB. In inter-region scenarios, where users roam between regions, the target region's gNB is responsible for user verification and certificate modification, functioning as a sanitiser to preserve user anonymity. This approach eliminates the need for active CN participation during handover procedures, thereby reducing CN transmission overhead, minimizing handover latency, and enhancing security and user privacy.

4.1.2 Threat Model

Our security model is designed to capture security notions recommended by the 3GPP group [4] and [56, 15, 33] for 5G authentication and handover protocols. It also addresses the desirable security and privacy properties identified in Section 3.4.1. In this model, users communicate with other network entities, specifically HgNBs and gNBs, via public and potentially insecure channels. This model, based on the Dolev-Yao framework [35], grants an adversary ($\mathcal{A}$) complete control over these public channels. Consequently, $\mathcal{A}$ can intercept, delete, insert, and modify any transmitted message. Moreover, $\mathcal{A}$ can compromise long-term and per-session secrets, simulating device-compromising attackers.

User privacy is considered an essential property in 5G-based mobile communication, so $\mathcal{A}$ may also try to break user anonymity by linking distinct "challenge" protocol executions of the same user, capturing linkability attacks. We allow $\mathcal{A}$ to schedule session initialisation to strengthen our adversary model. Thus, all sessions are initialised with owners chosen by the adversary, except the "challenge" sessions, which are instead initialised with a pair of potential owners, i.e., $\mathcal{A}$ must distinguish which party owns the "challenge" session.

Furthermore, we aim to ensure that an adversary cannot break authentication or learn session keys established during the proposed handover schemes. In Section 4.3, we present security experiments capturing these notions.

4.2 Secure Region-based Handover Scheme

Here we introduce our proposed handover scheme consisting of four principal phases/protocols: *Registration* Phase, *Initial Authentication*, *Intra-region HO*, and *Inter-region HO* protocols.

The ***Registration*** phase is responsible for the enrolment of gNBs, HgNBs, and new users into the network, as well as configuring the initialization parameters for network components (CN, gNB, and HgNB). During this phase, the CN generates SanSig key pairs for gNBs and issues certificates for HgNBs. Furthermore, the CN establishes and shares a long-term secret key with each user, along with pseudo-identities (pid and T_{ID}) to ensure user anonymity. The CN also initializes the user membership revocation list (RL), which is initially empty.

The ***Initial Authentication*** protocol is the first step that any registered user must execute to gain access to network services. This protocol verifies the user's identity and establishes their initial secure connection to the network. **Figure 4.1(a)** illustrates the message flow of this protocol, involving all network components: CN, gNB, HgNB, and UE. Initially, a UE employs its pseudo-identity, which was assigned in the preceding registration phase. The UE then requests a certificate from the CN to connect to the network. This request is routed through the responsible HgNB and gNB for that region. Upon receipt of the UE request, the CN issues a certificate (via SanSig) for the user using new temporary identities and transmits it to the UE. Section 4.2.2 provides a detailed explanation of this protocol.

The ***Intra-Region Handover*** protocol handles user mobility within a region. Any roaming user moving between two cells belonging to the same region needs to execute this protocol. This process occurs between two HgNBs controlled by a single gNB. It enables mutual authentication with the target HgNB using certificates that were issued to the user after a successful execution of the initial authentication protocol. These certificates are then

used to establish shared secret keys. **Figure 4.1(b)** delineates the message flow of this protocol, involving only the target HgNB and UE. The HgNB authenticates and verifies the user utilizing their certificates during this protocol. The verification process employs the SanSig.Verify() algorithm, which requires the public keys of the CN and gNB that manage the region. Section 4.2.3 comprehensively explains this protocol.

The ***Inter-Region Handover*** protocol manages user mobility between regions. Any roaming user moving between two cells belonging to different regions needs to execute this protocol. This process occurs between two HgNBs controlled by different gNBs, enabling mutual authentication with the target gNB using users' certificates obtained from the initial authentication. These certificates are then used to establish shared secret keys. **Figure 4.1(c)** illustrates the message flow of this protocol, involving only the target HgNB, gNB, and UE. During this protocol, the gNB authenticates the user using the received certificate obtained previously and updates it for the new region. The verification process is executed using the SanSig.Verify() algorithm, while the SanSig.Sanit() algorithm is employed to update the user certificate. Section 4.2.4 provides a detailed exposition of this protocol.

4.2.1 Registration Phase

In the registration phase, we assume secure communication channels between all participating entities (CN, gNB, HgNB, UE). A secure channel is a communication link that ensures confidentiality, integrity, and replay protection, preventing unauthorised access and data manipulation. During this phase, the CN generates and distributes the required credentials for all participants. These credentials encompass HgNB certificates, HgNB sanitising keys, gNB sanitising keys, UE pseudo-identities (T_{ID}, pid), UE long-term keys (k_i), and the revocation list (RL). The registration phase is structured into three distinct components: UE registration, gNB/HgNB registration, and accumulator initialisation. These components are delineated as follows:

1. **UE Registration**: In order to register into the network, each UE_i needs to share their essential information with the CN via a secure channel. Upon receiving the registration request, the CN then generates a long-term secret key k_i, a pseudo-identity (pid_i) and a temporary ID (T_{IDi}) for each user, where pid_i, $T_{IDi} \xleftarrow{\$} \{0,1\}^\lambda$, and λ is the bit-length of k_i.

2. **gNB/HgNB Registration**: Each gNB and HgNB needs to register to the network and share the essential registration information with the CN. Upon receiving the registration request, the CN then generates a signing key pair for itself and HgNBs, i.e., $(hpk_{sig}^{\text{CN}}, hsk_{sig}^{\text{CN}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{sig}(1^\lambda)$, $(pk_{san}^{\text{HgNB}}, sk_{san}^{\text{HgNB}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{san}(1^\lambda)$. These pairs of keys are used to enable HgNBs to sanitise their certificates and prove their legitimacy to the UEs. Next, CN signs the HgNB_i certificate ($\mathbb{C}_H = \mathbb{C}_{fix}^H || \mathbb{C}_{\mathsf{MOD}}^H$) for each HgNB in the network: $\sigma_H \xleftarrow{\$} \mathsf{SanSig.Sign}(\mathbb{C}_H, hsk_{sig}^{\text{CN}}, pk_{san}^{\text{HgNB}}, \mathsf{ADM}(\mathbb{C}_{\mathsf{MOD}}^H))$. Hereafter, CN generates signing and sanitising keys for itself and gNBs: $(gpk_{sig}^{\text{CN}}, gsk_{sig}^{\text{CN}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{sig}(1^\lambda)$,$(pk_{san}^{\text{gNB}}, sk_{san}^{\text{gNB}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{san}(1^\lambda)$. These pairs of keys will be used to sign users' certificates ($\mathbb{C}_U$) in the initial authentication protocol and sanitise

users' certificates in the inter-region protocol. To expedite the registration process, CN can execute this step offline.

3. **Accumulator Initialisation**: To initialise the accumulator/RL, first the CN generates a secret accumulator key $(sk) \xleftarrow{\$} \mathsf{KGen}(1^\lambda)$ and also creates the revocation list $RL \xleftarrow{\$} \mathsf{Gen}(sk, X)$, where X is initially empty.

4.2.2 Initial Authentication

Each registered user who wants to join the network must execute this protocol. As part of this process, the CN generates credentials for new users, which will be used in subsequent HO protocols. In this context, gNBs function as passive intermediaries, which are only responsible for forwarding messages between HgNBs and CNs. Consequently, we consider the HgNB and gNB as a single entity for this protocol. Figure 4.2 illustrates the protocol's steps, which are as follows:

Step $\mathbf{A_1}$: HgNB $\rightarrow$ UE: $\mathbf{M_1}$: $[\mathbb{C}^*_H, \sigma^*_H, g^h]$.
When a new UE enters the coverage area of HgNB, HgNB samples h and computes g^h. Next, HgNB updates their certificate (the modifiable part) , i.e. $\mathbb{C}^{*H}_{mod} = \text{HID} \| g^h$ (preventing replay attacks). Then HgNB sanitises the updated certificate $\mathbb{C}_H = \mathbb{C}^H_{fix} \| \mathbb{C}^{*H}_{mod}$, using the sanitising algorithm $\mathsf{SanSig.Sanit}$, and composes a message $\mathbf{M_1}$, sending $\mathbf{M_1}$ to UE.

Step $\mathbf{A_2}$: UE $\rightarrow$ HgNB/gNB: $\mathbf{M_2}$:$[\mathsf{AE.Enc}\{k_s, M_{A_0} || T_{ID}\}, g^{r_u}]$.

Upon receiving $\mathbf{M_1}$, UE verifies the HgNB sanitised certificate $\mathbb{C}_H$ using the SanSig verification algorithm $\mathsf{SanSig.Verify}$, containing g^h (preventing MITM attacks). If successful, UE samples (r_{id}, r_u), and computes the session keys (sk_i, k_s). Next, UE encrypts $(pid || r_{id})$ using the long-term key k_i shared with CN, to generate the message $\mathbf{M_{A_0}}$. Afterwards, UE encrypts $(M_{A_0} || T_{ID})$ with k_s (preventing linkability), and sends the message $\mathbf{M_2}$ to HgNB.

Step $\mathbf{A_3}$: HgNB/gNB $\rightarrow$ CN: $\mathbf{M_3}$: $[M_{A_0}, T_{ID}, \text{HID}, \text{EID}, \text{RID}]$.
Upon receiving the response message $\mathbf{M_2}$, HgNB computes (sk_i, k_s) to decrypt $\mathbf{M_2}$. Next, HgNB forwards the decrypted message along with the user's pseudo-identities and region identities to CN.

***Note**: In this step, we use the notation $\mathsf{Dec}(M_2)$ to refer specifically to the decryption of the encrypted portion $(M_{A_0} \| T_{ID})$, while other parts (g^{r_u}) remain in plaintext. This is a slight abuse of notation avoids introducing extra symbols.*

Step $\mathbf{A_4}$: CN $\rightarrow$ HgNB/gNB: $\mathbf{M_4}$: $[\mathsf{AE.Enc}\{k_i, \sigma_U \| \mathbb{C}_U \| \omega_U \| v)\}]$.
CN retrieves the long term key k_i of UE using T_{ID}, and decrypts $\mathbf{M_{A_0}}$, to recover (pid, r_{id}). Next, CN computes a new temporary user identifier ${T_{ID}}^*$, and generates a user ID (ZUID_i), which will be the user's identifier in the revocation list RL. CN creates and signs $\mathbb{C}_U$ by generating the "fixed" part of the UE certificate $\mathbb{C}^U_{fix} = \text{ZUID}_i \| T_U$ (where T_U is a user subscription validity period), and the "modifiable" region-specific part of the UE certificate $\mathbb{C}^U_{mod} = \text{RUID} \| \text{RID}$ (where RUID a region-user ID and RID is the region ID). Then CN signs both parts of the UE certificate generating $\mathbb{C}_U \leftarrow$

$\mathsf{SanSig.Sign}$. Next, CN generates a non-membership witness $(\omega_U) \leftarrow \mathsf{NonWitCreate}$, and specifies the version v of RL, corresponding to the version of RL from which ω_U was generated. CN then stores ZUID_i, $T_{ID}i$ and $T_{ID}i^*$ (to prevent de-synchronisation), and encrypts ω_U, UE certificate and its signature using k_i, to generate the message $\mathbf{M_4}$ sending $\mathbf{M_4}$ to the HgNB/gNB.

Step $\mathbf{A_5}$: HgNB/gNB $\rightarrow$ UE: $\mathbf{M_5}$: $[\mathsf{AE.Enc}\{k_s, M_4\}]$

The HgNB/gNB encrypts $\mathbf{M_4}$ using the session key k_s, thereby ensuring unlinkability, to produce the message $\mathbf{M_5}$, which is subsequently transmitted to the UE.

Step $\mathbf{A_6}$: UE $\rightarrow$ HgNB/gNB: **ACK** :$[\mathsf{AE.Enc}(k_s, (\mathsf{AE.Enc}(k_i, flag), T_{ID}))]$

Upon receiving $\mathbf{M_5}$, UE recovers $(\sigma_U, \mathbb{C}_U, \omega_U, v)$, and verifies their certificate, using $\mathsf{SanSig.Verify}(\mathbb{C}_U, \sigma_U, \cdots)$. If verification fails, UE terminates the execution of the protocol. Otherwise, the user then updates $T_{ID}{}^*$, and sends an acknowledgement **ACK**, encryption of an acknowledgement flag $flag$ and the user's T_{ID}, which is encrypted using the user long-term key k_i and T_{ID}, and then encrypted again using the ephemeral key k_s to HgNB.

Step $\mathbf{A_7}$: HgNB/gNB $\rightarrow$ CN: ACK' : $[ACK', T_{ID}]$

The HgNB/gNB decrypts **ACK** using the session key k_s, to produce the message ACK', which is then transmitted along with T_{ID} to the CN.

Step $\mathbf{A_8}$: Upon receiving ACK, CN recovers k_i (using the old T_{ID}), and uses it to decrypt ACK, then CN updates the $T_{ID}{}^*$. If ACK was not received within the prespecified time window, CN deletes $T_{ID}{}^*$. CN will continue to maintain both T_{ID} and $T_{ID}{}^*$. Details of this protocol is depicted in Figure 4.2.

Remark 1 *The primary purpose of sending the ACK message during the initial authentication process is for the CN to ensure that* $\mathbf{M_5}$ *has been successfully delivered to the UE. Only after the successful delivery of message* $\mathbf{M_5}$ *will both the User and the CN update the* T_{ID}*. If the message is not successfully delivered, the CN might update its* T_{ID} *while the User does not, or vice versa, leading to a desynchronization between the UE and the CN. To prevent such desynchronization, the CN maintains both (*T_{ID}*,* $T_{ID}{}^*$*) values until it receives the ACK message. While this approach is effective, another research work has proposed an alternative method. Gope et al. [40] have suggested a solution to prevent desynchronization attacks without compromising the privacy of the UE. In this approach, the UE does not send an ACK. Instead, both the UE and CN maintain a set of single-use pseudo-identities, which are discarded from their memory after use. This method ensures that both parties remain synchronized while preserving the UE's privacy.*

4.2.3 Intra-region Handover

The intra-region HO protocol is executed when a user remains within the same region but moves to a different small cell under the control of a different HgNB, i.e., between HgNBs

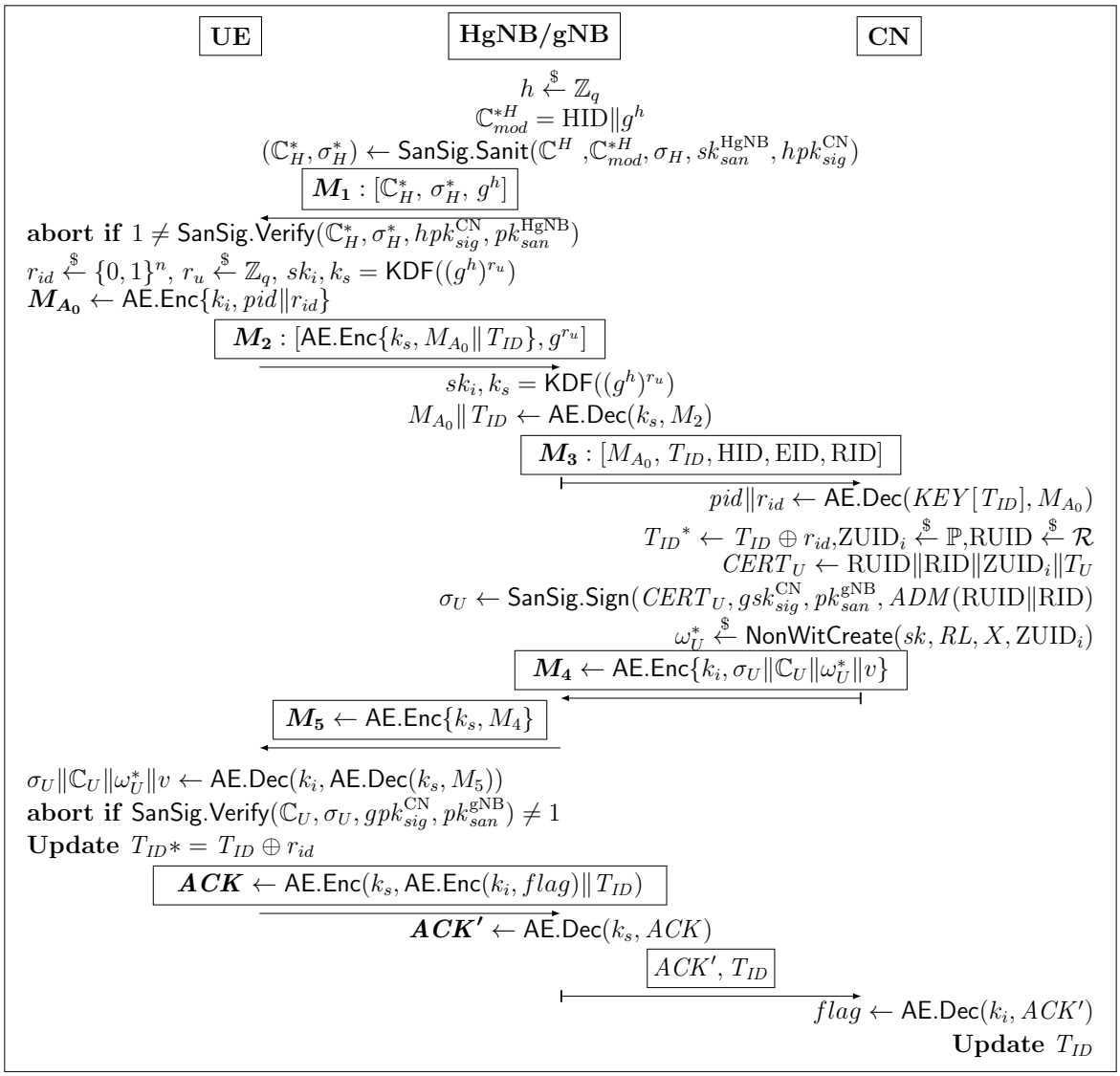

Figure 4.2: *Initial authentication protocol of the proposed region-based HO scheme. Detailed algorithm descriptions are provided in Chapter 2.*

within the same region. This protocol ensures seamless connectivity as the user transitions from one small cell to another under a different HgNB's supervision. The detailed steps of the intra-region HO protocol are outlined below and depicted in Figure 4.3.

Step $\mathbf{B_1}$: HgNB $\rightarrow$ UE: $\mathbf{M_1}$: $[\mathbb{C}^*_H, \sigma^*_H, g^h]$.
This step proceeds identically to **Step $\mathbf{A_1}$** of the initial authentication protocol. In this regard, when a new user enters into the coverage area of new HgNB, the HgNB sanitises their certificate and composes a message $\mathbf{M_1}$, then sending $\mathbf{M_1}$ to UE.

Step $\mathbf{B_2}$: UE $\rightarrow$ HgNB: $\mathbf{M_2}$:$[\mathsf{AE.Enc}\{k_s, \mathbb{C}_U || \sigma_U || \omega_U || v\}, g^{r_u}]$.
Upon receiving the message $\mathbf{M_1}$, UE verifies the HgNB sanitised certificate $\mathbb{C}_H$ and the

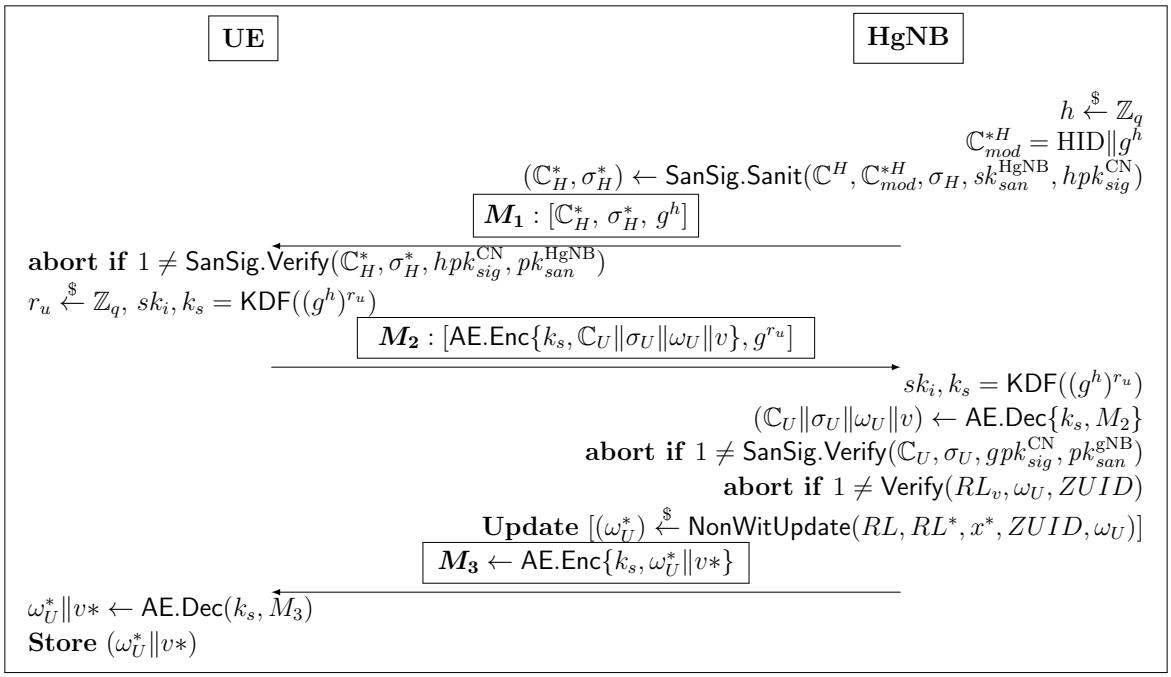

Figure 4.3: *Intra-region handover protocol of the proposed region-based HO scheme. Detailed algorithm descriptions are provided in Chapter 2.*

DH public keyshare g^h, using $\mathsf{SanSig.Verify}(\mathbb{C}^*_H, \sigma^*_H, \cdots)$. If successful, UE samples r_u and computes session keys sk_i, k_s. Next, UE composes a message $\mathbf{M_2}$ and encrypts it using k_s. The encrypted part of $\mathbf{M_2}$ consist of $\mathbb{C}_U$, σ_U, ω_U and v, which is the user's certificate, certificate signature, non-membership witness and the accumulator version of which ω_U was created from, respectively. Finally, UE sends $\mathbf{M_2}$ to HgNB.

Step $\mathbf{B_3}$: HgNB $\rightarrow$ UE: $\mathbf{M_3}$: $[\mathsf{AE.Enc}\{k_s, \omega^*_U||v^*\}]$.
Upon receiving the response message $\mathbf{M_2}$, HgNB generates the session keys sk_i, k_s, to decrypt $\mathbf{M_2}$. Subsequently, HgNB verifies UE's certificate using $\mathsf{SanSig.Verify}(\mathbb{C}_U, \sigma_{UE}, \cdots)$. If successful, HgNB recovers the accumulator version v and checks if $v_i = v_{RL}$, to check if RL has been updated. If not, HgNB checks if the UE has been revoked by calling $\mathsf{Verify}(ZUID_i, ..)$. Otherwise, if the revocation list has been updated, where $v_i \neq v_{RL}$, HgNB checks if $ZUID_i$ has been accumulated in the updated RL. If not, HgNB updates the non-membership witness ω^*_U (where x^* is the new unrevoked UE). Finally, HgNB encrypts and sends $\mathbf{M_3}$ to UE, which they will maintain for future communications. Details of this protocol is depicted in Figure 4.3.

***Note**: In this step, we use the notation $\mathsf{Dec}(M_2)$ to refer specifically to the decryption of the encrypted portion $(\mathbb{C}_U\|\sigma_U\|\omega_U\|v)$, while other parts (g^{r_u}) remain in plaintext. This is a slight abuse of notation to avoid introducing extra symbols .*

4.2.4 Inter-region Handover

The inter-region HO protocol is executed when a user moves between cells belonging to different regions, which involves a transition between HgNBs controlled by different gNBs.

This protocol ensures secure authentication and seamless connectivity as users roam across regions. The detailed steps of the inter-region HO protocol are outlined below and depicted in Figure 4.4.

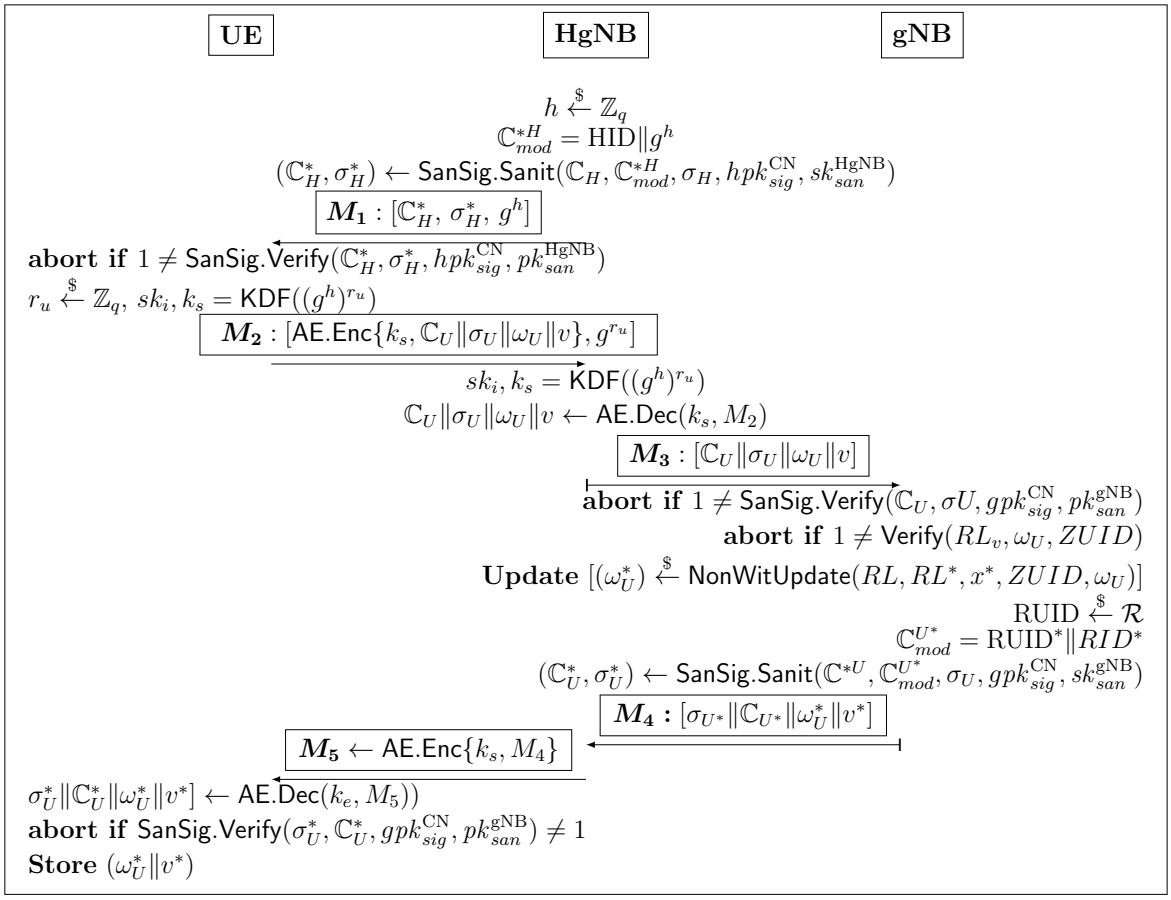

Figure 4.4: *Inter-region handover protocol of the proposed region-based HO scheme. Detailed algorithm descriptions are provided in Chapter 2.*

Step $\mathbf{C_1}$: HgNB → UE: $\mathbf{M_1}$: $[\mathbb{C}^*_H, \sigma^*_H, g^h]$.
This step proceeds as in **Step $\mathbf{A_1}$** of the initial authentication protocol. In this regard, for the new users entering the coverage area of HgNB, the HgNB sanitises his/her certificate and composes a message $\mathbf{M_1}$, then sending $\mathbf{M_1}$ to UE.

Step $\mathbf{C_2}$: UE → HgNB: $\mathbf{M_2}$:$[\mathsf{AE.Enc}\{k_s, \mathbb{C}_U||\sigma_U||\omega_U|||v\}, g^{r_u}]$.
This step proceeds identically to **Step $\mathbf{B_2}$** of the intra-region HO protocol. In this regard, UE verifies $\mathbf{M_1}$, compute a session key. UE then composes $\mathbf{M_2}$, encrypt it using the session key then send it to HgNB.

Step $\mathbf{C_3}$: HgNB → gNB: $\mathbf{M_3}$: $[\mathbb{C}_U||\sigma_U||\omega_U||v]$.
Upon the arrival of $\mathbf{M_2}$, HgNB generates the keys sk_i, k_s to decrypt $\mathbf{M_2}$, then forwards the decrypted message to gNB.

***Note**: In this step, we use the notation* $\mathsf{Dec}(M_2)$ *to refer specifically to the decryption of the encrypted portion* $(\mathbb{C}_U \| \sigma_U \| \omega_U \| v)$*, while other parts* (g^{r_u}) *remain in plaintext. This is a slight abuse of notation to avoid introducing extra symbols .*

Step $\mathbf{C_4}$: gNB $\rightarrow$ HgNB: $\mathbf{M_4}$: $[\sigma_U^* \| \mathbb{C}_U^* \| \omega_U^* \| v^*]$.
After receiving the response message $\mathbf{M_3}$, gNB verifies the user's certificate using the SanSig verification algorithm, i.e. $\mathsf{SanSig.Verify}(\mathbb{C}_U, \sigma_U, \cdots)$. If successful, gNB retrieves the accumulator version v and checks if $v_i = v_{RL}$, to see if RL has been updated. If not, gNB checks if the UE has been revoked by using $\mathsf{Verify}(ZUID_i)$. Otherwise, if the revocation list has been updated (and $v_i \neq v_{RL}$) gNB checks whether $ZUID_i$ is added in the later version of the RL. If not, gNB updates the non-membership witness $\omega_U^* \leftarrow \mathsf{NonWitUpdate}(.)$ (where x^* is the new non-revoked UE). Subsequently gNB updates the region-user identifier RUID_i^*, updates the "modifiable" region-specific part of the UE certificate $cert_{mod}^{*U}$, and updates the user certificate accordingly, where $\mathbb{C}_U^* = \mathbb{C}_{mod}^{*U} || \mathbb{C}_{fix}^U$. After, gNB sanitises UE $\mathbb{C}_U \leftarrow \mathsf{SanSig.Sanit}(.)$. Finally, gNB composes $\mathbf{M_4}$, sending $\mathbf{M_4}$ to HgNB.

Step $\mathbf{C_5}$: HgNB $\rightarrow$ UE: $\mathbf{M_5}$: $[\mathsf{AE.Enc}\{k_s, M_4\}]$.
Upon receiving the message $\mathbf{M_4}$, the HgNB encrypts it using the session key k_s to generate $\mathbf{M_5}$, which is then transmitted to the UE.

Step $\mathbf{C_6}$: Upon receiving the encrypted message $\mathbf{M_5}$, the UE recovers $(\sigma_U^*, \mathbb{C}_U^*, \omega_U^*, v^*)$, and verifies the sanitised certificate signature, using SanSig verification algorithm i.e. $\mathsf{SanSig.Verify}(\mathbb{C}_U^*, \sigma_U^*, \cdots)$. If verification fails, UE terminates the execution of the protocol. Otherwise, the user updates their certificate and RUID. Details of this protocol is depicted in Figure 4.

4.3 Security Analysis

This section presents formal proof demonstrating that our protocols achieve mutual authentication, key indistinguishability, and unlinkability, following the security framework provided in Section 2.3. Each proof is structured as a series of game-hops, where we incrementally modify the experiment and ultimately show that the adversary cannot succeed (or detect the changes) with non-negligible probability. We start by analysing the MA-security of each protocol individually.

4.3.1 Mutual Authentication Security

In this section, we analyse the mutual authentication (MA) security of our proposed scheme, demonstrating that it achieves MA security. We build upon the general MA-security model introduced in Chapter 2, Section 2.3.2. In the MA-security game, defined in Def.24, $\mathcal{A}$ wins (and $\mathbf{Exp}^{\mathsf{MA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if the adversary has caused a **clean** session to accept (and set $\pi_i^s.\alpha \leftarrow \texttt{accepted}$) and there either exists no matching subset session π_j^t (i.e. no CN session that outputs the messages received by π_i^s - Def.22), or no matching session π (i.e. no CN session that outputs the messages received by π_i^s - Def.23).

Here, we specify the adversary queries and cleanness predicate for our particular protocols.

Adversary Queries. Here, we define queries that represent the behaviours of the adversary $\mathcal{A}$ during the execution of the MA experiment:

- **Create**(i,s): allows $\mathcal{A}$ to initialize new UE sessions π_i^s.
- **Send**(m, i, s): allows $\mathcal{A}$ to send messages m to a session π_i^s. It produces a message m', which might be empty.
- **CorruptLTK**(i)$\rightarrow k_i$: allows $\mathcal{A}$ to leak the long-term key of UE_i.
- **StateReveal**(i,s)$\rightarrow \pi_i^s$: allows $\mathcal{A}$ to reveal the internal state of π_i^s.

Cleanness Predicate. It is important to note that forging messages to the π_i^s is trivial if the $\mathcal{A}$ has compromised the long-term key shared between CN and UE. To mitigate this type of attack, we introduce a cleanness predicate, which ensures that the adversary is prevented from issuing a **CorruptLTK** or **StateReveal** queries before the test session acceptance.

Definition 27 (Cleanness predicate) *A session π_i^s in the* MA *experiment, described in Section 2.3.2, is* **clean** *if* ***CorruptLTK***(i) *was not issued before $\pi_i^s.\alpha =$* `accepted`*,* and ***StateReveal***(i, s) *was not issued* and *for all j, s' such that $\pi_j^{s'}.msg_r = \pi_i^s.msg_s$,* ***StateReveal***$(j, s')$ *was not issued.*

4.3.1.1 MA-security of Initial Authentication protocol

Here we present our formal analysis and results for the MA-security of the initial authentication (IA) protocol.

Theorem 1 MA-***security of Initial Authentication***. *The initial authentication protocol depicted in Figure 4.2 is* MA*-secure under the cleanness predicate defined in Def.27. For any PPT algorithm $\mathcal{A}$ against the* MA *experiment, $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is negligible assuming the EUFCMA security of* SanSig*, INT-CTXT security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the MA-security experiment, that $\mathcal{A}$ cannot issue a **CorruptLTK**(i) query before a session π_i^s accepts such that $\mathcal{C}$ terminates the game and outputs 1, nor can it issue a **StateReveal**(i, s), nor a **StateReveal**(CN, s) query (where $\pi_{\text{CN}}^{s'}$ received messages from π_i^s).

We divide the proof into two cases: the first where the UE accepts messages $\mathbf{M_1}$ and $\mathbf{M_4}$ without an honest matching CN partner. We denote $\mathcal{A}$'s advantage in **Case 1** as $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$. The second case is when the CN accepts messages $\mathbf{M_3}$ and **ACK** without an honest matching UE partner. We denote $\mathcal{A}$'s advantage in **Case 2** as $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$, where λ is the bit-length of k_i. It is clear that:

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},IA}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

Case 1: UE accepts messages $\mathbf{M_1}$ and $\mathbf{M_4}$ without an honest matching CN partner. Here we provide the security analysis through a series of game-based reductions:

Game $A_{1.0}$: This is the original mutual authentication experiment defined in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{A_{1.0}}}$

Game $A_{1.1}$: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under pk^{CN}_{sig} and pk^{HgNB}_{san}. At the beginning of the experiment, we initialise a SanSig challenger $\mathcal{C}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN and HgNB respectively. Then any time CN or HgNB needs to generate a signature, we query $\mathsf{SanSig.Sign}$ to sign a message m. Now, we trigger the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17: $\mathbf{Adv}_{G_{A_{1.0}}} \leq \mathbf{Adv}_{G_{A_{1.1}}} + \mathbf{Adv}^{\text{EUFCMA}}_{\mathsf{SanSig}}(\mathcal{A})$.

Game $A_{1.2}$: In this game we guess the first session π^s_i to accept without a matching partner, such that $\pi_i.role = UE$. Since there are at most n_P parties running n_S sessions, the probability of session π^s_i accepts without a matching partner is: $\mathbf{Adv}_{G_{A_{1.1}}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{A_{1.2}}}$.

Game $A_{1.3}$: Here we introduce another abort event. That is triggered if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π^s_i, i.e. session π^s_i receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by **Game** $A_{1.1}$ we abort if $\mathcal{A}$ generates a valid signature over any message m, it follows that: $\mathbf{Adv}_{G_{A_{1.2}}} \leq \mathbf{Adv}_{G_{A_{1.3}}}$.

Game $A_{1.4}$: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from a DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus any $\mathcal{A}$ that can distinguish **Game** $A_{1.3}$ from **Game** $A_{1.4}$ can break the DDH assumption, as formalised in Def.10. Thus it follows that: $\mathbf{Adv}_{G_{A_{1.3}}} \leq \mathbf{Adv}_{G_{A_{1.4}}} + \mathbf{Adv}^{G,g,q}_{\mathsf{DDH}}(\mathcal{A})$.

Game $A_{1.5}$: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ by interacting with a KDF challenger. Since $sk_i, k_s \leftarrow \mathsf{KDF}(g^c)$ and by **Game** $A_{1.4}$ g^c is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game** $A_{1.4}$ from **Game** $A_{1.5}$ can be used to break KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{A_{1.4}}} \leq \mathbf{Adv}_{G_{A_{1.5}}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game $A_{1.6}$: In this game, we introduce an abort event that occurs if π^s_i decrypts a valid ciphertext keyed by $\hat{sk_i}$, but the ciphertext was not produced by an honest CN session. Specifically, we initialise an AE challenger that is queried whenever the challenger needs to encrypt or decrypt with $\hat{sk_i}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AE challenger to break the security of the AE scheme. By **Game** $A_{1.5}$ $\hat{sk_i}$ is already uniformly random and independent and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can break the INT-CTXT security of the AE scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{A_{1.5}}} \leq \mathbf{Adv}_{G_{A_{1.6}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $A_{1.7}$: In this game, the session π_i will only accept $\mathbf{M_1}$ from HgNB and $\mathbf{M_4}$ from CN if they are honest partners. $\mathcal{A}$ cannot produce a valid ciphertext by **Game** $A_{1.6}$, and $\mathcal{A}$ cannot produce a valid signature by **Game** $A_{1.1}$. Thus the advantage of $\mathcal{A}$ in winning the MA-security experiment is negligible. $\mathbf{Adv}_{G_{A_{1.7}}} = 0$.

Case 2: CN accepts messages $\mathbf{M_3}$ and **ACK** without an honest matching UE partner. In this case, we assume that the first session to accept without a matching partner is owned by CN. Here we provide the security analysis through a series of game-based reductions:

Game $A_{2.0}$: This is the original mutual authentication game described in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{A_{2.0}}}$.

Game $A_{2.1}$: In this game, we guess the index i of the first CN session that accepts without a matching partner such that their partner is owned by UE_i, i.e. $\pi^s_{\mathrm{CN}}.pid = \mathrm{UE}_i$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_{A_{2.0}}} \leq n_P \cdot \mathbf{Adv}_{G_{A_{2.1}}}$.

Game $A_{2.2}$: As per the definition of cleanness predicate (27), $\mathcal{A}$ cannot issue a $\mathbf{CorruptLTK}(i)$ query before the CN session accepts without a matching partner. In this game, we introduce an abort event that triggers if the first π^t_{CN} to accept without a matching partner accepts a ciphertext $\mathbf{M_3}$ that was not output from a matching partner session π^s_i. Specifically, we initialise an INT-CTXT challenger that is queried whenever $\mathcal{C}$ needs to encrypt or decrypt with k_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AEchallenger to break the INT-CTXT security of the AE scheme. Since k_i is uniformly random and cannot be leaked to $\mathcal{A}$, this replacement is sound. Thus, any $\mathcal{A}$ that triggers this abort event can be used to break the INT-CTXT security of AE, as formalised in Def.6. Thus: $\mathbf{Adv}_{G_{A_{2.1}}} \leq \mathbf{Adv}_{G_{A_{2.2}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $A_{2.3}$: Here we introduce a similar abort event that triggers if π^t_{CN} accepts a ciphertext **ACK** that was not output from a matching partner session π^s_i. The changes introduced to **Game** $A_{2.3}$ follow from **Game** $A_{2.2}$, and thus introduces no new advantage for $\mathcal{A}$. Thus: $\mathbf{Adv}_{G_{A_{2.2}}} \leq \mathbf{Adv}_{G_{A_{2.3}}}$.

Game $A_{2.4}$: In this game, the π^t_{CN} only accepts $\mathbf{M_3}$ and **ACK** from an honest matching partner. Thus, summing the probabilities we find that the $\mathcal{A}$ has a negligible advantage in winning the MA-security experiment.: $\mathbf{Adv}_{G_{A_{2.4}}} = 0$

4.3.1.2 MA-security of Intra-region Handover

Here we present our formal analysis and results for the MA-security of the intra-region handover (IRH) protocol.

Theorem 2 MA-*security of Intra-region Handover*. *The intra-region handover protocol depicted in Figure 4.3 is* MA-*secure under the cleanness predicate defined in Def.27. For any PPT algorithm $\mathcal{A}$ against the* MA *experiment,* $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of SanSig, INT-CTXT security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the MA security experiment, that $\mathcal{A}$ cannot issue a **CorruptLTK**(i) query before a session π_i^s accepts such that $\mathcal{C}$ terminate the game and outputs 1, nor can it issue a **StateReveal**(i, s), nor a **StateReveal**(HgNB, s) query (where $\pi_{\text{HgNB}}^{s'}$ received messages from π_i^s).

Here we divide the games into two cases: the first where the UE accepts messages m_{B_1} and m_{B_3} without an honest matching HgNB partner. The second case is when the HgNB accepts a message (m_{B_2}) without an honest matching UE partner. It is clear that:

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},IRH}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

Case 1: In this case, we assume that there exists the first session to accept without a matching partner is a UE. We choose the first session because if we can prove that an adversary $\mathcal{A}$ has a negligible opportunity of making some first session to accept without a matching partner, it must follow that $\mathcal{A}$ cannot cause any session to accept without a matching partner. Here we provide the security analysis through a series of game-based reductions:

Game $B_{1.0}$: This mutual authentication experiment is adapted from the original definition presented in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{B_{1.0}}}$.

Game $B_{1.1}$: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^{\text{CN}}_{sig}\&pk^{HgNB}_{san}$. At the beginning of this game, we initialize a SanSig challenger that outputs $pk^{\mathcal{C}}_{sig}\&pk^{\mathcal{C}}_{san}$, which we embed into CN and HgNB. Then every time CN or HgNB needs to generate a signature, we query $\mathsf{SanSig.Sign}$ to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of $SanSig$, as formalised in Def.17: $\mathbf{Adv}_{G_{B_{1.0}}} \leq \mathbf{Adv}_{G_{B_{1.1}}} + \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game $B_{1.2}$: In this game, we guess the first session π_i^s to accept without a matching partner, such that $\pi_i.role = UE$. Since there are n_P parties running at most n_S sessions, the probability of session π_i^s accepts without a matching partner is: $\mathbf{Adv}_{G_{B_{1.1}}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{B_{1.2}}}$.

Game $B_{1.3}$: Here we introduce another abort event. That is triggered if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_i^s, i.e. session π_i^s receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by Game $B_{1.2}$ we abort if $\mathcal{A}$ generates a valid signature over any message m, it follows that: $\mathbf{Adv}_{G_{B_{1.2}}} \leq \mathbf{Adv}_{G_{B_{1.3}}}$.

Game $B_{1.4}$: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game $B_{1.3}$ from Game $B_{1.4}$ without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10: $\mathbf{Adv}_{G_{B_{1.3}}} \leq \mathbf{Adv}_{B_{A_{1.4}}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game $B_{1.5}$: In this game the challenger replaces the encryption and session keys sk_i, sk_i' with uniformly random values $\hat{sk_i}, \hat{sk_i'}$ by interacting with a KDF challenger. Since $sk_i, sk_i' \leftarrow \mathsf{KDF}(g^c)$ and by **Game** $B_{1.4}$ g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{B_{1.4}}} \leq \mathbf{Adv}_{G_{B_{1.5}}} + \mathbf{Adv}^{KDF}_{KDF}(\mathcal{A})$.

Game $B_{1.6}$: In this game, we introduce an abort event that occurs if π_i^s decrypts a valid ciphertext keyed by $\hat{sk_i}$. Specifically, we initialise an AE challenger that is queried whenever the challenger needs to encrypt or decrypt with sk_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AE challenger to break the security of the security of the AE scheme. By **Game** $B_{1.5}$, this replacement is sound and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{B_{1.5}}} \leq \mathbf{Adv}_{G_{B_{1.6}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $B_{1.7}$: In this game, the session π_i^s will only accepts m_{B_1} and m_{B_3} from an honest HgNB. Knowing that $\mathcal{A}$ cannot produce a valid ciphertext from Game $B_{1.6}$, and $\mathcal{A}$ cannot produce a valid signature from Game $B_{1.1}$. Thus the advantage of $\mathcal{A}$ in winning the MA security experiment is negligible.: $\mathbf{Adv}_{G_{B_{1.7}}} = 0$.

Case 2: In this case, we assume that there is a session where HgNB accepts a message (m_{B_2}) without a matching partner. Here we provide the security analysis through a series of game-based reductions:

Game $B_{2.0}$: This mutual authentication experiment is adapted from the original definition presented in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{B_{2.0}}}$.

Game $B_{2.1}$: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^U_{sig}\&pk^U_{san}$. At the beginning of this game, we initialize a SanSig challenger that outputs $pk^{\mathcal{C}}_{sig}\&pk^{\mathcal{C}}_{san}$, which we embed into UE. Then every time UE needs to generate a signature, we query $\mathsf{SanSig.Sign}$ to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of $SanSig$, as formalised in Def.17. Thus: $\mathbf{Adv}_{G_{B_{1.0}}} \leq \mathbf{Adv}_{G_{B_{1.1}}} + \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game $B_{2.2}$: In this game, we guess i such that the first HgNB session that accepts without a matching partner sets $\pi^s_{\text{HgNB}}.pid = \text{UE}_i$, introducing a factor of n_P and n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_{B_{2.0}}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{B_{2.1}}}$.

Game $B_{2.3}$: Here we introduce another abort event. That is triggers if $\mathcal{A}$ sends a public key shared g^{r_u} to the session π_i^s, i.e. session π_i^s receives g^{r_u} that is not from an honest UE. Since this requires a signature over g^{r_u}, and by Game $B_{2.1}$ we abort if $\mathcal{A}$ generates a valid signature over any message m, it follows that: Thus: $\mathbf{Adv}_{G_{B_{2.2}}} \leq \mathbf{Adv}_{G_{B_{2.3}}}$.

Game $B_{2.4}$: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game $B_{2.3}$ from Game

$B_{2.4}$ without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. Thus: $\mathbf{Adv}_{G_{B_{2.3}}} \leq \mathbf{Adv}_{G_{B_{2.4}}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game $B_{2.5}$: In this game the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk}_i, \hat{sk}'_i$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** $B_{2.4}$ g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{B_{2.4}}} \leq \mathbf{Adv}_{G_{B_{2.5}}} + \mathbf{Adv}^{KDF}_{KDF}(\mathcal{A})$.

Game $B_{2.6}$: In this game, we introduce an abort event that occurs if π^s_i decrypts a valid ciphertext keyed by $\hat{sk}_i$. Specifically, we initialise an AE challenger that is queried whenever the challenger needs to encrypt or decrypt with sk'_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AE challenger to break the security of the security of the AE scheme. By **Game** $B_{2.5}$ this replacement is sound and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{B_{2.5}}} \leq \mathbf{Adv}_{G_{B_{2.6}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $B_{2.7}$: In this game, the session π^s_i will only accepts m_{B_2} from an honest UE. Knowing that $\mathcal{A}$ cannot produce a valid ciphertext from Game $B_{2.6}$, and $\mathcal{A}$ cannot produce a valid signature from Game $B_{2.1}$. Thus the advantage of $\mathcal{A}$ in winning the MA security experiment is negligible: $\mathbf{Adv}_{G_{B_{2.7}}} = 0$.

4.3.1.3 MA-security of Inter-region Handover

Here, we present our formal analysis and results for the MA-security of the inter-region handover (ERH) protocol.

Theorem 3 MA-*security of Inter-region Handover*. *The inter-region handover protocol depicted in Figure 4.4 is* MA-*secure under the cleanness predicate defined in Def.27. For any PPT algorithm* $\mathcal{A}$ *against the* MA *experiment,* $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of SanSig, INT-CTXT security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the MA security experiment, that $\mathcal{A}$ cannot issue a **CorruptLTK**(i) query before a session π^s_i accepts such that $\mathcal{C}$ terminate the game and outputs 1, nor can it issue a **StateReveal**(i, s), nor a **StateReveal**(HgNB, s) query (where $\pi^{s'}_{\text{HgNB}}$ received messages from π^s_i).

Here we divide the games into two cases: the first where the UE accepts messages m_{C_1} and m_{C_4} without an honest matching gNB partner. The second case is when the HgNB accepts a message m_{C_2} without an honest matching UE partner.It is clear that:

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},ERH}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

Case 1: In this case, we assume that there exists the first session to accept without a matching partner is a UE. We choose the first session because if we can prove that an adversary $\mathcal{A}$ has a negligible opportunity of making some first session to accept without a matching partner, it must follow that $\mathcal{A}$ cannot cause any session to accept without a

matching partner. Here we provide the security analysis through a series of game-based reductions:

Game $C_{1.0}$: This is the original mutual authentication experiment defined in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{C_{1.0}}}$.

Game $C_{1.1}$: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^{\text{CN}}_{sig}\&pk^{HgNB}_{san}$ or $pk^{\text{CN}}_{sig}\&pk^{gNB}_{san}$. At the beginning of this game, we initialize a pair of SanSig challengers that output $pk^{\mathcal{C}}_{sig}\&pk^{\mathcal{C}}_{san}$, which we embed into CN, HgNB and gNB. Then every time CN, HgNB or gNB needs to generate a signature, we query $\mathsf{SanSig.Sign}$ to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of $SanSig$, as formalised in Def.17. Thus: $\mathbf{Adv}_{G_{C_{1.0}}} \leq \mathbf{Adv}_{G_{C_{1.1}}} + 2 \cdot \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game $C_{1.2}$: In this game, we guess the first session π^s_i to accept without a matching partner, such that $\pi_i.role = UE$. Since there are n_P parties running at most n_S sessions, the probability of session π^s_i accepts without a matching partner is: $\mathbf{Adv}_{G_{C_{1.1}}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{C_{1.2}}}$.

Game $C_{1.3}$: Here we introduce another abort event. That is triggers if $\mathcal{A}$ sends a Diffie-Hellman public key share g^h to the session π^s_i, i.e. session π^s_i receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by Game $C_{1.2}$ we abort if $\mathcal{A}$ generates a valid signature over any message m, it follows that: $\mathbf{Adv}_{G_{C_{1.2}}} \leq \mathbf{Adv}_{G_{C_{1.3}}}$.

Game $C_{1.4}$: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game $C_{1.3}$ from Game $C_{1.4}$ without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. it follows that: $\mathbf{Adv}_{G_{C_{1.3}}} \leq \mathbf{Adv}_{G_{C_{1.4}}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game $C_{1.5}$: In this game the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk_i}, \hat{sk'_i}$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** $C_{1.4}$ g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{C_{1.4}}} \leq \mathbf{Adv}_{G_{C_{1.5}}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game $C_{1.6}$: In this game, we introduce an abort event that occurs if π^s_i decrypts a valid ciphertext keyed by $\hat{sk_i}$. Specifically, we initialise an AE challenger that is queried whenever the challenger needs to encrypt or decrypt with sk_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AE challenger to break the security of the security of the AE scheme. By **Game** $C_{1.5}$ this replacement is sound and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{C_{1.5}}} \leq \mathbf{Adv}_{G_{C_{1.6}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $C_{1.7}$: In this game, the session π_i will only accepts m_{C_1} and m_{C_4} from an honest HgNB. Knowing that $\mathcal{A}$ cannot produce a valid ciphertext from Game $C_{1.6}$, and $\mathcal{A}$ cannot produce a valid signature from Game $C_{1.1}$. Thus the advantage of $\mathcal{A}$ in winning the MA security experiment is negligible: $\mathbf{Adv}_{G_{C_{1.7}}} = 0$.

Case 2: In this case, we assume that there is a session where gNB accepts the message m_{C_2} without a matching partner. Here we provide the security analysis through a series of game-based reductions:

Game $C_{2.0}$: This is the original mutual authentication defined in Def.24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{C_{2.0}}}$.

Game $C_{2.1}$: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^{\text{CN}}_{sig}\&pk^{HgNB}_{san}$. At the beginning of this game, we initialize a SanSig challenger that outputs $pk^{\mathcal{C}}_{sig}\&pk^{\mathcal{C}}_{san}$, which we embed into CN and HgNB. Then every time CN or HgNB needs to generate a signature, we query SanSig.Sign to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of $SanSig$, as formalised in Def.17. This implies:: $\mathbf{Adv}_{G_{C_{1.0}}} \leq \mathbf{Adv}_{G_{C_{1.1}}} + \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game $C_{2.2}$: In this game, we guess i such that the first gNB session that accepts without a matching partner sets $\pi^s_{\text{gNB}}.pid = \text{UE}_i$, introducing a factor of n_P and n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_{C_{2.0}}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{C_{2.1}}}$.

Game $C_{2.3}$: Here we introduce another abort event. That is triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^{r_u} to the session π^s_i, i.e. session π^s_i receives g^{r_u} that is not from an honest UE. Since this requires a signature over g^{r_u}, and by Game $C_{2.1}$ we abort if $\mathcal{A}$ generates a valid signature over any message m, it follows that:$\mathbf{Adv}_{G_{C_{2.2}}} \leq \mathbf{Adv}_{G_{C_{2.3}}}$.

Game $C_{2.4}$: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game $C_{1.3}$ from Game $C_{1.4}$ without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{C_{2.3}}} \leq \mathbf{Adv}_{G_{C_{2.4}}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game $C_{2.5}$: In this game the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk}_i, \hat{sk}'_i$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** $C_{2.4}$ g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{C_{2.4}}} \leq \mathbf{Adv}_{G_{C_{2.5}}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game $C_{2.6}$: In this game, we introduce an abort event that occurs if π^s_i decrypts a valid ciphertext keyed by $\hat{sk}_i$. Specifically, we initialise an AE challenger that is queried whenever the challenger needs to encrypt or decrypt with sk_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext, and we can submit the valid ciphertext to the AE challenger to break the security of the security of the AE scheme. By

Game $C_{2.5}$ this replacement is sound and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{C_{2.5}}} \leq \mathbf{Adv}_{G_{C_{2.6}}} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AE}}(\mathcal{A})$.

Game $C_{2.7}$: In this game, the session π_i will only accepts m_{C_2} from an honest UE. Knowing that $\mathcal{A}$ cannot produce a valid ciphertext from Game $C_{2.6}$, and $\mathcal{A}$ cannot produce a valid signature from Game $C_{2.1}$. Thus the advantage of $\mathcal{A}$ in winning the MA security experiment is negligible: $\mathbf{Adv}_{G_{C_{2.7}}} = 0$

4.3.2 Unlinkability Security

In this section, we analyse the unlinkability (Unlink) security of our proposed scheme, demonstrating that they achieve unlinkability security. We build upon the general Unlink-security model introduced in Chapter 2, Section 2.3.4. A protocol Π is Unlink-secure, if there exist no PPT algorithms $\mathcal{A}$ that can win the Unlink security game with non-negligible advantage. Here, we specify the adversary capabilities and cleanness predicate for our particular protocols.

Adversary Queries In addition to the **Create**, **CorruptLTK**, **Send** and **StateReveal** queries listed in Section 4.3.1, we define two additional queries: **Test**, which allows the adversary to initialise one of two sessions (depending on a bit b sampled by the challenger), and **SendTest**, which allows the adversary to interact with that session without revealing which party owns it.

- **Test**(s,i,s',i') $\rightarrow m$: allows $\mathcal{A}$ to begin a new session π_b, where $(\pi_0 = \pi_i^s)$ or $(\pi_1 = \pi_{i'}^{s'})$, where b is sampled by $\mathcal{C}$, and both π_i^s and $\pi_{i'}^{s'}$ are **clean**. **Test** query is only allowed to be issued by $\mathcal{A}$ if $\pi_b.\alpha \neq in-progress$ and **Send** queries to sessions owned by UE_i or $\mathrm{UE}_{i'}$ are only allowed to be issued by $\mathcal{A}$ until π_b has rejected or accepted the experiment execution.

- **SendTest**$(m) \rightarrow (m')$: allows $\mathcal{A}$ to send a message m to π_b after issuing **Test**. The $\mathcal{C}$ returns a $\perp$ if $\pi_b.\alpha \neq in-progress$.

Cleanness Predicate. It is important to note that it is trivial for $\mathcal{A}$ to determine which of UE_i or $\mathrm{UE}_{i'}$ owns session π_b if the $\mathcal{A}$ has compromised the long-term key. To mitigate this type of attack, we introduce a cleanness predicate, which ensures that the adversary is prohibited from issuing a **CorruptLTK** or **StateReveal** queries before the session's acceptance.

Definition 28 (Cleanness predicate) *A session π_i^s in the* Unlink *experiment, described in Section 2.3.4, is* **clean** *if* ***CorruptLTK***(i) *or* ***CorruptLTK***(i') *was not issued before* $\pi_b.\alpha =$ `accepted`, and ***StateReveal***(i,s), ***StateReveal***(i',s') *was not issued* and *for all s' such that $\pi_{\mathrm{CN}}^{s'}.msg_r = \pi_b.msg_s$,* ***StateReveal***$(\mathrm{CN},s')$ *was not issued.*

4.3.2.1 Unlink-security of Initial Authentication protocol

Here we present our formal analysis and results for the Unlink-security of the initial authentication protocol.

Theorem 4 Unlink-***security of Initial Authentication***. *The initial authentication protocol depicted in Figure 4.2 is unlinkable under the cleanness predicate defined in Def.28. For any PPT algorithm $\mathcal{A}$ against the* Unlink *experiment,* $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig*, IND-CPA security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the MA security experiment, that $\mathcal{A}$ cannot issue a $\mathbf{CorruptLTK}(i)$ query before a session π^s_i accepts such that $\mathcal{C}$ terminate the game and outputs 1, nor can it issue a $\mathbf{StateReveal}(i,s)$, nor a $\mathbf{StateReveal}(\mathrm{CN},s)$ query (where $\pi^{s'}_{\mathrm{CN}}$ received messages from π^s_i). Below we proceed via a sequence of games. We bound the adversary advantage difference of winning these games with the underlying cryptographic assumptions till the adversary reaches a game where the advantage of that game equals 0, which shows that adversary A cannot win. Here we provide the security analysis through a series of game-based reductions:

Game A_0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{A_0}}$.

Game A_1: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^{CN}_{sig}\&pk^{HgNB}_{san}$. At the beginning of this game, we initialize a SanSig challenger which outputs $pk^{\mathcal{C}}_{sig}\&pk^{\mathcal{C}}_{san}$, which we embed into CN and HgNB. Then every time CN or HgNB needs to generate a signature, we query SanSig.Sign to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of $SanSig$, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_{A_0}} \leq \mathbf{Adv}_{G_{A_1}} + \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game A_2: In this game, we guess the first session π^s_i to accept without a matching partner, such that $\pi_i.role = UE$. We also introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π^s_i, i.e. session π^s_i receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by Game A_1 we abort if $\mathcal{A}$ generates a valid signature over any message m, this introduces no additional bound. Since there are n_P parties running at most n_S sessions, the probability of session π^s_i accepts without a matching partner is: $\mathbf{Adv}_{G_{A_1}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{A_2}}$.

Game A_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game A_3 from Game A_2 without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{A_2}} \leq \mathbf{Adv}_{G_{A_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game A_4: In this game, the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk_i}, \hat{sk'_i}$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** A_3 g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{A_3}} \leq \mathbf{Adv}_{G_{A_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game A_5: In this game, we replace the computation of the ciphertext $c = \mathsf{Enc}(sk_i, M_{A_0} \| T_{ID})$ with $\hat{c} = \mathsf{Enc}(\hat{sk_i}, rand)$, where $rand \xleftarrow{\$} \{0,1\}^L$ and $L = |M_{A_0} \| T_{ID}|$, and the computation of the ciphertext $c^* = \mathsf{Enc}(sk_i, M^*_{A_0} \| {T_{ID}}^*)$ with $\hat{c^*} = \mathsf{Enc}(\hat{sk_i}, rand^*)$ where $rand^* \xleftarrow{\$} \{0,1\}^{L^*}$ and $L^* = |M^*_{A_0} \| {T_{ID}}^*|$. We do so by interacting with an AE challenger whenever $\hat{sk_i}$ is used by the challenger to encrypt a message, and issuing either an Enc oracle call $(M_{A_0} \| T_{ID}, rand)$ or $(M^*_{A_0} \| {T_{ID}}^*, rand^*)$. Note that if the bit b sampled by AE is 0, then we are in **Game** A_4, and otherwise we are in **Game** A_5. Since $\hat{sk_i}$ is uniformly random and independent, this change is sound, and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{A_4}} \leq \mathbf{Adv}_{G_{A_5}} + \mathbf{Adv}^{IND-CPA}_{\mathsf{AE}}(\mathcal{A})$.

Game A_6: In this game we highlight that the channel between the HgNB and the gNB is assumed to be secure, thus $\mathcal{A}$ cannot compromise any underlying plaintext sent between HgNB and gNB, and all messages sent to and from π_b are random strings that are independent of the bit b sampled by the challenger. Thus it follows that $\mathcal{A}$ has no advantage in guessing the bit b : $\mathbf{Adv}_{G_{A_6}} = 0$.

4.3.2.2 Unlink-security of Intra-region Handover

Here, we present our formal analysis and results for the Unlink-security of the intra-region handover protocol.

Theorem 5 Unlink-***security of Intra-region Handover***. *The intra-region handover protocol depicted in Figure 4.3 is unlinkable under the cleanness predicate defined in Def.28. For any PPT algorithm $\mathcal{A}$ against the* Unlink *experiment,* $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of SanSig, IND-CPA security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the MA security experiment, that $\mathcal{A}$ cannot issue a $\mathbf{CorruptLTK}(i)$ query before a session π^s_i accepts such that $\mathcal{C}$ terminate the game and outputs 1, nor can it issue a $\mathbf{StateReveal}(i, s)$, nor a $\mathbf{StateReveal}(\text{HgNB}, s)$ query (where $\pi^{s'}_{\text{HgNB}}$ received messages from π^s_i). Next, we proceed via a sequence of games. We bound the adversary advantage difference of winning these games with the underlying cryptographic assumptions till the adversary reaches a game where the advantage of that game equals 0, which shows that adversary A cannot win.

Game B_0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{B_0}}$.

Game B_1: Here we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under $pk^{CN}_{sig} \& pk^{HgNB}_{san}$. At the beginning of this game, we initialize a SanSig challenger which outputs $pk^{\mathcal{C}}_{sig} \& pk^{\mathcal{C}}_{san}$, which we embed into CN and HgNB. Then every time CN or HgNB needs to generate a signature, we query $\mathsf{SanSig.Sign}$ to sign a message m. Next, we define the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ wins is bounded by the EUFCMA security of *SanSig*, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_{B_0}} \leq \mathbf{Adv}_{G_{B_1}} + \mathbf{Adv}^{EUFCMA}_{SanSig}(\mathcal{A})$.

Game B_2: In this game, we guess the first session π_i^s to accept without a matching partner, such that $\pi_i.role = UE$. We also introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_i^s, i.e. session π_i^s receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by Game B_1 we abort if $\mathcal{A}$ generates a valid signature over any message m, this introduces no additional bound. Since there are n_P parties running at most n_S sessions, the probability of session π_i^s accepts without a matching partner is: $\mathbf{Adv}_{G_{B_1}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{B_2}}$.

Game B_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game B_3 from Game B_2 without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{B_2}} \leq \mathbf{Adv}_{G_{B_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game B_4: In this game the challenger replaces the encryption and session keys sk_i, sk_i' with uniformly random values $\hat{sk}_i, \hat{sk}_i'$ by interacting with a KDF challenger. Since $sk_i, sk_i' \leftarrow \mathsf{KDF}(g^c)$ and by **Game** B_3 g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{B_3}} \leq \mathbf{Adv}_{G_{B_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game B_5: In this game, we replace the computation of the ciphertext $c = \mathsf{Enc}(sk_i, \mathbb{C}_U \| \sigma_U \| \omega_U \| v)$ with $\hat{c} = \mathsf{Enc}(\hat{sk}_i, rand)$, where $rand \xleftarrow{\$} \{0,1\}^L$ and $L = |\mathbb{C}_U \| \sigma_U \| \omega_U \| v|$, and the computation of the ciphertext $c^* = \mathsf{Enc}(sk_i, \mathbb{C}_U^* \| \sigma_U^* \| \omega_U^* \| v^*)$ with $\hat{c^*} = \mathsf{Enc}(\hat{sk}_i, rand^*)$ where $rand^* \xleftarrow{\$} \{0,1\}^{L^*}$ and $L^* = |\mathbb{C}_U^* \| \sigma_U^* \| \omega_U^* \| v^*|$. We do so by interacting with an AE challenger whenever $\hat{sk}_i$ is used by the challenger to encrypt a message, and issuing either an Enc oracle call $(\mathbb{C}_U \| \sigma_U \| \omega_U \| v, rand)$ or $(\mathbb{C}_U^* \| \sigma_U^* \| \omega_U^* \| v^*, rand^*)$. Note that if the bit b sampled by AE is 0, then we are in **Game** B_4, and otherwise we are in **Game** B_5. Since $\hat{sk}_i$ is uniformly random and independent, this change is sound, and adds advantage bound by the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{B_4}} \leq \mathbf{Adv}_{G_{B_5}} + \mathbf{Adv}^{IND-CPA}_{\mathsf{AE}}(\mathcal{A})$.

Game B_6: In this game we highlight that the channel between the HgNB and the gNB is assumed to be secure, thus $\mathcal{A}$ cannot compromise any underlying plaintext sent between HgNB and gNB, and all messages sent to and from π_b are uniformly random and independent of the bit b sampled by the challenger. Thus it follows that $\mathcal{A}$ has no advantage in guessing the bit b : $\mathbf{Adv}_{G_{B_6}} = 0$.

4.3.2.3 Unlink-security of Inter-region Handover

Here, we present our formal analysis and results for the Unlink-security of the inter-region handover protocol.

Theorem 6 Unlink-***security of Inter-region Handover***. *The inter-region handover protocol depicted in Figure 4.4 is unlinkable under the cleanness predicate defined in Def.28. For*

any PPT algorithm $\mathcal{A}$ against the Unlink *experiment,* $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of SanSig, IND-CPA security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: First, we recall that in order to win the Unlink-security experiment, that $\mathcal{A}$ cannot issue a $\mathbf{CorruptLTK}(i)$ query before a session π^s_i accepts such that $\mathcal{C}$ terminate the game and outputs 1, nor can it issue a $\mathbf{StateReveal}(i,s)$, nor a $\mathbf{StateReveal}(\text{HgNB},s)$ query (where $\pi^{s'}_{\text{HgNB}}$ received messages from π^s_i). As before, we proceed via a sequence of games.

Game C_0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{C_0}}$.

Game C_1: In this game, we introduce an abort event, where the challenger aborts if $\mathcal{A}$ produces a valid signature σ that verifies under hpk^{CN}_{sig} and pk^{HgNB}_{san} or gpk^{CN}_{sig} and pk^{gNB}_{san}. At the beginning of this game, we initialise a pair of SanSig challengers which output $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into CN, HgNB and gNB. Then every time CN, HgNB or gNB needs to generate a signature, we query SanSig.Sign to sign a message m. Now, we trigger the abort event that occurs whenever $\mathcal{A}$ produces a valid signature. Thus, the probability that $\mathcal{A}$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_{C_0}} \leq \mathbf{Adv}_{G_{C_1}} + 2 \cdot \mathbf{Adv}^{\text{EUFCMA}}_{\mathsf{SanSig}}(\mathcal{A})$.

Game C_2: In this game, we guess the first session π^s_i to accept without a matching partner, such that $\pi^s_i.role = \text{UE}$. We also introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public key share g^h to the session π^s_i, i.e. session π^s_i receives g^h that is not from an honest HgNB. Since this requires a signature over g^h, and by **Game** C_1 we abort if $\mathcal{A}$ generates a valid signature over any message m, this introduces no additional bound. Since there are n_P parties running at most n_S sessions, this introduces the following bound: $\mathbf{Adv}_{G_{C_1}} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_{C_2}}$.

Game C_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from a DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus any $\mathcal{A}$ that can distinguish **Game** C_2 from **Game** C_3 can break the DDH assumption, as formalised in Def.10. Thus it follows that: $\mathbf{Adv}_{G_{C_2}} \leq \mathbf{Adv}_{G_{C_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game C_4: In this game, the challenger replaces the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ by interacting with a KDF challenger. Since $sk_i, k_s \leftarrow \mathsf{KDF}(g^c)$ and by **Game** C_3 g^c is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game** C_3 from **Game** C_4 can be used to break KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{C_3}} \leq \mathbf{Adv}_{G_{C_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game C_5: In this game, we replace the computation of the ciphertext $c = \mathsf{Enc}(sk_i, \mathbb{C}_U \| \sigma_U \| \omega_U \| v)$ with $\hat{c} = \mathsf{Enc}(\hat{sk_i}, rand)$, where $rand \xleftarrow{\$} \{0,1\}^L$ and $L = |\mathbb{C}_U \| \sigma_U \| \omega_U \| v|$, and the computation of the ciphertext $c^* = \mathsf{Enc}(sk_i, \mathbb{C}^*_U \| \sigma^*_U \| \omega^*_U \| v^*)$ with $\hat{c^*} = \mathsf{Enc}(\hat{sk_i}, rand^*)$

where $rand^* \xleftarrow{\$} \{0,1\}^{L^*}$ and $L^* = |\mathbb{C}_U^* \| \sigma_U^* \| \omega_U^* \| v^*|$. We do so by interacting with an encryption challenger whenever $\hat{sk}_i$ is used by the challenger to encrypt a message, and issuing either an Enc oracle call $(\mathbb{C}_U \| \sigma_U \| \omega_U \| v, rand)$ or $(\mathbb{C}_U^* \| \sigma_U^* \| \omega_U^* \| v^*, rand^*)$. Note that if the bit b sampled by the challenger is 0, then we are in **Game** C_4, and otherwise we are in **Game** C_5. Since $\hat{sk}_i$ is (by **Game** C_5) uniformly random and independent, this change is sound. Any adversary $\mathcal{A}$ that can distinguish between **Game** C_4 and **Game** C_5 can be used to break the security of AE, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{C_4}} \leq \mathbf{Adv}_{G_{C_5}} + \mathbf{Adv}_{\mathsf{AE}}^{IND-CPA}(\mathcal{A})$.

GameC_6: In this game, we highlight that the channel between the HgNB and the gNB is assumed to be secure, thus $\mathcal{A}$ cannot compromise any underlying plaintext sent between HgNB&gNB, and all messages sent to and from π_b are uniformly random and independent of the bit b sampled by the challenger. Thus it follows that $\mathcal{A}$ has no advantage in guessing the bit b, and summing the probabilities $\mathcal{A}$ has a negligible advantage in winning the Unlink game. Thus:$\mathbf{Adv}_{G_{C_6}} = 0$

4.3.3 Key Indistinguishability Security

In this section, we analyse the key indistinguishability (KIND) security of our proposed scheme, demonstrating that they achieve KIND-security. We build upon the general KIND-security model introduced in Chapter 2, Section 2.3.3. Recall that in the KIND-security game, an adversary $\mathcal{A}$ attempts to distinguish between a real session key and a random key with a non-negligible advantage. Here, we define the following specific adversary query and cleanness predicate:

Adversary Queries. In addition to the **Create**, **Send**, **CorruptLTK** and **StateReveal** queries listed in Section 4.3.1, we define one additional query **Test** that allows the adversary $\mathcal{A}$ to either the real key derived by a **clean** session during the execution of the KIND experiment, or a random key sampled from the same distribution (depending on the challenge bit b).

- **Test**$(i, s) \rightarrow m$: When $\mathcal{C}$ receives a **Test** query, if **Test** has already been issued, $\pi_i^s.\alpha = \texttt{accepted}$, or π_i^s is not **clean**, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ sets $k_0 \leftarrow \pi_i^s.k$, and $k_1 \xleftarrow{\$} \{0,1\}^\lambda$, and returns k_b to $\mathcal{A}$ (where b was sampled by $\mathcal{C}$ at the beginning of the experiment).

Cleanness Predicate. Here, it is important to note that distinguishing a real session key becomes trivial for the $\mathcal{A}$ if they have compromised the long-term key, as the adversary can generate valid session keys using the compromised key. To mitigate this type of attack, we introduce a cleanness predicate, which ensures that the adversary is prevented from issuing a **CorruptLTK** or **StateReveal** query before the session's acceptance.

Definition 29 (Cleanness predicate) *A session π_i^s in the* KIND *experiment, described in Section 2.3.3, is* **clean** *if* ***CorruptLTK***(i) *was not issued before $\pi_i^s.\alpha = \texttt{accepted}$,* and ***StateReveal***$(i, s)$ *was not issued* and *for all j, s' such that $\pi_j^{s'}.msg_r = \pi_i^s.msg_s$,* ***StateReveal***$(j, s')$ *was not issued.*

4.3.3.1 KIND-security of Initial Authentication protocol

Here, we present our formal analysis and results for the KIND-security of the initial authentication protocol.

Theorem 7 KIND-*security of Initial Authentication*. *The initial authentication protocol depicted in Figure 4.2 achieves* KIND-*security under the cleanness predicate defined in Def.29. For any PPT algorithm $\mathcal{A}$ against the* KIND *experiment,* $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: We proceed via a sequence of games. We bound the adversary advantage difference of winning these games with the underlying cryptographic assumptions until the adversary reaches a game where the advantage of that game equals 0, which shows that adversary $\mathcal{A}$ cannot win.

Game A_0: This is the original Key Indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{A_0}}$.

Game A_1: In this game, we introduce an abort event that triggers if $\mathcal{C}$ guesses a session $\pi^{s^*}_{i^*}$ and $\mathcal{A}$ issues a **Test** query to a session π^s_i where $i \neq i^*$ and $s \neq s^*$. This gives us at most $(n_S \cdot n_P)$ UEs sessions, with probability at least:$\mathbf{Adv}_{G_{A_0}} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_{A_1}}$.

Game A_2: Here, we introduce a new abort event, if the test session π^s_i accepts any message from a non-honest CN / UE. This exactly matches the MA-experiment, where the $\mathcal{A}$ attempts to inject messages from CN or UE. Thus, this game is bounded by the probability of $\mathcal{A}$ breaking the MA-security of the Initial authentication protocol, and thus: $\mathbf{Adv}_{G_{A_1}} \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}_{G_{A_2}}$

Game A_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game A_3 from Game A_2 without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{A_2}} \leq \mathbf{Adv}_{G_{A_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game A_4: In this game the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk}_i, \hat{sk}'_i$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** A_3 g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{A_3}} \leq \mathbf{Adv}_{G_{A_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game A_5: Here we emphasise that as a result of these changes, the session key $\hat{sk}'_i$ is now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_{A_5}} = 0$.

4.3.3.2 KIND-security of Intra-region Handover

Here, we present our formal analysis and results for the KIND-security of the intra-region handover protocol.

Theorem 8 KIND-*security of Intra-region Handover*. *The intra-region handover protocol described in Figure 4.3 is* KIND*-secure under the cleanness predicate defined in Def.32. For any PPT algorithm $\mathcal{A}$ against the* KIND *experiment,* $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: We proceed via a sequence of games. We bound the adversary advantage difference of winning these games with the underlying cryptographic assumptions until the adversary reaches a game where the advantage of that game equals 0, which shows that adversary $\mathcal{A}$ cannot win.

Game B_0: This is the original KIND experiment defined in Def.25 of Chapter 2: $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{B_0}}$.

Game B_1: In this game, we introduce an abort event that triggers if $\mathcal{C}$ guesses a session $\pi^{s^*}_{i^*}$ and $\mathcal{A}$ issues a **Test** query to a session π^s_i where $i \neq i^*$ and $s \neq s^*$. Since there are at most $(n_S \cdot n_P)$ such sessions, this introduces the bound: $\mathbf{Adv}_{G_{B_0}} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_{B_1}}$.

Game B_2: Here, we introduce a new abort event if the test session π^s_i accepts any message from a non-honest HgNB / UE. This exactly matches the MA-experiment, where the $\mathcal{A}$ attempts to inject messages from HgNB or UE. Thus, this game is bounded by the probability of $\mathcal{A}$ breaking the MA-security of the Intra-region HO protocol, and thus: $\mathbf{Adv}_{G_{B_1}} \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}_{G_{B_2}}$

Game B_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus any $\mathcal{A}$ that can distinguish **Game** B_2 from **Game** B_3 can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{B_2}} \leq \mathbf{Adv}_{G_{B_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game B_4: In this game, the challenger replaces the encryption and session keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ by interacting with a KDF challenger. Since $sk_i, k_s \leftarrow \mathsf{KDF}(g^c)$ and by **Game** B_3 g^c is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game** B_3 from **Game** B_4 can be used to break KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{B_3}} \leq \mathbf{Adv}_{G_{B_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game B_5: We highlight that as a result of these changes, the session key $\hat{sk_i}$ is now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_{B_5}} = 0$.

4.3.3.3 KIND-security of Inter-region Handover

Here we present our formal analysis and results for the KIND-security of the inter-region handover protocol.

Theorem 9 KIND-*security of Inter-region Handover*. *The inter-region handover protocol depicted in Figure 4.4 achieves* KIND*-security under the cleanness predicate defined in Def.29. For any PPT algorithm $\mathcal{A}$ against the* KIND *experiment,* $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig *the KDF security of* KDF *and the* DDH *assumption.*

Proof: Next, we proceed via a sequence of games. We bound the adversary advantage difference of winning these games with the underlying cryptographic assumptions till the adversary reaches a game where the advantage of that game equals 0, which shows that adversary A cannot win.

Game C_0: This is the original Key Indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_{C_0}}$.

Game C_1: In this game, we introduce an abort event that triggers if $\mathcal{C}$ guesses a session $\pi^{s^*}_{i^*}$ and $\mathcal{A}$ issues a **Test** query to a session π^s_i where $i \neq i^*$ and $s \neq s^*$. This gives us at most $(n_S \cdot n_P)$ UEs sessions, with probability at least: $\mathbf{Adv}_{G_{C_0}} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_{C_1}}$.

Game C_2: Here we introduce a new abort event, if the test session π^s_i accepts any message from a non-honest HgNB / UE. This exactly matches the MA-experiment, where the $\mathcal{A}$ attempts to inject messages from HgNB or UE. Thus, this game is bounded by the probability of $\mathcal{A}$ breaking the MA-security of the Inter-region HO protocol, and thus: $\mathbf{Adv}_{G_{C_1}} \leq \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}_{G_{C_2}}$

Game C_3: In this game, we replace g^h, g^{r_u} and g^{hr_u} computed honestly in the protocol execution with g^a, g^b, g^c respectively, from DDH challenger. By the definition of Decisional Diffie-Hellman, a, b, c are sampled uniformly at random from $\mathbb{Z}_q$, and independent of the protocol execution. Thus the adversary cannot distinguish Game C_3 from Game C_2 without breaking DDH, which adds additional advantage bound by the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{C_2}} \leq \mathbf{Adv}_{G_{C_3}} + \mathbf{Adv}^{DDH}_{\mathsf{DDH}}(\mathcal{A})$.

Game C_4: In this game the challenger replaces the encryption and session keys sk_i, sk'_i with uniformly random values $\hat{sk}_i, \hat{sk}'_i$ by interacting with a KDF challenger. Since $sk_i, sk'_i \leftarrow \mathsf{KDF}(g^c)$ and by **Game** C_3 g^c is already uniformly random and independent, this change is sound and introduces additional advantage bound by the KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{C_3}} \leq \mathbf{Adv}_{G_{C_4}} + \mathbf{Adv}^{KDF}_{\mathsf{KDF}}(\mathcal{A})$.

Game C_5: Here we emphasise that as a result of these changes, the session key $\hat{sk}'_i$ is now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_{C_5}} = 0$.

4.4 Performance Evaluation and Comparison

In this section, we conduct a comprehensive evaluation of the proposed scheme in comparison to existing state-of-the-art protocols [4, 37, 71, 69]. Our analysis encompasses three key aspects: feature set comparison, computational overhead, and communication cost. This multifaceted approach assesses the scheme's efficacy and efficiency relative to current standards in the field.

4.4.1 Security Features Comparison

Our analysis compares the proposed scheme with existing protocols based on the desired security and privacy features discussed in Chapter 3, Section 3.4.1. These features are crucial for overcoming several security and privacy issues found in the current version of 5G protocols and are essential for any AKA and HO protocols in cellular networks. Table 4.2 provides a comparative analysis of the achieved security and privacy properties in the proposed scheme against state-of-the-art protocols [4, 37, 71, 69]. This comparison reveals that existing protocols fail to simultaneously guarantee all required security features, particularly Unlink and Perfect Forward Secrecy (PFS).

While most previous works achieve mutual authentication and anonymity, some present challenges in subscriber identification. For instance, LSHA protocol [69] lacks a clear mechanism for the system to identify registered subscribers during HO. During the execution of the HO scheme, the UE does not send any identifier to the system (gNB), and there are no parameters that can help the system identify the user. Consequently, the system must resort to trial decryption with all maintained keys to recover the plaintext of the UE, an impractical and time-consuming process when a gNB may interact with thousands of UE requests.

Among the analysed protocols, only ReHand [37] and our proposed scheme achieve Unlink. However, ReHand exhibits limitations in other critical areas, notably the absence of secure inter-region HO support and failure to achieve PFS. Indeed, full PFS achievement is absent in all previous works examined, including the conventional 5G-AKA protocol, which also lacks Unlink.

This deficiency in the 5G-AKA protocol has significant implications for schemes utilizing it for initial authentication. Specifically, RUSH [71] and LSHA [69], both employing the conventional 5G-AKA protocol in their initial protocol, can only attain partial PFS, which is denoted by $\circledcirc$ in Table 4.2. While these schemes ensure PFS in their HO protocols, their reliance on the standard 5G-AKA for initial authentication precludes full PFS and Unlink guarantees in the overall system.

Concerning secure inter-region HO, a key contribution of this study, RUSH [71] stands as the sole existing protocol among those analysed to support this functionality. However, RUSH's implementation necessitates blockchain infrastructure, introducing additional complexity. In contrast, this study presents a method for achieving secure privacy-preserving inter-region HO without blockchain dependency. Furthermore, RUSH exhibits deficiencies in Unlink. Its HO preparation phase requires users to broadcast a Chameleon hash value (CH_{UE}) to neighbouring Access Points (APs). The CH_{UE} value in RUSH remains constant across sessions for each user. This invariability allows adversaries to track user movements by correlating this persistent identifier across different access points and times, thus compromising unlinkability and user privacy.

Table 4.2: *Features comparison.*

Features	MA	UA	Unlink	PFS	SRM	ERH	PM
5G [4]	✓	✓	✗	✗	✗	✗	CF
ReHand[37]	✓	✓	✓	✗	✓	✗	CF
RUSH [71]	✓	✓	✗	◎	✗	✓	BAN+AVISPA
LSHA [69]	✓	N/P	✗	◎	✗	✗	BAN+Scyther
Ours	✓	✓	✓	✓	✓	✓	CF

MA:Mutual Authentication, **UA**:User Anonymity,
Unlink:Unlinkability, **PFS**:Perfect Forward Secrecy, **SRM**:Secure
Revocation Management, **ERH**:Secure Inter-region HO supports,
PM: Proof Method, **N/P**: no information provided, ◎: Partial
CF: Computational Formal security analysis

Our scheme also incorporates an efficient user revocation mechanism, which is lacking in some previous works like [71]. While ReHand [37] uses Nyberg's accumulator based on symmetric hash functions, our proposed scheme employs an efficient asymmetric-key-based accumulator, offering better scalability for managing a significant number of users. We provide comprehensive formal security proof to validate our claims, demonstrating that our proposed scheme achieves all desirable features. This approach offers more robust validation compared to the BAN logic, used in RUSH [71] and LSHA [69], which has known weaknesses in protocol idealization [25] and limitations in proving privacy-related features [60]. Lastly, our scheme avoids relying on timestamps to prevent replay attacks, eliminating the need for continuous synchronization of distributed clocks across cells, a challenge in some previous works like RUSH.

4.4.2 Computational Cost

This section evaluates and compares the performance of the proposed scheme with existing related works in terms of computational cost. It is assumed that all aggregated network entities (CN, HgNB, gNB) possess higher computational capabilities than the UE. For a fair comparison between related existing work and our proposed scheme, we conducted a theoretical computation analysis by implementing each cryptographic primitive. Simulations of these cryptographic operations employed by various schemes (including our proposals) were conducted on a Dell Inspiron machine with an i7 core, 2.30GHz CPU and 16.0 GB RAM (operating as the aggregated network entities per the scheme). To measure the computational cost at UE, we employ a smartphone running the Android-10 mobile operating system, equipped with octa-core 1.8GHz Quad-Core ARM Cortex-A55, 2.7GHz Quad-Core Mongoose M3 processors and 6GB RAM. Implementing the required cryptographic operations for our proposed scheme and the related works utilises the Java Cryptography Extension (JCE) [66] libraries.

Table 4.3 presents the computation cost of the underlying cryptographic primitives, which serves as the basis for measuring the overall computational cost of the protocols. In implementing sanitizable signatures, we adopted the methodology proposed by [29], employing a nested RSA signature scheme comprising both outer and inner signature components. This

Table 4.3: *Time costs of cryptography operations.*

Notation	T_{UE}(ms)	T_{Sys}(ms)	**Notation**	T_{UE}(ms)	T_{Sys}(ms)
T_{DH}	0.18	0.09	T_{MAC}	0.195	0.071
T_{SM}	1.148	0.235	T_{H}	0.402	0.089
T_{ELG}	1.176	0.648	T_{PRG}	0.467	0.1273
T_{E}	0.859	0.340	T_{AES}	0.559	0.385
$T_{S.ver}$	0.084	0.046	T_{Mod}	0.0021	0.001
$T_{S.sign}$	1.43	0.87	$T_{S.san}$	.705	0.384
$T_{Nwit.gen}$	2.29	1.12	$T_{Nwit.ver}$	4.01	2.23
T_{KDF}	0.063	0.022			

T_{UE}: Computational time on user device, T_{Sys}: Computational time on system, T_{DH}: Elliptic Curve Diffie-Hellman operation, T_{SM}: scalar multiplication, T_{E}: exponentiation operation, T_{MAC}: Massage authentication operations(Hmac-SHA256), T_{H}: Hash operations(SHA-256), T_{PRG}: Random number generators, T_{AES}:Symmetric encryption/decryption operations, T_{ELG}: Elgamal Asymmetric encryption/decryption operations, T_{Mod}: Modular operations, $T_{S.sign}, T_{S.san}, T_{S.ver}$: sanitizable signature, sanitisation and verification operations, $T_{Nwit.gen}, T_{Nwit.ver}$: Non-witness generation and verification T_{KDF}: Key Derivation Function

framework defines $T_{S.sign}$ as the cumulative time required to generate the inner and outer signatures. Conversely, $T_{S.san}$ represents the time allocated solely to the outer signature generation, which aligns with the conceptual sanitisation process. The verification procedure, denoted by $T_{S.ver}$, encompasses validating both signature layers. Concerning the accumulator implementation, we have adopted a simplified variant of the dynamic universal accumulator scheme proposed by Li, Li, and Xue [50], which is based on RSA cryptographic assumptions. Appendix A contains the complete code implementation and a snapshot of the output.

The initial authentication computation costs, as presented in Table 4.4, demonstrate our proposed protocol's efficiency compared to existing solutions. Our approach exhibits T_{UE} costs of $\approx$ 4.72 ms and T_{Sys} costs of $\approx$ 4.832 ms for initial authentication. Whilst these numbers are slightly higher than some conventional methods, such as the 5G-AKA [4] protocol ($\approx 2.03ms$ for T_{UE} and $\approx 1.09ms$ for T_{Sys}), they remain competitive within the broader landscape. Notably, our protocol outperforms RUSH [71] in T_{UE} costs ($\approx 8.95ms$) and shows comparable efficiency to ReHand [37] in T_{UE} costs ($\approx 3.190ms$). This suggests that our protocol achieves a balanced approach to initial authentication, maintaining competitive computational costs whilst incorporating enhanced security features.

Regarding handover computation costs, Table 4.5 illustrates the efficiency of our protocol across various scenarios. For intra-region handovers, our solution demonstrates notable performance, with T_{UE} costs of $\approx 1.926ms$, surpassing the efficiency of all existing solutions including conventional methods ($\approx 2.366ms$). In inter-region scenarios, our protocol exhibits costs of $\approx 2.010ms$ for T_{UE} and $\approx 4.53ms$ for T_{Sys}, showing marked improvement over RUSH [58] ($\approx 5.42ms$ for T_{UE}). Whilst our T_{Sys} costs for intra-region handovers ($\approx 4.802ms$) are marginally higher than some alternatives, they remain within an acceptable range defined by ETSI [3] of 1 ms to 100 ms. These figures collectively underscore the computational effi-

Table 4.4: *Performance comparison based on computational cost (Initial authentication).*

Protocol	**Entity**	**Initial authentication**	**Total time (ms)**
Conventional 5G-AKA[4]	T_{UE}	$T_{PRG} + T_{MAC} + 3T_{KDF} + T_{ELG}$	≈ 2.03
	T_{Sys}	$T_{PRG} + 1T_{MAC} + 2T_H + T_{ELG} + 3T_{KDF}$	≈ 1.09
ReHand[37]	T_{UE}	$2T_{AES} + 4T_H + T_{PRG}$	≈ 3.190
	T_{Sys}	$3T_{AES} + 5T_H + T_{PRG}$	≈ 1.75
RUSH[71]	T_{UE}	$4T_{PRG} + T_{MAC} + 3T_{KDF} + 3T_H + T_E$ $+5T_{Mod} + 3T_{SM} + T_{ELG}$	≈ 8.95
	T_{Sys}	$2T_{PRG} + T_{MAC} + 3T_H + T_{ELG}$ $+3T_{KDF} + 2T_{SM} + 3T_{Mod}$	≈ 1.78
LSHA[69]	T_{UE}	$T_{PRG} + T_{MAC} + 3T_{KDF} + T_{ELG}$	≈ 2.03
	T_{Sys}	$T_{PRG} + 1T_{MAC} + 2T_H + T_{ELG} + 3T_{KDF}$	≈ 1.09
OURs	T_{UE}	$2T_{S.ver} + +2T_{PRG} + T_{KDF} + 6T_{AES} + T_{DH}$	≈ 4.72
	T_{Sys}	$3T_{PRG} + T_{KDF} + T_{S.san} + T_{S.sign}$ $+T_{Nwit.gen} + 5T_{AES} + T_{DH}$	≈ 4.832

ciency of our protocol in handover scenarios, particularly when considering the comprehensive security features it offers.

In conclusion, our proposed scheme balances computational efficiency and enhanced security competitively. For initial authentication, it shows comparable costs to existing solutions, while in handover scenarios, it demonstrates superior efficiency, particularly for intra-region and inter-region UE costs. Although system costs for intra-region handovers are marginally higher, the overall performance remains competitive. Notably, our scheme introduces a novel secure inter-region handover solution without relying on a central party or blockchain technology. This approach offers an efficient, scalable solution, marking a significant advancement in secure mobile communications by effectively balancing efficiency and security.

4.4.3 Communication Cost

This section presents analysis and comparison of our proposed scheme with existing handover protocols, including the conventional 5G [4], ReHand [37], RUSH [71], and LSHA [69], in terms of communication cost. To ensure an accurate evaluation, we consider both the propagation and transmission time of the message size, as well as the network's data rate, to measure the transmission delay across all protocols. In accordance with the 3GPP specification [4], we assume a wide-area scenario with an uplink data rate of 25 Mbps and a downlink rate of 50 Mbps. The transmission delay for all protocols is measured using the platforms as mentioned earlier and the relevant message sizes of the proposed protocol, as listed in Table 4.6. The table enumerates the sizes of various cryptographic components the proposed protocol, which are instrumental in computing transmission delays within the 5G network. The certificate, comprising a region-user ID, region ID, zone user ID, and user subscription validity period—each occupying 8 bytes—totals 32 bytes. The sanitizable signature is composed of two nested RSA signatures, each determined by the key size of 256-byte, resulting in a 256-byte signature. For the Elliptic Curve Diffie-Hellman (ECDH), the secp256r1 (NIST

Table 4.5: *Performance comparison based on computational cost (Handover).*

Protocol	**Type**	**Entity**	**Handover Cost**	**Total time (ms)**
Conventional 5G-HO[4]	Intra region	T_{UE}	$4T_{AES} + 2T_{KDF}$	≈ 2.366
		T_{Sys}	$4T_{AES} + 2T_{KDF}$	≈ 1.60
ReHand [37]	Intra region	T_{UE}	$T_{PGR} + 3T_H + T_{AES}$	≈ 2.23
		T_{Sys}	$T_{PRG} + 5T_H + 2T_{AES}$	≈ 1.359
RUSH [71]	Inter region	T_{UE}	$3T_{PRG} + T_{SM} + 5T_H + T_E + T_{Mod}$	≈ 5.42
		T_{Sys}	$2T_{PRG} + T_{SM} + 6T_H + T_E + T_{Mod}$	≈ 1.376
LSHA [69]	Intra region	T_{UE}	$T_{PRG} + T_{MAC} + T_H + 5T_{AES} + T_{KDF}$	≈ 3.92
		T_{Sys}	$6T_{PRG} + T_{MAC} + T_H + 6T_{AES} + T_{ELG} + T_{KDF}$	≈ 3.95
OURs	Intra region	T_{UE}	$T_{S.ver} + T_{PRG} + T_{KDF} + T_{DH} + 2T_{AES}$	≈ 1.926
		T_{Sys}	$T_{PRG} + T_{DH} + T_{S.san} + 2T_{AES} + T_{KDF} + T_{S.ver} + T_{Nwit.ver} + T_{Nwit.gen}$	≈ 4.802
	Inter region	T_{UE}	$2T_{S.ver} + T_{PRG} + T_{KDF} + T_{DH} + 2T_{AES}$	≈ 2.010
		T_{Sys}	$2T_{PRG} + T_{DH} + T_{KDF} + 2T_{S.san} + T_{S.ver} + T_{Nwit.ver} + T_{Nwit.gen}$	≈ 4.53

P-256) curve is utilised [65]. The public key shared between participants is 68 bytes, encompassing x and y coordinates plus metadata, whilst the resulting shared secret (session key- k_s) is 32 bytes, derived from the x-coordinate of the computed point. This size discrepancy is because the public key represents a full curve point. Lastly, the non-membership witness occupies 32 bytes. These parameter lengths collectively inform the communication and transmission requirements of the protocol. To calculate the propagation delay, we utilise the spectrum wave speed and the distance between the user and the nearest connected gNB. The 3GPP specification [4] indicates that in wireless communication, the propagation wave speed is 3×10^8 m/s, with an approximate maximum cell size of 200 m in 5G networks. Consequently, we derive a propagation delay of a single message, which is $\approx 0.67\mu s$.

Table 4.7 presents a comparative analysis of the communication cost between the related works. It is evident that our proposed scheme introduces additional communication overhead compared to the existing state-of-the-art protocols [4],[37],[71],[69]. This increased overhead is attributable to the transmission of a user certificate, along with its signature, to the communicating partner for mutual authentication purposes. Each certificate and signature occupy approximately 288 bytes. Notwithstanding this additional overhead, the combination of signatures and certificates enables our scheme to achieve inter-region handover without requiring assistance from the CN. This approach effectively reduces the overall communication and computation burden on the CN. Furthermore, the utilization of the SanSig algorithm for signing certificates allows our scheme to maintain user privacy throughout the roaming process, eliminating the need for re-authentication.

In conclusion, while our proposed scheme introduces some additional communication overhead, the benefits in terms of enhanced security, reduced CN dependency, and improved user privacy offer a compelling trade-off in the context of next-generation mobile networks.

Table 4.6: *Length of parameters.*

Parameters	**Size (in bytes)**
$\mathbb{C}_U : [\text{RUID}\|\|\text{RID}\|\|\text{ZUID}_i\|\|T_U]$	$8 \times 4 = 32$
SanSig Signature(σ)	256
ECDH public key size	68
Non-membership witness size	32

Table 4.7: *Performance comparison based on communication cost.*

Protocol	**Link**	**Total message size (bits)**	**Transmission time (ms)**	**Propagation time (ms)**	**Total time (ms)**
Conventional 5G-HO[4]	UP	640	0.0256	0.00201	0.02764
	Down	256	0.00512	0.00133	0.00646
ReHand[37]	UP	832	0.03328	0.00134	0.03462
	Down	768	0.01536	0.00067	0.0398
RUSH[71]	UP	896	0.03584	0.00133	0.03717
	Down	896	0.01792	0.00067	0.01859
LSHA[69]	UP	256	0.01024	0.00067	0.01091
	Down	387	0.00774	0.00133	0.00907
OURs (Intra-Region HO)	UP	3,104	0.12416	0.00067	0.12483
	Down	3,104	0.06208	0.00134	0.06342
OURs (Inter-Region HO)	UP	3,104	0.12416	0.00067	0.12483
	Down	5,408	0.10816	0.00134	0.1095

4.5 Summary

This chapter has made significant contributions to secure authentication and handover protocols in 5G SCN roaming environments, successfully addressing the first research question ($\mathbf{RQ}_1$) of this thesis. In response to this question, we have developed the *first* solution for secure, privacy-preserving inter-region 5G handover for roaming users, addressing a critical gap in the existing literature. Our approach ensures seamless user mobility within and between regions and employs an efficient user membership revocation scheme.

The efficacy of our proposed solution has been validated through rigorous formal security analysis and comparative studies. These evaluations have demonstrated the scheme's effectiveness in achieving MA, UA,Unlink, PFS, SRM and ERH. Furthermore, the analysis has shown our approach's computational and communicational efficiency relative to existing methods while highlighting the comprehensive security features achieved.*

While these contributions improve the authentication and handover protocols, they also reveal opportunities for further enhancement. The next chapter will build upon this foundation and introduce a more robust security framework. Specifically, we will address two critical aspects: Key Compromise Impersonation (KCI) resilience [32] and a Key Escrow-Free (KEF)

*This work has undergone peer review and has been accepted for publication in the ***IEEE Transactions on Information Forensics and Security*** in 2023.

approach [7]. These additions will further strengthen the system against potential vulnerabilities, enhancing its overall security posture. Moreover, our research will evolve from focusing on intra- and inter-region handovers to proposing a universal handover approach. The universal approach enables seamless, secure transitions across any network domain, regardless of geographical boundaries. By addressing these critical aspects of security, privacy, and universality, the subsequent chapter seeks to elevate our solution to new heights of efficacy and reliability in the realm of 5G security. We will explore how this universal approach enhances user experience through truly borderless connectivity, simplifies network management, and bolsters overall network security and privacy. This progression from a region-based to a universal model represents a natural evolution in our research, promising to deliver a more comprehensive and secure solution for the 5G environment.

Chapter 5

UniHand: Privacy-Preserving Universal Handover for 5G

The previous chapter introduced a region-based handover protocol that addressed significant security and privacy challenges in 5G Small Cell Networks (SCNs). Building upon that foundation, this chapter presents a more comprehensive solution that enhances security and privacy features and introduces a universal handover approach. Supporting universal handover eliminates regional boundaries, expands the coverage area, and allows users to move seamlessly across wider geographical zones.

This chapter introduces the UniHand scheme, a new approach to authenticated key exchange and handover protocols in 5G environments. UniHand is designed to achieve a set of security and privacy objectives, including Mutual Authentication (MA), User Anonymity (UA), unlinkability (Unlink), Perfect Forward Security (PFS), and Key Escrow-Free (KEF). Most importantly, UniHand is also resilient against Key Compromise Impersonation (KCI) attacks, a significant advancement in 5G security that has not been adequately addressed in existing protocols.

Ensuring protection against KCI attacks is paramount in 5G environments. In a KCI attack, an adversary who has compromised the private key of an entity can impersonate other entities to the compromised one [32]. This vulnerability is particularly dangerous in the context of 5G SCNs, where the proliferation of connected devices increases the attack surface and the potential impact of a successful compromise. A KCI attack on user equipment (UE), for instance, could allow an adversary to impersonate legitimate network entities in the compromised UE, potentially leading to man-in-the-middle attacks, data interception, and privacy breaches for that user. By providing KCI resilience, UniHand significantly enhances the overall security 5G networks, ensuring that even if an entity is compromised, the integrity of that entity's communications and the broader network interactions remain protected.

Another crucial feature of UniHand is its KEF [7]. In traditional key escrow systems, a third party (often the core network) holds copies of encryption keys, possibly for lawful interception purposes. However, this approach introduces significant privacy and security risks, as the escrowed keys become a prime target for attackers and can be subject to misuse. UniHand's KEF design ensures that secret keys are jointly computed between users and Core Network (CN), without being fully controlled by any single party. This approach not only enhances user privacy but also reduces the risk of large-scale key compromises. By designing

a KEF scheme, it addresses critical security concerns in modern telecommunications, aligning with recent advancements in cryptographic key management that protect against potential misuse by key generation authorities [36].

Similar to the previous chapter, UniHand scheme leverages innovative cryptographic schemes, including Sanitisable Signatures (SanSig) [12] and universal accumulators [50], to achieve advanced security features. Both schemes are detailed in Chapter 2, in Sections 2.2.8 and 2.2.9, respectively. These schemes enable the UniHand to maintain high levels of security and privacy while supporting efficient user revocation management.

Motivation and Contributions

The Authentication and Key Agreement (AKA) process forms the cornerstone of security in 5G networks, serving as the critical first step in validating users before granting access to network services. Equally important is the Handover (HO) procedure, which re-authenticates roaming users and ensures service continuity. Therefore, maintaining robust security and privacy measures in both AKA and HO protocols are paramount for the integrity of 5G networks.

However, current 5G-AKA and HO protocols exhibit vulnerabilities to various security and privacy threats. Although numerous studies have attempted to address these issues, a comprehensive solution that tackles all security and privacy concerns in 5G AKA and HO protocols remains lacking. These concerns include susceptibility to linkability attacks, identity confusion attacks, lack of PFS, and confidentiality breaches of sequence numbers. A significant limitation of many previous solutions is their reliance on the existing 5G-AKA as a foundation for HO protocols, thereby inheriting its weaknesses. Moreover, to the best of our knowledge, no existing protocols have achieved both KCI resilience and KEF properties in 5G networks. This oversight creates a critical security vulnerability in 5G networks, particularly when considering active attackers capable of impersonating honest parties to compromised entities. Both KCI resilience and KEF are essential for enhancing security, ensuring fairness in key generation, and mitigating the risk of protocol failure due to a single key compromise.

To address these challenges, we propose UniHand, a Universal Handover scheme that achieves seamless user mobility while fulfilling all requisite security and privacy properties, as detailed in Section 3.4.1. Our primary contributions are as follows:

- Development of the *first* standalone solution achieving KCI resilience with KEF for roaming users in 5G SCNs, providing privacy-preserving and secure authentication and universal HO protocols.
- Propose a novel secure privacy-preserving user membership revocation scheme for efficiently managing a large number of users in 5G, leveraging dynamic universal accumulators [50].
- Conduct of a rigorous formal security analysis of the proposed UniHand scheme, demonstrating its achievement of all desired security and privacy properties.
- Conduct of a comprehensive comparative analysis between UniHand and existing literature, illustrating that UniHand achieves all required security properties.

Chapter Organisation

The rest of the chapter is organised as follows:

- Section 5.1 outlines the system and threat models that form the foundation of UniHand, detailing the network architecture and potential threats.
- Section 5.2 presents the core UniHand scheme, elucidating the initial authentication and universal handover protocols.
- Section 5.3 offers a rigorous security analysis, providing formal proofs of UniHand's security properties and demonstrating its security and privacy strength.
- Section 5.4 evaluates UniHand's performance and security, comparing it with other existing protocols in terms of security, computational and communication costs.
- Finally, Section 5.5 concludes the chapter, summarizing key contributions and discussing future research directions.

We provide the notation used in this chapter in Table 5.1.

5.1 System and Threat Model

This section provides the foundational framework of our proposed UniHand scheme, encompassing both the system architecture and security considerations. We begin by detailing the system model, which outlines the key components, their interactions, and the overall architecture of our proposed solution. Following this, a comprehensive threat model that delineates potential threats and attack vectors our scheme is designed to counter.

5.1.1 System Model

This subsection delineates the System Model of our proposed UniHand scheme. The architecture comprises three principal components: the Core Network (CN), the 5G radio base station (Next Generation NodeB, gNB), and the User Equipment (UE), as illustrated in Figure 5.1. These components are the key participants in two fundamental protocols: initial authentication and universal handover.

The CN encapsulates the core entities of the 5G infrastructure, including the AMF, UPF, SMF, and AUSF. The CN is responsible for configuring all network participants, encompassing both the gNB and UE, as well as generating certificates essential for authentication processes. This central role is particularly crucial in the initial authentication protocol, where the CN authenticates UEs, generates certificates for future HOs, and establishes session keys for subsequent communications. The gNB serves as the intermediary, facilitating seamless interaction between the UE and the CN. The gNB plays a pivotal role in the network's communication infrastructure, particularly during the universal handover protocol. A key innovation of the UniHand scheme is that during handover, the gNB can authenticate users without assistance from the CN. This autonomous authentication is achieved through the utilization of sanitizable signatures, which allow the gNB to verify certificates and authenticate users independently. This capability significantly reduces network overhead and latency during HOs, enhancing the overall efficiency and scalability of the 5G network.

Table 5.1: *Notation and cryptographic functions*

Symbol	Description
Notation :	
CN	Core network
gNB	Base station
UE	User equipment
UID_i	Universal user ID
RUID	Random user ID
ω_U	non-membership witness
pid, T_{ID}	User pseudo IDs
T_U	User subscription validity period
$RL_v\, RL_{new}$	Revocation list
k_i	Long-term key
k_s	Session key
$\mathsf{AE.Enc}\{k_i, m\}$	Authenticated encryption
$\mathsf{AE.Dec}\{k_i, m\}$	Authenticated decryption
$\mathbb{C}_G$	gNB certificate
$\mathbb{C}_U$	UE certificate
pk_{sig}^{CN}, sk_{sig}^{CN}	CN public and secret signing keys
pk_{san}^{gNB}, sk_{san}^{gNB}	gNB public and secret sanitising keys
ACK	Acknowledgement
σ	SanSig Signature
CryptographicFunctions :	
AE.Enc/AE.Dec	Authenticated encryption/decryption
KDF	Key derivation function
SanSig.Sign	Sanitizable signature (sign)
SanSig.Sanit	Sanitizable signature (sanitise)
DDH	Directional Diffie-Hellman

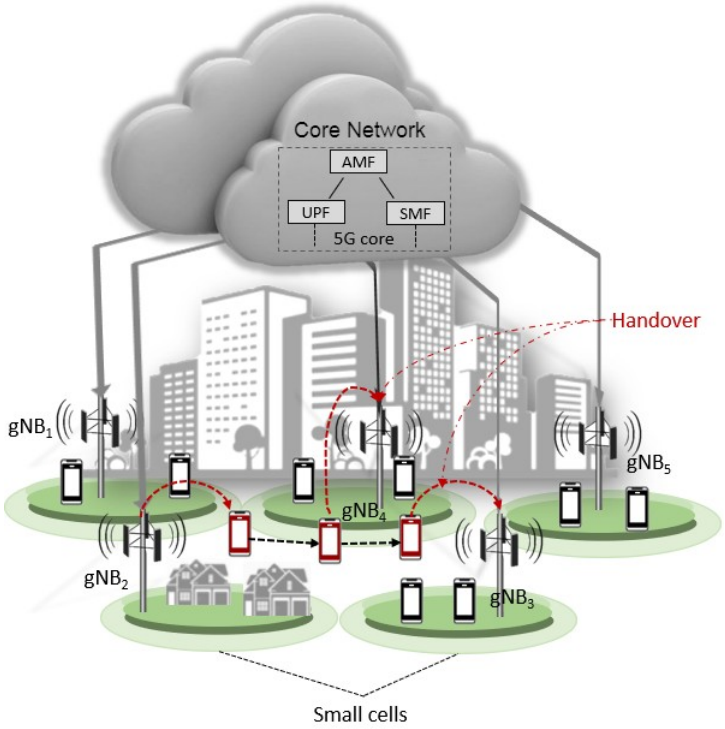

Figure 5.1: *System Model*

5.1.2 Threat Model

The threat model is designed to capture security notions recommended by the 3GPP group [4] and [56, 15, 33] for 5G authentication and handover protocols. It also addresses the desirable security and privacy properties identified in Section 3.4.1. During the execution of UniHand, all communication channels are public, i.e. the adversary can control the public channels fully. Our analysis combines three types of adversaries: Type 1 adversary $\mathcal{A}_1$ controls the network and can intercept, insert, modify and delete any message. Type 2 adversary $\mathcal{A}_2$ tries to break the linkability and key-indistinguishability proprieties of communicating parties. Type 3 adversary $\mathcal{A}_3$ captures KCI attacks and is capable of compromising at most one of the following secret keys:

1. Long-term keys (LTK) shared between UE and CN.
2. Asymmetric secret keys (ASK), signing and sanitising keys of the protocol participants, i.e. UE, gNB and CN
3. Session keys established between protocol participants.

5.2 Proposed UniHand Scheme

The UniHand scheme consists of three principal phases/protocols: *system initialisation* phase, an *initial authentication* protocol and *universal HO* protocol. The ***system initialisation*** phase is responsible for registering gNBs and new users into the network, as well as generating

the initialisation parameters for UE and gNB. During this phase, CN generates certificates for all gNBs in the network. Additionally, the CN generates a long-term secret key and shares it with the user, along with pseudo-identities (pid and T_{ID}) to preserve users' anonymity. The CN also initialise the accumulator to manage user revocation efficiently. In the ***initial authentication*** protocol, the CN generates certificates for joining users, which are created using SanSig algorithm using their pid and users' sanitising public keys. Finally, the ***universal HO*** protocol facilitates network access for roaming users while upholding network security and user privacy.

5.2.1 System Initialisation

In this phase, we assume all communication channels are secure, ensuring confidentiality, integrity, and replay protection to prevent unauthorised access and data manipulation. The system initialisation phase is responsible for generating all requisite parameters for $\mathsf{UniHand}$, including the registration of UEs and gNBs in the network and the initialisation of the accumulator. This phase can be divided into three main parts:

1. **gNB Registration**: Each gNB in the network must undergo registration and authentication with the CN. Initially, the gNB generates an asymmetric sanitising key pair $(pk_{san}^{\text{gNB}}, sk_{san}^{\text{gNB}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{san}(1^n)$. Subsequently, the gNB transmits a registration message to the CN, encompassing its sanitising public key (pk_{san}^{gNB}). Upon receipt of this message, the CN authenticates the gNB and generates a certificate $(\mathbb{C}_G, \sigma_G)$ via the SanSig algorithm: $\sigma_G \xleftarrow{\$} \mathsf{SanSig.Sign}(\mathbb{C}_G, sk_{sig}^{\text{CN}}, pk_{san}^{\text{gNB}}, \mathsf{ADM}(\mathbb{C}_{mod}^{G}))$. To optimise efficiency, the CN may execute this step offline.

2. **UE Registration**: Analogously, each user in the network must register to access the provided network services. A UE initiates this phase by generating an asymmetric sanitising key pair $(pk_{san}^{\text{UE}}, sk_{san}^{\text{UE}}) \xleftarrow{\$} \mathsf{SanSig.KGen}_{san}(1^n)$. The UE then transmits a registration request to the CN via a secure channel, including all UE credentials and their sanitising public key pk_{san}^{UE}. Upon receiving this registration message, the CN generates two independent pseudo identities, pid and T_{ID} (with no direct relationship between these aliases), in addition to a symmetric long-term key (k_i) for the intended user. These are subsequently shared with the UE, where $pid_i, T_{IDi} \xleftarrow{\$} \{0,1\}^n$.

3. **Accumulator Initialisation**: To establish the revocation list (RL) using an accumulator ($\ddot{C}$), the CN first generates a secret key for the accumulator via $\mathsf{KGen}(1^n) \xrightarrow{\$} (sk)$. Subsequently, it generates the accumulator using $\mathsf{Gen}(sk, X) \xrightarrow{\$} RL$, where X is initially an empty set.

5.2.2 Initial Authentication

Each registered user is required to execute this protocol to securely join the network. This protocol begins by facilitating a mutual authentication between communicating parties, leading to the generation of a new certificate for the intended UE by the CN. This process is illustrated in Figure 5.2 and comprises the following key steps:

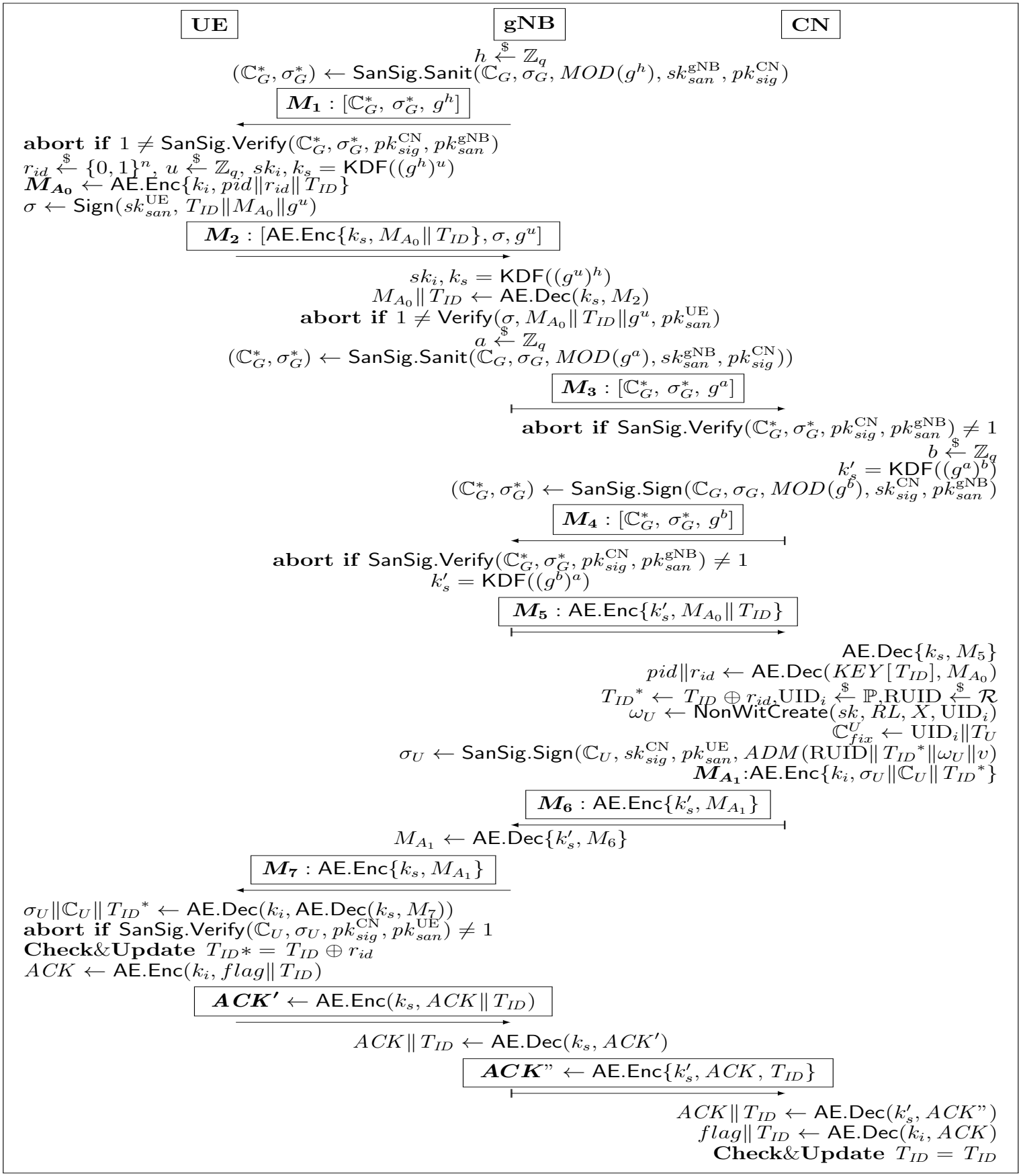

Figure 5.2: *Initial authentication protocol of* UniHand *scheme. Detailed descriptions of each algorithm are provided in Chapter 2.*

Step 1: gNB → UE. $\mathbf{M_1}$:$[\mathbb{C}^*_G, \sigma^*_G, g^h]$.
The first step is establishing a secure session key between UE and gNB via exchanging signed Diffie–Hellman (DH) values. In this step, the gNB randomly samples an ephemeral key (h) and computes (g^h). Then the gNB utilises the $\mathsf{SanSig.Sanit}(.)$ algorithm to sanitise their certificate to include the generated DH ephemeral ($MOD(g^h)$) for key integrity and to prevent the known MITM attack on DH [63]. The updated/sanitised certificate with (g^h) are sent to the UE via M_1.

Step 2: UE → gNB. $\mathbf{M_2}$:$[\mathsf{AE.Enc}\{k_s, M_{A_0} \| T_{ID}\}, \sigma, g^u]$
Upon receiving M_1, UE first verifies the sanitised signature using ($\mathsf{SanSig.Verify}()$) and checks the message integrity (i.e. $g^h \stackrel{?}{=} g^h$). If both verification hold then UE samples (r_{id} *and* u) randomly and computes session keys (sk_i, k_s). Next UE encrypts r_{id}, pid and T_{ID} using k_i, where *pid and* T_{ID} is user pseudo identities and k_i is the symmetric long-term key shared between the UE and CN to construct message M_{A_0}(to preserve message confidentiality/privacy in case of honest but curious gNB).Then the UE signs M_{A_0}, along with T_{ID} and g^u using user's secret sanitising key (sk^{UE}_{san}). Finally, the UE composes a message M_2 which consists of the signature, ephemeral DH (g^u) along with the encryption of $M_{A_0} \| T_{ID}$ (using the session key (k_s)), then sends it to gNB.

Step 3: gNB → CN. $\mathbf{M_3}$: $[\mathbb{C}^*_G, \sigma^*_G, g^a]$
After receiving the message M_2, gNB first computes session keys (sk_i, k_s) and decrypt M_2. Subsequently, the gNB verifies the signature σ and checks the message integrity (i.e. $g^u \stackrel{?}{=} g^u$). If both verifications hold, then gNB randomly samples a temporary key (a) and computes (g^a). Then the gNB utilises the $\mathsf{SanSig.Sanit}(.)$ algorithm to sanitise his/her certificate to include the generated DH ephemeral ($MOD(g^a)$) for key integrity and to prevent the known MITM attack on DH. The updated/sanitised certificate with (g^a) are sent to the CN via M_3.

***Note**: In this step, we use the notation* $\mathsf{Dec}(M_2)$ *to refer specifically to the decryption of the encrypted portion* ($M_{A_0} \| T_{ID}$)*, while other parts* (σ, g^u) *remain in plaintext. This is a slight abuse of notation to avoid introducing extra symbols.*

Step 4: CN → gNB. $\mathbf{M_4}$: $[\mathbb{C}^*_G, \sigma^*_G, g^b]$
Upon receiving M_3, CN first verifies the sanitised signature using ($\mathsf{SanSig.Verify}$) and checks the message integrity (i.e. $g^a \stackrel{?}{=} g^a$). If both verification hold, then CN samples randomly (b) and computes the session key (k'_s). Next the CN re-sign $\mathbb{C}^*_G$ using $\mathsf{SanSig.Sign}$ and includes the generated DH ephemeral. Then the CN constructs a message M_4 and sends it to gNB.

Step 5: gNB → CN. $\mathbf{M_5}$: $\mathsf{AE.Enc}\{k'_s, M_{A_0} \| T_{ID}\}$
After receiving the message M_4, gNB first verifies the certificate $\mathbb{C}^*_G$ and checks the message integrity (i.e. $g^b \stackrel{?}{=} g^b$). If both verification hold then gNB computes session key (k'_s) to encrypt UEs information ($M_{A_0} \| T_{ID}$) and send the ciphertext to the CN.

Step 6: CN → gNB. $\mathbf{M_6}$: $\mathsf{AE.Enc}\{k'_s, M_{A_1}\}$
Upon receiving M_5, the CN decrypts message M_5 using the session key (k'_s) and recovers the long term key k_i via T_{ID} to decrypts M_{A_0}, and obtain (pid, r_{id}). The CN then

computes a new temporary user identifier $T_{ID}{}^*$, and generates a universal user ID (UID_i), which will be the user's identifier during HOs and in RL. Next, CN generates a non-membership witness ($\omega_U \leftarrow \mathsf{NonWitCreate}(.)$, and specifies the version number (v) of RL. CN creates and signs the ($\mathbb{C}_U$) by generating "fixed" part ($\mathbb{C}^U_{fix}$) and "modifiable" part $\mathbb{C}^U_{mod}$, where $\mathbb{C}^U_{fix} = \text{UID}_i \| T_U$ (T_U is a user subscription validity period) and $\mathbb{C}^U_{mod} = \text{RUID} \| T_{ID}{}^* \| \omega_U \| v$ (RUID a random-user ID). Then CN signs the entire certificate using $\mathsf{SanSig.Sign}(.)$ algorithm. CN then stores UID_i, T_{IDi} and T^*_{IDi} (to prevent de-synchronisation) and encrypts user certificate $\mathbb{C}_U$ and σ_U using (k_i), to compose message M_{A_1}. Finally, the CN encrypts M_6 using the session key and sends it to the gNB.

Step 7: gNB $\rightarrow$ UE. $\mathbf{M_7} : \mathsf{AE.Enc}\{k_s, M_{A_1}\}$
Upon the receipt of the message from CN, the gNB decrypt M_6 and re-encrypt it using (k_s), then forwards it to the UE via M_7.

Step 8: UE $\xrightarrow{ACK'}$ gNB $\xrightarrow{ACK"}$ CN.
Upon the reception of M_7, first, the UE decrypts the message using the session key (k_s), then using the long-term key (k_i). Next, the UE verifies his/her certificate using $\mathsf{SanSig.Verify}(.)$ and checks the integrity of T_{ID} (i.e. $T_{ID}{}^* \stackrel{?}{=} T_{ID} \oplus r_{id}$). If both verifications hold, then UE sends an acknowledgement message to the CN to confirm the new temporary identity and delete the old one.

5.2.3 Universal Handover

This protocol must be executed by each roaming user and facilitates mutual authentication between communicating parties, thereby ensuring secure transitions for roaming users between 5G small cells. The process, illustrated in Figure 5.3, encompasses the following key steps:

Step 1: gNB $\rightarrow$ UE. $\mathbf{M_1}$: $[\mathbb{C}^*_G, \sigma^*_G, g^h]$.
Analogous to the initial authentication protocol's *first step*. gNB generates ephemeral key h, computes g^h, and sanitises its certificate to include $MOD(g^h)$. This establishes a secure session key and mitigates MITM attacks. gNB sends the sanitised certificate and g^h to UE.

Step 2: UE $\rightarrow$ gNB. $\mathbf{M_2}$:$[\mathsf{AE.Enc}\{k_s, \mathbb{C}_U \| \sigma_U \| \omega_U \| v\}, g^u]$
Upon receiving the message M_1, UE verifies gNB sanitised certificate ($\mathbb{C}^*_G$) using the SanSig verification algorithm $\mathsf{SanSig.Verify}(\mathbb{C}^*_G, \sigma^*_G)$. Then the UE checks the message integrity (i.e. $g^h \stackrel{?}{=} g^h$). If both verification hold, UE samples u and RUID randomly, then computes session keys (sk_i, k_s). Next, UE updates their certificate (the modifiable part), i.e. $MOD(\text{RUID} \| \omega \| v \| g^u)$ (preventing replay attacks), where the modifiable part consists of random-user id, non-membership witness, the accumulator version number and Ephemeral DH, respectively. Then UE sanitises the updated certificate using the sanitising algorithm $\mathsf{SanSig.Sanit}(.)$, and encrypt it using k_s to compose a message M_2. Finally, UE sends M_2 along with g^u (Ephemeral DH) to the gNB.

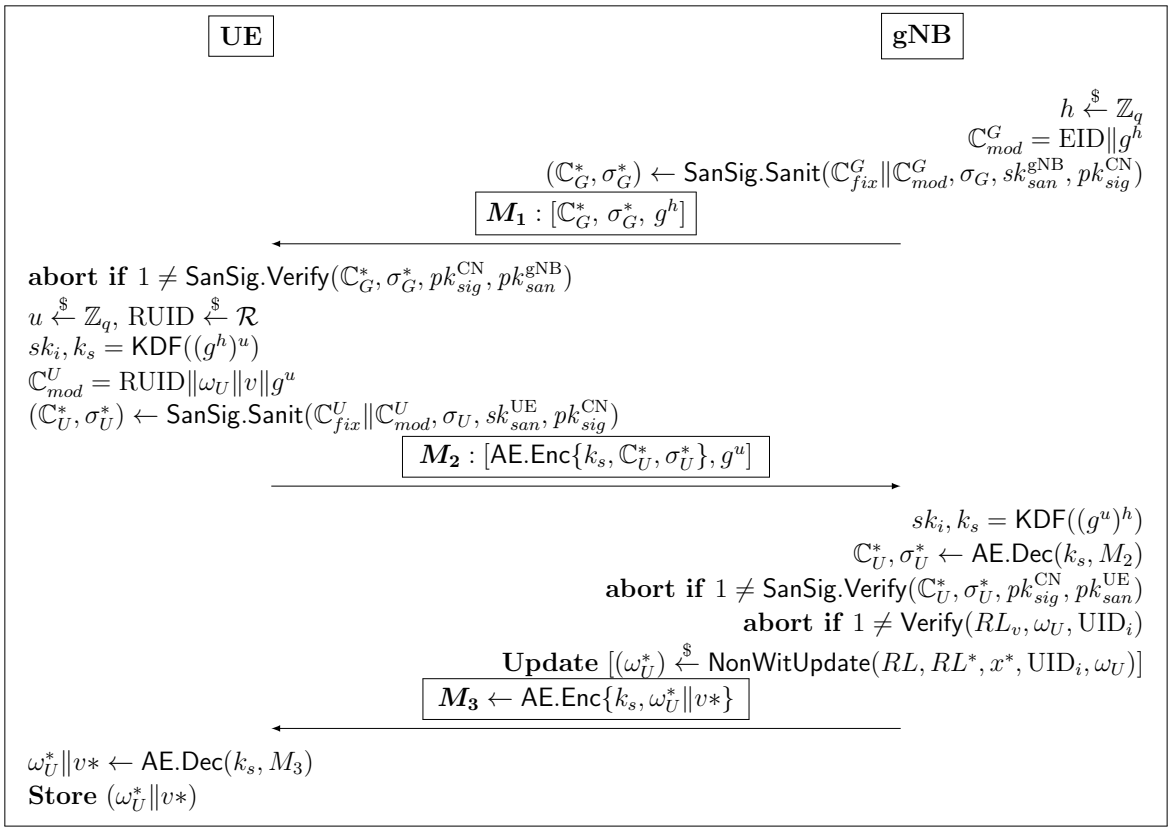

Figure 5.3: *Universal Handover protocol of* UniHand *scheme. Detailed descriptions of each algorithm are provided in Chapter 2.*

Step 3: gNB $\rightarrow UE$ $\mathbf{M_3}$:$\leftarrow \mathsf{AE.Enc}\{k_s, \omega^*_U \| v*\}$
Upon receiving the response message M_2, gNB generates the session keys (sk_i, k_s), to decrypt M_2. Subsequently, gNB verifies UE's certificate using the verification algorithm $\mathsf{SanSig.Verify}(\mathbb{C}_U, \sigma_U)$. If SanSig verification holds, gNB retrieves the accumulator version v and checks if $v_i = v_{RL}$, to see if RL has been updated. If RL has not been updated, gNB checks whether the UE is in the revocation list by $\mathsf{Verify}(\text{UID}_i, v_i, ..)$. Otherwise, if the revocation list has been updated, where $v_i \neq v_{RL}$, gNB checks whether UID is added in the later version of the RL or not, by $(\mathsf{Verify}(\text{UID}_i, v_{RL}, ..))$. If not, gNB updates the non-membership witness (ω^*_U) (where x^* is the new revoked UE). Finally, gNB encrypts and sends M_3 to UE, which he will maintain for future communications.

***Note**: In this step, we use the notation* $\mathsf{Dec}(M_2)$ *to refer specifically to the decryption of the encrypted portion* $(\mathbb{C}^*_U \| \sigma^*_U)$*, while other parts* (g^u) *remain in plaintext. This is a slight abuse of notation to avoid introducing extra symbols .*

5.3 Security Analysis

In this section, we formally prove the security of our protocols. We begin by showing that our protocols achieve mutual authentication. Then, we demonstrate that the session keys

produced by the proposed UniHand scheme cannot be distinguished from random keys. Lastly, we prove that outside attackers cannot link different sessions belonging to the same party, which protects user privacy.

5.3.1 Mutual Authentication Security

In this section, we analyse the mutual authentication (MA) security of UniHand scheme, demonstrating that it achieves MA-security. We build upon the general MA-security model introduced in Chapter 2, Section 2.3.2. Recall that in the MA security game, defined in Def.24, $\mathcal{A}$ wins (and $\mathbf{Exp}^{\mathsf{MA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if the adversary has caused a **clean** session to accept (and set $\pi^s_i.\alpha \leftarrow$ `accepted`) and there either exists no matching subset session π^t_j (i.e. no CN session that outputs the messages received by π^s_i - Def.22), or no matching session π (i.e. no CN session that outputs the messages received by π^s_i - Def.23).

Here, we specify our particular protocols' adversary capabilities and cleanness predicate.

Adversary Queries. Here, we define queries that represent the behaviours of the adversary $\mathcal{A}$ during the execution of the MA experiment:

- **Create**(i,s): allows $\mathcal{A}$ to initialize new UE sessions π^s_i.
- **Send**(m, i, s): allows $\mathcal{A}$ to send messages m to a session π^s_i. It produces a message m', which might be empty.
- **CorruptLTK**(i)$\rightarrow k_i$: allows $\mathcal{A}$ to leak the long-term key of UE_i.
- **CorruptASK**$(i, \rho) \rightarrow (sk)$: allows $\mathcal{A}$ to leak the long-term asymmetric keys of a party.
- **StateReveal**(i,s)$\rightarrow \pi^s_i$: allows $\mathcal{A}$ to reveal the internal state of π^s_i.
- **Reveal**$(i, s, \rho) \rightarrow k$: allows $\mathcal{A}$ to reveal the secret session key k computed during session π^s_i where $\pi^s_i.\rho = \rho$.

Cleanness Predicate. We now turn to describe our cleanness predicates. It is important to note that forging messages to the π^s_i becomes trivial if the $\mathcal{A}$ has compromised both the long-term key shared between CN and UE, as well as the long-term asymmetric key of the partner session, or if the $\mathcal{A}$ compromised the long-term asymmetric key of gNB when $\pi^t_j.\rho = \text{gNB}$. To mitigate this type of attack, we introduce a cleanness predicate that ensures the $\mathcal{A}$ is prevented from issuing **CorruptLTK**, **CorruptASK**, or **StateReveal** queries before the test session's acceptance.

Definition 30 (Initial authentication cleanness) *A session π^s_i in the* MA *experiment, described in Section 2.3.2, is* $\mathbf{clean}_{IA}$ *if the following conditions hold:*

1. ***StateReveal***$(i, s, \pi^s_i.\rho)$ *has not been issued* and *for all sessions π^t_j such that π^t_j is a matching subset of π^s_i* ***StateReveal***$(j, t, \pi^t_j.\rho)$ *has not been issued and for all sessions π^t_j such that π^t_j is a matching session of π^s_i* ***StateReveal***$(j, t, \pi^t_j.\rho)$ *has not been issued.*
2. *If $\pi^s_i.\rho \neq \text{gNB}$, and there exists no $(j, t) \in n_P \times n_S$ such that π^t_j is a matching session of π^s_i,* ***CorruptLTK***(i) *or* ***CorruptASK***$(\pi^s_i.pid, (\text{CN}, \text{UE})\backslash\pi^s_i.\rho)$ *have not both been issued before $\pi^s_i.\alpha =$* `accepted`.

3. *If there exists no* $(j,t) \in n_P \times n_S$ *such that* π_j^t *is a matching subset for* π_i^s*, and* $\pi_i^s.\rho \neq$ gNB*,* ***CorruptASK***$(\pi_i^s.pid, \text{gNB})$ *has not been issued before* $\pi_i^s.\alpha = \texttt{accepted}$*. Else, if there exists no* $(j,t) \in n_P \times n_S$ *such that* π_j^t *is a matching subset for* π_i^s*, and* $\pi_i^s.\rho =$ gNB*,* ***CorruptASK***$(\pi_i^s.pid, \text{UE})$ *and* ***CorruptASK***$(, \text{CN})$ *have not been issued before* $\pi_i^s.\alpha = \texttt{accepted}$*.*

Definition 31 (Universal Handover cleanness) *A session* π_i^s *in the* MA *experiment, described in Section 2.3.2, is* $\mathbf{clean}_{UH}$ *if the following conditions hold:*

1. ***StateReveal***$(i, s, \pi_i^s.\rho)$ *has not been issued* and *for all sessions* π_j^t *such that* π_j^t *is a matching subset of* π_i^s ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*

2. *If there exists no* $(j,t) \in n_P \times n_S$ *such that* π_j^t *is a matching subset for* π_i^s*,* ***CorruptASK***$(\pi_i^s.pid, (\text{gNB}, \text{UE}) \backslash (\pi_i^s.\rho))$ *has not been issued before* $\pi_i^s.\alpha = \texttt{accepted}$*.*

5.3.1.1 MA-security of Initial Authentication protocol

Here we present our formal analysis and results for the MA-security of the initial authentication (IA) protocol.

Theorem 10 MA-***security of Initial Authentication****. Initial Authentication depicted in Figure 5.2 is* MA*-secure under the cleanness predicate* $\mathbf{clean}_{IA}$ *defined in Def.30. For any PPT algorithm* $\mathcal{A}$*,* $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{IA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming that the* SanSig*,* Enc *and* KDF *schemes achieve* EUFCMA*,* AE *and* KDF *security and under the* DDH *assumption.*

Proof: Our proof is divided into three cases, denoted by

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda), \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{IA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$$

.

Case 1: We assume the first **clean** session π_i^s to accept without a matching session or subset sets $\pi_i^s.\rho = \text{UE}$. Now we begin by bounding the advantage of $\mathcal{A}$ in Case 1.1.

Case 1.1: According to the definition of this case, UE either accepts messages $\mathbf{M_1}$ and $\mathbf{M_7}$ without a matching subset (i.e. without honest gNB partner), or $\mathbf{M_{A_1}}$ without a matching session identifier (i.e. without honest CN). In this subcase $\mathcal{A}$ cannot corrupt the long-term UE symmetric secret or the $\pi_i^s.pid_{\text{gNB}}$ asymmetric key. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1.1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first UE session that accepts without a matching session or subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $j \in n_P$ of the gNB party such that $\pi_i^s.pid$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_P \mathbf{Adv}_{G_2}$.

Game 3: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk_{sig}^{\mathcal{C}}$ and $pk_{san}^{\mathcal{C}}$, which we embed into the CN's pk_{sig}^{CN} and gNB's pk_{san}^{gNB} respectively. Anytime CN or gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$, then $\mathcal{A}$ must have produced a message $\mathbf{M_1} = C_G^*, \sigma_G^*, g^h$ such that $\mathsf{SanSig.Verify}(C_G^*, \sigma_G^*, pk_{sig}^{\mathcal{C}}, pk_{san}^{\mathcal{C}}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C_G^*, σ_G^* and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 4 : In this game, we guess the index $t \in n_S$ of the gNB session π_j^t that output $\mathbf{M_1}$ received by π_i^s, introducing a factor of n_S in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_3} \leq n_P \cdot n_S \mathbf{Adv}_{G_4}$.

Game 5: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_i^s, i.e. session π_i^s receives g^h that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ_G^* in $\mathbf{M_1}$ to verify over g^h, and by **Game 3** we already abort if σ_G^* comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5}$.

Game 6: In this game, we replace g^{hu} computed honestly in π_i^s with a uniformly random and independent value $\hat{g^{hu}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{hu} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $c = ab$ and we are in **Game 5**; otherwise, $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 6**. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can break the DDH assumption, as formalised in Def.10. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 7: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_i^s and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 6** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 6** from **Game 7** can be used to break KDF security, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 8: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_7}$ (keyed by $\hat{k_s}$), but $\mathbf{M_7}$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{M_7}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 7** $\hat{k_s}$ is already uniformly random and independent, and this

replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 9: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_{A_1}}$ (keyed by k_i), but $\mathbf{M_{A_1}}$ was not output by a session owned by CN. We do so by defining a reduction $\mathcal{B}_5$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_5$ queries whenever $\mathcal{C}$ needs to encrypt or decrypt with k_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under k_i, and we can submit $\mathbf{M_{A_1}}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By the definition of the MA security game, k_i is uniformly randomly sampled at the beginning of the experiment, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_5,\mathsf{AE}}(\mathcal{A})$.

Game 10: In this game, the session π_i^s will only accept $\mathbf{M_1}$ and $\mathbf{M_7}$ from gNB and $\mathbf{M_{A_1}}$ from CN if they were generated honestly by gNB and CN sessions, and thus π_i^s has both matching sessions and matching subsets. Thus the advantage of $\mathcal{A}$ in winning the MA-security experiment is negligible. $\mathbf{Adv}_{G_{10}} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 1.2**.

Case 1.2: According to the definition of this case, UE either accepts messages $\mathbf{M_1}$ and $\mathbf{M_7}$ without a matching subset (i.e. without honest gNB partner), or $\mathbf{M_{A_1}}$ without a matching session identifier (i.e. without honest CN). In this subcase, $\mathcal{A}$ cannot corrupt the long-term CN asymmetric key or the $\pi_i^s.pid_{\text{gNB}}$ asymmetric key. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1.2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first UE session that accepts without a matching session or subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$. We do so by defining a reduction $\mathcal{B}_6$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime CN or gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$, then $\mathcal{A}$ must have produced a message $\mathbf{M_1} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_6$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_6$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_6,\mathsf{SanSig}}(\mathcal{A})$.

Game 3 : In this game, we guess the index $(j,t) \in n_P \times n_S$ of the gNB session π_j^t that output $\mathbf{M_1}$ received by π_i^s, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_P \cdot n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_i^s, i.e. session π_i^s receives g^h that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ_G^* in $\mathbf{M_1}$ to verify over g^h, and by **Game 2** we already abort if σ_G^* comes from $\mathcal{A}$, it follows that: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{hu} computed honestly in π_i^s with a uniformly random and independent value $g^{\hat{hu}}$. We do so by defining a reduction $\mathcal{B}_7$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{hu} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $c = ab$, and we are in **Game 4**. Otherwise, $c \xleftarrow{\$} \mathbb{Z}_q$, and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_7 \mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$. We do so by defining a reduction $\mathcal{B}_8$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $g^{\hat{uh}}$ and replacing the computation of sk_i, k_s in π_i^s and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(g^{\hat{uh}})$ and by **Game 5** $g^{\hat{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_8,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_7}$ (keyed by $\hat{k_s}$), but $\mathbf{M_7}$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_9$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_9$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{M_7}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 6** $\hat{k_s}$ is already uniformly random and independent, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_9$ to break the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_9,\mathsf{AE}}(\mathcal{A})$.

Game 8: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_{A_1}}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_{A_1}}$ was not output by a session owned by CN. We do so by defining a reduction $\mathcal{B}_{10}$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and UE_i's pk^{UE}_{san} respectively. Anytime CN or UE needs to generate a signature over a message m, $\mathcal{B}_{10}$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_{A_1}}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_{A_1}}$ was not output by a session owned by CN, then $\mathcal{A}$ must have produced a message $\mathbf{M_{A_1}} = \mathsf{AE.Enc}(k_i, \sigma_U \| C_U \| T^*_{ID} \| \omega \| v)$ such that $\mathsf{SanSig.Verify}(C_U, \sigma_U, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_{10}$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C_U, σ_U and triggers the abort event. Thus, the probability that $\mathcal{B}_{10}$

triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_{10},\mathsf{SanSig}}(\mathcal{A})$.

Game 9: In this game, the session π_i^s will only accept $\mathbf{M_1}$ and $\mathbf{M_7}$ from gNB and $\mathbf{M_{A_1}}$ from CN if they have matching subsets and sessions. Thus the advantage of $\mathcal{A}$ in winning the MA-security experiment is negligible. $\mathbf{Adv}_{G_9} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**, where We assume the first **clean** session π_i^s to accept without a matching session or subset sets $\pi_i^s.\rho = \text{CN}$.

Case 2.1: According to the definition of this case, CN either accepts messages $\mathbf{M_3}$, $\mathbf{M_5}$ or $\mathbf{ACK}''$ without a matching subset (i.e. without honest gNB partner), or $\mathbf{M_{A_0}}$, ACK without a matching session identifier (i.e. without honest UE). In this subcase $\mathcal{A}$ cannot corrupt the long-term UE symmetric key or the $\pi_i^s.pid_{\text{gNB}}$ asymmetric key. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2.1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1: In this game, we guess the index $s \in n_S$ of the first CN session that accepts without a matching session or subset, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_3}$ without setting $\pi_i^s.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_3}$ was not output by a session owned by $\pi_i^s.pid_{\text{gNB}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_3}$ without setting $\pi_i^s.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_3}$ (resp. $\mathbf{M_5}$ was not output by a session owned by $\pi_i^s.pid_{\text{gNB}}$, then $\mathcal{A}$ must have produced a message $\mathbf{M_3} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 3 : In this game, we guess the index $(j,t) \in n_P \times n_S$ of the gNB session π_j^t that output $\mathbf{M_3}$ received by π_i^s, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_P \cdot n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^a to the session π_i^s, i.e. session π_i^s receives g^a that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_3}$ to verify over g^a, and by **Game 2** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{ab} computed honestly in π_i^s with a uniformly random and independent value $g^{\hat{ab}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^a, g^b and g^{ab} computed by π_i^s and π_j^t with the

outputs of $\mathcal{C}_{\mathsf{DDH}}$. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $g^{\hat{ab}} = g^{ab}$ and we are in **Game 4**, otherwise $\hat{ab} \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the session and encryption keys k'_s with uniformly random values $\hat{k_s}'$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $g^{\hat{ab}}$ and replacing the computation of k'_s in π^s_i and π^t_j with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{k_s}'$. Since $k_s \leftarrow \mathsf{KDF}(g^{\hat{ab}})$ and by **Game 5** $g^{\hat{ab}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if π^s_i decrypts $\mathbf{M_5}$ (keyed by $\hat{k_s}'$), but $\mathbf{M_5}'$ was not output by π^t_j. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π^s_i needs to encrypt or decrypt with $\hat{k_s}'$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}'$, and we can submit $\mathbf{M_5}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 6** $\hat{k_s}'$ is already uniformly random and independent, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 8 : In this game, we guess the index $i \in n_P$ of the UE$_i$ key k_i π^s_i uses to decrypt $\mathbf{M_{A_0}}$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_7} \leq n_P \mathbf{Adv}_{G_8}$.

Game 9: In this game, we introduce an abort event that triggers if π^s_i decrypts $\mathbf{M_{A_0}}$, **ACK** (all keyed by k_i), but $\mathbf{M_{A_0}}$ or **ACK** was not output by an honest UE session. We do so by defining a reduction $\mathcal{B}_5$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_5$ queries when $\mathcal{B}_5$ needs to encrypt or decrypt with k_i. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under k_i, and we can submit $\mathbf{M_{A_0}}$ or **ACK** to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By the definition of the execution environment, k_i is generated uniformly random and independent,o and by the definition of **Case 2.1** $\mathcal{A}$ cannot issue **CorruptLTK**(i, UE) and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the AE security of Enc, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_5,\mathsf{AE}}(\mathcal{A})$.

Game 10: In this game, the π^s_i only accepts $\mathbf{M_3}$, $\mathbf{M_5}$ and **ACK**$''$ from an matching subset gNB and $\mathbf{M_{A_0}}$, **ACK** from an matching session UE. Thus, summing the probabilities, we find that the $\mathcal{A}$ has a negligible advantage in winning the MA-security experiment.: $\mathbf{Adv}_{G_9} = 0$

We turn to bound the advantage of $\mathcal{A}$ in **Case 2.2**.

Case 2.2: According to the definition of this case, CN either accepts messages $\mathbf{M_3}$, $\mathbf{M_5}$ or **ACK**$''$ without a matching subset (i.e. without honest gNB partner), or $\mathbf{M_{A_0}}$, ACK without a matching session identifier (i.e. without honest UE). In this subcase, $\mathcal{A}$ cannot corrupt the long-term asymmetric keys of UE or the $\pi^s_i.pid_{\text{gNB}}$ asymmetric key. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2.2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $s \in n_S$ of the first CN session that accepts without a matching session or subset, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π^s_i receives a message $\mathbf{M_3}$ without setting $\pi^s_i.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_3}$ was not output by a session owned by $\pi^s_i.pid_{\text{gNB}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π^s_i receives a message $\mathbf{M_3}$ without setting $\pi^s_i.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_3}$ (resp. $\mathbf{M_5}$ was not output by a session owned by $\pi^s_i.pid_{\text{gNB}}$, then $\mathcal{A}$ must have produced a message $\mathbf{M_3} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 3 : In this game, we guess the index $(j,t) \in n_P \times n_S$ of the gNB session π^t_j that output $\mathbf{M_3}$ received by π^s_i, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_P \cdot n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^a to the session π^s_i, i.e. session π^s_i receives g^a that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_3}$ to verify over g^a, and by **Game 2** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{ab} computed honestly in π^s_i with a uniformly random and independent value $\hat{g^{ab}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^a, g^b and g^{ab} computed by π^s_i and π^t_j with the outputs of $\mathcal{C}_{\mathsf{DDH}}$. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{ab}} = g^{ab}$ and we are in **Game 4**, otherwise $\hat{ab} \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the encryption keys k'_s with uniformly random values $\hat{k_s}'$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{ab}}$ and replacing the computation of k'_s in π^s_i and π^t_j with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{k_s}'$. Since $k_s \leftarrow \mathsf{KDF}(\hat{g^{ab}})$ and by **Game 5** $\hat{g^{ab}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_5}$ (keyed by $\hat{k_s}'$), but $\mathbf{M_5}'$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}'$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}'$, and we can submit $\mathbf{M_5}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 6** $\hat{k_s}'$ is already uniformly random and independent, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of Enc. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 8 : In this game, we guess the index $l \in n_P$ of the UE_l party π_i^s communicates with, such that $\pi_i^s.pid_{\mathrm{UE}} = l$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_7} \leq n_P\mathbf{Adv}_{G_8}$.

Game 9: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_j^t receives a message $\mathbf{M_2}$, but $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\mathrm{UE}}$. We do so by defining a reduction $\mathcal{B}_5$ that initialises a **Signature** challenger $\mathcal{C}_{\mathsf{Sign}}$, that outputs $pk^{\mathcal{C}}_{sig}$, which we embed into the UE_l's pk^{UE}_{sig}. Anytime UE needs to generate a signature over a message m, $\mathcal{B}_5$ instead queries $\mathcal{C}$ with m. Now, if π_j^t receives a message $\mathbf{M_2}$ where $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\mathrm{UE}}$, then $\mathcal{A}$ must have produced a message $T_{ID}\|\mathbf{M_{A_0}}\|g^u$ such that $\mathsf{Sign.Verify}(T_{ID}\|\mathbf{M_{A_0}}\|g^u, \sigma, pk^{\mathcal{C}}_{sig}) = 1$, which is a valid signature forgery. $\mathcal{B}_5$ responds to $\mathcal{C}_{\mathsf{Sign}}$ with $T_{ID}\|\mathbf{M_{A_0}}\|g^u, \sigma$ and triggers the abort event. Thus, the probability that $\mathcal{B}_5$ triggers the abort event is bounded by the EUFCMA security of Sign: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_5,\mathsf{Sign}}(\mathcal{A})$.

Game 10 : In this game, we guess the index $v \in n_P$ of the UE_l session that π_j^t receives $\mathbf{M_2}$ from, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_9} \leq n_P\mathbf{Adv}_{G_{10}}$.

Game 11: In this game, we replace g^{uh} computed honestly in π_j^t and π_v^l with a uniformly random and independent value $\hat{g^{uh}}$. We do so by defining a reduction $\mathcal{B}_6$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{uh} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{uh}} = g^{ab}$ and we are in **Game 10**, otherwise $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 11**. Any $\mathcal{A}$ that can distinguish **Game 10** from **Game 11** can break the DDH assumption. Thus: $\mathbf{Adv}_{G_{10}} \leq \mathbf{Adv}_{G_{11}} + \mathbf{Adv}^{DDH}_{\mathcal{B}_6,\mathsf{DDH}}(\mathcal{A})$.

Game 12: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ in π_j^t and π_v^l. We do so by defining a reduction $\mathcal{B}_7$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_v^l and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 11** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 11** from **Game 12** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{11}} \leq \mathbf{Adv}_{G_{12}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_7,\mathsf{KDF}}(\mathcal{A})$.

Game 13: In this game, we introduce an abort event that triggers if π_j^t decrypts $\mathbf{ACK}'$ (keyed by $\hat{k_s}$), but $\mathbf{ACK}'$ was not output by π_v^l. We do so by defining a

reduction $\mathcal{B}_8$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_8$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{ACK'}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 12** $\hat{k_s}$ is already uniformly random and independent and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_8$ to break the AE security of Enc, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_{12}} \leq \mathbf{Adv}_{G_{13}} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_8,\mathsf{AE}}(\mathcal{A})$.

Game 14: In this game, all messages sent to π_j^t come from an UE session without modification. Similarly, π_i^s only accepts $\mathbf{M_3}, \mathbf{M_5}, \mathbf{ACK''}$ from an matching subset session π_j^t. Thus, it follows that all messages added to the session identifier by π_i^s come from some matching UE session π_v^l. Thus we find that the $\mathcal{A}$ has a negligible advantage in winning the MA-security experiment.: $\mathbf{Adv}_{G_{14}} = 0$

We turn to bound the advantage of $\mathcal{A}$ in **Case 3**:

Case 3: In this case, gNB either accepts messages $\mathbf{M_2}$, $\mathbf{M_4}$, $\mathbf{M_6}$, or $\mathbf{ACK''}$, without a matching subset (i.e. without honest UE or CN partners). Note that in this case, $\mathcal{A}$ cannot corrupt the asymmetric long-term keys of either the CN or the $\pi_i^s.pid$. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C3}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first gNB session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_4}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_4}$ was not output by a session owned by $\pi_i^s.pid_{\text{CN}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime CN needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_4}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_4}$ was not output by a session owned by $\pi_i^s.pid_{\text{CN}}$, then $\mathcal{A}$ must have produced a message $\mathbf{M_4} = C^*_G, \sigma^*_G, g^b$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 3 : In this game, we guess the index $j \in n_S$ of the CN session π_j^t that output $\mathbf{M_6}$ received by π_i^s, introducing a factor of n_S in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^b to the session π_i^s, i.e. session π_i^s receives g^b that was not output from a CN session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_4}$ to verify over g^b, and by **Game 2** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{ab} computed honestly in π_i^s with a uniformly random and independent value $\hat{g^{ab}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^a, g^b and g^{ab} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{ab}} = g^{ab}$ and we are in **Game 4**, otherwise $\hat{ab} \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the encryption keys k_s' with uniformly random values $\hat{k_s}'$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{ab}}$ and replacing the computation of k_s' in π_i^s and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{k_s}'$. Since $k_s \leftarrow \mathsf{KDF}(\hat{g^{ab}})$ and by **Game 5** $\hat{g^{ab}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_6}$ (keyed by $\hat{k_s}'$), but $\mathbf{M_6}'$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}'$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}'$, and we can submit $\mathbf{M_6}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 6** $\hat{k_s}'$ is already uniformly random and independent, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of Enc. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 8 : In this game, we guess the index $l \in n_P$ of the UE_l party π_i^s communicates with, such that $\pi_i^s.pid_{\mathrm{UE}} = l$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_7} \leq n_P\mathbf{Adv}_{G_8}$.

Game 9: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_j^t receives a message $\mathbf{M_2}$, but $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\mathrm{UE}}$. We do so by defining a reduction $\mathcal{B}_5$ that initialises a **Signature** challenger $\mathcal{C}_{\mathsf{Sign}}$, that outputs $pk^{\mathcal{C}}_{sig}$, which we embed into the UE_l's pk^{UE}_{sig}. Anytime UE needs to generate a signature over a message m, $\mathcal{B}_5$ instead queries $\mathcal{C}$ with m. Now, if π_j^t receives a message $\mathbf{M_2}$ where $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\mathrm{UE}}$, then $\mathcal{A}$ must have produced a message $T_{ID}\|\mathbf{M_{A_0}}\|g^u$ such that $\mathsf{Sign.Verify}(T_{ID}\|\mathbf{M_{A_0}}\|g^u, \sigma, pk^{\mathcal{C}}_{sig}) = 1$, which is a valid signature forgery. $\mathcal{B}_5$ responds to $\mathcal{C}_{\mathsf{Sign}}$ with $T_{ID}\|\mathbf{M_{A_0}}\|g^u, \sigma$ and triggers the abort event. Thus, the probability that $\mathcal{B}_5$ triggers the abort event is bounded by the EUFCMA security of Sign: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_5,\mathsf{Sign}}(\mathcal{A})$.

Game 10 : In this game, we guess the index $v \in n_P$ of the UE_l session that π_j^t receives $\mathbf{M_2}$ from, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_9} \leq n_P\mathbf{Adv}_{G_{10}}$.

Game 11: In this game, we replace g^{uh} computed honestly in π_i^s and π_v^l with a uniformly random and independent value $\hat{g^{uh}}$. We do so by defining a reduction $\mathcal{B}_6$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{uh} computed by π_i^s

and π_v^l with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{uh}} = g^{ab}$ and we are in **Game 10**, otherwise $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 11**. Any $\mathcal{A}$ that can distinguish **Game 10** from **Game 11** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_{10}} \leq \mathbf{Adv}_{G_{11}} + \mathbf{Adv}^{DDH}_{\mathcal{B}_6,\mathsf{DDH}}(\mathcal{A})$.

Game 12: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ in π_i^s and π_v^l. We do so by defining a reduction $\mathcal{B}_7$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_v^l and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 11** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 11** from **Game 12** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_{11}} \leq \mathbf{Adv}_{G_{12}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_7,\mathsf{KDF}}(\mathcal{A})$.

Game 13: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{ACK'}$ (keyed by $\hat{k_s}$), but $\mathbf{ACK'}$ was not output by π_v^l. We do so by defining a reduction $\mathcal{B}_8$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_8$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{ACK'}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 12** $\hat{k_s}$ is already uniformly random and independent, and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_8$ to break the AE security of Enc. This implies: $\mathbf{Adv}_{G_{12}} \leq \mathbf{Adv}_{G_{13}} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_8,\mathsf{AE}}(\mathcal{A})$.

Game 14: In this game, all messages sent to π_i^s come from an UE or CN session without modification. Thus, it follows that all messages sent to π_i^s come from some matching subset UE session π_v^l or a matching subset CN session π_j^t, and thus we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment:$\mathbf{Adv}_{G_{14}} = 0$

5.3.1.2 MA-security of Universal Handover

Here we present our formal analysis and results for the MA-security of the Universal Handover (UH) protocol.

Theorem 11 MA-***security of the Universal Handover***. *Universal Handover depicted in Figure 5.3 is* MA-*secure under the cleanness predicate defined in Def.31. For any PPT algorithm $\mathcal{A}$ against the* MA *experiment,* $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{UH}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EU-FCMA security of* SanSig*, INT-CTXT security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof: Our proof is divided into two cases

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{UH}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$$

.

Case 1: In this case, UE accepts messages $\mathbf{M_1}$ and $\mathbf{M_3}$ without an honest matching partner (gNB) (no matching subset). In this case, the $\mathcal{A}$ cannot corrupt the $\pi_i^s.pid_{\text{gNB}}$ asymmetric key. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first UE session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $j \in n_P$ of the gNB party such that $\pi_i^s.pid = j$, introducing a factor of n_P in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_1} \leq n_P \mathbf{Adv}_{G_2}$.

Game 3: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_1}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_i^s.pid$, then $\mathcal{A}$ must have produced a message $\mathbf{M_1} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 4 : In this game, we guess the index $t \in n_S$ of the gNB session π_j^t that output $\mathbf{M_1}$ received by π_i^s, introducing a factor of n_S in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_3} \leq n_S \mathbf{Adv}_{G_4}$.

Game 5: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_i^s, i.e. session π_i^s receives g^h that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_1}$ to verify over g^h, and by **Game 3** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5}$.

Game 6: In this game, we replace g^{hu} computed honestly in π_i^s with a uniformly random and independent value $g^{\hat{hu}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{hu} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $c = ab$ and we are in **Game 5**, otherwise, $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 6**. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 7: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with ϵ and replacing the computation of sk_i, k_s in π_i^s and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game**

6 $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 6** from **Game 7** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 8: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_3}$ (keyed by $\hat{k_s}$), but $\mathbf{M_3}$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{M_3}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 7** $\hat{k_s}$ is already uniformly random and independent and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of the symmetric encryption scheme, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 9: In this game, the π_i^s only accepts $\mathbf{M_1}$, $\mathbf{M_3}$ from an honest matching partner. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_9} = 0$

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**:

Case 2: In this case, gNB accepts messages $\mathbf{M_2}$ without an honest matching partner (UE) (no matching subset). Note that in this case, $\mathcal{A}$ cannot corrupt the asymmetric long-term keys of the $\pi_i^s.pid$. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original mutual authentication experiment defined in 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first gNB session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $j \in n_P$ of the UE party such that $\pi_i^s.pid = j$, introducing a factor of n_P in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_1} \leq n_P \mathbf{Adv}_{G_2}$.

Game 3: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_i^s receives a message $\mathbf{M_2}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\text{UE}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and UE's pk^{UE}_{san} respectively. Anytime UE needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_i^s receives a message $\mathbf{M_2}$ without setting $\pi_i^s.\alpha \leftarrow$ `rejected` but $\mathbf{M_2}$ was not output by a session owned by $\pi_i^s.pid_{\text{UE}}$, then $\mathcal{A}$ must have produced a message $\mathbf{M_2}$ such that $\mathsf{SanSig.Verify}(\mathbb{C}^*_U, \sigma^*_U, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with $\mathbb{C}^*_U, \sigma^*_U$ and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 4 : In this game, we guess the index $j \in n_S$ of the UE session that π_i^s receives $\mathbf{M_2}$ from, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_3} \leq n_S \mathbf{Adv}_{G_4}$.

Game 5: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^u to the session π_i^s, i.e. session π_i^s receives g^u that was not output from a UE session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ_U^* in $\mathbf{M_2}$ to verify over g^u, and by **Game 2** we already abort if σ_U^* comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5}$.

Game 6: In this game, we replace g^{uh} computed honestly in π_i^s and π_j^t with a uniformly random and independent value $\hat{g^{uh}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{uh} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{uh}} = g^{ab}$ and we are in **Game 5**, otherwise $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 6**. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 7: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ in π_i^s and π_j^t. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_j^t and π_i^s with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 6** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 6** from **Game 7** can be used to break KDF security of the KDF scheme., as formalised in Def.2. Thus: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 8: In this game, we introduce an abort event that triggers if π_i^s decrypts $\mathbf{M_2}$ (keyed by $\hat{k_s}$), but $\mathbf{M_2}$ was not output by π_j^t. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{AE}}$, which $\mathcal{B}_4$ queries when π_i^s needs to encrypt or decrypt with $\hat{k_s}$. The abort event only triggers if $\mathcal{A}$ can produce a valid ciphertext that decrypts under $\hat{k_s}$, and we can submit $\mathbf{M_2}$ to $\mathcal{C}_{\mathsf{AE}}$, breaking the AE security of the Enc scheme. By **Game 7** $\hat{k_s}$ is already uniformly random and independent and this replacement is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_4$ to break the AE security of Enc, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 9: In this game, the π_i^s only accepts M_2 from an honest matching partner. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_9} = 0$

5.3.2 Key Indistinguishability Security

In this section, we analyse the key indistinguishability (KIND) security of UniHand scheme, demonstrating that it achieves KIND-security. We build upon the general KIND-security model introduced in Chapter 2, Section 2.3.3. A protocol Π is considered KIND-secure if no PPT adversary can win the game with non-negligible advantage under the conditions specified below. Here we define the following specific adversary query and cleanness predicate:

Adversary Queries. In addition to the **Create**, **Send**, **CorruptLTK**, **CorruptASK**, **Reveal**, and **StateReveal** queries listed above, we define one additional query **Test** that allows the adversary $\mathcal{A}$ to either the real key derived by a **clean** session during the execution

of the KIND experiment, or a random key sampled from the same distribution (depending on the challenge bit b).

- **Test**$(i, s) \rightarrow m$: When $\mathcal{C}$ receives a **Test** query, if **Test** has already been issued, $\pi_i^s.\alpha = \texttt{accepted}$, or π_i^s is not **clean**, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ sets $k_0 \leftarrow \pi_i^s.k$, and $k_1 \xleftarrow{\$} \{0,1\}^\lambda$, and returns k_b to $\mathcal{A}$ (where b was sampled by $\mathcal{C}$ at the beginning of the experiment.

Cleanness Predicate. We now turn to describe our cleanness predicates. It is important to note that distinguishing a real session key becomes trivial for the $\mathcal{A}$ if they have compromised both the long-term key shared between CN and UE, as well as the long-term asymmetric key of the partner session. To mitigate this type of attack, we introduce a cleanness predicate that ensures the $\mathcal{A}$ is prevented from issuing **CorruptLTK**, **CorruptASK**, **Reveal** or **StateReveal** queries before the test session's acceptance.

Definition 32 (Cleanness Predicate) *A session π_i^s in the* KIND *experiment described above is* **clean**$_{IA}$& **clean**$_{UH}$ *if the following conditions hold:*

1. ***Reveal***(i, s, ρ) *has not been issued, and if a matching session π_j^t exists,* ***Reveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
2. *The query* ***StateReveal***(i, s, ρ) *has not been issued* and *for all j, t such that π_j^t has a matching subset with π_i^s,* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
3. *If there is no $(j, t) \in n_P \times n_S$ such that π_j^t is a matching subset for π_i^s,* ***CorruptLTK***(i) *and* ***CorruptASK***$(\pi_i^s.pid)$ *have not been both issued before $\pi_i^s.\alpha =$* `accepted`.

5.3.2.1 KIND-security of Initial Authentication protocol

Here we present our formal analysis and results for the key indistinguishability (KIND) security of the initial authentication (IA) phase.

Theorem 12 KIND-***security of*** UniHand. UniHand *scheme depicted in Figure 5.2 is* KIND-*secure under the cleanness predicate defined in Def.32. For any PPT algorithm $\mathcal{A}$ against the* KIND *experiment,* $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig, *the KDF security of* KDF *and the* DDH *assumption.*

Proof: Our proof is divided into two cases, donated by

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

.

Case 1: π_i^s accepts without a matching session (or subset). Here we provide the security analysis of **Case 1** through a series of game-based reductions:

Game 0: This is the original key indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i,s,\rho)$ such that π^s_i accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, by **Case 1**, π^s_i has no matching session (or subset), and by **Game 1**, we abort if π^s_i accepts without matching session, it follows that $\mathcal{A}$ can never query $\mathbf{Test}(i,s\rho)$ and thus the KIND game proceeds identically regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**:

Case 2 : π^s_i accepts with a matching session and subset. First, we recall that cleanness predicate(32) prevents the $\mathcal{A}$ from issuing a $\mathbf{Reveal}(i,s,\pi^s_i.\rho)$ query π^s_i (and to any session π^t_j such that π^t_j is a matching session or subset with π^s_i), nor can it issue a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$, nor to any session π^t_j such that π^t_j is a matching session or subset with π^s_i. We proceed via the following sequence of games.

Game 0: This is the original key indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{IA},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{G_0}$.

Game 1: In this game, we guess the index $(i,s) \in n_P \times n_S$, and abort if $\mathcal{A}$ issues a $\mathbf{Test}(i^*,s^*,\pi^{s^*}_{i^*})$ query such that $i \neq i^*$ and $s \neq s^*$. This introduces the following bound: $\mathbf{Adv}_{G_0} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this game, we guess the index $(j,t) \in n_P \times n_S$, and abort if π^t_j is not the matching subset of π^s_i, which must exist by **Case 2**. This introduces the following bound: $\mathbf{Adv}_{G_1} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_2}$.

Game 3: In this game, we replace g^{uh} computed honestly in π^s_i and π^t_j with a uniformly random and independent value $g^{\hat{uh}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{uh} computed by π^s_i and π^t_j with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $g^{\hat{uh}} = g^{ab}$ and we are in **Game 2**, otherwise $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 3**. Any $\mathcal{A}$ that can distinguish **Game 2** from **Game 3** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{DDH}_{\mathcal{B}_1,\mathsf{DDH}}(\mathcal{A})$.

Game 4: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ in π^s_i and π^t_j. We do so by defining a reduction $\mathcal{B}_2$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $g^{\hat{uh}}$ and replacing the computation of sk_i, k_s in π^t_j and π^s_i with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(g^{\hat{uh}})$ and by **Game 3** $g^{\hat{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 3** from **Game 4** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_1,\mathsf{KDF}}(\mathcal{A})$.

Here we emphasise that as a result of these changes, the session key $\hat{sk_i}$ is now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_4} = 0$.

5.3.2.2 KIND-security of Universal Handover

Here we present our formal analysis and results for the key indistinguishability (KIND) security of the Universal Handover.

Theorem 13 KIND-*security of* UniHand. UniHand *scheme depicted in Figure 5.3 is* KIND-*secure under the cleanness predicate defined in Def.32. For any PPT algorithm $\mathcal{A}$ against the* KIND *experiment,* $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EUFCMA security of* SanSig, *the KDF security of* KDF *and the* DDH *assumption.*

Proof: Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to:

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

Case 1: π^s_i accepts without a matching subset. Here we provide the security analysis through a series of game-based reductions:

Game 0: This is the original key indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{G_0}$.

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i,s,\rho)$ such that π^s_i accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, by **Case 1**, π^s_i has no matching session (or subset), and by **Game 1**, we abort if π^s_i accepts without matching session, it follows that $\mathcal{A}$ can never query $\mathbf{Test}(i,s\rho)$ and thus the KIND game proceeds identically regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**.

Case 2: π^s_i accepts with a matching subset. First, we recall that cleanness predicate 32 prevents the $\mathcal{A}$ from issuing a $\mathbf{Reveal}(i,s,\pi^s_i.\rho)$ query π^s_i (and to any session π^t_j such that π^t_j is a matching session or subset with π^s_i), nor can it issue a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$, nor to any session π^t_j such that π^t_j is a matching session or subset with π^s_i. We proceed via the following sequence of games.

Game 0: This is the original key indistinguishability experiment defined in Def.25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{G_0}$.

Game 1: In this game, we guess the index $(i,s) \in n_P \times n_S$, and abort if $\mathcal{A}$ issues a $\mathbf{Test}(i^*,s^*,\pi^{s^*}_{i^*})$ query such that $i \neq i^*$ and $s \neq s^*$. This introduces the following bound: $\mathbf{Adv}_{G_0} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this game, we guess the index $(j,t) \in n_P \times n_S$, and abort if π^t_j is not the matching subset of π^s_i, which must exist by **Case 2**. This introduces the following bound: $\mathbf{Adv}_{G_1} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_2}$.

Game 3: In this game, we replace g^{uh} computed honestly in π_i^s and π_j^t with a uniformly random and independent value $\hat{g^{uh}}$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{uh} computed by π_i^s and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{uh}} = g^{ab}$ and we are in **Game 2**, otherwise $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 3**. Any $\mathcal{A}$ that can distinguish **Game 2** from **Game 3** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{DDH}_{\mathcal{B}_1,\mathsf{DDH}}(\mathcal{A})$.

Game 4: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$ in π_i^s and π_j^t. We do so by defining a reduction $\mathcal{B}_2$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_j^t and π_i^s with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 3** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 3** from **Game 4** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_1,\mathsf{KDF}}(\mathcal{A})$.

Here we emphasise that as a result of these changes, the session key $\hat{sk_i}$ is now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_4} = 0$.

5.3.3 Unlinkability Security

In this section, we analyse the unlinkability (Unlink) security of our proposed scheme, demonstrating that they achieve Unlink-security. We build upon the general Unlink-security model introduced in Chapter 2, Section 2.3.4. A protocol Π is Unlink-secure, if there exist no PPT algorithms $\mathcal{A}$ that can win the Unlink security game with non-negligible advantage. Here, we specify the adversary capabilities and cleanness predicate for our particular protocols.

Adversary Queries. In the Unlink game, $\mathcal{A}$ has access **Create**, **Send**, **CorruptLTK**, **CorruptASK**, **Reveal**, **StateReveal**, **Test** and **SendTest** queries described above. Unlike in the KIND game, **Test** in Unlink allows the adversary to initialise one of two sessions (depending on a bit b sampled by the challenger), and **SendTest**, which allows the adversary to interact with that session without revealing which party owns it.

- **Test**$(s, i, s', i') \rightarrow m$: allows $\mathcal{A}$ to begin a new session π_b, where $(\pi_0 = \pi_i^s)$ or $(\pi_1 = \pi_{i'}^{s'})$, where b is sampled by $\mathcal{C}$, and both π_i^s and $\pi_{i'}^{s'}$ are **clean**. **Test** query is only allowed to be issued by $\mathcal{A}$ if $\pi_b.\alpha \neq in - progress$ and **Send** queries to sessions owned by UE_i or $\mathrm{UE}_{i'}$ are only allowed to be issued by $\mathcal{A}$ until π_b has rejected or accepted the experiment execution.

- **SendTest**$(m) \rightarrow (m')$: allows $\mathcal{A}$ to send a message m to π_b after issuing **Test**. The $\mathcal{C}$ returns a $\perp$ if $\pi_b.\alpha \neq in - progress$.

Cleanness Predicate. We now turn to define our cleanness predicate for the unlinkability game. It is important to note that it is trivial for $\mathcal{A}$ to determine which of UE_i or $\mathrm{UE}_{i'}$ owns session π_b if the $\mathcal{A}$ has compromised the long-term asymmetric key. To mitigate this type of attack, we introduce a cleanness predicate, which ensures that the adversary is prohibited from issuing a **CorruptASK** or **StateReveal** queries before the session's acceptance.

Definition 33 (Cleanness predicate) *A session π_i^s in the* Unlink *experiment is* **clean** *if the following conditions hold:*

1. *The query* ***StateReveal***(i, s, ρ) *has not been issued* and *for all j, t such that π_j^t is a matching session (or has a matching subset) with π_i^s,* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*

2. *If there is no $(j, t) \in n_P \times n_S$ such that π_j^t is a matching subset for π_i^s,* ***CorruptASK***$(\pi_i^s.pid, (\text{gNB}, \text{UE}) \backslash \pi_i^s.\rho)$ *has not been issued before $\pi_i^s.\alpha =$* `accepted`.

5.3.3.1 Unlink-security of Initial Authentication protocol

Here we present our formal analysis and results for the unlinkability (Unlink) security of the initial authentication phase.

Theorem 14 Unlink-***security of Initial Authentication.*** *The Initial Authentication depicted in Figure 5.2 is unlinkable under the cleanness predicate defined in Def.33. For any PPT algorithm $\mathcal{A}$ against the* Unlink *experiment, $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is negligible assuming the EUFCMA security of* SanSig*, Confidentiality (IND-CPA) security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

Case 1: The test session π_b (such that $\mathcal{A}$ issues $\mathbf{Test}(i, s, i^*, s^*)$) accepts messages without a matching subset. Here we provide the security analysis of **Case 1** through a series of game-based reductions:

Game 0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{G_0}$.

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i, s, i^*, s^*)$ and π_b accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, by **Case 1**, π has no matching session (or subset). By **Game 1**, we abort if π_b accepts without matching session, it follows that $\mathcal{A}$ can never terminate and output a guess bit b' and thus the Unlink game proceeds identically regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**.

Case 2: The test session π_b (such that $\mathcal{A}$ issues $\mathbf{Test}(i, s, i^*, s^*)$) accepts messages with a matching subset. First, we recall that the cleanness predicate defined in Def.33 prevents the $\mathcal{A}$ from issuing a $\mathbf{StateReveal}(i, s, \pi_i^s.\rho)$, nor to any session π_j^t such that π_j^t is a matching session or subset with π_i^s. Here we provide the security analysis of **Case 2** through a series of game-based reductions:

Game 0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i, s) \in n_P \times n_S$ of the π_b session, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_b receives a message $\mathbf{M_1}$ without setting $\pi_b.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_b.pid$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime CN or gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_b receives a message $\mathbf{M_1}$ without setting $\pi_b.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_b.pid$, then $\mathcal{A}$ must have produced a message $\mathbf{M_1} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 3: In this game, we guess the index $(j, t) \in n_P \times n_S$ of the gNB session π_j^t that output $\mathbf{M_1}$ received by π_b, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_P \cdot n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_b, i.e. session π_b receives g^h that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_1}$ to verify over g^h, and by **Game 2** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{hu} computed honestly in π_b with a uniformly random and independent value $\hat{g^{hu}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{hu} computed by π_b and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $c = ab$ and we are in **Game 4**. Otherwise, $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_b and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by

Game 5 $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we replace the plaintexts in messages $\mathbf{M_2}$, $\mathbf{M_7}$ and $\mathbf{M_8}$ with uniformly random values of the same length. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{Conf}}$, which $\mathcal{B}_4$ queries (p, p^*) (where p is the original plaintext message and $p^* \xleftarrow{\$} \{0,1\}^{|p|}$) when it needs to encrypt with $\hat{k_s}$. By **Game 6** $\hat{k_s}$ is already uniformly random and independent, and this replacement is sound. If the bit b sampled by $\mathcal{C}_{\mathsf{Conf}}$ is 0, then we are in **Game 6**, otherwise, we are in **Game 7**. Any $\mathcal{A}$ that can distinguish between **Game 6** and **Game 7** can be used by $\mathcal{B}_4$ to break the Conf security of Enc, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{IND-CPA}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Game 8: In this game, we guess the index $s \in n_S$ of the CN session (which we denote π_{CN} that is a matching session of π_b, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_7} \leq n_S \mathbf{Adv}_{G_8}$.

Game 9: Here we introduce an abort event, where $\mathcal{C}$ aborts if π^t_j receives a message $\mathbf{M_4}$ without setting $\pi^t_j.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_4}$ was not output by a session owned by CN. We do so by defining a reduction $\mathcal{B}_5$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime CN or gNB needs to generate a signature over a message m, $\mathcal{B}_5$ instead queries $\mathcal{C}$ with m. Now, if π^t_j receives a message $\mathbf{M_4}$ without setting $\pi^t_j.\alpha \leftarrow \texttt{rejected}$ but $\mathbf{M_4}$ was not output by a session owned by CN, then $\mathcal{A}$ must have produced a message $\mathbf{M_4} = C^*_G, \sigma^*_G, g^b$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_5$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_5$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_5,\mathsf{SanSig}}(\mathcal{A})$.

Game 10: In this game, we replace g^{ab} computed honestly in π^t_j and π_{CN} with a uniformly random and independent value $\hat{g^{ab}}$. We do so by defining a reduction $\mathcal{B}_6$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^a, g^b and g^{ab} computed by π^t_j and π^s_iCN with the outputs of $\mathcal{C}_{\mathsf{DDH}}$. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $\hat{g^{ab}} = g^{ab}$ and we are in **Game 9**, otherwise $\hat{ab} \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 10**. Any $\mathcal{A}$ that can distinguish **Game 9** from **Game 10** can break the DDH assumption. Thus: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{10}} + \mathbf{Adv}^{DDH}_{\mathcal{B}_6,\mathsf{DDH}}(\mathcal{A})$.

Game 11: In this game we replace the session and encryption keys k'_s with uniformly random values $\hat{k_s}'$. We do so by defining a reduction $\mathcal{B}_7$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{ab}}$ and replacing the computation of k'_s in π^t_j and π_{CN} with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{k_s}'$. Since $k_s \leftarrow \mathsf{KDF}(\hat{g^{ab}})$ and by **Game 10** $\hat{g^{ab}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 9** from **Game 10** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{10}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_7,\mathsf{KDF}}(\mathcal{A})$.

Game 12: In this game, we replace the plaintexts in messages $\mathbf{M_5}$, $\mathbf{M_6}$ and $\mathbf{ACK}''$ with uniformly random values of the same length. We do so by defining a reduction $\mathcal{B}_8$ that initialises an AE challenger $\mathcal{C}_{\mathsf{Conf}}$, which $\mathcal{B}_8$ queries with (p, p^*) (where p is the original plaintext message and $p^* \xleftarrow{\$} \{0,1\}^{|p|}$) when it needs to encrypt with $\hat{k_s}'$. By **Game 11** $\hat{k_s}'$ is already uniformly random and independent, and this replacement is sound. If the bit b sampled by $\mathcal{C}_{\mathsf{Conf}}$ is 0, then we are in **Game 11**, otherwise, we are in **Game 12**. Any $\mathcal{A}$ that can distinguish between **Game 11** and **Game 12** can be used by $\mathcal{B}_8$ to break the Conf security of Enc. This implies: $\mathbf{Adv}_{G_{11}} \leq \mathbf{Adv}_{G_{12}} + \mathbf{Adv}^{\mathsf{Conf}}_{\mathcal{B}_8,\mathsf{AE}}(\mathcal{A})$.

Now we highlight that all messages sent across the network to and from π_b and its matching session and subsets are uniformly random and independent of the bit b sampled by the challenger. Thus it follows that $\mathcal{A}$ has no advantage in guessing the bit b, and summing the probabilities $\mathcal{A}$ has a negligible advantage in winning the Unlink game. Thus: $\mathbf{Adv}_{G_{12}} = 0$

5.3.3.2 Unlink-security of Universal Handover

Here we present our formal analysis and results for the Unlink-security of the Universal Handover protocol.

Theorem 15 Unlink***-security of Universal Handover***. *The Universal Handover depicted in Figure 5.3 is unlinkable under the cleanness predicate defined in Def.33. For any PPT algorithm $\mathcal{A}$ against the* Unlink *experiment,* $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the EU-FCMA security of* SanSig*, Confidentiality (IND-CPA) security of* AE*, the KDF security of* KDF *and the* DDH *assumption.*

Proof:
Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{UH},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{UH},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

.

Case 1: The test session π_b (such that $\mathcal{A}$ issues $\mathbf{Test}(i, s, i^*, s^*)$) accepts messages without a matching subset. Here we provide the security analysis of **Case 1** through a series of game-based reductions:

Game 0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}^{G_0}$.

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i, s, i^*, s^*)$ and π_b accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have : $\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{IA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, by **Case 1**, π has no matching session (or subset), and by **Game 1**, we abort if π_b accepts without matching session, it follows that $\mathcal{A}$ can never terminate and output a guess bit b' and thus the Unlink game proceeds identically regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

We turn to bound the advantage of $\mathcal{A}$ in **Case 2**:

Case 2: The test session π_b (such that $\mathcal{A}$ issues $\mathbf{Test}(i, s, i^*, s^*)$) accepts messages with a matching subset. First, we recall that the cleanness predicate defined in Def.26 prevents the $\mathcal{A}$ from issuing a $\mathbf{StateReveal}(i, s, \pi_i^s.\rho)$, nor to any session π_j^t such that π_j^t is a matching subset with π_i^s. Here we provide the security analysis of **Case 2** through a series of game-based reductions:

Game 0: This is the original unlinkability experiment defined in Def.26 of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i, s) \in n_P \times n_S$ of the π_b session, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2: Here we introduce an abort event, where $\mathcal{C}$ aborts if π_b receives a message $\mathbf{M_1}$ without setting $\pi_b.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_b.pid$. We do so by defining a reduction $\mathcal{B}_1$ that initialises a SanSig challenger $\mathcal{C}_{\mathsf{SanSig}}$, that outputs $pk^{\mathcal{C}}_{sig}$ and $pk^{\mathcal{C}}_{san}$, which we embed into the CN's pk^{CN}_{sig} and gNB's pk^{gNB}_{san} respectively. Anytime gNB needs to generate a signature over a message m, $\mathcal{B}_1$ instead queries $\mathcal{C}$ with m. Now, if π_b receives a message $\mathbf{M_1}$ without setting $\pi_b.\alpha \leftarrow$ `rejected` but $\mathbf{M_1}$ was not output by a session owned by $\pi_b.pid$, then $\mathcal{A}$ must have produced a message $\mathbf{M_1} = C^*_G, \sigma^*_G, g^h$ such that $\mathsf{SanSig.Verify}(C^*_G, \sigma^*_G, pk^{\mathcal{C}}_{sig}, pk^{\mathcal{C}}_{san}) = 1$, which is a valid signature forgery. $\mathcal{B}_1$ responds to $\mathcal{C}_{\mathsf{SanSig}}$ with C^*_G, σ^*_G and triggers the abort event. Thus, the probability that $\mathcal{B}_1$ triggers the abort event is bounded by the EUFCMA security of SanSig, as formalised in Def.17. This implies: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{\mathsf{EUFCMA}}_{\mathcal{B}_1,\mathsf{SanSig}}(\mathcal{A})$.

Game 3: In this game, we guess the index $(j, t) \in n_P \times n_S$ of the gNB session π_j^t that output $\mathbf{M_1}$ received by π_b, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_2} \leq n_P \cdot n_S \mathbf{Adv}_{G_3}$.

Game 4: Here we introduce another abort event that triggers if $\mathcal{A}$ sends a Diffie-Hellman public keyshare g^h to the session π_b, i.e. session π_b receives g^h that was not output from a gNB session, but instead from $\mathcal{A}$. Since this trigger event requires the signature σ^*_G in $\mathbf{M_1}$ to verify over g^h, and by **Game 2** we already abort if σ^*_G comes from $\mathcal{A}$, it follows that $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4}$.

Game 5: In this game, we replace g^{hu} computed honestly in π_b with a uniformly random and independent value $g^{\hat{hu}}$. We do so by defining a reduction $\mathcal{B}_2$ that initialises a DDH challenger $\mathcal{C}_{\mathsf{DDH}}$, and replaces g^u, g^h and g^{hu} computed by π_b and π_j^t with the outputs of $\mathcal{C}_{\mathsf{DDH}}$, g^a, g^b, g^c. We note that if the bit b sampled by $\mathcal{C}_{\mathsf{DDH}}$ is 1, then $c = ab$ and we are in **Game 4**. Otherwise, $c \xleftarrow{\$} \mathbb{Z}_q$ and we are in **Game 5**. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can break the DDH assumption, as formalised in Def.10. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{DDH}_{\mathcal{B}_2,\mathsf{DDH}}(\mathcal{A})$.

Game 6: In this game we replace the session and encryption keys sk_i, k_s with uniformly random values $\hat{sk_i}, \hat{k_s}$. We do so by defining a reduction $\mathcal{B}_3$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{g^{uh}}$ and replacing the computation of sk_i, k_s in π_b and π_j^t with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{sk_i}, \hat{k_s}$. Since $sk_i, k_s \leftarrow \mathsf{KDF}(\hat{g^{uh}})$ and by **Game 5** $\hat{g^{uh}}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme, as formalised in Def.2. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we replace the plaintexts in messages $\mathbf{M_2}$, $\mathbf{M_7}$ and $\mathbf{M_8}$ with uniformly random values of the same length. We do so by defining a reduction $\mathcal{B}_4$ that initialises an AE challenger $\mathcal{C}_{\mathsf{Conf}}$, which $\mathcal{B}_4$ queries (p, p^*) (where p is the original plaintext message and $p^* \xleftarrow{\$} \{0,1\}^{|p|}$) when it needs to encrypt with $\hat{k_s}$. By **Game 6** $\hat{k_s}$ is already uniformly random and independent, and this replacement is sound. If the bit b sampled by $\mathcal{C}_{\mathsf{Conf}}$ is 0, then we are in **Game 6**, otherwise, we are in **Game 7**. Any $\mathcal{A}$ that can distinguish between **Game 6** and **Game 7** can be used by $\mathcal{B}_4$ to break the Conf security of Enc, as formalised in Def.6. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{IND-CPA}_{\mathcal{B}_4,\mathsf{AE}}(\mathcal{A})$.

Now we highlight that all plaintext messages sent across the network to and from π_b and its matching subset are uniformly random and independent of the bit b sampled by the challenger. Thus it follows that $\mathcal{A}$ has no advantage in guessing the bit b, and summing the probabilities $\mathcal{A}$ has a negligible advantage in winning the Unlink game. Thus: $\mathbf{Adv}_{G_7} = 0$.

5.4 Performance Evaluation and Comparison

In this section, we conduct a comprehensive evaluation of the proposed scheme in comparison to existing state-of-the-art protocols [4, 31, 37, 47, 71]. Our analysis encompasses three key aspects: feature set comparison, computational overhead, and communication cost. This multifaceted approach allows for a thorough assessment of the scheme's security and efficiency relative to current works in the field.

5.4.1 Security Features Comparison

Our analysis compares the proposed scheme with existing protocols based on the desired security and privacy features discussed in Chapter 3, Section 3.4.1. These features are crucial for overcoming several security and privacy issues found in the current version of 5G protocols and are essential for any AKA and HO protocols in cellular networks. Table 5.2 provides a comparative analysis of the achieved security and privacy properties in UniHand against state-of-the-art protocols [4, 31, 37, 47, 71]. This comparison reveals that existing protocols fail to simultaneously guarantee all required security features, particularly Unlink and Perfect Forward Secrecy (PFS), Key Compromise Impersonation resilience (KCI) and Key Escrow-Free (KEF).

Although all compared schemes provide MA and user anonymity (UA), Unlink is not universally supported, with the 5G protocol [4] and RUSH [71] lacking this feature. Moreover, Table 5.2 further reveal that previous schemes have not fully considered protecting prior

Table 5.2: *Features comparison.*

Features	MA	UA	Unlink	PFS	KEF	KCI	SRM	UHO
5G [4]	✓	✓	✗	✗	✗	✗	✗	✗
CPPHA [31]	✓	✓	✓	✗	✗	✗	✗	✓
ReHand[37]	✓	✓	✓	✗	✗	✗	✓	✗
RUSH[71]	✓	✓	✗	◎	✓	✗	✗	✓
AKA$^+$ [47]	✓	✓	✓	✗	✗	✗	✗	✗
UniHand	✓	✓	✓	✓	✓	✓	✓	✓

MA:Mutual Authentication, **UA**:User Anonymity,
Unlink: Unlinkability, **PFS**: Perfect Forward Secrecy,
KEF: Key Escrow-Free, **KCI**: Key Compromise Impersonation,
SRM: Secure Revocation Management, **UHO**: Universal HO

communications against adversaries with compromised long-term keys (PFS), with RUSH offering only partial support due to its reliance on the 5G-AKA protocol, which does not support PFS. KEF feature is limited to RUSH and UniHand. Notably, UniHand stands alone in its resilience against KCI attacks, addressing a critical vulnerability in current systems. KCI resilience is crucial as it prevents impersonation attacks even if an adversary obtains long-term keys, thereby maintaining system security in worst-case scenarios.

Secure Revocation Management (SRM) is supported solely by ReHand [37] and UniHand, a crucial capability for efficient subscriber management in 5G networks. However, the revocation mechanism in ReHand, based on Nyberg's one-way accumulator [54], employs a static base accumulator, necessitating regeneration for each addition/deletion to the revocation list, thus introducing computational overhead and adversely affecting network efficiency. Universal Handover (UHO) support varies among the protocols. CPPHA [31], RUSH [71], and UniHand offer this feature to enhance seamless connectivity across cells. However, it's worth noting that CPPHA relies on software-defined networks, while RUSH utilises blockchain technology. These approaches may be considered additional technologies or third-party solutions. While they offer certain advantages, it seems that the potential security and privacy implications of these technologies have not been fully addressed in the respective protocols. This reliance on external systems could potentially introduce new vulnerabilities or privacy concerns that may need further investigation. In contrast, UniHand appears to offer UHO support without depending on such additional technologies, which might provide a more self-contained and potentially more secure approach to universal handover.

UniHand provides comprehensive support for all eight security features appears to set it apart from existing protocols. It seemingly combines strong privacy protections (UA, Unlink) with robust key management mechanisms (PFS, KEF, KCI resilience), while also addressing efficient subscriber management (SRM) and seamless connectivity (UHO).

5.4.2 Computational Cost

This section evaluates and compares the performance of the proposed scheme with existing related works in terms of computational cost. It is assumed that all aggregated network entities (CN, gNB) possess higher computational capabilities than the UE. For a fair comparison between related existing work and our proposed scheme, we conducted a theoretical

Table 5.3: *Time costs of cryptography operations.*

Notation	T_{UE}(ms)	T_{Sys}(ms)	**Notation**	T_{UE}(ms)	T_{Sys}(ms)
T_{DH}	0.18	0.09	T_{MAC}	0.195	0.071
T_{SM}	1.148	0.235	T_{H}	0.402	0.089
T_{ELG}	1.176	0.648	T_{PRG}	0.467	0.1273
T_{E}	0.859	0.340	T_{AES}	0.559	0.385
$T_{S.ver}$	0.084	0.046	T_{Mod}	0.0021	0.001
$T_{S.sign}$	1.43	0.87	$T_{S.san}$	0.705	0.384
$T_{Nwit.gen}$	2.29	1.12	$T_{Nwit.ver}$	4.01	2.23
T_{KDF}	0.063	0.022	$T_{DS.sign}$	0.7	0.4
$T_{DS.ver}$	0.04	0.024			

T_{UE}: Computational time on user device, T_{Sys}: Computational time on system, T_{DH}: Elliptic Curve Diffie-Hellman operation, T_{SM}: scalar multiplication, T_{E}: exponentiation operation, T_{MAC}: Massage authentication operations(Hmac-SHA256), T_{H}: Hash operations(SHA-256), T_{PRG}: Random number generators, T_{AES}:Symmetric encryption/decryption operations, T_{ELG}: Elgamal Asymmetric encryption/decryption operations, T_{Mod}: Modular operations, $T_{S.sign}, T_{S.san}, T_{S.ver}$: sanitizable signature, sanitisation and verification operations, $T_{Nwit.gen}, T_{Nwit.ver}$: Non-witness generation and verification T_{KDF}: Key Derivation Function, $T_{DS.sign}, T_{DS.ver}$: RSA digital signature and verification

computation analysis by implementing each cryptographic primitive. Simulations of these cryptographic operations employed by various schemes (including UniHand) were conducted on a Dell Inspiron machine with an i7 core, 2.30GHz CPU and 16.0 GB RAM (operating as the aggregated network entities per the scheme). To measure the computational cost at UE, we employ a smartphone running the Android-10 mobile operating system, equipped with octa-core 1.8GHz Quad-Core ARM Cortex-A55, 2.7GHz Quad-Core Mongoose M3 processors and 6GB RAM. The implementation of the required cryptographic operations for our proposed scheme and the related works utilises the Java Cryptography Extension (JCE) [66] libraries.

Table 5.3 presents the computation cost of the underlying cryptographic primitives, which serves as the basis for measuring the overall computational cost of the protocols. In our implementation of sanitizable signatures and accumulators, we adopted the same methodology discussed in Section 4.4.2. For SanSig we employ a nested RSA signature scheme comprising both outer and inner signature components. As for the accumulator, we have adopted a simplified variant of the dynamic universal accumulator, which is based on RSA cryptographic assumptions. Appendix A contains the complete code implementation and a snapshot of the output.

The computational costs for UniHand scheme demonstrate a balance between security and efficiency. For initial authentication, UniHand shows a moderate computational cost for the UE at approximately 5.42 ms, positioning it between the less demanding conventional 5G [4] protocol and the more computationally intensive RUSH [71]. However, UniHand exhibits the highest system-side (T_{Sys}) computational cost at 8.943 ms, which may be attributed to its comprehensive security features. In terms of universal handover, UniHand shows competitive

Table 5.4: *Performance comparison based on computational cost*

Protocol	Entity	Initial Authentication	Total (ms)	Entity	Universal HO	Total (ms)
Conventional 5G[4]	T_{UE}	$T_{PRG} + T_{MAC} + 3T_{KDF} + T_{ELG}$	≈ 2.03	T_{UE}	$4T_{AES} + 2T_{KDF}$	≈ 2.366
	T_{Sys}	$T_{PRG} + 1T_{MAC} + 2T_H + T_{ELG} + 3T_{KDF}$	≈ 1.09	T_{Sys}	$4T_{AES} + 2T_{KDF}$	≈ 1.60
CPPHA[31]	T_{UE}	$5T_{PRG} + 2T_{MAC} + T_{AES} + T_H$	≈ 3.687	T_{UE}	$2T_{AES} + T_{PRG} + 4T_H$	≈ 3.19
	T_{Sys}	$4T_{PRG} + 1T_{MAC} + 3T_H + 2T_{ELG} + T_{AES}$	≈ 2.536	T_{Sys}	$T_{AES} + T_{PGR} + 6T_H + 3T_{ELG}$	≈ 3.04
ReHand[37]	T_{UE}	$2T_{AES} + 4T_H + T_{PRG}$	≈ 3.19	T_{UE}	$T_{PGR} + 3T_H + T_{AES}$	≈ 2.23
	T_{Sys}	$3T_{AES} + 5T_H + T_{PRG}$	≈ 1.75	T_{Sys}	$T_{PRG} + 5T_H + 2T_{AES}$	≈ 1.359
RUSH[71]	T_{UE}	$4T_{PRG} + T_{MAC} + 3T_{KDF} + 3T_H + T_E + 5T_{Mod} + 3T_{SM} + T_{ELG}$	≈ 8.95	T_{UE}	$3T_{PRG} + T_{SM} + 5T_H + T_E + T_{Mod}$	≈ 5.42
	T_{Sys}	$2T_{PRG} + T_{MAC} + 3T_H + T_{ELG} + 3T_{KDF} + 2T_{SM} + 3T_{Mod}$	≈ 1.78	T_{Sys}	$2T_{PRG} + T_{SM} + 6T_H + T_E + T_{Mod}$	≈ 1.376
UniHand	T_{UE}	$2T_{S.ver} + 2T_{PRG} + T_{KDF} + T_{DH} + 6T_{AES} + T_{DS.sign}$	≈ 5.42	T_{UE}	$T_{S.ver} + 2T_{PRG} + T_{KDF} + T_{DH} + T_{S.sign} + 2T_{AES}$	≈ 3.10
	T_{Sys}	$5T_{PRG} + 2T_{DH} + 2T_{S.san} + 11T_{AES} + 3T_{KDF} + 2T_{S.ver} + 2T_{S.sign} + T_{Nwit.gen} + T_{DS.ver}$	≈ 8.943	T_{Sys}	$T_{S.san} + T_{PRG} + T_{DH} + T_{KDF} + T_{S.ver} + T_{Nwit.ver} + T_{Nwit.gen} + 2T_{AES}$	≈ 4.80

performance with UE-side computation at 3.10 ms, comparable to other protocols and slightly more efficient than CPPHA and RUSH. The system-side of UHO computation for UniHand (4.80 ms) is higher than other protocols, potentially reflecting the trade-off for enhanced security features.

Although these figures suggest some increased computational costs, particularly on the system side, UniHand demonstrates significant improvements in overall network efficiency, especially concerning the CN involvement. UniHand substantially reduces the computational burden on the CN during handovers compared to the conventional 5G-HO protocol [4]. This reduction is crucial given the CN's critical role in mobile networks, managing functions such as connectivity, mobility, authentication, authorization, subscriber data, and policy management. The importance of this reduction becomes apparent when considering the increased handover frequency in 5G networks due to the higher density of small cells. UniHand addresses this challenge by eliminating the need for CN involvement during handover execution. This is achieved because UniHand requires only gNB involvement during handovers, not the Core Network (CN), thus significantly reducing the overall overhead on the CN.

5.4.3 Communication Cost

This section presents analysis and comparison of our proposed scheme with existing handover protocols, including the conventional 5G [4], ReHand [37], RUSH [71] and CPPHA [31], in terms of communication cost. To ensure an accurate evaluation, we consider both the propagation and transmission time of the message size, as well as the network's data rate, to measure the transmission delay across all protocols.

Following the specifications and message sizes introduced in Chapter 4, Section 4.4.3, the analysis adheres to the 3GPP specification [4], assuming a wide-area scenario with an uplink data rate of 25 Mbps and a downlink rate of 50 Mbps. Transmission delays for all protocols are measured using the previously mentioned platforms and the relevant message sizes of the proposed protocol, as detailed in Table 5.5. This table details the sizes of cryptographic components used in transmission delay calculations. Key elements include a 32-byte certifi-

Table 5.5: *Length of parameters.*

Parameters	**Size (in bytes)**
$\mathbb{C}_U : [\text{RUID} \| \text{UID} \| T_{ID} \| T_U]$	$8 \times 4 = 32$
SanSig Signature(σ)	256
ECDH public key size	68
non-membership witness size	32

Table 5.6: *Performance comparison based on communication cost.*

Protocol	**Link**	**Total message size (bits)**	**Transmission time (ms)**	**Propagation time (ms)**	**Total time (ms)**
Conventional 5G[4]	UP	640	0.0256	0.00201	0.02764
	Down	256	0.00512	0.00133	0.00646
CPPHA[31]	UP	1728	0.06912	0.00268	0.0718
	Down	1056	0.02112	0.00133	0.02245
ReHand[37]	UP	832	0.03328	0.00134	0.03462
	Down	768	0.01536	0.00067	0.0398
RUSH[71]	UP	896	0.03584	0.00133	0.03717
	Down	896	0.01792	0.00067	0.01859
UniHand	UP	2848	0.11391	0.00067	0.11458
	Down	3104	0.06208	0.00133	0.06341

cate, a 256-byte sanitizable signature, and a 68-byte ECDH public key using the secp256r1 curve [65]. The resulting 32-byte shared secret (k_s) is derived from the x-coordinate of the computed point. The non-membership witness is 32 bytes. For propagation delay, we use the 3GPP specification [4]: given a wave speed of 3×10^8 m/s and a maximum 5G cell size of 200 m, we calculate a delay of $0.67\mu s$.

Table 5.6 presents a comparative analysis of communication costs across existing handover protocols, including the conventional 5G [4], ReHand [37], RUSH [71] and CPPHA [31]. The conventional 5G approach exhibits the lowest communication cost, with total times of 0.02764 ms for uplink (UP) and 0.00646 ms for downlink (Down). Conversely, UniHand demonstrates the highest communication cost, at 0.11458 ms for UP and 0.06341 ms for Down, primarily due to its larger message sizes. However, this increased cost is mitigated by several significant advantages. UniHand incorporates advanced security and privacy features, enhancing the overall handover process. Moreover, it implements a universal handover mechanism, a characteristic shared by only a few advanced protocols in the field. The distinguishing feature of UniHand lies in its complete elimination of CN communication during the universal handover protocol. This approach enables gNBs to authenticate users autonomously without CN assistance. This elimination of CN assistance results in a substantial reduction of CN overhead. These features collectively contribute to improved overall network architecture efficiency, offsetting the higher communication cost.

5.5 Summary

This chapter has successfully addressed the second research question ($\mathbf{RQ}_2$) of this thesis by introducing UniHand, a universal handover scheme for 5G SCN with KEF and KCI resilience. This solution tackles critical security and privacy challenges in 5G roaming environments. UniHand utilises innovative cryptographic techniques, including sanitizable signatures and universal accumulators, to achieve a comprehensive set of security properties, such as MA, UA, Unlink and PFS. Notably, UniHand is the first scheme to provide resilience against KCI attacks with KEF in this context.

The chapter provided a detailed description of the UniHand protocols for initial authentication and universal handover. Rigorous security proofs were presented to demonstrate the scheme's fulfilment of the desired security and privacy properties. Performance evaluations showed that while UniHand has slightly higher computational/communicational costs in some areas, it significantly improves overall network efficiency by reducing CN involvement during handovers. This makes UniHand particularly well-suited for dense 5G small cell deployments with frequent handovers*.

While UniHand offers significant advancements, it relies on *asymmetric cryptography*. The next chapter explores an alternative approach more aligned with the *symmetric cryptography* primitives predominant in the current 5G infrastructure. It introduces PGUP, an innovative symmetric-based scheme designed to enhance security and privacy in 5G Authentication and handover protocols. Notably, PGUP achieves both PFS and perfect forward privacy within a symmetric cryptography framework, which is considered a notable challenge in the field. These properties enhance the security and privacy of the current 5G-AKA, ensuring past communications and user privacy remains protected even if long-term keys are compromised. The following chapter will detail PGUP's design, security analysis, and performance evaluation, demonstrating its advancements in 5G network security using infrastructure-compatible techniques.

*This work has undergone peer review and has been accepted for publication in the ***IEEE Computer Security Foundations Symposium (IEEE CSF)*** in 2024.

Chapter 6

PGUP: Pretty Good User Privacy for 5G Security

The rapid evolution of cellular network standards has led to the development of 5G technology, which promises unprecedented improvements in network capacity, speed, and latency. However, this advancement brings with it new challenges in security and privacy, particularly in the realms of user authentication and handover procedures. As we continue to explore solutions to these challenges, it becomes crucial to consider approaches that not only enhance security but also align with the existing infrastructure and capabilities of User Equipment (UE). In our ongoing research into secure and private 5G communications, we have explored various approaches. The previous chapter introduced UniHand, a comprehensive universal handover scheme for 5G small cell networks that addressed critical security and privacy challenges through innovative asymmetric cryptographic techniques. While UniHand offers significant advancements, its reliance on asymmetric cryptography necessitates substantial changes to the existing 5G infrastructure.

Building upon these insights, this chapter explores an alternative approach more closely aligned with the symmetric cryptography primitives predominant in current 5G systems. We introduce PGUP (Pretty Good User Privacy), a symmetric-based scheme designed to enhance security and privacy in 5G Authentication and Key Agreement (AKA) and Handover (HO) protocols. PGUP addresses key weaknesses in current 5G security by proposing a symmetric-based primitive called Puncturable Key Wrapping (PKW^{+}), tailored specifically for the 5G environment. This approach allows PGUP to achieve both Perfect Forward Secrecy (PFS) and Perfect Forward Privacy (PFP) within a symmetric cryptography framework, which is considered a notable challenge in the field. In addition, PGUP tackles the significant linkability vulnerability posed by active adversaries, which allows an adversary to track a user's activities throughout their network connection.

By utilizing symmetric cryptography, PGUP aims to strengthen security and privacy aspects without imposing significant overhead on communication parties. This design choice facilitates a smoother transition to enhanced protocols without disrupting the user experience or requiring significant hardware upgrades, addressing the need for solutions compatible with existing infrastructure and UE capabilities.

Motivation and Contributions

The security and privacy of AKA and HO protocols are critical in 5G and mobile communication systems. These protocols ensure confidentiality, user anonymity, and seamless transitions between base stations. However, current solutions fall short in addressing all security and privacy challenges simultaneously, particularly when considering the limited computational capabilities of UE. UEs often operate under significant constraints in terms of processing power, memory, and energy consumption. This limitation makes it crucial to develop security solutions that are not only robust but also computationally efficient. Symmetric cryptography, which forms the basis of existing 5G security protocols, requires significantly less computational resources compared to asymmetric alternatives, making it particularly suitable for resource-constrained UEs.

Our research identifies a critical gap: the need for a comprehensive solution that enhances security and privacy while considering the computational limitations of UEs. Current approaches either fail to address all security challenges or impose substantial computational overhead, potentially compromising the user experience and device performance. To address this gap, we propose PGUP, an innovative symmetric-based scheme designed to enhance 5G-AKA and HO protocols. PGUP utilises a novel Puncturable Key Wrapping (PKW^+) primitive, specifically tailored for the 5G environment. This approach enables PGUP to achieve critical security properties, including PFS, PFP, and user unlinkability, all within a symmetric cryptography framework suitable for UE resource constraints. By leveraging symmetric cryptography, our solution seeks to provide advanced security features while remaining computationally efficient for resource-limited devices.

Our key contributions include:

- The *first* standalone symmetric-based solution achieves PFS, PFP, user unlinkability, secure revocation, and seamless universal handover;
- Design a *new variant* of puncturable key wrapping (i.e., PKW^+), tailored specifically for 5G, to achieve PFS, PFP and unlinkability in a symmetric base setting;
- The proposed solution aims to maintain conceptual alignment with 5G protocol structures, potentially facilitating integration with minimal modifications to existing infrastructure;
- A comprehensive formal security analysis of our proposed scheme;
- A comparative performance evaluation of PGUP with the conventional 5G-AKA and HO protocols demonstrating the cost-effectiveness of the proposed PGUP scheme.

Chapter Organisation

The rest of the chapter is organised as follows:

- Section 6.1 introduces the proposed Puncturable Key Wrapping+ (PKW^+) scheme, detailing its design, security considerations, and formal definitions.
- Section 6.2 outlines the threat model considered for PGUP, describing the capabilities of potential adversaries and the security assumptions.

- Section 6.3 presents the core PGUP scheme, elucidating the authentication and the universal handover protocols.
- Section 6.4 offers a comprehensive security analysis of PGUP, providing formal proofs for mutual authentication, key indistinguishability, and unlinkability properties.
- Section 6.5 evaluates the performance of PGUP, comparing it with existing protocols in terms of security features, computational overhead, and communication costs.
- Finally, the chapter concludes with a summary of the key contributions and findings of the PGUP scheme.

6.1 Proposed Puncturable Key Wrapping

The original Puncturable Key Wrapping (PKW) scheme, as described in Section 2.2.10, offers PFS in symmetric-key environments. However, its direct application to 5G-enabled mobile communication presents substantial challenges, primarily arising from potential security reductions associated with key-reuse [55]. This section outlines these challenges and introduces our proposed solution: the PKW^{+} scheme.

6.1.1 Security Considerations and Potential Vulnerabilities

Cryptographic algorithms are designed with specific security properties and underlying assumptions. Applying a key designed for one algorithm in a different context may introduce vulnerabilities, potentially compromising the overall security of the system. To illustrate this concern, consider the following use case in the 5G-AKA protocol, as depicted in Figure Chapter 3- Figure 3.5. In the 5G-AKA protocol, a single key K serves multiple purposes. If the PKW scheme were applied directly within the 5G-AKA protocol, this same key K would be employed across various cryptographic operations, potentially leading to security vulnerabilities. Specifically, K would be used for:

1. PKW : Wrapping/encrypting the user's identity (SUPI);
2. KDF: Deriving secondary keys (AK, HK, K_{SEAF});
3. MAC and Hash: Generating the Message Authentication Code (MAC) and the expected response ($xRES^{*}$), respectively.

This multifunctional use of K across various cryptographic schemes may introduce potential vulnerabilities. An attacker who successfully compromises the derived keys (e.g., AK, HK, K_{SEAF}) might gain partial information about K, potentially weakening the security of the Subscription Concealed Identifier (SUCI). Conversely, if K were to be compromised, it could lead to the exposure of the derived keys, undermining the security of multiple protocol components simultaneously. This interdependence of security properties, stemming from the use of a single key, could significantly impact the overall strength and resilience of the protocol.

6.1.2 The PKW$^+$ Solution

To address the security concerns associated with key-reuse in the original PKW scheme, we propose PKW$^+$, a new variant of PKW specifically tailored for the 5G environment. PKW$^+$ retains the essential functionalities and PFS properties of PKW while incorporating key derivation capabilities, effectively mitigating the risks associated with key-reuse.

The key modifications in PKW$^+$ include:

- Integration of a Key Derivation Function (KDF) into the existing Wrap and Unwrap algorithms.
- Introduction of a new algorithm (Drv) for deriving keys for use in other contexts.

These enhancements, illustrated in Figure 6.1, strengthen the overall security of the scheme. The PKW$^+$ scheme incorporates a separate key derivation function, which effectively separates the key used for wrapping from the keys used in other parts of the protocol:

- The long-term key K is used exclusively for PKW$^+$ scheme algorithms (Wrap, Unwrap, Drv, and Punc).
- Separate derived keys are employed for other cryptographic algorithms within the protocol.

This separation ensures mutual protection: compromise of the derived keys reveals no information about the long-term key K, and compromise of K reveals nothing about the derived keys. By introducing PKW$^+$, we maintain the PFS guarantees of the original PKW while mitigating the risks associated with *key-reuse*. This approach aligns with the principle of key separation in cryptographic protocol design [55], enhancing the overall security of the 5G-AKA protocol.

Definition 34 (Puncturable Key Wrapping$^+$) *The* PKW$^+$ *consists of a tuple of five algorithms, denoted as* PKW$^+$*:*{KGen, Wrap, Unwrap, Punc, Drv}*, associated with five sets: the secret-key space $\mathcal{SK}$, the tag space $\mathcal{T}$, the additional data space $\mathcal{AD}$, message space $\mathcal{M}$, and the wrap-key space $\mathcal{K}$.*

- KGen$() \rightarrow sk$*: a probabilistic algorithm that takes no inputs and outputs a secret key* $sk \in \mathcal{SK}$.
- Wrap$(sk, T, AD, m) \rightarrow C/\perp$*: a probabilistic wrapping algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a tag* $T \in \mathcal{T}$*, an additional data* $AD \in \mathcal{AD}$*, and a message* $m \in \mathcal{M}$ *and outputs either a ciphertext* $C \in \{0,1\}^*$ *or* $\perp$ *for failure.*
- Unwrap$(sk, T, AD, C) \rightarrow m/\perp$*: a deterministic unwrapping algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *, a tag* $T \in \mathcal{T}$*, an additional data* $AD \in \mathcal{AD}$*, and a ciphertext* $C \in \{0,1\}^*$ *and outputs either a message* $m \in \mathcal{M}$ *or* $\perp$ *for failure.*
- Punc$(sk, T) \rightarrow sk'$*: a deterministic puncturing algorithm takes an input of a secret key* $sk \in \mathcal{SK}$ *and a tag* $T \in \mathcal{T}$ *and returns an updated secret key* $sk' \in \mathcal{SK}$.
- Drv$(sk, T, AD) \rightarrow k$*: a deterministic key derivation function algorithm takes an input of a secret key* $sk \in \mathcal{SK}$*, a tag* $T \in \mathcal{T}$ *and additional data* $AD \in \mathcal{AD}$ *and returns a derived secret key* k.

Correctness: The PKW^+requires that the wrapped message can be successfully unwrapped from the wrapping ciphertext, even if the secret key has been punctured on various tags, except for the specific tag utilized in the wrapping of that particular message. Formally, we require that for all $T \in \mathcal{T}, AD \in \mathcal{AD}, m \in \mathcal{M}$, where $\overline{T}_1, \overline{T}_2 \in \mathcal{T}^*$ and $T \neq (\overline{T}_1 \cup \overline{T}_2)$,

$$\mathbf{Pr}[\mathsf{Unwrap}(sk_{\setminus \overline{T}_1}, T, AD, \mathsf{Wrap}(sk_{\setminus \overline{T}_2}, T, AD, m)) = m | sk \xleftarrow{\$} \mathsf{KGen}()] = 1 \tag{6.1}$$

Similarly, PKW^+requires that the derivation algorithm drives the same key regardless of the punctured state of the secret key sk, as shown in Def. 36. sk is the secret key generated by puncturing on $(T_1, ..., T_n \in \mathcal{T})$, where puncturing on sk is not affected by the order of T (PPRF *puncture invariance* Def. 15). We also redefine this concept for the PKW^+,

Definition 35 (PKW^+puncture invariance) *The $\mathsf{PKW}^+$is puncture invariance for all secret keys $sk \in \mathcal{SK}$ and all tags $T_i \in \mathcal{T}$, where $i \in \{1, \ldots, n\}$, and the sk is solely influenced by the tags on which punctures have occurred, with no regard for the sequence in which these punctures were executed.*

$$\mathsf{Punc}(\mathsf{Punc}(sk, T_0), T_1) = \mathsf{Punc}(\mathsf{Punc}(sk, T_1), T_0) \tag{6.2}$$

Similarly, the key derivation and wrapping remain consistent regardless of the punctured state of the key,

Definition 36 (PKW^+consistency) *The $\mathsf{PKW}^+$is consistent if the output of the derivation and wrapping algorithms solely depends on the tag, additional data and the wrapped message but not the punctured state of the secret key unless the output is $\perp$ due to puncturing. Thus, if all messages $m \in \mathcal{M}$, additional data $AD \in \mathcal{AD}$ and all tags $(T_1, .., T_n) \in \mathcal{T}^*$ and $T \in \mathcal{T}$, where $T \neq (T_1, .., T_n)$, it holds that,*

$$\mathbf{Pr}[\mathsf{Drv}(sk, T, AD) = \mathsf{Drv}(sk_{\setminus (T_1, \ldots, T_n)}, T, AD) | sk \xleftarrow{\$} \mathsf{KGen}()] = 1 \tag{6.3}$$

$$\mathbf{Pr}[\mathsf{Wrap}(sk, T, AD, m) = \mathsf{Wrap}(sk_{\setminus (T_1, \ldots, T_n)}, T, AD, m) | sk \xleftarrow{\$} \mathsf{KGen}()] = 1 \tag{6.4}$$

6.1.3 PKW^+ Security

In accordance with the work of Rogaway and Shrimpton [59], we adopt the concept of "DAE security" (Deterministic Authenticated Encryption), which combines confidentiality and integrity notions. They demonstrate that this combined notion is equivalent to two separate notions in their context, as well as in authenticated encryption overall. Here, we extend this finding to the PKW^+ context and formally validate that a similar equivalence exists for our forward security notions. This combined confidentiality and integrity framework, outlined in Figure 6.2, captures *ind*\$*-cca* and KIND. It also ensures forward security, meaning that confidentiality assurances persist even after compromising the secret key, provided it has been appropriately punctured before corruption to prevent trivial wins. Our combined security notion (confidentiality and integrity) is tailored to the PKW^+ setting. In-game $\mathbf{G}^{\mathbf{ind\$-cca}}_{\mathsf{PKW}^+}(\mathcal{A})$, the adversary $\mathcal{A}^{\mathcal{Q}}$ has access to a set of algorithms

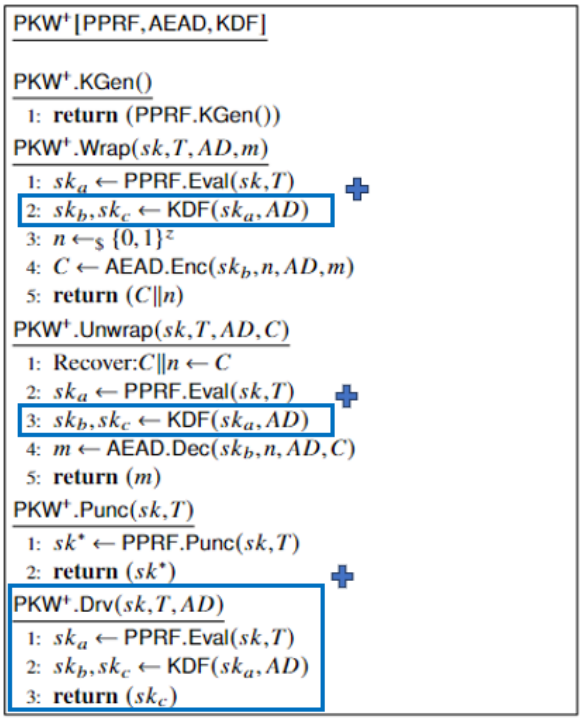

Figure 6.1: PKW$^+$ *algorithms. Blue boxes represents the modifications* PKW$^+$ *introduces to* PKW .

$\mathcal{Q}$ = {New, Wrap, Wrap**$**, Unwrap, Punc, Drv, **Corrupt}**, Wrap$, which, given a key index i, a tag T, additional data AD, and a message m chosen by the adversary, responds with either a legitimate wrapping of m under the secret key sk_i or a random bit-string of length $|m|$. Similarly, the adversary can use real wrappings and key derivation, i.e. Wrap, Unwrap and Drv, without the need for puncturing through an additional oracle Wrap. However, in the case of key indistinguishability ($\mathbf{G}^{\mathbf{KIND}}_{\mathsf{PKW}^+}(\mathcal{A})$), the adversary has access to $\mathcal{Q}$ = {New, Wrap, Punc, Drv, Drv$}, where (Drv$) is challenge key deriving oracle, which, given a key index i, a tag T and additional data AD, responds with either a legitimate wrapping of k under the secret key sk_i or a random bit-string of length $|k|$.

The idea behind this approach is to ensure forward security for all messages wrapped using keys that have been punctured at the time of compromise. It also aims to prevent any potential information leakage from unpunctured ciphertexts that the adversary gains insight into during corruption. In other words, we aim to guarantee a degree of independence among key wrappings produced with distinct tags. The concept of forward security is implemented through a corruption oracle **Corrupt**, allowing the adversary to compromise the current version of a secret key sk_i, provided that all tags used in challenge queries under sk_i must be punctured at the time of corruption, as dictated by the puncturing oracle Punc. Furthermore, in this context, we explicitly treat ciphertexts under punctured tags as valid forgery attempts, even if they were previously produced by Wrap. This ensures that once a tag is punctured, no ciphertext with that tag will be accepted, providing a form of replay protection. The adversary wins if he is able to create a ciphertext (along with a T and AD)

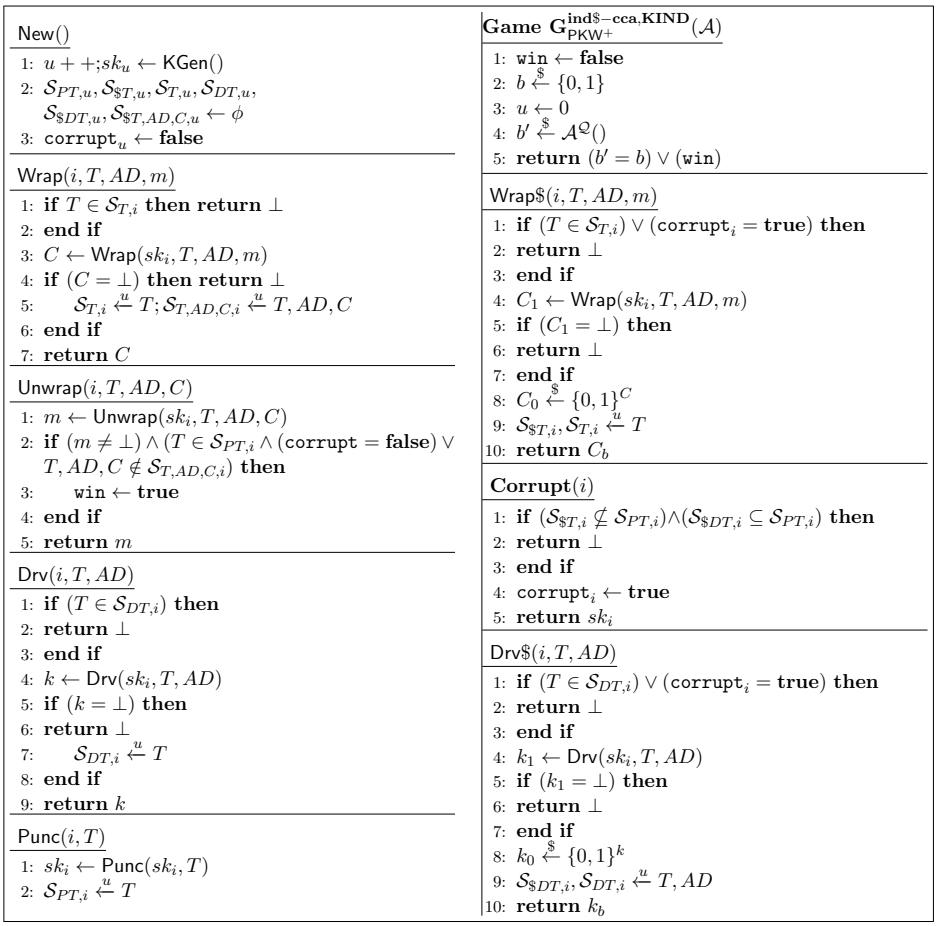

Figure 6.2: *PKW+ security*

that was not generated by Wrap or for which the tag was punctured through Punc, and that, upon unwrapping, does not result in $\perp$.

Definition 37 (PKW^+confidentiality and integrity) *Assuming PKW^+ is a puncturable key-wrapping scheme, we specify the advantage for a PPT algorithm $\mathcal{A}$ has in terms of forward indistinguishability as:*

$$\mathbf{Adv}^{ind\$-cca,\mathsf{KIND}}_{\mathsf{PKW}^+}(\mathcal{A}) = 2|\mathbf{Pr}[\mathbf{G}^{\mathbf{ind\$-cca},\mathsf{KIND}}_{\mathsf{PKW}^+}(\mathcal{A}) \Rightarrow \mathbf{true}] - \frac{1}{2}| \tag{6.5}$$

, where $\mathbf{G}^{\mathbf{ind\$-cca},\mathsf{KIND}}_{\mathsf{PKW}^+}$, is defined in Figure 6.2. We say that $\mathsf{PKW}^+$is (ind\$-cca, KIND) secure if, for all $\mathcal{A}$, the advantage is negligible in the security parameter λ.

6.1.4 Instantiation of PKW^+

Here, we present the construction framework for a PKW^+ scheme, illustrated in Figure 6.1. This framework utilizes an Authenticated Encryption scheme with Associated Data (AEAD) to ensure a secure encryption of messages. Each wrapping algorithm involves a secret key, additional data (AD), a tag (T), and the message to be wrapped. The AEAD secret key is generated by a KDF, which takes as input a key produced by PPRF and AD. The PPRF key is, in turn, generated using the secret key and tag. Additionally, the introduced derivation algorithm in PKW^+ (Drv()) utilizes the key generated by the PPRF, combined with the wrap tag, as input to the KDF, producing a new key for future use. This design approach facilitates the forward security of secret keys by puncturing the PPRF key, ensuring the unrecoverability of ciphertexts.

Next, we demonstrate that, under specific attributes of the foundational PPRF, AEAD and KDF schemes, our design $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$ attains both puncture invariance and consistency (as per Theorem 16). Additionally, it achieves forward indistinguishability contingent upon the inherent strength of the PPRF (Theorem 18) and ensures the confidentiality of ciphertexts (Theorem 17).

Theorem 16 *(PKW^+ is consistent and puncture invariant) The new key-wrapping scheme $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$, illustrated in Figure 6.1, is consistent, as specified in Def. 36. Furthermore, assuming the puncture invariance property of the PPRF, defined in Def. 15, it follows that $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$ also holds the property of puncture invariance, in accordance with Def. 35.*

Proof: The puncture invariance of $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$ is a direct consequence of the puncture invariance inherent in PPRF. The consistency of $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$ is based on several crucial components: Firstly, the correctness of PPRF guarantees that its evaluation at a specific point remains unchanged even when other points are punctured, ensuring that the KDF keys derived from a tag T remain unaffected by the puncturing of other tags. Secondly, the security of KDF guarantee the robustness of the keys generated for both AEAD encryption and future processes, none of which are influenced by the puncturing of other tags. Lastly, the encryption algorithm in the AEAD scheme ensures that the ciphertext relies solely on the inputs to AEAD encryption. Collectively, these factors conclusively establish the consistency and puncture invariance of $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$.

To accomplish this, we refer to the correctness definition of PPRF in Def. 13, where sk_n is derived by executing $sk_i \xleftarrow{\$} \mathsf{PPRF.Punc}(sk_{i-1}, T_i)$ for $i \in \{1, \ldots, n\}$. Now, to ensure consistency, the requirement precisely states that for all $AD \in \mathcal{AD}$ and all $m \in \mathcal{M}$,

$$\mathbf{Pr}[\mathsf{Wrap}(sk_0, T, AD, m) = \mathsf{Wrap}(sk_n, T, AD, m) | sk_0 \xleftarrow{\$} \mathsf{KGen}()] = 1 \tag{6.6}$$

Thanks to PPRF correctness this condition is satisfied for PKW^+[PPRF, KDF, AEAD], as per the definition of the construction, where:

1. $\mathsf{PKW}^+.\mathsf{KGen}() := \mathsf{PPRF.KGen}()$
2. $\mathsf{PKW}^+.\mathsf{Punc}(sk, T) := \mathsf{PPRF.Punc}(sk, T)$
3. $\mathsf{PKW}^+.\mathsf{Drv}(sk, T, AD) := \mathsf{KDF}(\mathsf{PPRF.Eval}(sk, T), AD)$
4. $\mathsf{PKW}^+.\mathsf{Wrap}(sk_0, T, AD, m) :=$
   $\mathsf{AEAD.Enc}(\mathsf{KDF}(\mathsf{PPRF.Eval}(sk_0, T), AD), T, AD, m) =$
   $\mathsf{AEAD.Enc}(\mathsf{KDF}(\mathsf{PPRF.Eval}(sk_n, T), AD), T, AD, m)$
   $:= \mathsf{PKW}^+.\mathsf{Wrap}(sk_n, T, AD, m)$

Theorem 17 *(PKW^+ is ind\$−cca secure) Consider the new key-wrapping scheme* PKW^+[PPRF, KDF, AEAD] *illustrated in Figure 6.1. For any adversary $\mathcal{A}$ attempting to compromise the ind\$ − cca security of* PKW^+[PPRF, KDF, AEAD]*, defined in Def. 37, by querying oracles* New, Wrap\$, Wrap, Unwrap, Punc, ***Corrupt*** *and* Drv *there exist corresponding adversaries ($\mathcal{B}_{pprf}, \mathcal{B}_{kdf}$, and $\mathcal{B}_{aead}$) that operate approximately at the same time as $\mathcal{A}$, such that:*

$$\mathbf{Adv}^{ind-cca}_{\mathsf{PKW}^+}(\mathcal{A}) \leq \mathbf{Adv}_{\mathsf{PPRF}}(\mathcal{B}_{pprf}) + \mathbf{Adv}_{\mathsf{KDF}}(\mathcal{B}_{kdf}) + \mathbf{Adv}_{\mathsf{AEAD}}(\mathcal{B}_{aead}) \tag{6.7}$$

Proof: To prove theorem 17, we divide it into two cases, denoted by $\mathbf{Adv}^{c_1}_{\mathsf{PKW}^+}(\lambda)$ and $\mathbf{Adv}^{c_2}_{\mathsf{PKW}^+}(\lambda)$. In c_1, we demonstrate that PKW^+ ensures the integrity of ciphertexts by proving that the probability of the adversary $\mathcal{A}$ winning in game $\mathrm{G}(ind\$ - cca)$ is negligible. Similarly, in c_2, we establish that PKW^+ provides confidentiality, ensuring that ciphertexts are indistinguishable from random strings. As a result, the probability of the adversary $\mathcal{A}$ correctly guessing the bit b in game $\mathrm{G}(ind\$ - cca)$ is also negligible. We then bound the advantage of $\mathcal{A}$ winning the game to:

$$\mathbf{Adv}^{ind\$-cca}_{\mathsf{PKW}^+}(\mathcal{A}) \leq \mathbf{Adv}^{C_1}_{\mathsf{PKW}^+}(\mathcal{A}) + \mathbf{Adv}^{C_2}_{\mathsf{PKW}^+}(\mathcal{A})$$

.

Case 1: Let game G_0 be equivalent to $\mathrm{G}(ind\$-cca)$, with the algorithms of PKW^+[PPRF; KDF; AEAD] implemented directly using the underlying PPRF, KDF and AEAD schemes. We begin by exploiting the fpr security property of PPRF and substitute the output of PPRF keys with random keys. Subsequently, in the following Game, we utilize the randomized PPRF keys generated previously to derive keys using $\mathsf{PKW}^+.\mathsf{Drv}()$, arguing that the resulting keys are indistinguishable from random keys. Next, we exploit the unforgeability/ integrity security of AEAD ciphertexts and substitute the output of ciphertext with random, and we argue that the $\mathcal{A}$ has a negligible advantage of forging a valid ciphertext. The reduction works as follows:

Game 0: This the original game $ind\$ - cca$ described in Figure 6.2: $\mathbf{Adv}^{ind\$-cca}_{\mathsf{PKW}^+}(\mathcal{A}) \leq \mathbf{Adv}_{G_0}$

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ sends $\mathsf{Unwrap}(j, ..)$ query under index j, where $j \in \{0, \ldots, n\}$ and setting `win` to **true**. We begin this game by guessing the key index (i) and ($i \neq j$) of the first successful forgery (i.e. `win` = **true**). Thus, $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this security game, we exploit the fpr security property of PPRF by replacing the output of PPRF key (i.e. sk_a in PKW^+ construction) with a random key. To accomplish this, we introduce a reduction $\mathcal{B}_1$ which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PPRF challenger $\mathcal{C}_{\mathsf{PPRF}}$. $\mathcal{B}_1$ simulates ($G^{fpr-ro\$}_{\mathsf{PPRF}}$). For all key indices $j \neq i$, $\mathcal{C}_{\mathsf{PPRF}}$ generates sk_j and responds to queries using real oracles. For key index i $\mathcal{B}_1$ uses $Ro\$ - \mathsf{Eval}$ to generate keys for Wrap, Unwrap and Drv queries. Specifically: Upon receiving wrap/unwrap queries $\mathsf{Wrap}/\mathsf{Unwrap}(i, T, AD, m/c)$ from $\mathcal{A}$, $\mathcal{B}_1$ queries $\mathcal{C}_{\mathsf{PPRF}}$ to obtain a real or random k_i using $Ro\$ - \mathsf{Eval}(i, T)$. Subsequently, $\mathcal{B}_1$ utilises the output key from $\mathcal{C}_{\mathsf{PPRF}}$ to generate another key using KDF. This generated key is then employed to encrypt/decrypt the message/ciphertext using AEAD algorithms. Finally, $\mathcal{B}_1$ replies to the $\mathcal{A}$ with the resulting message/ciphertext. However, when $\mathcal{A}$ calls derive query (Drv), we obtain the random key similarly and use the KDF to output a distinct key that has been used in the AEAD. Note that, the $\mathcal{A}$ cannot distinguish between multiple calls to $RO\$ - \mathsf{Eval}$ as it is handled internally in the PPRF game. The $\mathcal{A}$ simulation in Game 1 is essentially the same as Game 2, with the exception of replacing PPRF keys with random keys. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 1** and **Game 2**, then $\mathcal{A}$ has effectively breaks the fpr security of PPRF, as formalised in Def. 14. Thus: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{fpr}_{\mathcal{B}_1,\mathsf{PPRF}}(\mathcal{A})$

Game 3: In this game, we replace the output of the KDF (i.e. sk_b, sk_c in PKW^+ construction) with uniformly random value. We do so by interacting with a $\mathcal{B}_2$ that interacts with $\mathcal{C}_{\mathsf{KDF}}$ challenger. Similar to Game 2, for key index i $\mathcal{B}_2$ uses the output of KDF to generate keys for Wrap, Unwrap and Drv queries. Whenever sk_a is used in Wrap, Unwrap or Drv to generate sk_b, sk_c, $\mathcal{B}_2$ first we check if Wrap, Unwrap and Drv were called since the last puncture ($\mathsf{Punc}(i, ..)$) query. If this condition holds, $\mathcal{B}_2$ then calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $\hat{sk_b}, \hat{sk_c}$ and add the output to the lookup table. Otherwise we do a lookup for $\mathrm{T}[sk_a, AD]$ if it returns $\perp$, then $\mathcal{B}_2$ calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $\hat{sk_b}, \hat{sk_c}$ and add the output to the lookup table. Alternatively, $\mathcal{B}_2$ uses *out* as output values. since by **Game 2** the sk_a is already uniformly random, this change is sound. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 2** and **Game 3**, then $\mathcal{A}$ has effectively breaks the security of KDF, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{KDF}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$

Game 4: Here, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C that decrypts/unwraps correctly and was not generated by Wrap, breaking the unforgability/ integrity security of AEAD ciphertexts. We do so by defining a reduction $\mathcal{B}_3$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$. Upon receiving wrap/unwrap queries $\mathsf{Wrap}/\mathsf{Unwrap}(sk, T, AD, m/c)$ from $\mathcal{A}$, $\mathcal{B}_3$ first check if Wrap, Unwrap were called since the last puncture ($\mathsf{Punc}(i, ..)$) query, if not we retrieve the previously computed output. Otherwise, $\mathcal{B}_3$ queries $\mathcal{C}_{\mathsf{AEAD}}$ to obtain a real or random ciphertext C using $Ro\$(sk_b, n, AD, m)$. Finally, $\mathcal{B}_3$ replies to the $\mathcal{A}$ with the resulting ciphertext. This way $\mathcal{A}$ wins if it submits a valid forgery (C) to Unwrap, where correctly output m (i.e., $\mathsf{Unwrap}(i, T, AD, C) \rightarrow m : m \neq \perp$), and if either C has never been output from Wrap (i.e $T, AD, C \notin \mathcal{S}_{T,AD,C,i}$) or if tag T was previously punctured

via Punc (i.e., $T \in \mathcal{S}_{PT,i}$). since by **Game 3** the sk_b is already uniformly random, this change is sound. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 3** and **Game 4**, then $\mathcal{A}$ has effectively broken the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_3,\mathsf{AEAD}}(\mathcal{A})$

Since as per Case 1 the $\mathcal{A}$ has a negligible advantage of breaking AEAD security, where we abort if the $\mathcal{A}$ can forge a valid ciphertext, Thus

$$\mathbf{Adv}^{c_1}_{\mathsf{PKW}^+}(\mathcal{A}) = 0$$

We now bound the advantage of $\mathcal{A}$ in **Case 2**.

Case 2: Let game G_0 be equivalent to G($ind\$-cca$), with the algorithms of PKW$^+$[PPRF; KDF; AEAD] implemented directly using the underlying PPRF, KDFand AEAD schemes. we begin by exploiting the forward pseudorandomness (fpr) security property of PPRF and substitute the output of PPRF keys with random keys. Subsequently, in the following Game, we utilize the randomized PPRF keys generated previously to derive keys using PKW$^+$.Drv(), arguing that the resulting keys are indistinguishable from random keys. Following this, we employ the derived KDF keys to encrypt messages using PKW$^+$.Wrap\$(), and we argue that the resulting ciphertext is indistinguishable from random. The reduction works as follows:

Game 0: This the original game $ind\$ - cca$ described in Figure 6.2: $\mathbf{Adv}^{ind\$-cca}_{\mathsf{PKW}^+}(\mathcal{A}) \leq \mathbf{Adv}_{G_0}$

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ sends Unwrap$(j, ..)$ query under index j, where $j \in \{0, \ldots, n\}$ and setting `win` to **true**. We begin this game by guessing the key index (i) and ($i \neq j$) of the first successful forgery (i.e. `win` = **true**). Thus, $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this security game, we exploit the fpr security property of PPRF by replacing the output of PPRF key (i.e. sk_a in PKW$^+$ construction) with a random key. To accomplish this, we introduce a reduction $\mathcal{B}_1$ which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PPRF challenger $\mathcal{C}_{\mathsf{PPRF}}$. $\mathcal{B}_1$ simulates ($G^{fpr-ro\$}_{\mathsf{PPRF}}$). For all key indices $j \neq i$, $\mathcal{C}_{\mathsf{PPRF}}$ generates sk_j and responds to queries using real oracles. For key index i $\mathcal{B}_1$ uses $Ro\$ - \mathsf{Eval}$ to generate keys for Wrap, Unwrap and Drv queries. Specifically: Upon receiving wrap/unwrap queries Wrap/Unwrap$(i, T, AD, m/c)$ from $\mathcal{A}$, $\mathcal{B}_1$ queries $\mathcal{C}_{\mathsf{PPRF}}$ to obtain a real or random k_i using $Ro\$ - \mathsf{Eval}(i, T)$. Subsequently, $\mathcal{B}_1$ utilises the output key from $\mathcal{C}_{\mathsf{PPRF}}$ to generate another key using KDF. This generated key is then employed to encrypt/decrypt the message/ciphertext using AEAD algorithms. Finally, $\mathcal{B}_1$ replies to the $\mathcal{A}$ with the resulting message/ciphertext. However, when $\mathcal{A}$ calls derive query (Drv), we obtain the random key similarly and use the KDF to output a distinct key that has been used in the AEAD. Note that, the $\mathcal{A}$ cannot distinguish between multiple calls to $RO\$ - \mathsf{Eval}$ as it is handled internally in the PPRF game. The $\mathcal{A}$ simulation in Game 1 is essentially the same as Game 2, with the exception of replacing PPRF keys with random keys. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 1** and **Game 2**, then $\mathcal{A}$ has effectively breaks the fpr security of PPRF, as formalised in Def. 14. Thus: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{fpr}_{\mathcal{B}_1,\mathsf{PPRF}}(\mathcal{A})$.

Game 3: In this game, we replace the output of the KDF (i.e. sk_b, sk_c in PKW$^+$ construction) with uniformly random value. We do so by interacting with a $\mathcal{B}_2$ that interacts with $\mathcal{C}_{\mathsf{KDF}}$ challenger. Similar to Game 2, for key index i $\mathcal{B}_2$ uses the output of KDF to generate keys for Wrap, Unwrap and Drv queries. Whenever sk_a is used in Wrap, Unwrap or Drv to generate sk_b, sk_c, $\mathcal{B}_2$ first we check if Wrap, Unwrap and Drv were called since the last puncture

($\mathsf{Punc}(i,..)$) query. If this condition holds, $\mathcal{B}_2$ then calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $\hat{sk_b}, \hat{sk_c}$ and add the output to the lookup table. Otherwise we do a lookup for $\mathrm{T}[sk_a, AD]$ if it returns $\perp$, then $\mathcal{B}_2$ calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $\hat{sk_b}, \hat{sk_c}$ and add the output to the lookup table. Alternatively, $\mathcal{B}_2$ uses *out* as output values. since by **Game 2** the sk_a is already uniformly random, this change is sound. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 2** and **Game 3**, then $\mathcal{A}$ has effectively breaks the security of KDF, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{KDF}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ distinguishes a real ciphertext from a random ciphertext C, breaking the confidentiality security of AEAD. We do so by defining a reduction $\mathcal{B}_3$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$. Upon receiving wrap/unwrap queries $\mathsf{Wrap}/\mathsf{Unwrap}(sk, T, AD, m/c)$ from $\mathcal{A}$, $\mathcal{B}_3$ first check if $\mathsf{Wrap}, \mathsf{Unwrap}$ were called since the last puncture ($\mathsf{Punc}(i,..)$) query, if not we retrieve the previously computed output. Otherwise, $\mathcal{B}_3$ queries $\mathcal{C}_{\mathsf{AEAD}}$ to obtain a real or random ciphertext C using $Ro\$(sk_b, n, AD, m)$. Finally, $\mathcal{B}_3$ replies to the $\mathcal{A}$ with the resulting ciphertext. This way $\mathcal{A}$ wins if he manages to distinguish the result from real or random. since by **Game 3** the sk_b is already uniformly random, this change is sound. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 3** and **Game 4**, then $\mathcal{A}$ has effectively breaks the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}^{unf}_{G_4} + \mathbf{Adv}_{\mathcal{B}_3,\mathsf{AEAD}}$

Here, we emphasise that as a result of these changes, the generated ciphertext is indistinguishable from a random ciphertext, thus $\mathcal{A}$ has a negligible advantage in winning the game:

$$\mathbf{Adv}^{c_2}_{\mathsf{PKW}^+}(\mathcal{A}) = 0$$

Theorem 18 *(PKW^+ is KIND secure) Consider the new key-wrapping scheme $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$ illustrated in Figure 6.1. For any adversary $\mathcal{A}$ attempting to compromise the KIND security of $\mathsf{PKW}^+[\mathsf{PPRF}, \mathsf{KDF}, \mathsf{AEAD}]$, defined in 37, by querying oracles $\mathsf{New}, \mathsf{Wrap}, \mathsf{Punc}, \mathsf{Drv}\$$ and Drv, there exist corresponding adversaries ($\mathcal{B}_{pprf}$ and $\mathcal{B}_{kdf}$) that operate approximately at the same time as $\mathcal{A}$, such that:*

$$\mathbf{Adv}^{\mathsf{KIND}}_{\mathsf{PKW}^+}(\mathcal{A}) \leq \mathbf{Adv}_{\mathsf{PPRF}}(\mathcal{B}_{pprf}) + \mathbf{Adv}_{\mathsf{KDF}}(\mathcal{B}_{kdf}) \tag{6.8}$$

Proof: To prove theorem 18, we demonstrate that PKW^+ ensures key indistinguishability by proving that the probability of the adversary $\mathcal{A}$ winning in game $\mathrm{G}(\mathsf{KIND})$ is negligible.

Game 0: This the original game KIND described in Figure 6.2: $\mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{A}}(\mathcal{A}) \leq \mathbf{Adv}_{G_0}$

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ sends $\mathsf{Unwrap}(j,..)$ query under index j, where $j \in \{0, \ldots, n\}$ and setting `win` to **true**. We begin this game by guessing the key index (i) and ($i \neq j$) of the first successful forgery (i.e. $\texttt{win} = \textbf{true}$). Thus, $\mathcal{A}$'s advantage:$\mathbf{Adv}_{G_0} \leq n \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this security game, we exploit the fpr security property of PPRF by replacing the output of PPRF key (i.e. sk_a in PKW^+ construction) with a random key. To accomplish this, we introduce a reduction $\mathcal{B}_1$ which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PPRF challenger $\mathcal{C}_{\mathsf{PPRF}}$. $\mathcal{B}_1$ simulates ($G^{fpr-ro\$}_{\mathsf{PPRF}}$). For all key indices $j \neq i$, $\mathcal{C}_{\mathsf{PPRF}}$ generates sk_j and responds to queries using real oracles. For key index i $\mathcal{B}_1$ uses $Ro\$ - \mathsf{Eval}$ to generate keys for $\mathsf{Wrap}, \mathsf{Unwrap}$ and Drv queries. Specifically: Upon receiving wrap/unwrap queries $\mathsf{Wrap}/\mathsf{Unwrap}(i, T, AD, m/c)$ from $\mathcal{A}$, $\mathcal{B}_1$ queries $\mathcal{C}_{\mathsf{PPRF}}$ to obtain a real or random k_i using $Ro\$ - \mathsf{Eval}(i, T)$. Subsequently, $\mathcal{B}_1$ utilises the output key from $\mathcal{C}_{\mathsf{PPRF}}$ to

generate another key using KDF. This generated key is then employed to encrypt/decrypt the message/ciphertext using AEAD algorithms. Finally, $\mathcal{B}_1$ replies to the $\mathcal{A}$ with the resulting message/ciphertext. However, when $\mathcal{A}$ calls derive query (Drv), we obtain the random key similarly and use the KDF to output a distinct key that has been used in the AEAD. Note that, the $\mathcal{A}$ cannot distinguish between multiple calls to $RO\$-\mathsf{Eval}$ as it is handled internally in the PPRF game. The $\mathcal{A}$ simulation in Game 1 is essentially the same as Game 2, with the exception of replacing PPRF keys with random keys. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 1** and **Game 2**, then $\mathcal{A}$ has effectively breaks the fpr security of PPRF. Thus: $\mathbf{Adv}_{G_1} \leq \mathbf{Adv}_{G_2} + \mathbf{Adv}^{fpr}_{\mathcal{B}_1,\mathsf{PPRF}}(\mathcal{A})$.

Game 3: In this game, we replace the output of the KDF (i.e. sk_b, sk_c in PKW$^+$ construction) with uniformly random value. We do so by interacting with a $\mathcal{B}_2$ that interacts with $\mathcal{C}_{\mathsf{KDF}}$ challenger. Similar to Game 2, for key index i $\mathcal{B}_2$ uses the output of KDF to generate keys for Wrap, Unwrap and Drv queries. Whenever sk_a is used in Wrap, Unwrap or Drv to generate sk_b, sk_c, $\mathcal{B}_2$ first we check if Wrap, Unwrap and Drv were called since the last puncture ($\mathsf{Punc}(i, ..)$) query. If this condition holds, $\mathcal{B}_2$ then calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $s\hat{k}_b, s\hat{k}_c$ and add the output to the lookup table. Otherwise we do a lookup for $\mathrm{T}[sk_a, AD]$ if it returns $\perp$, then $\mathcal{B}_2$ calls $\mathcal{C}_{\mathsf{KDF}}$ to obtain random keys $s\hat{k}_b, s\hat{k}_c$ and add the output to the lookup table. Alternatively, $\mathcal{B}_2$ uses *out* as output values. since by **Game 2** the sk_a is already uniformly random, this change is sound. Therefore, if $\mathcal{A}$ manages to distinguish between **Game 2** and **Game 3**, then $\mathcal{A}$ has effectively breaks the security of KDF, thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{KDF}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$

Here we emphasise that as a result of these changes, the generated key in **Game 3** is now uniformly random and independent regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b:

$$\mathbf{Adv}^{\mathsf{KIND}}_{\mathsf{PKW}^+}(\mathcal{A}) = 0.$$

6.1.5 PKW$^+$ Performance

We also evaluated the performance of our new PKW$^+$ scheme against the original PKW*. Table 6.1 shows the execution times for key operations in both schemes.

Table 6.1: *Performance comparison of PKW and* PKW$^+$

Function	**PKW** (ms)	**PKW$^+$** (ms)	**Difference**
Wrap$(\cdot)$	1.1895	1.21468	+2.12%
Unwrap$(\cdot)$	1.14117	1.23913	+8.58%
Punc$(\cdot)$	2.75626	2.7733	+0.62%
Drv$(\cdot)$	-	0.00537	N/A
Total	5.08693	5.23248	+2.86%

As shown in Table 6.1, PKW$^+$ introduces a slight overhead compared to the original PKW, with a total increase in execution time of 2.86%. This minor performance cost is primarily due to the addition of the Drv function and small increases in Wrap and Unwrap times. However, the enhanced security features provided by PKW$^+$, including improved key separation and reduced risks associated with key-reuse, justify this modest performance trade-off. Notably,

*The source code for PKW$^+$ implementation can be found here.

the Punc algorithm has not been modified in PKW^+, and its slight increase is not directly related to the new features implemented in PKW^+. Thus, PKW^+ achieves enhanced security with no impact on this critical algorithm.

6.2 Threat Model

Our security model incorporates concepts from both the 3GPP group [4] and recent literature [56, 15, 33] for 5G authentication and handover protocols. It also addresses the desirable security and privacy properties identified in Section 3.4.1. Throughout the execution of the proposed scheme, all communication channels between the user and the network are public, signifying that the adversary has complete control over these public channels. For the channels connecting the gNBs and the CN (specifically the N2 interface), TS 33.501 [4] explicitly defines security requirements including confidentiality and integrity protection of signaling and user data, as well as authentication between the gNBs and the CN.

Our threat model encompasses three distinct types:

1. $\mathcal{A}_\mathbf{1}$: A Dolev-Yao adversary that possesses control over the network, allowing for the interception, insertion, modification, and deletion of any message.
2. $\mathcal{A}_\mathbf{2}$: Seeks to compromise the key-indistinguishability and linkability properties of the communicating parties. It is important to consider this adversary to protect user privacy, prevent tracking, and ensure robust key management.
3. $\mathcal{A}_\mathbf{3}$: Focuses on compromising forward Secrecy and forward privacy. This adversary has the ability to compromise various secret keys, including:
   (a) Long-term keys (LTK) shared between UE and CN
   (b) Asymmetric secret keys (ASK) of the CN
   (c) Session keys established between protocol participants

Protecting against $\mathcal{A}_3$ adversary ensures that protocols can safeguard past communications even if future key compromises occur, maintaining long-term security of the system.

6.3 Proposed PGUP Scheme

In this section, we introduce PGUP scheme that consists of an *authentication and key agreement* protocol and *handover* protocol. We provide the notation used in PGUP scheme in Table 6.2.

6.3.1 Authentication and Key Agreement

All registered users in the network seeking secure access are required to execute this protocol. While we adhere to a similar registration procedure as the current 5G-AKA, our approach differs in the method of SUPI (Subscription Permanent Identifier) protection. In 5G-AKA, this protection is achieved through encrypting the SUPI. In our protocol, we introduce SUTI (Subscription Temporary Identifier), an independent value from SUPI, which provides an extra layer of user identity protection. SUTI is a temporary random identifier generated

Table 6.2: *Notation*

Parameter	Content/Description
R	128 bit Random Number
SQN	48 bit Sequence Number
AK,MK	$f_5(rk, R)$
RK,SK	$f_3(AK, SUTI)$
MAC	$f_1(MK, SQN_{\text{CN}} \| R \| SUCI)$
SUTI	Subscription Temporary Identifier
CONC	$SQN_{\text{CN}} \oplus AK$
AUTN	$< CONC, MAC >$
$xRES^*$	$KDF(RK, R \| \text{gNB}_{name})$
$HxRES^*$	$SHA256(R \| xRES^*)$
HK, K_{SEAF}	$KDF(SK; R \| \text{gNB}_{name} \| SQN_{\text{CN}})$
ek, tk	$f_5(HK, \theta)$
HK^*	$f_3(tk, r)$
$hk, \widehat{HK}$	$f_5(HK^*, t)$
ck, C_0	KEM public and private keys
T	a tag $T \in \mathcal{T}$

by the CN and maintained by both the CN and the UE. This approach enhances privacy by further obfuscating the user's permanent identity, going beyond the protection offered in standard 5G-AKA. Following the 5G specifications (TS 33.501) [4], we assume that the public key of the CN is preserved in the USIM. This assumption is crucial for the security of the authentication process. Below, we present a detailed description of the sequence of messages essential for the PGUP-AKA protocol, as illustrated in Figures 6.3 and 6.4.

Step 1: UE $\rightarrow$ CN : [SUCI, ID_{CN}]
The proposed AKA follows a similar pattern to the conventional 5G-AKA in its initial steps. It begins by using the same ECIES key encapsulation algorithm ($\mathsf{KEM.Encaps}$) used in 5G-AKA to generate keys (ck, C_0). Subsequently, the UE introduces a random salt (Δ) to update its temporary identity (SUTI) by computing $\text{SUTI}^* = \mathsf{Hash}(\text{SUTI} \| \Delta)$. Subsequently, the UE encrypts the original SUTI using ck, producing ciphertext denoted as C_1. Next, the UE generates a tag $(T \xleftarrow{\$} \mathcal{T})$ and wrap the new temporary identity (SUTI^*) using ($\mathsf{PKW^+.Wrap}$) algorithm with the long-term key K, a tag T, and additional data $(AD \leftarrow C_0 \| C_1))$ to form C_2. Next, we derive a key (rk) using the ($\mathsf{PKW^+.Drv}$) algorithm and update the long-term key by puncturing it on the tag T. Finally, compose a SUCI which consists of (C_2) and send them along with the ID of the supported CN to the gNB, which he will forward to the specified CN.

Step 2: CN $\rightarrow$ gNB : $[R', AUTN, HxRES^*, K_{SEAF}, HK]$
In this step, the process closely resembles the conventional 5G-AKA, wherein the CN employs the ECIES key decapsulation algorithm ($\mathsf{KEM.Decaps}$) to generate ck, mirroring the UE's action. This key is then utilised to decrypt C_1, facilitating the retrieval of (SUTI). The SUTI serves as an index, allowing the CN to identify and access the corresponding long-term key K. The CN derives rk using $\mathsf{PKW^+.Drv}(K, T, C_0 \| C_1)$ and unwraps C_2 to obtain $\text{SUTI}^* \| \Delta$. Then, the CN performs a key update (K^*) by

puncturing on K using the tag T. The CN checks if SHA256(SUTI) equals SUTI* and updates SUTI to SUTI*. Following this, the CN randomly selects R and encrypts it using ck (to prevent linkability attacks). The subsequent steps closely mirror those of the conventional 5G-AKA, incorporating minor adjustments for the generation of secure values. By utilising the f_5 function, the CN derives the authentication key (AK) and MAC key (MK). Subsequently, employing f_3, the CN generates the session key (SK) and response key (RK). Further, the MK is utilised in f_1 to generate the Message Authentication Code (MAC). The CN then obfuscates the sequence number through an XOR operation with AK. Subsequently, the CN employs the KDF to generate the expected response ($xRES$) and hashes it using SHA256. Crucially, the CN generates both the session key and handover key, denoted as k_{SEAF} and HK, respectively, facilitating communication between the gNB and the user. Next, the CN increments the sequence number (SQN). Finally, the CN sends $(R^{'}, AUTN, HxRES^*, K_{SEAF}, HK)$ to the gNB, which he/she forwards $(R^{'}, AUTN)$ only to the UE.

Step 3: UE $\rightarrow$ gNB : $[RES^*]$
Upon message reception, the UE decrypts $R^{'}$ and generates authentication and MAC keys (AK, MK) using the received R and rk. Extracting $(xCONS, XMAC)$ from the Authentication Token ($AUTN$). UE deconceals ($xCONC$) to retrieve (SQN_{CN}) through XOR operations. Next, UE uses MK to generate a MAC for the $xSQN_{\text{CN}}, R$ and $SUCI$ using (f_1) function. Then UE compares the generated MAC with $XMAC$ and SQN_{CN} with SQN_{UE}, if both verifications hold UE computes the response key RK and session key SK using the authentication key ($f_3(AK, \text{SUTI})$). Then UE generate another set of keys, K_{SEAF} and HK keys to be used during future communications and in handover protocol.

Step 4: gNB $\rightarrow$ CN : $[RES^*, SUCI]$
Upon receiving RES^*, the gNB compares $HxRES$ with the hash value of RES^*. If both hash values are equal, the gNB sends RES^* together with the corresponding $SUCI$ to the CN. The rest of the protocol then proceeds according to 5G-AKA.

Remark 2 *We consistently update the long-term key (K) shared between the* UE *and the* CN *during each execution. These updates are achieved through the* $\mathsf{Punc}()$ *algorithm, which serves two critical purposes. Firstly, it ensures PFS, guaranteeing that the confidentiality of past communications is maintained even if the current key is compromised. Secondly, it significantly mitigates risks associated with key-desynchronization issues between the* UE *and* CN*. This mitigation is effective due to the puncture invariance of* $\mathsf{PKW}^+.\mathsf{Punc}()$*, defined in Def. 35. This property guarantees that the order of punctures does not affect the outcome, resulting in consistent key derivation regardless of potential message reordering or loss. Consequently, this characteristic strengthens the robustness of the key update mechanism.*

6.3.1.1 Integration of PGUP-AKA Protocol with 5G-AKA

It's important to highlight that the PGUP-AKA protocol follows exactly the same message call flow as the current 5G-AKA protocol: identical messages are delivered in the same sequence. The only difference is the content of these messages. This alignment demonstrates

the seamless and straightforward integration of the PGUP-AKA protocol into the 5G environment.

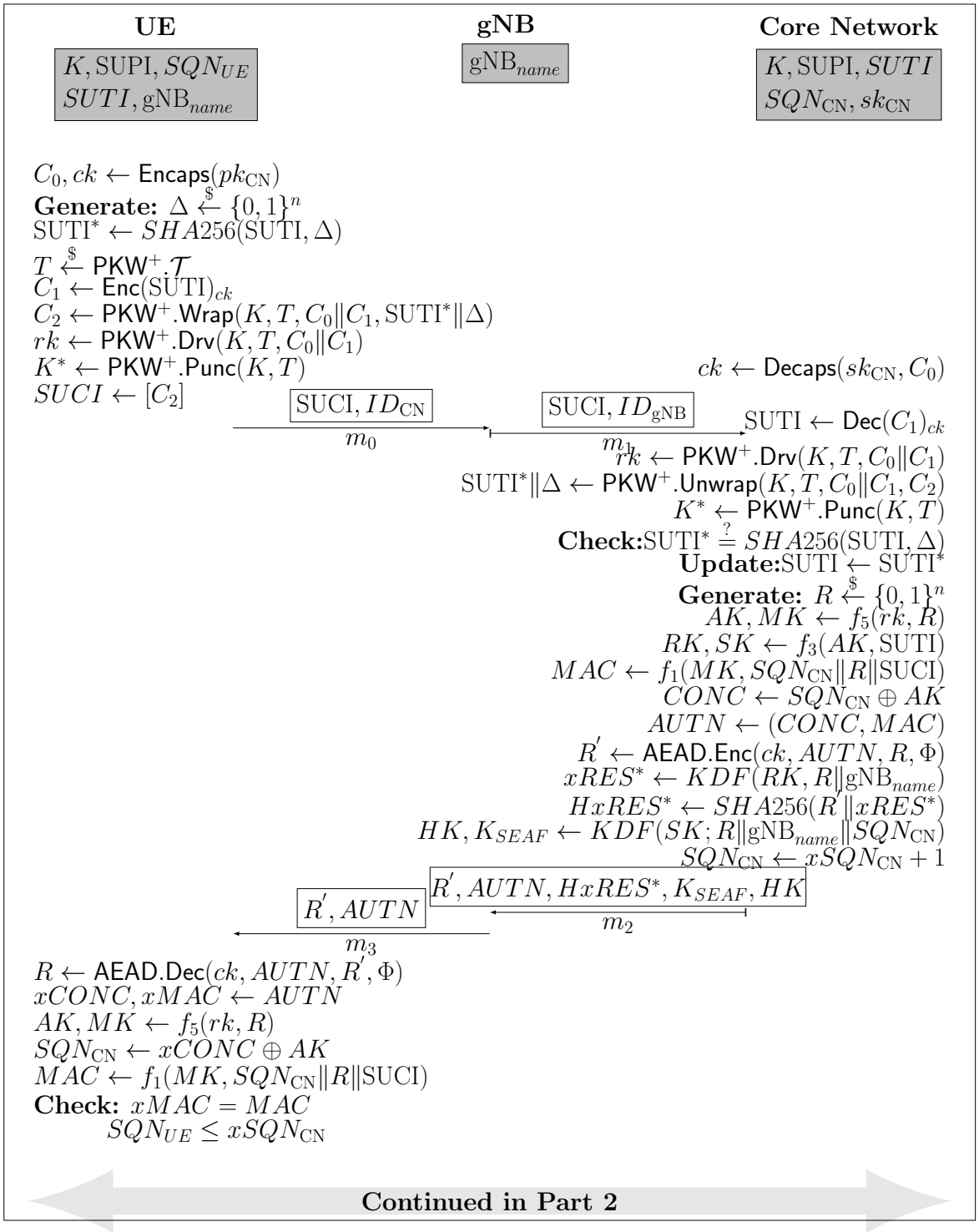

Figure 6.3: *PGUP-AKA protocol (part 1)*

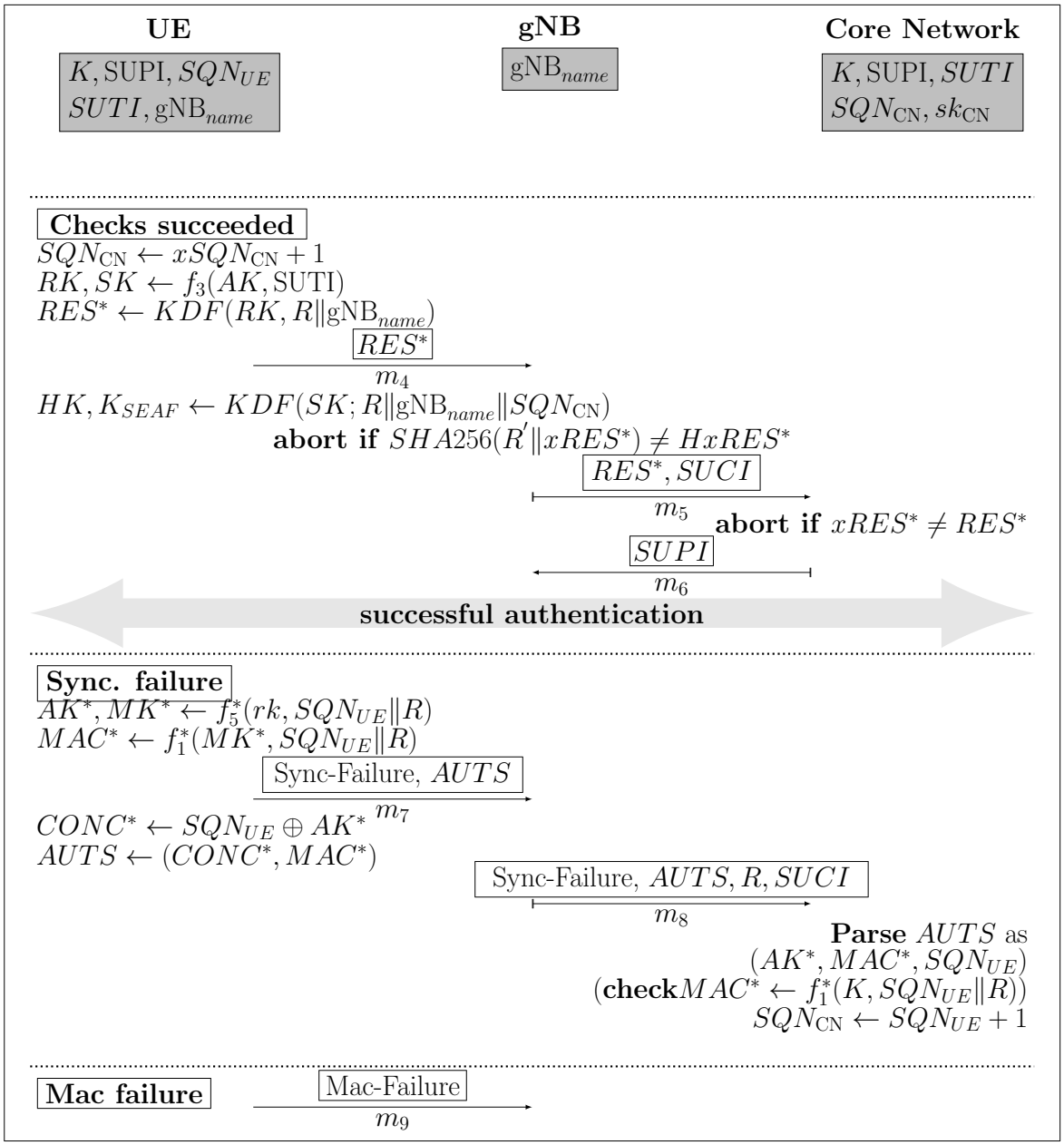

Figure 6.4: *PGUP-AKA protocol (part 2)*

6.3.2 Universal Handover

This protocol is triggered when a UE's current base station (SgNB) is no longer capable of serving it, or when a more suitable base station (TgNB) is discovered. In such cases, the SgNB executes the process of handing over the UE. Below, we present a detailed description of the sequence of messages essential for the HO protocol, as illustrated in Figure 6.5.

Step 1: SgNB $\rightarrow$ TgNB/UE : $[C_{BS}, C]/[C_{\text{UE}}]$
This step begins when SgNB sends a HO request and notification to TgNB and UE, respectively. First, SgNB generates encryption and temporary keys (ek & tk) from a nonce θ and HK (generated during the AKA protocol) using $f_5(\cdot)$. Next, SgNB randomly samples (r, a) and then uses the temporary key tk and r to update the HK using $f_3(\cdot)$. Additionally, SgNB generates a session key sk using $\mathsf{KGen}()$ algorithm. These keys encrypt all user information maintained by the SgNB and are used to send the ciphertext to the UE and TgNB. The encryption process involves two distinct operations: first, the encryption key (ek) is used with $\mathsf{AEAD.Enc}(\cdot)$ to encrypt $(SUCI|SQN_{HO}|r|H)$, where H is a header, resulting in ciphertext C_{UE}; concurrently, the session key (sk) encrypts $(SUCI|SQN_{HO}|HK^*|a|H)$, producing C_{BS}. Following these encryptions, SgNB uses the Base Key (BK), shared between base stations beforehand, to wrap the session key with $\mathsf{PKW}^{+}.\mathsf{Wrap}(BK, T, a, sk)$, generating C, and then punctures BK on tag T. Finally, SgNB initiates the HO request by sending the HO request (C_{BS}, C) to TgNB and simultaneously transmitting the HO notification (C_{UE}) to UE.

Step 2: TgNB $\rightarrow$ UE : $[HO_{Res}]$
Upon receiving the HO request (C_{BS}, C) from the SgNB, TgNB retrieves sk by unwrapping C using BK. TgNB then uses sk to decrypt C_{BS} and retrieve the UE's information with the updated handover key $(SUCI\|SQN_{HO}\|HK^*\|a)$. Next, TgNB punctures BK on tag T and checks if SQN_{HO} is less than the threshold n. If the condition holds, TgNB increments SQN_{HO} by one and then randomly samples t to update the handover key HK, generating hk and $\widehat{HK}$. The hk is then used to encrypt the UE's information using the AEAD encryption algorithm ($AEAD.\mathsf{Enc}(hk, SUCI \|SQN_{HO}\|t\|H)$).

Step 3: Upon receiving the HO notification (C_{UE}) from the SgNB, UE generates encryption and temporary keys (ek&tk) from a fixed nonce θ and HK (generated during the AKA protocol) using $f_5(\cdot)$. Then UE uses the temporary key tk and r to update the HK using $f_3(\cdot)$. Next, UE decrypts C_{UE} using ek and checks if his information matches and the SQN_{HO} is less than the threshold n.

Step 4: Upon receiving the HO response from TgNB, the UE updates the handover key HK^*, generating hk and $\widehat{HK}$. The hk is then used to decrypt HO_{Res} using the AEAD decryption algorithm. The UE then checks if the decrypted information matches its own records.

Step 5: **Threshold** $[SQN_{HO} \geq n]$ In case the roaming user exceeds the handover threshold (i.e., a user has executed the HO protocol y times, where $SQN_{HO} = y$ and $y \geq n$), the UE resets the handover sequence to zero, setting $SQN_{HO} = 0$. Similarly,

TgNB also resets SQN_{HO}, but first checks with the CN if the user is revoked. If the user is revoked, Steps 2 and 4 will not be executed, and the session will be aborted.

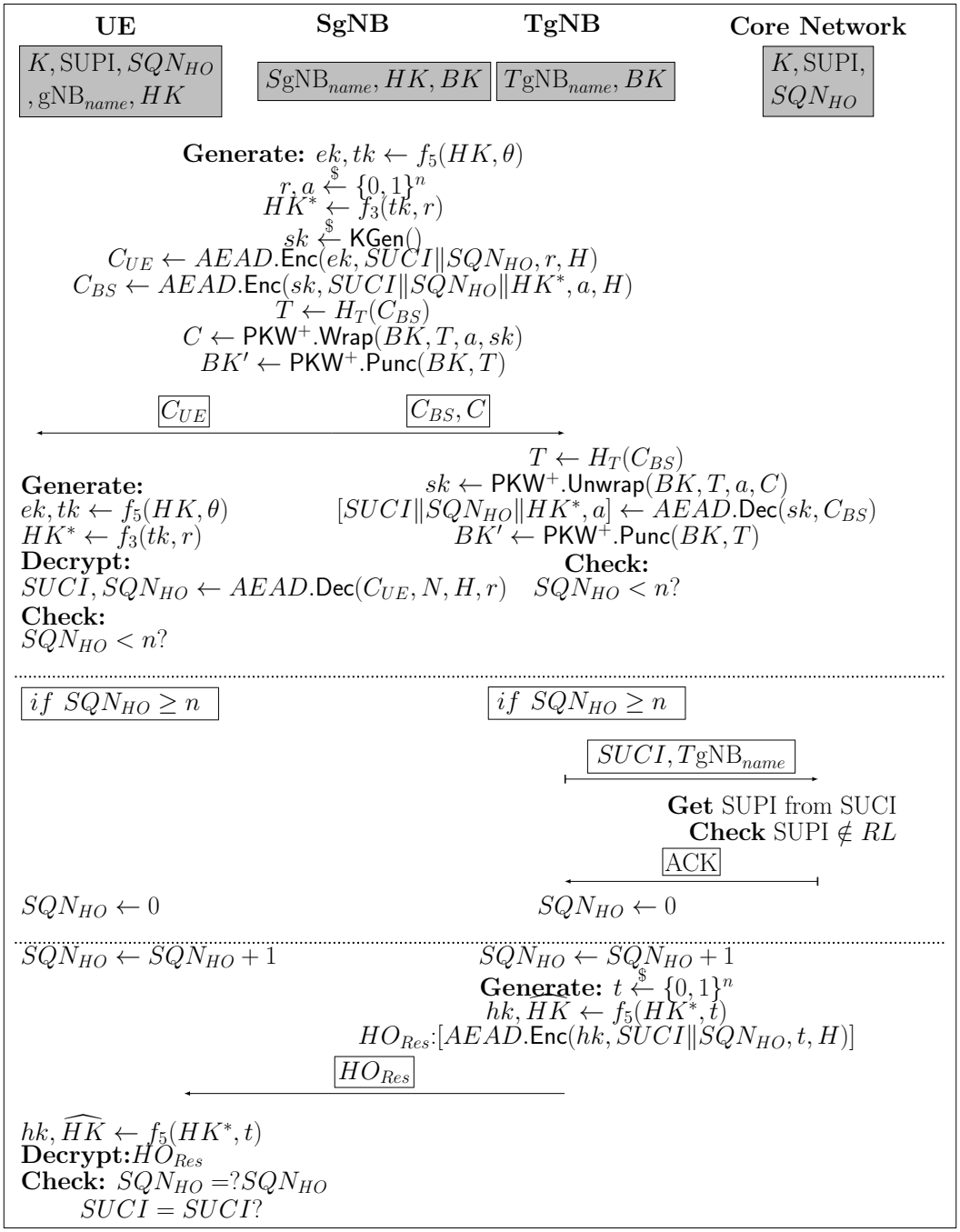

Figure 6.5: *PGUP-HO protocol*

6.3.2.1 Integration of PGUP-HO Protocol with 5G-HO

Here, we demonstrate the seamless integration of the PGUP-HO protocol into the existing 5G-HO protocol, as illustrated in Figure 6.6. The call flow depicted in blue font in Figure 6.6 outlines precisely where PGUP-HO messages can be incorporated without necessitating any infrastructural adjustments. It's worth noting that the figure does not include revocation messages from the PGUP-HO protocol, as 5G currently lacks efficient handover support. Unless there's a deliberate effort from the 5G group to incorporate revocation checks, our revocation mechanism cannot be seamlessly deployed. However, aside from this consideration, the PGUP-HO protocol can be integrated into the current 5G-HO protocol without requiring any additional infrastructure or modifications.

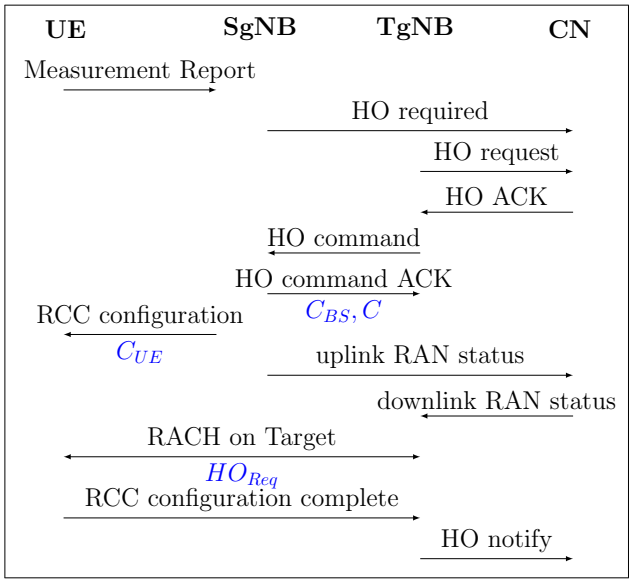

Figure 6.6: *PGUP 5G-HO compatibility*

6.4 Security Analysis

In this section, we formally prove the security of our protocols. We begin by showing that our protocols achieve mutual authentication. Then, we demonstrate that the session keys produced by the proposed PGUP scheme cannot be distinguished from random keys. Lastly, we prove that outside attackers cannot link different sessions belonging to the same party, which protects user privacy.

6.4.1 Mutual Authentication Security

In this section, we analyse the mutual authentication (MA) security of PGUP scheme, demonstrating that it achieves mutual authentication security. We maintain the definition of MA-security as presented in Def. 24 of Chapter 2. Recall that in the MA security game, defined in 24, $\mathcal{A}$ wins (and $\mathbf{Exp}^{\mathsf{MA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ outputs 1), if the adversary has caused a **clean** session to accept (and set $\pi_i^s.\alpha \leftarrow \texttt{accepted}$) and there either exists no matching subset session π_j^t

(i.e. no CN session that outputs the messages received by π_i^s - Def. 22), or no matching session π (i.e. no CN session that outputs the messages received by π_i^s - Def. 23). Here, we specify the adversary capabilities and cleanness predicate for our particular protocols.

Adversary Queries. Here, we define queries that represent the behaviours of the adversary $\mathcal{A}$ during the execution of the MA experiment:

- **Create**(i,s): allows $\mathcal{A}$ to initialize new UE sessions π_i^s.
- **Send**(m, i, s): allows $\mathcal{A}$ to send messages m to a session π_i^s. It produces a message m', which might be empty.
- **CorruptLTK**(i)$\rightarrow k_i$: allows $\mathcal{A}$ to leak the long-term key of UE$_i$.
- **CorruptASK**(i,ρ)$\rightarrow (sk)$: allows $\mathcal{A}$ to leak the long-term asymmetric keys of a party.
- **StateReveal**(i,s)$\rightarrow \pi_i^s$: allows $\mathcal{A}$ to reveal the internal state of π_i^s.
- **Reveal**$(i, s, \rho) \rightarrow k$: allows $\mathcal{A}$ to reveal the secret session key k computed during session π_i^s where $\pi_i^s.\rho = \rho$.

Cleanness Predicate. We now turn to describe our cleanness predicates. It is important to note that the $\mathcal{A}$ can easily forge messages to π_i^s if he has compromised both the long-term key shared between CN and UE, as well as the long-term asymmetric key of the CN , or if the $\mathcal{A}$ compromised the long-term key of the partner session, where$\pi_j^t.\rho =$ UE. To mitigate this type of attack, we introduce a cleanness predicate that ensures the $\mathcal{A}$ is prevented from issuing **CorruptLTK**, **CorruptASK**, or **StateReveal** queries before the test session's acceptance.

Definition 38 (PGUP-AKA cleanness) *A session π_i^s in the* MA *experiment described above is* **clean**$_{AKA}$ *if the following conditions hold:*

1. ***StateReveal***$(i, s, \pi_i^s.\rho)$ *has not been issued* and *for all sessions π_j^t such that π_j^t is a matching subset of π_i^s* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
2. *If ($\pi_i^s.\rho =$* gNB*) or ($\pi_i^s.\rho =$* CN*), and there exists no $(j,t) \in n_P \times n_S$ such that π_j^t is a matching session of π_i^s,* ***CorruptLTK***$(\pi_i^s.pid)$ *have not been issued before $\pi_i^s.\alpha =$* `accepted`.
3. *If $\pi_i^s.\rho =$* UE*, and there exists no $(j,t) \in n_P \times n_S$ such that π_j^t is a matching session of π_i^s,* ***CorruptLTK***(i) *or* ***CorruptASK***(CN) *have not both been issued before $\pi_i^s.\alpha =$* `accepted`.

Definition 39 (PGUP-HO cleanness) *A session π_i^s in the* MA *experiment described above is* **clean**$_H$ *if the following conditions hold:*

1. ***StateReveal***$(i, s, \pi_i^s.\rho)$ *has not been issued* and *for all sessions π_j^t such that π_j^t is a matching subset of π_i^s* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*

A protocol Π is considered MA-secure if no probabilistic polynomial-time (PPT) algorithms $\mathcal{A}$ capable of winning the MA security game against a clean session with a non-negligible advantage. We formalise this notion below.

6.4.1.1 MA-security of PGUP-AKA protocol

Here we present our formal analysis and results for the MA-security of the AKA protocol.

Theorem 19 MA-*security of PGUP-AKA. PGUP-AKA protocol depicted in Figures 6.3 and 6.4 is* MA-*secure under the cleanness predicate* $\mathbf{clean}_{AKA}$ *defined in Def. 38. For any PPT algorithm* $\mathcal{A}$, $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the security of* PKW$^+$, AEAD, MAC, KEM *and* KDF *schemes.*

Proof: Our proof is divided into three cases, denoted by

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda), \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{and} \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)).$$

Case 1: In this case assume that the first **clean** session π^s_i to accept without a matching session (i.e. $\pi^s_i.\rho = \text{gNB}$). Accordingly, gNB either accepts messages $\mathbf{m_0}$, $\mathbf{m_4}$ or $\mathbf{m_7}$, without a matching subset (i.e. without honest UE partner). Note that in this case, $\mathcal{A}$ cannot corrupt the long-term UE symmetric-key.

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first gNB session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index t of the CN session π^{CN}_t and we abort if there exists $\pi^{\text{CN}}_{t^*}$ that matches the π^s_i and $(t \neq t^*)$, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_S \mathbf{Adv}_{G_2}$.

Game 3 : In this game, we guess UE party index j such that $\pi^{\text{CN}}_t.pid = j$, introducing a factor of n_P in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_2} \leq n_P \mathbf{Adv}_{G_3}$.

Game 4 : In this game, we guess session index $r \in n_S$ and aborts if π^j_r registers a ciphertext in $C_{txt} \leftarrow (C, T, AD, m, j)$ and π^{CN}_t computes $rk \leftarrow \mathsf{Drv}(k_i, T', C_0)$ and $(., T', ., .) \in C_{txt}$ but $T' \neq T$, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_3} \leq n_S \mathbf{Adv}_{G_4}$.

Game 5 : In this game, we replace the computation of rk in π^j_r and π^{CN}_t sessions with uniformly random value ($\hat{rk}$). We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW$^+$ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls New(), $\mathcal{B}_1$ calls to PKW$^+$.New(), then we assign i to each $SUTI$ and a table $(SUTI, i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T, AD, m, i) \rightarrow C$ on a new entry, such that $(\cdot, T, AD, m, i) \notin C_{txt}$. Similarly, the R_k register will be updated whenever we call $\mathsf{Drv}(T, AD, i) \rightarrow rk$ on new entries, such that $(\cdot, T, AD, i) \notin R_k$. Finally, T_p will be updated whenever we call $\mathsf{Punc}(T, i, \rho)$. Whenever $\mathbf{Send}(i, s, m, \rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T, i, \rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0\|C_1, SUTI||\Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0\|C_1, SUTI||\Delta)$ by $\mathcal{B}_1$. and if Unwrap was queried, the computation of $m \leftarrow \mathsf{Unwrap}(K, T, C_0\|C_1, C_2)$ is replaced with a call to $\mathsf{Unwrap}(i, T, C_0\|C_1, C_2)$ by $\mathcal{B}_1$. Similarly, whenever $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0\|C_1)$.
- IF $(T, i, \overline{\rho}) \in T_p$: Depending on the algorithm that was called, we check its registry for an output. If Unwrap or Drv were queried, we recover the output from the register (C_{txt}, R_k), respectively. However, if Wrap was queried, then the adversary has either issued a **Corrupt**, hence $\mathcal{B}_1$ can simulate Wrap, or the adversary has forged a message between UE and CN, hence breaking the PKW^+ security.
- IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0\|C_1, SUTI||\Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0\|C_1, SUTI||\Delta)$ by $\mathcal{B}_1$. Then, the output is added to the register as $(C, i, T, C_0\|C_1, SUTI||\Delta) \rightarrow C_{txt}$. Similarly, whenever $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0\|C_1)$. and adds it to the register as $\mathsf{Drv}(i, T, C_0\|C_1) \rightarrow R_K$.

We note that this process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability game, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1, \mathsf{PKW}^+}(\mathcal{A})$.

Game 6: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_2 that decrypts correctly and is accepted by π_t^{CN}, but was not generated by π_r^j, breaking the INT-CTXT security of AEAD in PKW^+. In this game, no replacement is required because if the $\mathcal{A}$ could forge a cipher text, then they would win **Game 5**. Note that $(C_0\|C_1)$ is securely authenticated by $\mathsf{PKW}^+.\mathsf{Wrap}()$ as additional data, and by the definition of **Case 1** $\mathcal{A}$ cannot issue **CorruptLTK**(UE). Any $\mathcal{A}$ that can trigger this abort event can be used to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{PKW}^+}(\mathcal{A})$.

Game 7: In this game we replace the authentication and mac keys AK, MK with uniformly random values $\hat{AK}, \hat{MK}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{rk}$ and replacing the computation of AK, MK in any session that computes AK, MK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{AK}, \hat{MK}$. Since $AK, MK \leftarrow \mathsf{KDF}(rk, R)$ and by **Game 5** $\hat{rk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of KDF, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2, \mathsf{KDF}}(\mathcal{A})$.

Game 8: In this game, we introduce an abort event that triggers if the same hash value exists for different inputs, thereby introducing hash collisions during the challenger execution with an honest session. We do so by defining a reduction $\mathcal{B}_3$ that initialises an Hash challenger $\mathcal{C}_{\mathsf{Hash}}$ and maintains a lookup table. Whenever $\mathcal{B}_3$ needs to hash a value, he queries the lookup table on x and recovers (out). The abort event only triggers if $\mathcal{A}$ can get the same output hash values from two distinct input values, thus

finding a collision and breaking the security of the Hash scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of Hash. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{collision}_{\mathcal{B}_3,\mathsf{Hash}}(\mathcal{A})$.

Game 9: In this game, we replace the response and session keys RK, SK with uniformly random values $\hat{RK}, \hat{SK}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{AK}$ and replacing the computation of RK, SK in any session that computes RK, SK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}, \hat{SK}$. Since $RK, SK \leftarrow \mathsf{KDF}(\hat{AK}, SUTI)$ and by **Game 7**,$\hat{AK}$ is already uniformly random, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 8** from **Game 9** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 10: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a valid tag (MAC), but (MAC) was not output by π^t_j. We do so by defining a reduction $\mathcal{B}_5$ that initialises an MAC challenger $\mathcal{C}_{\mathsf{MAC}}$, which $\mathcal{B}_5$ queries when π^t_j needs to MAC (SQN, R) with $(\hat{MK})$ and from **Game 7** $\hat{MK}$ is already uniformly random and independent. This abort event occurs if MAC is verified correctly under $\hat{MK}$, but MAC was never produced by the $\mathcal{C}_{\mathsf{MAC}}$, breaking the security of the MAC scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the existential unforgeability security of the MAC scheme, as formalised in Def. 4. This implies: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{19}} + \mathbf{Adv}^{EUFCMA}_{\mathcal{B}_5,\mathsf{MAC}}(\mathcal{A})$.

Game 11: In this game, we replace the response (RES^*) and (K_{SEAF}, HK) with uniformly random value $(\hat{RES^*})$ and $(\hat{K_{SEAF}}\hat{HK})$, respectively. We do so by defining a reduction $\mathcal{B}_6$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}, \hat{RK}$ and replacing the computation of (RES^*) and (K_{SEAF}, HK) in any session that computes RES^*, K_{SEAF}, HK, with the outputs from the $(\mathcal{C}_{\mathsf{KDF}}\ \hat{RK})$ and $(\mathcal{C}_{\mathsf{KDF}}\ \hat{K_{SEAF}}\hat{HK})$. Since $(RES^* \leftarrow \mathsf{KDF}(\hat{RK}, R\|\text{gNB}_{name}))$ and $(\mathcal{C}_{\mathsf{KDF}}\ \hat{K_{SEAF}}\hat{HK})$ and by **Game 9** $\hat{RK}$ and $\hat{SK}$ are already uniformly random, so these changes are sound. Any $\mathcal{A}$ that can distinguish **Game 10** from **Game 11** can be used to break KDF security of the KDF scheme, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_{10}} \leq \mathbf{Adv}_{G_{11}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_6,\mathsf{KDF}}(\mathcal{A})$.

By now, it becomes evident that the $\mathcal{A}$ is unable to tamper with messages exchanged between the π^s_i and π^t_j. Specifically, π^s_i exclusively accepts message $\mathbf{m_0}$ from an honest matching partner, and $\mathcal{A}$ cannot modify the content of C_2 without compromising the security of PKW^+. Additionally, $C_0\|C_1$ remains unaltered since it is authenticated using C_2. Especially since the $\mathcal{A}$ cannot corrupt the long-term UE symmetric-key, according to the definition of this case. Similarly, the adversary cannot modify $\mathbf{m_4}, \mathbf{m_7}$ without compromising the security of KDF&MAC. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_{11}} = 0$

We now bound the advantage of $\mathcal{A}$ in **Case 2**.

Case 2.1: In this case assume that the first **clean** session π^s_i to accept without a matching session(i.e. $\pi^s_i.\rho = \text{UE}$). Accordingly, UE accepts message $\mathbf{m_3}$ without a matching subset (i.e. without honest CN partner). In this subcase, $\mathcal{A}$ cannot corrupt the long-term asymmetric secret key of CN.

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1: In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first UE session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2: In this game, we guess the index $(j) \in n_S$ of the first CN session that accepts without a matching subset, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_S \mathbf{Adv}_{G_2}$.

Game 3: In this game, we replace the key ck with uniformly random and independent value $\hat{ck}$.We do so by interacting with a $\mathcal{B}_1$ challenger $\mathcal{C}^{ind-cpa}_{\mathsf{KEM}}$, and replacing pk_{CN} with a public key pk recieved from $\mathcal{B}_1$. By the definition of the execution environment, pk_{CN} is generated uniformly random and independent, and by the definition of **Case 2.1** $\mathcal{A}$ cannot issue **CorruptASK**(CN) and this replacement is sound. Any $\mathcal{A}$ that can detect the replacement can be used to break $ind-cpa$ security of KEM. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{ind-cpa}_{\mathcal{B}_1,\mathsf{KEM}}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext $R^{'}$ (keyed by $\hat{ck}$) that decrypts correctly and was not generated by CN, breaking the authentication security of AEAD. We do so by defining a reduction $\mathcal{B}_2$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$, which $\mathcal{B}_2$ queries when he needs to encrypt/decrypt with $\hat{ck}$. By **Game 3** $\hat{ck}$ is already uniformly random and independent, and this replacement is undetectable. Note that $AUTN$ is securely authenticated by AEAD security since it is included in the AEAD functions as additional data. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_2$ to break the security of AEAD. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_2,\mathsf{AEAD}}(\mathcal{A})$.

In this stage, it becomes evident that the $\mathcal{A}$ is unable to tamper with messages exchanged between the π^s_i and π^t_j. Specifically, π^s_i exclusively accepts message $\mathbf{m_3}$ from an honest matching partner, and $\mathcal{A}$ cannot modify the content of m_3 without compromising the security of AEAD and KEM. Additionally, $AUTN$ remains unaltered since it is authenticated using $R^{'}$. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_4} = 0$

We turn to bound the advantage of $\mathcal{A}$ in **Case 2.2**.

Case 2.2: According to the definition of this case, UE accepts message $\mathbf{m_3}$ without a matching subset (i.e. without honest CN partner). In this subcase, $\mathcal{A}$ cannot corrupt the UE symmetric secret key (K).

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C2}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first UE session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $(j) \in n_S$ of the first CN session that accepts without a matching subset, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_S \mathbf{Adv}_{G_2}$.

Game 3 : In this game, we replace the computation of rk in π_r^j and π_t^{CN} sessions with uniformly random value ($\hat{rk}$). We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW^+ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls $\mathsf{New}()$, $\mathcal{B}_1$ calls to $\mathsf{PKW}^+.\mathsf{New}()$, then we assign i to each $SUTI$ and a table $(SUTI, i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T, AD, m, i) \rightarrow C$ on a new entry, such that $(\cdot, T, AD, m, i) \notin C_{txt}$. Similarly, the R_k register will be updated whenever we call $\mathsf{Drv}(T, AD, i) \rightarrow rk$ on new entries, such that $(\cdot, T, AD, i) \notin R_k$. Finally, T_p will be updated whenever we call $\mathsf{Punc}(T, i, \rho)$. Whenever $\mathbf{Send}(i, s, m, \rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T, i, \rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0 \| C_1,$ $SUTI || \Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0 \| C_1, SUTI || \Delta)$ by $\mathcal{B}_1$. and if Unwrap was queried, the computation of $m \leftarrow \mathsf{Unwrap}(K, T, C_0 \| C_1, C_2)$ is replaced with a call to $\mathsf{Unwrap}(i, T, C_0 \| C_1, C_2)$ by $\mathcal{B}_1$. Similarly, whenever $\mathsf{Drv}(K, T, C_0 \| C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0 \| C_1)$.
- IF $(T, i, \overline{\rho} \in T_p$:Depending on the algorithm that was called, we check its registry for an output. If Unwrap or Drv were queried, we recover the output from the register (C_{txt}, R_k), respectively. However, if Wrap was queried, then the adversary has either issued a **Corrupt**, hence $\mathcal{B}_1$ can simulate Wrap, or the adversary has forged a message between UE and CN, hence breaking the PKW^+ security.
- IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0 \| C_1, SUTI || \Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0 \| C_1, SUTI || \Delta)$ by $\mathcal{B}_1$. Then, the output is added to the register as $(C, i, T, C_0 \| C_1, SUTI || \Delta) \rightarrow C_{txt}$. Similarly, whenever $\mathsf{Drv}(K, T, C_0 \| C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0 \| C_1)$. and adds it to the register as $\mathsf{Drv}(i, T, C_0 \| C_1) \rightarrow R_K$.

We note that this process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability security, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1, \mathsf{PKW}^+}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_2 that decrypts correctly and is accepted by π_t^{CN}, but was not generated by π_r^j, breaking the authentication security of AEAD in PKW^+. In this game, no replacement is required because if the $\mathcal{A}$ could forge a cipher text, then they would win **Game 5**. Note that $(C_0 \| C_1)$ is securely authenticated by $\mathsf{PKW}^+.\mathsf{Wrap}()$ as additional data, and by the definition of **Case 1** $\mathcal{A}$ cannot issue **CorruptLTK**(UE). Any $\mathcal{A}$ that can trigger this abort event can be used to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{PKW}^+}(\mathcal{A})$.

Game 5: In this game we replace the authentication and mac keys AK, MK with uniformly random values $\hat{AK}, \hat{MK}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{rk}$ and replacing the computation of AK, MK in any session that computes AK, MK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{AK}, \hat{MK}$. Since $AK, MK \leftarrow \mathsf{KDF}(rk, R)$ and by **Game 3** $\hat{rk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of KDF, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$.

Game 6: In this game, we introduce an abort event that triggers if the same hash value exists for different inputs, thereby introducing hash collisions during the challenger execution with an honest session. We do so by defining a reduction $\mathcal{B}_3$ that initialises an Hash challenger $\mathcal{C}_{\mathsf{Hash}}$ and maintains a lookup table. Whenever $\mathcal{B}_3$ needs to hash a value, he queries the lookup table on x and recovers (out). The abort event only triggers if $\mathcal{A}$ can get the same output hash values from two distinct input values, thus finding a collision and breaking the collision security of the Hash scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of Hash. This implies: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{collision}_{\mathcal{B}_3,\mathsf{Hash}}(\mathcal{A})$.

Game 7: In this game, we replace the response and session keys RK, SK with uniformly random values $\hat{RK}, \hat{SK}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{AK}$ and replacing the computation of RK, SK in any session that computes RK, SK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}, \hat{SK}$. Since $RK, SK \leftarrow \mathsf{KDF}(\hat{AK}, SUTI)$ and by **Game 5**,$\hat{AK}$ is already uniformly random, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 6** from **Game 7** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 8: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a valid tag (MAC), but (MAC) was not output by π^t_j. We do so by defining a reduction $\mathcal{B}_5$ that initialises an MAC challenger $\mathcal{C}_{\mathsf{MAC}}$, which $\mathcal{B}_5$ queries when π^t_j needs to MAC (SQN, R) with $(\hat{MK})$ and from **Game 5** $\hat{MK}$ is already uniformly random and independent. This abort event occurs if MAC is verified correctly under $\hat{MK}$, but MAC was never produced by the $\mathcal{C}_{\mathsf{MAC}}$, breaking the security of the MAC scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the existential unforgeability security of the MAC scheme, as formalised in Def. 4. This implies: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{EUFCMA}_{\mathcal{B}_5,\mathsf{MAC}}(\mathcal{A})$.

Game 9: In this game, we replace the response (RES^*) with uniformly random value $(\hat{RES^*})$. We do so by defining a reduction $\mathcal{B}_6$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}, \hat{RK}$ and replacing the computation of (RES^*) in any session that computes RES^*, with the outputs from the ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}$). Since $(RES^* \leftarrow \mathsf{KDF}(\hat{RK}, R \| \mathrm{gNB}_{name}))$ and by **Game 7** $\hat{RK}$ is already uniformly random, so these changes are sound. Any $\mathcal{A}$ that can distinguish **Game 8** from **Game 9** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_6,\mathsf{KDF}}(\mathcal{A})$.

Game 10: In this game, we replace the (K_{SEAF}, HK) with uniformly random value $(\hat{K_{SEAF}}, \hat{HK})$. We do so by defining a reduction $\mathcal{B}_7$ that interacts with a KDF challenger

$\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}$ and replacing the computation of (K_{SEAF}, HK) in any session that computes K_{SEAF}, HK, with the outputs from the $(\mathcal{C}_{\mathsf{KDF}}\ \hat{K_{SEAF}HK})$. Since $(\mathcal{C}_{\mathsf{KDF}}\ \hat{K_{SEAF}HK})$ and by **Game 7** $\hat{SK}$ is already uniformly random, so these changes are sound. Any $\mathcal{A}$ that can distinguish **Game 9** from **Game 10** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{10}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_7,\mathsf{KDF}}(\mathcal{A})$.

By this stage, it is evident that the adversary cannot tamper with the messages exchanged between π_i^s and π_j^t sessions. In particular, the test session only accepts message m_3 from an honest matching partner because the $\mathcal{A}$ cannot modify the content of $AUTN$ without compromising the security of $\mathsf{AEAD}, \mathsf{KDF}, \mathsf{MAC}$,and PKW^+. Furthermore, if the $\mathcal{A}$ attempts to alter $R^{'}$, the π_i^s will reject it since the decryption of $R^{'}$ is authenticated using MAC. This holds true since the adversary cannot corrupt the long-term UE symmetric key, as per the definition of this case. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_{10}} = 0$

Case 3: In this case, we assume that the first **clean** session π_i^s to accept without a matching session(i.e. $\pi_i^s.\rho = \text{CN}$). Accordingly, CN accepts messages $\mathbf{m_1}$ or $\mathbf{m_5}$, without a matching subset (i.e. without honest UE partner).

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C3}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(s) \in n_S$ of the first CN session that accepts without a matching subset, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_S\mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $(t, j) \in n_P \times n_S$ of the first UE session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_P \cdot n_S\mathbf{Adv}_{G_2}$.

Game 3 : In this game, we replace the computation of rk in π_r^j and π_t^{CN} sessions with uniformly random value $(\hat{rk})$. We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW^+ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls $\mathsf{New}()$, $\mathcal{B}_1$ calls to $\mathsf{PKW}^+.\mathsf{New}()$, then we assign i to each $SUTI$ and a table $(SUTI, i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T, AD, m, i) \rightarrow C$ on a new entry, such that $(\cdot, T, AD, m, i) \notin C_{txt}$. Similarly, the R_k register will be updated whenever we call $\mathsf{Drv}(T, AD, i) \rightarrow rk$ on new entries, such that $(\cdot, T, AD, i) \notin R_k$. Finally, T_p will be updated whenever we call $\mathsf{Punc}(T, i, \rho)$. Whenever $\mathbf{Send}(i, s, m, \rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T, i, \rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0\|C_1, SUTI||\Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0\|C_1, SUTI||\Delta)$ by $\mathcal{B}_1$. and if Unwrap was queried, the computation of $m \leftarrow \mathsf{Unwrap}(K, T, C_0\|C_1, C_2)$ is replaced with a call to $\mathsf{Unwrap}(i, T, C_0\|C_1, C_2)$ by $\mathcal{B}_1$. Similarly, whenever $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0\|C_1)$.

- IF $(T, i, \overline{\rho} \in T_p$:Depending on the algorithm that was called, we check its registry for an output. If Unwrap or Drv were queried, we recover the output from the register (C_{txt}, R_k), respectively. However, if Wrap was queried, then the adversary has either issued a **Corrupt**, hence $\mathcal{B}_1$ can simulate Wrap, or the adversary has forged a message between UE and CN, hence breaking the PKW^+ security.
- IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0 \| C_1, SUTI || \Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0 \| C_1, SUTI || \Delta)$ by $\mathcal{B}_1$. Then, the output is added to the register as $(C, i, T, C_0 \| C_1, SUTI || \Delta) \rightarrow C_{txt}$. Similarly, whenever $\mathsf{Drv}(K, T, C_0 \| C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0 \| C_1)$. and adds it to the register as $\mathsf{Drv}(i, T, C_0 \| C_1) \rightarrow R_K$.

We note that this process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability security, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1, \mathsf{PKW}^+}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_2 that decrypts correctly and is accepted by π_t^{CN}, but was not generated by π_r^j, breaking the authentication security of AEAD in PKW^+. In this game, no replacement is required because if the $\mathcal{A}$ could forge a cipher text, then they would win **Game 5**. Note that $(C_0 \| C_1)$ is securely authenticated by $\mathsf{PKW}^+.\mathsf{Wrap}()$ as additional data, and by the definition of **Case 1** $\mathcal{A}$ cannot issue **CorruptLTK**(UE). Any $\mathcal{A}$ that can trigger this abort event can be used to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{PKW}^+}(\mathcal{A})$.

Game 5: In this game we replace the authentication and mac keys AK, MK with uniformly random values $\hat{AK}, \hat{MK}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{rk}$ and replacing the computation of AK, MK in any session that computes AK, MK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{AK}, \hat{MK}$. Since $AK, MK \leftarrow \mathsf{KDF}(\hat{rk})$ and by **Game 5** $\hat{rk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of KDF. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2, \mathsf{KDF}}(\mathcal{A})$.

Game 6: In this game, we introduce an abort event that triggers if the same hash value exists for different inputs, thereby introducing hash collisions during the challenger execution with an honest session. We do so by defining a reduction $\mathcal{B}_3$ that initialises an Hash challenger $\mathcal{C}_{\mathsf{Hash}}$ and maintains a lookup table. Whenever $\mathcal{B}_3$ needs to hash a value, he queries the lookup table on x and recovers (out). The abort event only triggers if $\mathcal{A}$ can get the same output hash values from two distinct input values, thus finding a collision and breaking the collision security of the Hash scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of Hash. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{collision}_{\mathcal{B}_3, \mathsf{Hash}}(\mathcal{A})$.

Game 8: In this game, we replace the response and session keys RK, SK with uniformly random values $\hat{RK}, \hat{SK}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{AK}$ and replacing the computation of RK, SK in any session that computes RK, SK, with the outputs from the

$\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}, \hat{SK}$. Since $RK, SK \leftarrow \mathsf{KDF}(\hat{AK})$ and by **Game 5**,$\hat{AK}$ is already uniformly random, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 7** from **Game 8** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 9: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a valid tag (MAC), but (MAC) was not output by π^t_j. We do so by defining a reduction $\mathcal{B}_5$ that initialises an MAC challenger $\mathcal{C}_{\mathsf{MAC}}$, which $\mathcal{B}_5$ queries when π^t_j needs to MAC (SQN, R) with $(\hat{MK})$ and from **Game 5** $\hat{MK}$ is already uniformly random and independent. This abort event occurs if MAC is verified correctly under $\hat{MK}$, but MAC was never produced by the $\mathcal{C}_{\mathsf{MAC}}$, breaking the security of the MAC scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the existential unforgeability security of the MAC scheme, as formalised in Def. 4. This implies: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{EUFCMA}_{\mathcal{B}_5,\mathsf{MAC}}(\mathcal{A})$.

Game 10: In this game, we replace the response (RES^*) and (K_{SEAF}, HK) with uniformly random value $(\hat{RES^*})$ and $(\hat{K_{SEAF}}\hat{HK})$, respectively. We do so by defining a reduction $\mathcal{B}_6$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}, \hat{RK}$ and replacing the computation of (RES^*) and (K_{SEAF}, HK) in any session that computes RES^*, K_{SEAF}, HK, with the outputs from the ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}$) and ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{K_{SEAF}}\hat{HK}$). Since ($RES^* \leftarrow \mathsf{KDF}(\hat{RK})$) and ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{K_{SEAF}}\hat{HK}$) and by **Game 8** $\hat{RK}$ and $\hat{SK}$ are already uniformly random, so these changes are sound. Any $\mathcal{A}$ that can distinguish **Game 9** from **Game 10** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{10}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_6,\mathsf{KDF}}(\mathcal{A})$.

By this stage, it is evident that the adversary is incapable of tampering with the messages exchanged between π^s_i and π^t_j sessions. In particular, the test session only accepts message m_3 from an honest matching partner because the $\mathcal{A}$ cannot modify the content of $AUTN$ without compromising the security of AEAD, KDF, MAC,and PKW^+. Furthermore, if the $\mathcal{A}$ attempts to alter R', the π^s_i will reject it since the decryption of R' is authenticated using MAC. This holds true since the adversary cannot corrupt the long-term UE symmetric-key, as per the definition of this case. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_{10}} = 0$

6.4.1.2 MA-security of Handover Protocol

Here we present our formal analysis and results for the MA-security of the PGUP-HO protocol.

Theorem 20 MA-***security of PGUP Handover***. *The PUGP Handover protocol depicted in Figure 6.5, is* MA*-secure under the cleanness predicate in Def. 39. For any PPT algorithm* $\mathcal{A}$ *against the* MA *experiment,* $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ *is negligible assuming the security of* PKW^+, AEAD, MAC, KEM *and* KDF *schemes.*

Proof: Our proof is divided into three cases, denoted by

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda), \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{and} \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{HO}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$$

.

Case 1: In this case, we assume that the first **clean** session π^s_i to accept without a matching session(i.e. $\pi^s_i.\rho = \text{TgNB}$). Accordingly, TgNB accepts message$\mathbf{C}, \mathbf{C_{BS}}$ without an honest matching partner (SgNB) (no matching subset). In this case, the $\mathcal{A}$ cannot corrupt the Bootstrap-key (BK) between SgNB&TgNB.

Here we provide the security analysis:

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first TgNB session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $j \in n_P$ of the SgNB party such that $\pi^s_i.pid = j$, introducing a factor of n_P in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_1} \leq n_P \mathbf{Adv}_{G_2}$.

Game 3 : In this game, we replace the computation of C with uniformly random value ($\hat{C}$). We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW^+ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls $\mathsf{New}()$, $\mathcal{B}_1$ calls to $\mathsf{PKW}^+.\mathsf{New}()$, then we assign i to each $SUTI$ and a table $(SUTI, i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T, AD, m, i) \rightarrow C$ on a new entry, such that $(\cdot, T, AD, m, i) \notin C_{txt}$. Similarly, T_p will be updated whenever we call $\mathsf{Punc}(T, i, \rho)$. Whenever $\mathbf{Send}(i, s, m, \rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T, i, \rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C \leftarrow \mathsf{Wrap}(BK, T, a, sk)$ is replaced with a call to $\mathsf{Wrap}(i, T, a, sk)$ by $\mathcal{B}_1$. and if Unwrap was queried, the computation of $m \leftarrow \mathsf{Unwrap}(BK, T, a, C)$ is replaced with a call to $\mathsf{Unwrap}(i, T, a, C)$ by $\mathcal{B}_1$. Similarly, whenever $\mathsf{Drv}(BK, T, C_0 \| C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0 \| C_1)$.
- IF $(T, i, \overline{\rho} \in T_p$:Depending on the algorithm that was called, we check its registry for an output. If Unwrap was queried, we recover the output from the register (C_{txt}). However, if Wrap was queried, then the adversary has either issued a **Corrupt**, hence $\mathcal{B}_1$ can simulate Wrap, or the adversary has forged a message between UE and CN, hence breaking the PKW^+ security.
- IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, a, sk)$ is replaced with a call to $\mathsf{Wrap}(i, T, a, sk)$ by $\mathcal{B}_1$. Then, the output is added to the register as $(C, i, T, a, sk) \rightarrow C_{txt}$.

We note that this process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability security, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1,\mathsf{PKW}^+}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_{BS} that decrypts correctly and is accepted by π_i^s, but was not generated by π_t^j, breaking the authentication security of AEAD in PKW^+. In this game, no replacement is required because if the $\mathcal{A}$ could forge a cipher text, then they would win the game. Note that sk is randomly generated at the beginning of the session and C_{BS} is securely authenticated by $\mathsf{PKW}^+.\mathsf{Wrap}$. Any $\mathcal{A}$ that can trigger this abort event can be used to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{PKW}^+}(\mathcal{A})$.

By this stage, it is evident that the adversary is incapable of tampering with the messages exchanged between π_i^s and π_j^t sessions. In particular, the test session only accepts message C, C_{BS} from an honest matching partner because the $\mathcal{A}$ cannot modify the content of either ciphertext without compromising the security of AEAD,and PKW^+. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_4} = 0$

We now bound the advantage of $\mathcal{A}$ in **Case 2**.

Case 2: In this case, we assume that the first **clean** session π_i^s to accept without a matching session(i.e. $\pi_i^s.\rho = \text{UE}$). Accordingly, UE accepts message$\mathbf{C_{UE}}$ without an honest matching partner (UE) (no matching subset). Note here the $\mathcal{A}$ cannot corrupt the Handover-key (HK) between SgNB&UE. Here we provide the security analysis:

Game 0: This is the original mutual authentication game described in Def. 24 of Chapter 2: $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},\mathsf{C1}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i, s) \in n_P \times n_S$ of the first TgNB session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \cdot \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $j \in n_P$ of the SgNB party such that $\pi_i^s.pid = j$, introducing a factor of n_P in $\mathcal{A}$'s advantage. $\mathbf{Adv}_{G_1} \leq n_P \mathbf{Adv}_{G_2}$.

Game 3: In this game, we replace ek, tk with uniformly random values $\hat{ek}, \hat{tk}$. We do so by defining a reduction a $\mathcal{B}_1$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{HK}$ and replacing the computation of ek, tk in any session that computes ek, tk, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{ek}, \hat{tk}$. Since $ek, tk \leftarrow \mathsf{KDF}(\hat{HK})$ and by the definition of the execution environment, HK is generated uniformly random and independent, and by the definition of **Case 2** $\mathcal{A}$ cannot corrupt HK so this replacement is sound. Any $\mathcal{A}$ that can distinguish **Game 2** from **Game 3** can be used to break KDF security of the KDF scheme, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_1,\mathsf{KDF}}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_{UE} (keyed by $\hat{ek}$) that decrypts correctly and was not generated by SgNB, breaking the authentication security of AEAD. We do so by defining a reduction $\mathcal{B}_2$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$, which $\mathcal{B}_2$ queries when he needs to encrypt/decrypt with $\hat{ek}$. By **Game 3** $\hat{ek}$ is already uniformly random and independent, and this replacement is undetectable. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_2$ to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_2,\mathsf{AEAD}}(\mathcal{A})$.

Game 5: In this game, we replace handover keys HK^* with uniformly random values $\hat{HK^*}$. We do so by defining a reduction a $\mathcal{B}_3$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{tk}$ and replacing the computation of HK^* in any session that computes HK^*, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK^*}$. Since $HK^* \leftarrow \mathsf{KDF}(\hat{tk})$ and by **Game 3** $\hat{tk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 6: In this game, we replace hk with uniformly random value $\hat{hk}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{hk^*}$ and replacing the computation of hk in any session that computes hk, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK^*}$. Since $hk \leftarrow \mathsf{KDF}(\hat{HK^*})$ and by **Game 5** $\hat{HK^*}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext HO_{res} (keyed by $\hat{hk}$) that decrypts correctly and was not generated by TgNB, breaking the authentication security of AEAD. We do so by defining a reduction $\mathcal{B}_5$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$, which $\mathcal{B}_5$ queries when he needs to encrypt/decrypt with $\hat{hk}$. By **Game 6** $\hat{hk}$ is already uniformly random and independent, and this replacement is undetectable. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the security of AEAD. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{AEAD}}(\mathcal{A})$

In this stage, it becomes evident that the $\mathcal{A}$ is unable to tamper with messages exchanged between the π^s_i and π^t_j. Specifically, π^s_i exclusively accepts message $\mathbf{C_{UE}}, \mathbf{HO_{res}}$ from an honest matching partner, and $\mathcal{A}$ cannot modify the content of either without compromising the security of $\mathsf{PKW}^+, \mathsf{AEAD\&KDF}$. Thus, summing the probabilities we find that the $\mathcal{A}$ has negligible advantage in winning the MA-security experiment: $\mathbf{Adv}_{G_7} = 0$

6.4.2 Key Indistinguishability Security

In this section, we analyse the key indistinguishability (KIND) security of $\mathsf{UniHand}$ scheme, demonstrating that it achieves key indistinguishability security. We maintain the definition of Key Indistinguishability (KIND) security as presented in Def. 25 of Chapter 2. A protocol Π is considered KIND-secure if no PPT adversary can win the game with non-negligible advantage under the conditions specified below. Here we define the following specific adversary query and cleanness predicate:

Adversary Queries. In addition to the **Create**, **Send**, **CorruptLTK**, **CorruptASK**, **Reveal**, and **StateReveal** queries listed above, we define one additional query **Test** that allows the adversary $\mathcal{A}$ to either the real key derived by a **clean** session during the execution of the KIND experiment, or a random key sampled from the same distribution (depending on the challenge bit b).

- $\mathbf{Test}(i, s) \rightarrow m$: When $\mathcal{C}$ receives a **Test** query, if **Test** has already been issued, $\pi^s_i.\alpha = \texttt{accepted}$, or π^s_i is not **clean**, then $\mathcal{C}$ returns $\perp$. Otherwise, $\mathcal{C}$ sets $k_0 \leftarrow \pi^s_i.k$,

and $k_1 \xleftarrow{\$} \{0,1\}^\lambda$, and returns k_b to $\mathcal{A}$ (where b was sampled by $\mathcal{C}$ at the beginning of the experiment.

Cleanness Predicate. We now turn to describe our **clean**. It is important to note that distinguishing a real session key becomes trivial for the $\mathcal{A}$ if they have compromised both the long-term key shared between CN and UE, as well as the long-term asymmetric key of the partner session. Additionally, in case of handover protocol, the adversary can also forge messages between gNB*s* if they compromised the base key (BK). Therefore to prevent this attack, we introduce a cleanness predicate that ensures the $\mathcal{A}$ is prevented from issuing **CorruptLTK**, **CorruptASK**, **CorruptASK**, **Reveal** or **StateReveal** queries before the test session's acceptance.

Definition 40 (cleanness predicate) *A session π_i^s in the* KIND *experiment described above is* $\mathbf{clean}_{AKA}$ *if conditions (1,2,3) hold, and is* $\mathbf{clean}_H$ *if conditions (1,2,4) hold:*

1. ***Reveal***(i, s, ρ) *has not been issued, and if a matching session π_j^t exists,* ***Reveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
2. *The query* ***StateReveal***(i, s, ρ) *has not been issued* and *for all j, t such that π_j^t has a matching conversation with π_i^s,* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
3. *If there is no $(j, t) \in n_P \times n_S$ such that π_j^t is a matching subset for π_i^s,* ***CorruptLTK***(i) *and* ***CorruptASK***$(\pi_i^s.pid)$ *have not been both issued before $\pi_i^s.\alpha =$* `accepted`.
4. ***CorruptBK***(i, j) *has not been issued before $\pi_i^s.\alpha =$* `accepted`.

6.4.2.1 KIND-security of PGUP-AKA protocol

Here we present our formal analysis and results for the key indistinguishability (KIND) security of the PGUP-AKA protocol.

Theorem 21 KIND-***security of PGUP-AKA*** *. The PGUP-AKA protocol depicted in Figures 6.3 and 6.4 is considered* KIND*-secure and supports PFS, provided that it satisfies the cleanness predicate described in Def. 40. In other words, for any PPT algorithm $\mathcal{A}$ attempting to breach the* KIND *experiment, the probability of success is negligible. This holds true under the assumption* PKW^+, AEAD, MAC, KEM *and* KDF *.*

Proof: Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game subject to certain assumptions to

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{AKA},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{AKA},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

.

Case 1: Here we present the analysis of **Case 1**, where π_i^s accepts without a matching session.

Game 0: This is the original key indistinguishability experiment described in Def. 25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{AKA},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.

Game 1: In this game, we incorporate an abort event triggered by $\mathcal{A}$ issuing a query $\mathbf{Test}(i,s,\rho)$ where π^s_i accepts without a matching session. This scenario precisely mirrors the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, as per **Case 1**, π^s_i has no matching session, and following the logic of **Game 1** where we abort upon π^s_i accepting without a matching session, it can be deduced that $\mathcal{A}$ is incapable of querying $\mathbf{Test}(i,s,\rho)$. Consequently, the KIND game unfolds identically, regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

Case 2: Here we present the analysis of **Case 2**, where π^s_i accepts with a matching session.

First, we recall the cleanness predicate 32, which restricts $\mathcal{A}$ from issuing a $\mathbf{Reveal}(i,s,\pi^s_i.\rho)$ query to π^s_i (and to any session π^t_j such that π^t_j is a matching session with π^s_i). Additionally, $\mathcal{A}$ is prevented from issuing a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$ query or any query to any session π^t_j such that π^t_j is a matching session with π^s_i. We proceed through a series of games to illustrate the security property.

Game 0: This is the original key indistinguishability experiment described in Def. 25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{AKA},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.

Game 1: In this game, we guess the index $(i,s) \in n_P \times n_S$, and abort if $\mathcal{A}$ issues a $\mathbf{Test}(i^*,s^*,\pi^{s^*}_{i^*})$ query such that $i \neq i^*$ and $s \neq s^*$. This introduces the following bound: $\mathbf{Adv}_{G_0} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_1}$.

Game 2: In this game, we guess the index $(j,t) \in n_P \times n_S$, and abort if π^t_j is not the matching subset of π^s_i, which must exist by **Case 2**. This introduces the following bound: $\mathbf{Adv}_{G_1} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_2}$.

Game 3 : In this game, we replace the computation of rk in any sessions computes it with uniformly random value ($\hat{rk}$). We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW^+ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls $\mathsf{New}()$, $\mathcal{B}_1$ calls to $\mathsf{PKW}^+.\mathsf{New}()$, then we assign i to each $SUTI$ and a table $(SUTI,i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T,AD,m,i) \rightarrow C$ on a new entry, such that $(\cdot,T,AD,m,i) \notin C_{txt}$. Similarly, the R_k register will be updated whenever we call $\mathsf{Drv}(T,AD,i) \rightarrow rk$ on new entries, such that $(\cdot,T,AD,i) \notin R_k$. Finally, T_p will be updated whenever we call $\mathsf{Punc}(T,i,\rho)$. Whenever $\mathbf{Send}(i,s,m,\rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T,i,\rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Drv was queried, the computation of $\mathsf{Drv}(K,T,C_0\|C_1) \rightarrow rk$ is replaced with a call to $\mathsf{Drv}(i,T,C_0\|C_1)$ by $\mathcal{B}_1$.
- IF $(T,i,\overline{\rho} \in T_p$: If Drv were queried, we recover the output from the register (R_k).

– IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Drv was queried, the computation of $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is replaced with $\mathsf{Drv}(i, T, C_0\|C_1)$,by $\mathcal{B}_1$ and adds it to the register as $\mathsf{Drv}(i, T, C_0\|C_1) \rightarrow R_K$.

This process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability security, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1,\mathsf{PKW}^+}(\mathcal{A})$.

Game 4: In this game we replace the authentication and mac keys AK, MK with uniformly random values $\hat{AK}, \hat{MK}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{rk}$ and replacing the computation of AK, MK in any session that computes AK, MK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{AK}, \hat{MK}$. Since $AK, MK \leftarrow \mathsf{KDF}(\hat{rk})$ and by **Game 3** $\hat{rk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_2$ to break the security of KDF, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$.

Game 5: In this game, we replace the response and session keys RK, SK with uniformly random values $\hat{RK}, \hat{SK}$. We do so by defining a reduction a $\mathcal{B}_3$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{AK}$ and replacing the computation of RK, SK in π^s_i and π^t_j with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}, \hat{SK}$. Since $RK, SK \leftarrow \mathsf{KDF}(\hat{AK})$ and by **Game 4**,$\hat{AK}$ is already uniformly random, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 6: In this game, we replace the response (K_{SEAF}, HK) with uniformly random value $(\hat{K_{SEAF}}\hat{HK})$. We do so by reusing the reduction $\mathcal{B}_4$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}, \hat{RK}$ and replacing the computation of (K_{SEAF}, HK) in π^s_i and π^t_j with the outputs from the ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}$) and ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{K_{SEAF}}\hat{HK}$). Since $(K_{SEAF}, HK \leftarrow \mathsf{KDF}(\hat{SK}))$ and by **Game 5** $\hat{SK}$ is already uniformly random, so this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can break the KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Here we emphasise that as a result of these changes, the session keys $\hat{rk}, \hat{AK}, \hat{MK}, \hat{RK}, \hat{SK}$, $\hat{K_{SEAF}}, \hat{HK}$ are now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$, thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_6} = 0$.

6.4.2.2 KIND-security of Handover Protocol

Here we present our formal analysis and results for the key indistinguishability (KIND) security of the Handover protocol.

Theorem 22 KIND***-security of PGUP Handover*** *. The Handover protocol depicted in Figure 6.5, is considered* KIND*-secure and supports PFS, provided that it satisfies the cleanness predicate described in Def. 40. In other words, for any PPT algorithm $\mathcal{A}$ attempting*

to breach the KIND *experiment, the probability of success is negligible. This holds true under the assumption* PKW$^+$, AEAD, MAC, KEM *and* KDF *security.*

Proof: Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{HO}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$$

.

We begin by treating **Case 1**.

Case 1: Here we present the analysis of **Case 1**, where π^s_i accepts without a matching session.

> **Game 0**: This is the original key indistinguishability experiment described in Def. 25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{HO},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.
>
> **Game 1**: In this game, we incorporate an abort event triggered by $\mathcal{A}$ issuing a query $\mathbf{Test}(i,s,\rho)$ where π^s_i accepts without a matching session. This scenario precisely mirrors the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Since, as per **Case 1**, π^s_i has no matching session, and following the logic of **Game 1** where we abort upon π^s_i accepting without a matching session, it can be deduced that $\mathcal{A}$ is incapable of querying $\mathbf{Test}(i,s,\rho)$. Consequently, the KIND game unfolds identically, regardless of the bit b sampled by $\mathcal{C}$. Thus $\mathbf{Adv}_{G_1} = 0$.

Case 2: Here we present the analysis of **Case 2**, where π^s_i accepts with a matching session.

First, we recall the cleanness predicate 32, which restricts $\mathcal{A}$ from issuing a $\mathbf{Reveal}(i,s,\pi^s_i.\rho)$ query to π^s_i (and to any session π^t_j such that π^t_j is a matching session with π^s_i). Additionally, $\mathcal{A}$ is prevented from issuing a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$ query or any query to any session π^t_j such that π^t_j is a matching session with π^s_i. We proceed through a series of games to illustrate the security property.

> **Game 0**: This is the original key indistinguishability experiment described in Def. 25 of Chapter 2: $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}_{HO},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.
>
> **Game 1**: In this game, we guess the index $(i,s) \in n_P \times n_S$, and abort if $\mathcal{A}$ issues a $\mathbf{Test}(i^*,s^*,\pi^{s^*}_{i^*})$ query such that $i \neq i^*$ and $s \neq s^*$. This introduces the following bound: $\mathbf{Adv}_{G_0} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_1}$.
>
> **Game 2**: In this game, we guess the index $(j,t) \in n_P \times n_S$, and abort if π^t_j is not the matching subset of π^s_i, which must exist by **Case 2**. This introduces the following bound: $\mathbf{Adv}_{G_1} \leq n_S \cdot n_P \cdot \mathbf{Adv}_{G_2}$.

Game 3: In this game, we replace ek, tk with uniformly random values $\hat{ek}, \hat{tk}$. We do so by defining a reduction a $\mathcal{B}_1$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{HK}$ and replacing the computation of ek, tk in any session that computes ek, tk, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{ek}, \hat{tk}$. Since $ek, tk \leftarrow \mathsf{KDF}(\hat{HK})$ and by the definition of the execution environment, HK is generated uniformly random and independent, and by the definition of **Case 2** $\mathcal{A}$ cannot corrupt HK so this replacement is sound. Any $\mathcal{A}$ that can distinguish **Game 2** from **Game 3** can be used to break KDF security of the KDF scheme, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_1,\mathsf{KDF}}(\mathcal{A})$.

Game 4: In this game, we replace handover keys HK^* with uniformly random values $\hat{HK^*}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{tk}$ and replacing the computation of HK^* in any session that computes HK^*, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK^*}$. Since $HK^* \leftarrow \mathsf{KDF}(\hat{tk})$ and by **Game 3** $\hat{tk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 3** from **Game 4** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2,\mathsf{KDF}}(\mathcal{A})$.

Game 5: In this game, we replace $hk, \widehat{HK}$ with uniformly random values $\hat{hk}, \widehat{\hat{HK}}$. We do so by defining a reduction a $\mathcal{B}_3$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{HK^*}$ and replacing the computation of $hk, \widehat{HK}$ in any session that computes them, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK^*}$. Since $hk, \widehat{HK} \leftarrow \mathsf{KDF}(\hat{HK^*})$ and by **Game 4** $\hat{HK^*}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Here we emphasise that as a result of these changes, the session keys $\hat{ek}, \hat{tk}, \hat{HK^*}, \hat{hk}, \widehat{\hat{HK}}$ are now uniformly random and independent of the protocol execution regardless of the bit b sampled by $\mathcal{C}$; thus $\mathcal{A}$ has no advantage in guessing the bit b: $\mathbf{Adv}_{G_5} = 0$.

6.4.3 Unlinkability Security

In this section, we analyse our proposed scheme's unlinkability (Unlink) security, demonstrating that they achieve unlinkability security. We maintain the definition of Unlink-security as presented in Def. 26 of Chapter 2. A protocol Π is Unlink-secure, if no PPT algorithms $\mathcal{A}$ exist that can win the Unlink security game with a non-negligible advantage. Here, we specify our particular protocols' adversary capabilities and cleanness predicate.

Adversary Queries In the Unlink game, $\mathcal{A}$ has access **Create**, **Send**, **CorruptLTK**, **CorruptASK**, **Reveal**, **StateReveal**, **Test** and **SendTest** queries described above. Unlike in the KIND game, **Test** in Unlink allows the adversary to initialise one of two sessions (depending on a bit b sampled by the challenger), and **SendTest**, which allows the adversary to interact with that session without revealing which party owns it.

- **Test**(s,i,s',i') $\rightarrow m$: allows $\mathcal{A}$ to begin a new session π_b, where $(\pi_0 = \pi_i^s)$ or $(\pi_1 = \pi_{i'}^{s'})$, where b is sampled by $\mathcal{C}$, and both π_i^s and $\pi_{i'}^{s'}$ are **clean**. **Test** query is only allowed to be issued by $\mathcal{A}$ if $\pi_b.\alpha \neq in - progress$ and **Send** queries to sessions owned by UE_i or $\mathrm{UE}_{i'}$ are only allowed to be issued by $\mathcal{A}$ until π_b has rejected or accepted the experiment execution.

- **SendTest**$(m) \rightarrow (m')$: allows $\mathcal{A}$ to send a message m to π_b after issuing **Test**. The $\mathcal{C}$ returns a $\perp$ if $\pi_b.\alpha \neq in - progress$.

Cleanness Predicate. We now turn to describe our cleanness predicates. It is important to note that it is trivial for $\mathcal{A}$ to determine which of UE_i or $\text{UE}_{i'}$ owns session π_b if the $\mathcal{A}$ has compromised both the long-term key shared between CN and UE, as well as the long-term asymmetric key of the CN , or if the $\mathcal{A}$ compromised the long-term key of the partner session, where$\pi_j^t.\rho = \text{UE}$. To mitigate this type of attack, we introduce a cleanness predicate that ensures the $\mathcal{A}$ is prevented from issuing **CorruptLTK**, **CorruptASK**, or **StateReveal** queries before the test session's acceptance.

Definition 41 (Cleanness predicate) *A session π_i^s in the* Unlink *experiment is* **clean** *if the following conditions hold:*

1. ***StateReveal***$(i, s, \pi_i^s.\rho)$ *has not been issued* and *for all sessions π_j^t such that π_j^t is a matching subset of π_i^s* ***StateReveal***$(j, t, \pi_j^t.\rho)$ *has not been issued.*
2. *If ($\pi_i^s.\rho = \text{gNB}$) or ($\pi_i^s.\rho = \text{CN}$), and there exists no $(j, t) \in n_P \times n_S$ such that π_j^t is a matching session of π_i^s,* ***CorruptLTK***$(\pi_i^s.pid)$ *have not been issued before $\pi_i^s.\alpha =$* `accepted`.
3. *If $\pi_i^s.\rho = \text{UE}$, and there exists no $(j, t) \in n_P \times n_S$ such that π_j^t is a matching session of π_i^s,* ***CorruptLTK***(i) *or* ***CorruptASK***(CN) *have not both been issued before $\pi_i^s.\alpha =$* `accepted`.

6.4.3.1 Unlink-security of PGUP-AKA protocol

Here we present our formal analysis and results for the unlinkability (Unlink) security of the PGUP-AKA protocol.

Theorem 23 Unlink-***security of the PGUP scheme*** *. The proposed scheme, as shown in Figures 6.3 and 6.4, is considered* Unlink*-secure and supports PFP, provided that it satisfies the cleanness predicate described in Def. 41. In other words, for any PPT algorithm $\mathcal{A}$ attempting to breach the* Unlink *experiment, the probability of success is negligible. This holds true under the assumption* PKW^+, AEAD, MAC, KEM *and* KDF *security are all maintained.*

Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\text{Unlink},\mathbf{clean}_{AKA},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\text{Unlink},\mathbf{clean}_{AKA},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\text{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\text{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\text{Unlink},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

.

We begin by treating **Case 1**.

Case 1: In this case, we assume that π_b (such that $\mathcal{A}$ issues **Test**(i, s, i^*, s^*)) accepts without a matching subset.

Game 0: This is the original unlinkability experiment defined in (26) of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{AKA},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.

Game 1: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i,s,i^*,s^*)$ and π_b accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Considering **Case 1**, where π lacks a matching session, and according to **Game 1**, if π_b accepts without such a match, we abort. Consequently, $\mathcal{A}$ is unable to terminate and produce a guess bit b'. Consequently, the Unlink game progresses uniformly irrespective of the bit b sampled by $\mathcal{C}$. Thus, $\mathbf{Adv}_{G_1} = 0$.

We now turn to **Case 2**.

Case 2: In this case, we assume that π^s_i (such that $\mathcal{A}$ issues $\mathbf{Test}(i,s,i^*,s^*)$) accepts with a matching subset.

First, we recall that the cleanness predicate defined in Def. 41 prevents the $\mathcal{A}$ from issuing a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$, nor to any session π^t_j such that π^t_j is a matching session or subset with π^s_i. We proceed via the following sequence of games.

Game 0: This is the original unlinkability experiment defined in (26) of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{AKA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.

Game 1 : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the first session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.

Game 2 : In this game, we guess the index $(t,j) \in n_P \times n_S$ of the first UE session that accepts without a matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_P \cdot n_S \mathbf{Adv}_{G_2}$.

Game 3 : In this game, we guess session index $r \in n_S$ and aborts if π^j_r registers a ciphertext in $C_{txt} \leftarrow (C,T,AD,m,j)$ and π^{CN}_t computes $rk \leftarrow \mathsf{Drv}(k_i,T',C_0)$ and $(.,T',.,.) \in C_{txt}$ but $T' \neq T$, introducing a factor of n_S in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_2} \leq n_S \mathbf{Adv}_{G_3}$.

Game 4 : In this game, we replace the computation of rk in π^j_r and π^{CN}_t sessions with uniformly random value ($\hat{rk}$). We do so by introducing a reduction $\mathcal{B}_1$, which operates as follows: at the beginning of the game, $\mathcal{B}_1$ initiates a PKW^+ challenger $\mathcal{C}_{\mathsf{PKW}^+}$. Every time $\mathcal{A}$ calls $\mathsf{New}()$, $\mathcal{B}_1$ calls to $\mathsf{PKW}^+.\mathsf{New}()$, then we assign i to each $SUTI$ and a table $(SUTI,i)$ is maintained. Additionally, $\mathcal{B}_1$ maintains the following registers C_{txt}, R_k and T_p. C_{txt} will be updated whenever we call $\mathsf{Wrap}(T,AD,m,i) \rightarrow C$ on a new entry, such that $(\cdot,T,AD,m,i) \notin C_{txt}$. Similarly, the R_k register will be updated whenever we call $\mathsf{Drv}(T,AD,i) \rightarrow rk$ on new entries, such that $(\cdot,T,AD,i) \notin R_k$. Finally, T_p will be updated whenever we call $\mathsf{Punc}(T,i,\rho)$. Whenever $\mathbf{Send}(i,s,m,\rho)$ is called, we need first to check the T_p register and follow these conditions:

- IF $(T,i,\rho) \in T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K,T,C_0 \| C_1, SUTI || \Delta)$ is replaced with a call to

$\mathsf{Wrap}(i, T, C_0\|C_1, SUTI||\Delta)$ by $\mathcal{B}_1$. and if Unwrap was queried, the computation of $m \leftarrow \mathsf{Unwrap}(K, T, C_0\|C_1, C_2)$ is replaced with a call to $\mathsf{Unwrap}(i, T, C_0\|C_1, C_2)$ by $\mathcal{B}_1$. Similarly, whenever $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0\|C_1)$.

- IF $(T, i, \overline{\rho}) \in T_p$: Depending on the algorithm that was called, we check its registry for an output. If Unwrap or Drv were queried, we recover the output from the register (C_{txt}, R_k), respectively. However, if Wrap was queried, then the adversary has either issued a **Corrupt**, hence $\mathcal{B}_1$ can simulate Wrap, or the adversary has forged a message between UE and CN, hence breaking the PKW^+ security.
- IF $(T, i, \cdot) \notin T_p$: We forward the query to $\mathcal{C}_{\mathsf{PKW}^+}$. In case Wrap was queried, the computation of $C_2 \leftarrow \mathsf{Wrap}(K, T, C_0\|C_1, SUTI||\Delta)$ is replaced with a call to $\mathsf{Wrap}(i, T, C_0\|C_1, SUTI||\Delta)$ by $\mathcal{B}_1$. Then, the output is added to the register as $(C, i, T, C_0\|C_1, SUTI||\Delta) \rightarrow C_{txt}$. Similarly, whenever $\mathsf{Drv}(K, T, C_0\|C_1) \rightarrow rk$ is called, $\mathcal{B}_1$ replaces the computation with $\mathsf{Drv}(i, T, C_0\|C_1)$. and adds it to the register as $\mathsf{Drv}(i, T, C_0\|C_1) \rightarrow R_K$.

This process is precisely how all parties interact with their collective PKW^+ state, making this replacement sound. If $\mathcal{A}$ can distinguish between these two games, then $\mathcal{A}$ breaks the PKW^+ key indistinguishability security, as formalised in Theorem 18. Thus: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{\mathsf{KIND}}_{\mathcal{B}_1, \mathsf{PKW}^+}(\mathcal{A})$.

Game 5: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_2 that decrypts correctly and is accepted by π_t^{CN}, but was not generated by π_r^j, breaking the authentication security of AEAD in PKW^+. In this game, no replacement is required because if the $\mathcal{A}$ could forge a cipher text, then they would win **Game 5**. Note that $(C_0\|C_1)$ is securely authenticated by $\mathsf{PKW}^+.\mathsf{Wrap}()$ as additional data, and by the definition of **Case 1** $\mathcal{A}$ cannot issue **CorruptLTK**(UE). Any $\mathcal{A}$ that can trigger this abort event can be used to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{INT-CTXT.\mathsf{AEAD}}_{\mathsf{PKW}^+}(\mathcal{A})$.

Game 6: In this game we replace the authentication and mac keys AK, MK with uniformly random values $\hat{AK}, \hat{MK}$. We do so by defining a reduction a $\mathcal{B}_2$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{rk}$ and replacing the computation of AK, MK in any session that computes AK, MK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{AK}, \hat{MK}$. Since $AK, MK \leftarrow \mathsf{KDF}(rk)$ and by **Game 5** $\hat{rk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of KDF. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_2, \mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if the same hash value exists for different inputs, thereby introducing hash collisions during the challenger execution with an honest session. We do so by defining a reduction $\mathcal{B}_3$ that initialises an Hash challenger $\mathcal{C}_{\mathsf{Hash}}$ and maintains a lookup table. Whenever $\mathcal{B}_3$ needs to hash a value, he queries the lookup table on x and recovers (out). The abort event only triggers if $\mathcal{A}$ can get the same output hash values from two distinct input values, thus finding a collision and breaking the security of the Hash scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_3$ to break the security of Hash. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{collision}_{\mathcal{B}_3, \mathsf{Hash}}(\mathcal{A})$.

Game 8: In this game, we replace the response and session keys RK, SK with uniformly random values $\hat{RK}, \hat{SK}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{AK}$ and replacing the computation of RK, SK in any session that computes RK, SK, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}, \hat{SK}$. Since $RK, SK \leftarrow \mathsf{KDF}(\hat{AK})$ and by **Game 7**,$\hat{AK}$ is already uniformly random, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 8** from **Game 9** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_7} \leq \mathbf{Adv}_{G_8} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 9: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a valid tag (MAC), but (MAC) was not output by π^t_j. We do so by defining a reduction $\mathcal{B}_5$ that initialises an MAC challenger $\mathcal{C}_{\mathsf{MAC}}$, which $\mathcal{B}_5$ queries when π^t_j needs to MAC (SQN, R) with $(\hat{MK})$ and from **Game 7** $\hat{MK}$ is already uniformly random and independent. This abort event occurs if MAC is verified correctly under $\hat{MK}$, but MAC was never produced by the $\mathcal{C}_{\mathsf{MAC}}$, breaking the security of the MAC scheme. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the existential unforgeability security of the MAC scheme, as formalised in Def. 4. This implies: $\mathbf{Adv}_{G_8} \leq \mathbf{Adv}_{G_9} + \mathbf{Adv}^{EUFCMA}_{\mathcal{B}_5,\mathsf{MAC}}(\mathcal{A})$.

Game 10: In this game, we replace the response (RES^*) and (K_{SEAF}, HK) with uniformly random value $(\hat{RES^*})$ and $(\hat{K_{SEAF}HK})$, respectively. We do so by defining a reduction $\mathcal{B}_6$ that interacts with a KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{SK}, \hat{RK}$ and replacing the computation of (RES^*) and (K_{SEAF}, HK) in any session that computes RES^*, K_{SEAF}, HK, with the outputs from the ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{RK}$) and ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{K_{SEAF}HK}$). Since $(RES^* \leftarrow \mathsf{KDF}(\hat{RK}))$ and ($\mathcal{C}_{\mathsf{KDF}}$ $\hat{K_{SEAF}HK}$) and by **Game 9** $\hat{RK}$ and $\hat{SK}$ are already uniformly random, so these changes are sound. Any $\mathcal{A}$ that can distinguish **Game 9** from **Game 10** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_9} \leq \mathbf{Adv}_{G_{10}} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_6,\mathsf{KDF}}(\mathcal{A})$.

In this stage, it is important to highlight that all plaintext messages that are sent and received across the network to and from π_b and its corresponding session and subsets are uniformly random and independent of the bit b that is selected by the challenger. As a result, $\mathcal{A}$ has no advantage in predicting the bit b. Adding up the probabilities makes it evident that $\mathcal{A}$ has a negligible advantage in winning the Unlink game. Thus: $\mathbf{Adv}_{G_{10}} = 0$

6.4.3.2 Unlink-security of Universal Handover

Here we present our formal analysis and results for the unlinkability (Unlink) security of the proposed Handover protocol.

Theorem 24 Unlink-***security of the PGUP scheme*** . *The proposed scheme, as shown in Figure 6.5, is considered* Unlink*-secure and supports PFP, provided that it satisfies the cleanness predicate described in Def. 41. In other words, for any PPT algorithm $\mathcal{A}$ attempting to breach the* Unlink *experiment, the probability of success is negligible. This holds true under the assumption* PKW$^+$*,* AEAD*,* MAC*,* KEM *and* KDF *security are all maintained.*

Our proof is divided into two cases, denoted by

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{HO},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \text{ and } \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{HO},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$$

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to

$$\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda).$$

We begin by treating **Case 1**.

Case 1: In this case, we assume that π_b (such that $\mathcal{A}$ issues $\mathbf{Test}(i,s,i^*,s^*)$) accepts without a matching subset.

> **Game 0**: This is the original unlinkability experiment defined in (26) of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{HO},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq Adv_{G_0}$.
>
> **Game 1**: In this game, we introduce an abort event that triggers if $\mathcal{A}$ issues a query $\mathbf{Test}(i,s,i^*,s^*)$ and π_b accepts without a matching session or subset. This is exactly equal to the MA security experiment, and thus we have :$\mathbf{Adv}_{G_0} \leq \mathbf{Adv}_{G_1} + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{HO}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

Considering **Case 1**, where π lacks a matching session, and according to **Game 1**, if π_b accepts without such a match, we abort. Consequently, $\mathcal{A}$ is unable to terminate and produce a guess bit b'. Consequently, the Unlink game progresses uniformly irrespective of the bit b sampled by $\mathcal{C}$. Thus, $\mathbf{Adv}_{G_1} = 0$.

We now turn to **Case 2**.

Case 2: In this case, we assume that π^s_i (such that $\mathcal{A}$ issues $\mathbf{Test}(i,s,i^*,s^*)$) accepts with a matching subset.

First, we recall that cleanness predicate defined in Def. 41 prevents the $\mathcal{A}$ from issuing a $\mathbf{StateReveal}(i,s,\pi^s_i.\rho)$, nor to any session π^t_j such that π^t_j is a matching session or subset with π^s_i. We proceed via the following sequence of games.

> **Game 0**: This is the original unlinkability experiment defined in (26) of Chapter 2: $\mathbf{Adv}^{\mathsf{Unlink},\mathbf{clean}_{HO}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq \mathbf{Adv}_{G_0}$.
>
> **Game 1** : In this game, we guess the index $(i,s) \in n_P \times n_S$ of the test session, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_0} \leq n_P \cdot n_S \mathbf{Adv}_{G_1}$.
>
> **Game 2** : In this game, we guess the index $(t,j) \in n_P \times n_S$ of the test session's matching subset, introducing a factor of $n_P \times n_S$ in $\mathcal{A}$'s advantage: $\mathbf{Adv}_{G_1} \leq n_P \cdot n_S \mathbf{Adv}_{G_2}$.
>
> **Game 3**: In this game, we replace ek, tk with uniformly random values $\hat{ek}, \hat{tk}$. We do so by defining a reduction a $\mathcal{B}_1$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{HK}$ and replacing the computation of ek, tk in any session that computes ek, tk, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{ek}, \hat{tk}$. Since $ek, tk \leftarrow \mathsf{KDF}(\hat{HK})$ and by the definition of the execution environment, HK is generated uniformly random and independent, and by the definition of **Case 2** $\mathcal{A}$ cannot corrupt HK so this replacement is sound. Any $\mathcal{A}$ that can distinguish **Game 2** from **Game 3** can be used to break KDF security of the KDF scheme, as formalised in Def. 2. Thus: $\mathbf{Adv}_{G_2} \leq \mathbf{Adv}_{G_3} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_1,\mathsf{KDF}}(\mathcal{A})$.

Game 4: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext C_{UE} (keyed by $\hat{ek}$) that decrypts correctly and was not generated by SgNB, breaking the authentication security of AEAD. We do so by defining a reduction $\mathcal{B}_2$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$, which $\mathcal{B}_2$ queries when he needs to encrypt/decrypt with $\hat{ek}$. By **Game 3** $\hat{ek}$ is already uniformly random and independent, and this replacement is undetectable. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_2$ to break the security of AEAD, as formalised in Def. 8. This implies: $\mathbf{Adv}_{G_3} \leq \mathbf{Adv}_{G_4} + \mathbf{Adv}^{INT-CTXT}_{\mathcal{B}_2,\mathsf{AEAD}}(\mathcal{A})$.

Game 5: In this game, we replace handover keys HK^* with uniformly random values $\hat{HK}^*$. We do so by defining a reduction a $\mathcal{B}_3$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{tk}$ and replacing the computation of HK^* in any session that computes HK^*, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK}^*$. Since $HK^* \leftarrow \mathsf{KDF}(\hat{tk})$ and by **Game 3** $\hat{tk}$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 4** from **Game 5** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_4} \leq \mathbf{Adv}_{G_5} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_3,\mathsf{KDF}}(\mathcal{A})$.

Game 6: In this game, we replace hk with uniformly random value $\hat{hk}$. We do so by defining a reduction a $\mathcal{B}_4$, that initialises an KDF challenger $\mathcal{C}_{\mathsf{KDF}}$, querying $\mathcal{C}_{\mathsf{KDF}}$ with $\hat{hk}^*$ and replacing the computation of hk in any session that computes hk, with the outputs from the $\mathcal{C}_{\mathsf{KDF}}$ $\hat{HK}^*$. Since $hk \leftarrow \mathsf{KDF}(\hat{HK}^*)$ and by **Game 5** $\hat{HK}^*$ is already uniformly random and independent, this change is sound. Any $\mathcal{A}$ that can distinguish **Game 5** from **Game 6** can be used to break KDF security of the KDF scheme. Thus: $\mathbf{Adv}_{G_5} \leq \mathbf{Adv}_{G_6} + \mathbf{Adv}^{\mathsf{KDF}}_{\mathcal{B}_4,\mathsf{KDF}}(\mathcal{A})$.

Game 7: In this game, we introduce an abort event that triggers if $\mathcal{A}$ generates a ciphertext HO_{res} (keyed by $\hat{hk}$) that decrypts correctly and was not generated by TgNB, breaking the authentication security of AEAD. We do so by defining a reduction $\mathcal{B}_5$ that initialises an AEAD challenger $\mathcal{C}_{\mathsf{AEAD}}$, which $\mathcal{B}_5$ queries when he needs to encrypt/decrypt with $\hat{hk}$. By **Game 6** $\hat{hk}$ is already uniformly random and independent, and this replacement is undetectable. Any $\mathcal{A}$ that can trigger the abort event can be used by $\mathcal{B}_5$ to break the security of AEAD. This implies: $\mathbf{Adv}_{G_6} \leq \mathbf{Adv}_{G_7} + \mathbf{Adv}^{INT-CTXT}_{\mathsf{AEAD}}(\mathcal{A})$

In this stage, it is important to highlight that all plaintext messages that are sent and received across the network to and from π_b and its corresponding session and subsets are uniformly random and independent of the bit b that is selected by the challenger. As a result, $\mathcal{A}$ has no advantage in predicting the bit b. By adding up the probabilities, it becomes evident that $\mathcal{A}$ has a negligible advantage in winning the Unlink game. Thus: $\mathbf{Adv}_{G_7} = 0$

6.5 Performance Evaluation and Comparison

In this section, we conduct a comprehensive evaluation of the proposed PGUP in comparison to the conventional 5G-AKA and HO protocols. Our analysis encompasses feature set comparison, computational overhead, and communication cost. This multifaceted approach allows for a thorough assessment of the scheme's security and privacy relative to current works in the field.

Table 6.3: *Features comparison*

Type	Scheme	MA	PFS	PFP	Unlink	SRM	UHO
Symmetric-key-based	5G [4]	✓	✗	✗	✗	✗	✗
	AKA'[67]	✓	✗	✗	✓	✗	✗
	ReHand[37]	✓	✗	✗	✓	✓	‡
Asymmetric(Public)-key-based	*AAKA* [70]	✓	✗	✗	✓	✗	✗
	AKA$^+$ [47]	✓	✗	✗	✓	✗	✗
Ours (Symmetric-key-based)	PGUP	✓	✓	✓	✓	✓	✓

MA:Mutual Authentication, **PFS**: Perfect Forward Secrecy, **PFP**: Perfect Forward Privacy, **Unlink**: Unlinkability, **SRM**: Secure Revocation Management, **UHO**: Universal HO, ‡: Region-based HO

6.5.1 Security Features Comparison

Our analysis compares the PGUP scheme with existing protocols based on the desired security and privacy features discussed in Chapter 3, Section 3.4.1. These features are crucial for overcoming several security and privacy issues found in the current version of 5G protocols and are essential for any AKA and HO protocols in cellular networks.

Table 6.3 presents a comparison of the proposed PGUP scheme with several existing authentication schemes in 5G and mobile communication systems [4, 67, 37, 70, 47]. As illustrated, PGUP aims to address various security and privacy challenges by utilizing a symmetric-key cryptography, which is particularly well-suited for UE with limited computational resources and aligns with the existing 5G infrastructure. Although symmetric-key protocols like conventional 5G [4] and AKA' [67] provide MA, they fail to offer essential features such as PFS and PFP. Specifically, conventional 5G only supports MA, while AKA' improves upon this by achieving Unlink, though it still falls short in providing both forward secrecy and privacy. ReHand [37], another symmetric-key-based approach, supports Secure Revocation Management (SRM) and region-based handover (denoted by ‡), addressing certain limitations of its predecessors. However, it similarly lacks PFS, PFP, and Universal HO (UHO) capabilities.

Compared to asymmetric-key approaches such as AAKA [70] and AKA$^+$ [47], PGUP aims to achieve similar security goals while leveraging the efficiency of symmetric-key cryptography. Although AAKA and AKA$^+$ provide MA and Unlink, comparable to some symmetric-key protocols, they lack key features such as PFS, PFP, SRM, and UHO.

PGUP addresses the missing features found in many existing protocols by incorporating MA, Unlink, PFS, PFP, UHO and SRM, significantly enhancing the overall security and privacy properties of the 5G environment.

6.5.2 Computational and Communication Cost

To evaluate our PGUP scheme against conventional 5G-AKA and HO protocols, we established a testbed simulating network entities (CN, gNB) using a Dell Inspiron (i7 core, 2.30GHz CPU, 16.0 GB RAM) and UE using an Android-10 smartphone (octa-core 1.8GHz Quad-Core ARM Cortex-A55, 2.7GHz Quad-Core Mongoose M3, 6GB RAM). Cryptographic operations were implemented using Java Cryptography Extension (JCE) [66]. We analyzed both computational and communication costs, considering factors such as propagation and transmission

Table 6.4: *Performance comparison.*

Schemes	Entity	Comp.cost(ms) AKA	Comp.cost(ms) HO	Link	Comm.cost(ms) AKA	Comm.cost(ms) HO
Conventional-5G [4]	T_{UE}	2.03	2.366	UP	0.02764	0.04358
	T_{Sys}	1.09	1.6	Down	0.00646	0.04067
	Total	3.12	3.97	Total	0.0341	0.08425
PGUP	T_{UE}	7.01	0.342	UP	0.05862	0
	T_{Sys}	3.23	4.81	Down	0.008349	0.0535
	Total	10.24	5.152	Total	0.066969	0.0535
UE: User Equipment, **Sys**:System						

time, message dimensions, and network data rates. Following 3GPP specifications [4], we used 25 Mbps uplink and 50 Mbps downlink rates for a wide-area scenario. Propagation delay was calculated using a wave speed of $3 \times 10^8 m/s$ and a maximum 5G cell size of 200 m, resulting in a 0.67 μs delay.

To measure the communication cost for the PGUP scheme, we aligned cryptographic element sizes with 5G specifications where possible, balancing security and efficiency. The SUTI is 8 bytes, while its hashed version, SUTI*, is 32 bytes, consistent with the SHA-256 output. The Δ parameter, occupying 4 bytes, provides anonymization for the SUTI. The puncturable key wrapping tag is 16 bytes. For ECDH key exchange (used in ECIES-KEM), C_0 (public key) is 91 bytes, typical for a 256-bit ECDH key. The C_1 parameter is 8 bytes, while C_2 totals 163 bytes, comprising 64 bytes of ciphertext (12-byte nonce, 36-byte encrypted message, 16-byte authentication tag) and 99 bytes of associated data. Adhering to 5G specifications, AUTN, RES , and R are each 16 bytes. R', another ciphertext with AD, is 32 bytes. Both C_{UE} and HO_{res} mirror C_2 at 163 bytes. These carefully chosen sizes balance security requirements, 5G compatibility, and efficient communication in our PGUP scheme.

Table 6.4 shows a comparison of the performance based on communication and computation. It shows the time required for executing the AKA and HO protocols, capturing the total time required for protocol execution at both the User Equipment level (T_{UE}) and the system level (T_{Sys}) (i.e., gNB and CN). In terms of computation, although PGUP's AKA protocol shows increased computational costs compared to the 5-AKAG, it offers significant improvements in handover operations. Specifically, PGUP's HO protocol demonstrates an 85.5% reduction in UE-level computational cost (0.342 ms vs 2.366 ms for 5G-HO). This improvement particularly benefits UE with limited resources, such as processing power and memory. In terms of communication costs, PGUP's HO protocol achieves a 36.5% reduction in total communication cost (0.0535 ms vs 0.08425 ms for 5G-HO). A key advantage of PGUP's HO protocol is eliminating uplink communication costs, which can significantly reduce network load during handover operations. These results suggest that PGUP offers substantial improvements in handover efficiency, particularly beneficial for UE performance and network load management during frequent handovers in 5G networks. The minor increase in computational cost for PGUP-AKA may be justified by the enhanced security features, making PGUP a viable option for scenarios requiring robust protection against emerging threats in 5G networks.

6.6 Summary

This chapter has successfully addressed the third research question ($\mathbf{RQ}_3$) of this thesis by introducing PGUP, an innovative symmetric-based scheme designed to enhance security and privacy in 5G-AKA and HO protocols. Building upon the foundations laid by UniHand in the previous chapter, PGUP offers a complementary approach that aligns more closely with the symmetric cryptography primitives predominant in the current 5G infrastructure.

The core innovation of PGUP lies in the development of PKW^+, a new variant of puncturable key wrapping tailored for 5G environments. This novel technique enables the achievement of PFS and PFP within a symmetric framework, addressing a significant challenge in the field. Furthermore, PGUP effectively mitigates the linkability vulnerability posed by active adversaries, substantially improving user privacy.

This chapter provided a comprehensive description of the PGUP scheme, detailing both AKA and HO protocols. The design maintains conceptual alignment with existing 5G structures, facilitating potential seamless integration into current networks. Rigorous formal security analysis demonstrated PGUP's achievement of critical security properties, including MA, KIND, and Unlink. The proofs covered various scenarios and adversary capabilities, providing thorough validation of the scheme's security features.

Performance evaluations showed significant improvements over conventional 5G protocols, with an 85.5% reduction in UE-level computational cost and a 36.5% reduction in total communication cost for handover operations. These advancements, combined with enhanced security features, position PGUP as a promising solution for addressing current and emerging security challenges in 5G networks.

By leveraging symmetric cryptography, PGUP offers a unique balance between advanced security features and computational efficiency, making it particularly suitable for resource-constrained UEs in 5G environments. This work contributes to improving the security and privacy of mobile communication systems while considering the practical constraints of existing infrastructure, paving the way for more secure and efficient 5G networks. [†]

[†]**This work has been submitted to *Privacy Enhancing Technologies Symposium* (PETS 2025).**

Chapter 7

Conclusion and Future Work

This thesis has investigated critical security and privacy challenges in 5G networks, focusing on developing novel authentication and handover protocols. As established in the research context in Chapter 1, the exponential growth of connected devices and the evolution of mobile networks have created a continuous need for enhanced security measures, particularly in the context of 5G's heterogeneous architecture and the deployment of ultra-dense SCNs. This thesis has aimed to address the vulnerabilities identified in existing 5G-AKA and HO protocols while simultaneously considering the challenges posed by frequent handovers in SCNs. Our research has been guided by the need to develop protocols that enhance security, preserve user privacy, and ensure seamless connectivity for roaming users. The following discussion revisits our research questions and summarises this study's key findings and contributions.

In Chapter 4, we addressed our first research question **($\mathbf{RQ}_1$)** by proposing a privacy-preserving *inter-region* authentication and handover scheme for roaming users in 5G networks. This scheme leverages asymmetric-key-based authenticated key exchange and handover protocols that preserve user privacy and network security while providing a seamless region-based handover mechanism and effective membership revocation management. It balances robust security features with efficient user revocation management (SRM) by utilizing several cryptographic schemes, including sanitizable signatures and universal accumulators. The formal security analysis demonstrated that the proposed protocol achieves mutual authentication (MA), user unlinkability (Unlink), and key indistinguishability (KIND). Moreover, our comparative study showed that the scheme offers enhanced security features, although it introduces some computational and communication overhead compared to existing solutions. This contribution advances the field by addressing critical security and efficiency concerns in 5G handovers, particularly in inter-region scenarios.

In Chapter 5, we tackled our second research question **($\mathbf{RQ}_2$)** by introducing UniHand, a Universal Handover scheme. UniHand aims to achieve both Key Compromise Impersonation (KCI) resilience and Key-Escrow-Free (KEF) properties for roaming users in 5G SCN. The proposed universal handover approach attempts to transcend the limitations of region-based protocols, offering seamless transitions across geographical areas. The security analysis demonstrates that UniHand fulfils several critical security and privacy properties, including MA, KIND, and Unlink. The performance evaluation showed that while UniHand introduces some computational overhead, it significantly improves overall network efficiency, mainly by reducing Core Network involvement during handovers.

In Chapter 6, we addressed the third research question **($\mathbf{RQ}_3$)** through the development of PGUP, a symmetric-key-based scheme designed to enhance security and privacy in 5G-AKA and HO protocols. The key innovation in PGUP is the introduction of Puncturable Key Wrapping+ (PKW^+), a novel primitive tailored for 5G environments. This approach enabled PGUP to achieve Perfect Forward Secrecy (PFS) and Perfect Forward Privacy (PFP) within a symmetric cryptography framework, representing a notable challenge in the field. Our rigorous security analysis confirmed that PGUP achieves MA, KIND, and Unlink. Performance evaluations showed significant improvements over conventional 5G protocols, with an 85.5% reduction in UE-level computational cost and a 36.5% reduction in total communication cost for handover operations. These advancements, combined with enhanced security features, position PGUP as a promising solution for addressing current and emerging security challenges in 5G networks while aligning with existing infrastructure constraints.

After addressing these three research questions, it is valuable to revisit and expand upon the comparative analysis initially presented in Chapter 3, Section 3.5. Table 7.1 builds upon our earlier review of existing protocols, where we incorporate our proposed solutions to provide a comprehensive overview of the advancements made in this thesis. This expanded comparative analysis demonstrates the significance of our proposed schemes. Compared to the existing state-of-the-art protocols discussed earlier in this thesis, our contributions, the Privacy-Aware Secure Region-based Handover scheme, UniHand, and PGUP, offer more comprehensive set of security and privacy features. Notably, our proposed schemes achieve MA, UA, PFS, Unlink, and SRM. Additionally, UniHand further offers KCI resilience and KEF properties. PGUP, meanwhile, provides PFP while maintaining the efficiency of symmetric-key cryptography. Furthermore, both UniHand and PGUP support universal handover, a feature only present in a few existing protocols. This comparison not only demonstrates the significance of our research but also contextualizes our work within the broader landscape of 5G security solutions, highlighting how we've addressed the gaps identified in our initial analysis.

In conclusion, this thesis aims to contribute to the 5G security and privacy field by developing three novel protocols, each building upon the insights and limitations of its predecessors. The Privacy-Aware, Secure Region-based Handover scheme laid a foundation for

Table 7.1: *Summary of features comparison with State-of-the-art protocols.*

Type	Scheme	MA	UA	PFS	PFP	Unlink	KCI	KEF	SRM	UHO	PM
Symmetric	**5G** - [4]	✓	✓	✗	✗	✗	✗	✗	✗	✗	CF
	AKA' - [67]	✓	✓	✗	✗	✓	✗	✗	✗	✗	Tamarin
	ReHand - [37]	✓	✓	✗	✗	✓	✗	✗	✗	✓	‡	CF
	LSHA - [69]	✓	N/P	◎	✗	✗	✗	✗	✗	‡	BAN+Scyther
	Protocol of- [26]	✓	✓	✗	✗	✗	✗	✗	✗	✗	RUBIN
Asymmetric	**AKA$^+$** - [47]	✓	✓	✗	✗	✓	✗	✗	✗	✗	Bana-Comon
	AAKA - [70]	✓	✓	✗	✗	✓	✗	✓	✗	✗	CF
	RUSH - [71]	✓	✓	◎	✗	✗	✗	✓	✗	✓	BAN+AVISPA
	CPPHA - [31]	✓	✓	✗	✗	✓	✗	✗	✗	✓	BAN+Scyther
Proposed	**Region-based Handover(Chapter 4)**	✓	✓	✓	✗	✓	✗	✗	✓	‡	CF
	UniHand (Chapter 5)	✓	✓	✓	✗	✓	✓	✓	✓	✓	CF
	PGUP (Chapter 6)	✓	✓	✓	✓	✓	✗	✗	✓	✓	CF

MA:Mutual Authentication, **UA**:User Anonymity, **PFS**: Perfect Forward Secrecy, **PFP**: Perfect Forward Privacy, **Unlink**: Unlinkability, **KCI**: Key compromise impersonation resilience, **KEF**: Key-escrow Free , **SRM**: Secure Revocation Management, **UHO**: Universal HO, ◎: Partial, **N/P**: no information provided ‡: Neighbour/Region-based HO, **PM**: Proof method, **CF**: Computational Formal security analysis

secure inter-region handovers. UniHand expanded on this by introducing universal handover capabilities and addressing critical vulnerabilities like KCI attacks. Finally, PGUP demonstrated the potential of symmetric-key approaches in achieving advanced security properties within the constraints of 5G systems.

Future Directions: The schemes proposed in this thesis address key security and privacy challenges in 5G networks, contributing to the development of more secure, efficient, and privacy-preserving mobile communication systems. As 5G networks continue to expand and evolve, the techniques presented here may serve as a foundation for future security frameworks. This work also opens up several promising directions for further research and development, which could extend the approaches presented in this thesis. One potential direction is the development of solutions that are fully backwards-compatible with existing 5G infrastructure. Such approaches could potentially facilitate easier adoption, allowing network operators to enhance their security measures without requiring extensive modifications to their current systems.

Building on the concept of backward compatibility, future research directions could explore how 5G features, particularly network slicing, may address security and privacy challenges while ensuring interoperability with legacy systems. Instead of developing fully backwards-compatible solutions that might compromise security, future work could investigate how network slicing enables isolated environments for legacy protocols alongside more secure solutions. This approach would allow for a gradual transition, balancing the need for compatibility with enhanced security while addressing vulnerabilities in 5G-AKA and HO protocols.

In parallel with addressing these compatibility challenges, the process of reading 5G technical specifications has highlighted potential areas for improvement in the documentation of future network generations. The complex nature of these specifications, which often rely on cross-referencing multiple documents, presents a significant challenge for researchers working to secure these systems. Future work could explore strategies to develop more self-contained technical specifications for 6G and beyond. This effort might address some of the difficulties encountered in navigating 5G documentation, including instances where security and privacy requirements were underspecified, leading to security and privacy vulnerabilities in the network. By improving the documentation of security protocols and requirements, future specifications could address the vulnerabilities encountered due to underspecified requirements in 5G standards. This approach aligns with ongoing research efforts, such as those in the present thesis, which aim to overcome these vulnerabilities resulting from underspecification.

As cryptography evolves, exploring post-quantum security solutions is essential for future research. The potential future threat of quantum computers to current cryptographic methods suggests a need for investigation into quantum-resistant algorithms and protocols that could be tailored for mobile network environments. However, implementing post-quantum cryptography (PQC) in resource-constrained devices such as user equipment (UE) might introduce several challenges. With limited processing power, memory, and battery life, these devices could potentially struggle with the computational demands and larger key sizes that PQC algorithms may require. Addressing the possible impacts on latency, energy consumption, and overall performance might be crucial. Future research could explore ways to optimize these algorithms for constrained environments, possibly by investigating lightweight cryptographic schemes or considering using 5G technologies like network slicing to offload complex cryptographic operations to more capable network segments. This aligns with the

ongoing need for enhanced security measures in the face of evolving threats, as emphasized throughout the thesis.

Bibliography

[1] Getting to 5G: Comparing 4G and 5G System Requirements - Qorvo, September 2017.

[2] 3GPP. Feasibility Study on New Services and Markets Technology Enablers for Critical Communications. Technical Specification version 14, 2016.

[3] 3GPP. ETSI TS 133 320 V15.0.0 (2018-07) - Universal Mobile Telecommunications System (UMTS); LTE; Security of Home Node B (HNB) / Home evolved Node B (HeNB) (3GPP TS 33.320 version 15.0.0 Release 15). Technical Report V15.0.0, 2018.

[4] ETSI 3rd Generation Partnership Project (3GPP). Security architecture and procedures for 5G System. Technical Specification version 16.3.0 Release 16, 3GPP, August 2020.

[5] Martín Abadi and Mark R. Tuttle. A semantics for a logic of authentication (extended abstract). In *Proceedings of the tenth annual ACM symposium on Principles of distributed computing*, PODC '91, pages 201–216, New York, NY, USA, July 1991. Association for Computing Machinery.

[6] Mohammed Aly Abdrabou, Ashraf Diaa Eldien Elbayoumy, and Essam Abd El-Wanis. LTE Authentication Protocol (EPS-AKA) weaknesses solution. In *2015 IEEE Seventh International Conference on Intelligent Computing and Information Systems (ICICIS)*, pages 434–441, December 2015.

[7] Hal Abelson, Ross Anderson, Steven Michael Bellovin, Josh Benaloh, Matt Blaze, Whitfield Diffie, John Gilmore, Peter G. Neumann, Ronald L. Rivest, Jeffrey I. Schiller, and Bruce Schneier. The Risks of Key Recovery, Key Escrow, and Trusted Third-Party Encryption. 1997.

[8] Carlisle Adams, Guenther Kramer, Serge Mister, and Robert Zuccherato. On The Security of Key Derivation Functions. In Kan Zhang and Yuliang Zheng, editors, *Information Security*, pages 134–145, Berlin, Heidelberg, 2004. Springer.

[9] Jeffrey G. Andrews, Stefano Buzzi, Wan Choi, Stephen V. Hanly, Angel Lozano, Anthony C. K. Soong, and Jianzhong Charlie Zhang. What Will 5G Be? *IEEE Journal on Selected Areas in Communications*, 32(6):1065–1082, June 2014.

[10] Jari Arkko, Pasi Eronen, Vesa Lehtovirta, and Vesa Torvinen. Improved Extensible Authentication Protocol Method for 3GPP Mobile Network Authentication and Key Agreement (EAP-AKA, October 2020.

[11] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P. C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron. The AVISPA Tool for the Automated Validation of Internet Security Protocols and Applications. In Kousha Etessami and Sriram K. Rajamani, editors, *Computer Aided Verification*, pages 281–285, Berlin, Heidelberg, 2005. Springer.

[12] Giuseppe Ateniese, Daniel H. Chou, Breno de Medeiros, and Gene Tsudik. Sanitizable Signatures. In Sabrina de Capitani di Vimercati, Paul Syverson, and Dieter Gollmann, editors, *Computer Security – ESORICS 2005*, Lecture Notes in Computer Science, pages 159–177, Berlin, Heidelberg, 2005. Springer.

[13] Matilda Backendal, Felix Günther, and Kenneth G. Paterson. Puncturable Key Wrapping and Its Applications. In Shweta Agrawal and Dongdai Lin, editors, *Advances in Cryptology – ASIACRYPT 2022*, Lecture Notes in Computer Science, pages 651–681, Cham, 2022. Springer Nature Switzerland.

[14] Gergei Bana and Hubert Comon-Lundh. Towards Unconditional Soundness: Computationally Complete Symbolic Attacker. In Pierpaolo Degano and Joshua D. Guttman, editors, *Principles of Security and Trust*, pages 189–208, Berlin, Heidelberg, 2012. Springer.

[15] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. A Formal Analysis of 5G Authentication. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, pages 1383–1396, New York, NY, USA, October 2018. Association for Computing Machinery.

[16] Mihir Bellare, Ran Canetti, and Hugo Krawczyk. Keying Hash Functions for Message Authentication. In *Proceedings of the 16th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '96, pages 1–15, Berlin, Heidelberg, August 1996. Springer-Verlag.

[17] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm, 2000. Report Number: 025.

[18] Mihir Bellare and Chanathip Namprempre. Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. *Journal of Cryptology*, 21(4):469–491, October 2008.

[19] Mihir Bellare and Phillip Rogaway. The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, Lecture Notes in Computer Science, pages 409–426, Berlin, Heidelberg, 2006. Springer.

[20] Mihir Bellare, Igors Stepanovs, and Brent Waters. New Negative Results on Differing-Inputs Obfuscation. In Marc Fischlin and Jean-Sébastien Coron, editors, *Advances in Cryptology – EUROCRYPT 2016*, pages 792–821, Berlin, Heidelberg, 2016. Springer.

[21] Josh Benaloh and Michael de Mare. One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In Tor Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93*, Lecture Notes in Computer Science, pages 274–285, Berlin, Heidelberg, 1994. Springer.

[22] Dan Boneh. The Decision Diffie-Hellman problem. In Joe P. Buhler, editor, *Algorithmic Number Theory*, Lecture Notes in Computer Science, pages 48–63, Berlin, Heidelberg, 1998. Springer.

[23] Dan Boneh and Victor Shoup. *A Graduate Course in Applied Cryptography*. 0.5 edition, January 2020.

[24] Ravishankar Borgaonkar, Lucca Hirschi, Shinjo Park, and Altaf Shaik. New Privacy Threat on 3G, 4G, and Upcoming 5G AKA Protocols. Technical Report 1175, 2018.

[25] Colin Boyd and Wenbo Mao. On a Limitation of BAN Logic. In Tor Helleseth, editor, *Advances in Cryptology — EUROCRYPT '93*, Lecture Notes in Computer Science, pages 240–247, Berlin, Heidelberg, 1994. Springer.

[26] An Braeken. Symmetric key based 5G AKA authentication protocol satisfying anonymity and unlinkability. *Computer Networks*, 181:107424, November 2020.

[27] Christina Brzuska, Marc Fischlin, Tobias Freudenreich, Anja Lehmann, Marcus Page, Jakob Schelbert, Dominique Schröder, and Florian Volk. Security of Sanitizable Signatures Revisited. In Stanisław Jarecki and Gene Tsudik, editors, *Public Key Cryptography – PKC 2009*, Lecture Notes in Computer Science, pages 317–336, Berlin, Heidelberg, 2009. Springer.

[28] Christina Brzuska, Marc Fischlin, Anja Lehmann, and Dominique Schröder. Unlinkability of Sanitizable Signatures. pages 444–461, May 2010.

[29] Christina Brzuska, Henrich C. Pöhls, and Kai Samelin. Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures. In Sokratis Katsikas and Isaac Agudo, editors, *Public Key Infrastructures, Services and Applications*, Lecture Notes in Computer Science, pages 12–30, Berlin, Heidelberg, 2014. Springer.

[30] Jan Camenisch and Anna Lysyanskaya. Dynamic Accumulators and Application to Efficient Revocation of Anonymous Credentials. In Moti Yung, editor, *Advances in Cryptology — CRYPTO 2002*, Lecture Notes in Computer Science, pages 61–76, Berlin, Heidelberg, 2002. Springer.

[31] Jin Cao, Maode Ma, Yulong Fu, Hui Li, and Yinghui Zhang. CPPHA: Capability-Based Privacy-Protection Handover Authentication Mechanism for SDN-Based 5G HetNets. *IEEE Transactions on Dependable and Secure Computing*, 18(3):1182–1195, May 2021. Conference Name: IEEE Transactions on Dependable and Secure Computing.

[32] K. Chalkias, F. Baldimtsi, D. Hristu-Varsakelis, and G. Stephanides. Two Types of Key-Compromise Impersonation Attacks against One-Pass Key Establishment Protocols. In Joaquim Filipe and Mohammad S. Obaidat, editors, *E-business and Telecommunications*, Communications in Computer and Information Science, pages 227–238, Berlin, Heidelberg, 2009. Springer.

[33] Cas Cremers and Martin Dehnel-Wild. Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion. San Diego, CA, USA, February 2019.

[34] CJF (Cas) Cremers. Scyther:semantics and verification of security protocols. Technische Universiteit Eindhoven, 2006.

[35] D. Dolev and A. Yao. On the security of public key protocols. *IEEE Transactions on Information Theory*, 29(2):198–208, March 1983. Conference Name: IEEE Transactions on Information Theory.

[36] Keita Emura, Shuichi Katsumata, and Yohei Watanabe. Identity-based encryption with security against the KGC: A formal model and its instantiations. *Theoretical Computer Science*, 900:97–119, January 2022.

[37] Chun-I Fan, Jheng-Jia Huang, Min-Zhe Zhong, Ruei-Hau Hsu, Wen-Tsuen Chen, and Jemin Lee. ReHand: Secure Region-Based Fast Handover With User Anonymity for Small Cell Networks in Mobile Communications. *IEEE Transactions on Information Forensics and Security*, 15:927–942, 2020.

[38] Nils Fleischhacker, Johannes Krupp, Giulio Malavolta, Jonas Schneider, Dominique Schröder, and Mark Simkin. Efficient Unlinkable Sanitizable Signatures from Signatures with Re-randomizable Keys. In *Proceedings, Part I, of the 19th IACR International Conference on Public-Key Cryptography — PKC 2016 - Volume 9614*, pages 301–330, Berlin, Heidelberg, March 2016. Springer-Verlag.

[39] Pierre-Alain Fouque, Cristina Onete, and Benjamin Richard. Achieving Better Privacy for the 3GPP AKA Protocol. Technical Report 480, Darmstadt, Germany, 2016.

[40] Prosanta Gope. LAAP: Lightweight anonymous authentication protocol for D2D-Aided fog computing paradigm. *Computers & Security*, 86:223–237, September 2019.

[41] Bo Han, Vijay Gopalakrishnan, Lusheng Ji, and Seungjoon Lee. Network function virtualization: Challenges and opportunities for innovations. *IEEE Communications Magazine*, 53(2):90–97, February 2015. Conference Name: IEEE Communications Magazine.

[42] Christoph Hanser, Simon Moritz, Farjola Zaloshnja, and Qin Zhang. Security in Mobile Telephony: The Security Levels in the Different Handy Generations. 2014.

[43] Arne Holst. Number of Internet of Things (IoT) connected devices worldwide from 2019 to 2030, by communications technology, October 2021.

[44] R Jeya and Dr B Amutha. WIRELESS GENERATIONS - A SURVEY. 115:10, 2017.

[45] Haibat Khan, Benjamin Dowling, and Keith M. Martin. Identity Confidentiality in 5G Mobile Telephony Systems. Technical Report 876, 2018.

[46] Rabia Khan, Pardeep Kumar, Dushantha Nalin K. Jayakody, and Madhusanka Liyanage. A Survey on Security and Privacy of 5G Technologies: Potential Solutions, Recent Advancements, and Future Directions. *IEEE Communications Surveys Tutorials*, 22(1):196–248, 2020. Conference Name: IEEE Communications Surveys Tutorials.

[47] Adrien Koutsos. The 5G-AKA Authentication Protocol Privacy. In *2019 IEEE European Symposium on Security and Privacy (EuroS&P)*, pages 464–479, June 2019.

[48] Russell W. F. Lai, Tao Zhang, Sherman S. M. Chow, and Dominique Schröder. Efficient Sanitizable Signatures Without Random Oracles. In Ioannis Askoxylakis, Sotiris Ioannidis, Sokratis Katsikas, and Catherine Meadows, editors, *Computer Security – ESORICS 2016*, pages 363–380, Cham, 2016. Springer International Publishing.

[49] Brian LaMacchia, Kristin Lauter, and Anton Mityagin. Stronger Security of Authenticated Key Exchange. In Willy Susilo, Joseph K. Liu, and Yi Mu, editors, *Provable Security*, Lecture Notes in Computer Science, pages 1–16, Berlin, Heidelberg, 2007. Springer.

[50] Jiangtao Li, Ninghui Li, and Rui Xue. Universal Accumulators with Efficient Nonmembership Proofs. In Jonathan Katz and Moti Yung, editors, *Applied Cryptography and Network Security*, Lecture Notes in Computer Science, pages 253–269, Berlin, Heidelberg, 2007. Springer.

[51] Madhusanka Liyanage. *A Comprehensive guide to 5G security.* John Wiley & Sons, IEEE Xplore, Hoboken, New Jersey, Piscataqay, New Jersey], 1st edition. edition, 2018.

[52] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The TAMARIN Prover for the Symbolic Analysis of Security Protocols. In *Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013, Proceedings*, volume 8044, pages 696–701. Springer, 2013.

[53] Sarmistha Mondal, Anindita Sinha, and Jayati Routh. A Survey on Evolution of Wireless Generations 0G to 7G. *International Journal of Advance Research in Science and Engineering (IJARSE*, 2(1), 2015.

[54] Kaisa Nyberg. Fast accumulated hashing. In Dieter Gollmann, editor, *Fast Software Encryption*, Lecture Notes in Computer Science, pages 83–87, Berlin, Heidelberg, 1996. Springer.

[55] Christopher Patton and Thomas Shrimpton. Security in the Presence of Key Reuse: Context-Separable Interfaces and their Applications, 2019. Publication info: A major revision of an IACR publication in CRYPTO 2019.

[56] Aleksi Peltonen, Ralf Sasse, and David Basin. A comprehensive formal analysis of 5G handover. In *Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks*, WiSec '21, pages 1–12, New York, NY, USA, June 2021. Association for Computing Machinery.

[57] Phillip Rogaway. Authenticated-encryption with associated-data. In *Proceedings of the 9th ACM conference on Computer and communications security*, CCS '02, pages 98–107, New York, NY, USA, November 2002. Association for Computing Machinery.

[58] Phillip Rogaway, Mihir Bellare, and John Black. OCB: A block-cipher mode of operation for efficient authenticated encryption. *ACM Trans. Inf. Syst. Secur.*, 6(3):365–403, August 2003.

[59] Phillip Rogaway and Thomas Shrimpton. A Provable-Security Treatment of the Key-Wrap Problem. In Serge Vaudenay, editor, *Advances in Cryptology - EUROCRYPT 2006*, pages 373–390, Berlin, Heidelberg, 2006. Springer.

[60] Peter Ryan, Steve A. Schneider, Michael Goldsmith, and Gavin Lowe. *The Modelling and Analysis of Security Protocols: The CSP Approach*. Addison-Wesley Professional, 2001. Google-Books-ID: HXt_S3HwKKAC.

[61] Amit Sahai and Brent Waters. How to use indistinguishability obfuscation: deniable encryption, and more. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, STOC '14, pages 475–484, New York, NY, USA, May 2014. Association for Computing Machinery.

[62] Ankush Singla, Rouzbeh Behnia, Syed Rafiul Hussain, Attila Yavuz, and Elisa Bertino. Look Before You Leap: Secure Connection Bootstrapping for 5G Networks to Defend Against Fake Base-Stations. In *Proceedings of the 2021 ACM Asia Conference on Computer and Communications Security*, pages 501–515, Virtual Event Hong Kong, May 2021. ACM.

[63] William Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall Press, USA, 6th edition, February 2013.

[64] Paul Syverson and Iliano Cervesato. *The Logic of Authentication Protocols*, volume 2171. May 2001.

[65] National Institute of Standards and Technology. Digital Signature Standard (DSS). Technical Report Federal Information Processing Standard (FIPS) 186-4 (Withdrawn), U.S. Department of Commerce, July 2013.

[66] Oracle Technology Network. Java Cryptography Architecture (JCA) Reference Guide, September 2022.

[67] Yuchen Wang, Zhenfeng Zhang, and Yongquan Xie. {Privacy-Preserving} and {Standard-Compatible} {AKA} Protocol for 5G. pages 3595–3612, 2021.

[68] Wenfeng Xia, Yonggang Wen, Chuan Heng Foh, Dusit Niyato, and Haiyong Xie. A Survey on Software-Defined Networking. *IEEE Communications Surveys Tutorials*, 17(1):27–51, 2015. Conference Name: IEEE Communications Surveys Tutorials.

[69] Xiaobei Yan and Maode Ma. A lightweight and secure handover authentication scheme for 5G network using neighbour base stations. *Journal of Network and Computer Applications*, 193:103204, November 2021.

[70] Hexuan Yu, Changlai Du, Yang Xiao, Angelos Keromytis, Chonggang Wang, Robert Gazda, Y. Hou, and Wenjing Lou. AAKA: An Anti-Tracking Cellular Authentication Scheme Leveraging Anonymous Credentials. January 2024.

[71] Yinghui Zhang, Robert H. Deng, Elisa Bertino, and Dong Zheng. Robust and Universal Seamless Handover Authentication in 5G HetNets. *IEEE Transactions on Dependable and Secure Computing*, 18(2):858–874, March 2021.

Appendices

Appendix A

Source Code

A.1 Sanitizable Signatures Implementation

This appendix presents a Java implementation of Sanitizable Signatures, a specialized form of digital signatures that allow partial modification of signed data without the assistance from the original signer. This implementation serves a dual purpose: it provides a concrete example of how sanitizable signatures can be realized, and more importantly, it offers a foundation for performance analysis. The code measures and reports average execution times, as shown in Figure A.1, for critical operations such as signature generation (both first-level and nested) and verification, providing valuable insights into the computational costs associated with sanitizable signatures. By examining this implementation and its output, it offers a practical understanding of the performance characteristics of sanitizable signature schemes employed in the thesis.

```
1  /**
2   * This is the SanSig code
3   */
4
5
6  import java.security.*;
7  import java.util.Arrays;
8
9  public class NestedSignatures2 {
10
11     private static final int ROUNDS = 100; // Number of rounds for
           averaging
12
13     public static KeyPair generateKeyPair() throws
           NoSuchAlgorithmException {
14         KeyPairGenerator keyGen = KeyPairGenerator.getInstance("RSA")
               ;
15         keyGen.initialize(2048);
16         return keyGen.generateKeyPair();
17     }
18
```

```
19      public static byte[] sign(PrivateKey privateKey, byte[] message)
            throws Exception {
20          Signature signature = Signature.getInstance("SHA256withRSA");
21          signature.initSign(privateKey);
22          signature.update(message);
23          return signature.sign();
24      }
25
26      public static boolean verify(PublicKey publicKey, byte[] message,
            byte[] signatureBytes) throws Exception {
27          Signature signature = Signature.getInstance("SHA256withRSA");
28          signature.initVerify(publicKey);
29          signature.update(message);
30          return signature.verify(signatureBytes);
31      }
32
33      public static long measureTime(Runnable operation) {
34          long startTime = System.nanoTime();
35          operation.run();
36          long endTime = System.nanoTime();
37          return endTime - startTime;
38      }
39
40      public static void main(String[] args) {
41          try {
42              KeyPair keyPair1 = generateKeyPair();
43              KeyPair keyPair2 = generateKeyPair();
44
45              // Create a 256-bit (32-byte) message
46              byte[] message256Bits = new byte[32];
47              Arrays.fill(message256Bits, (byte) 'A');
48
49              // Arrays to store timing results
50              long[] firstSignatureTimes = new long[ROUNDS];
51              long[] secondSignatureTimes = new long[ROUNDS];
52              long[] firstVerificationTimes = new long[ROUNDS];
53              long[] outerVerificationTimes = new long[ROUNDS];
54              long[] innerVerificationTimes = new long[ROUNDS];
55              long[] totalNestedVerificationTimes = new long[ROUNDS];
56
57              for (int i = 0; i < ROUNDS; i++) {
58                  // First signature
59                  SignatureResult signResult1 = signAndMeasure(keyPair1
                        , message256Bits);
60                  firstSignatureTimes[i] = signResult1.time;
61
62                  // Create nested message
63                  byte[] nestedMessage = new byte[message256Bits.length
                        + 1 + signResult1.signature.length];
64                  System.arraycopy(message256Bits, 0, nestedMessage, 0,
                        message256Bits.length);
```

```
65                  nestedMessage[message256Bits.length] = '|';
66                  System.arraycopy(signResult1.signature, 0,
                        nestedMessage, message256Bits.length + 1,
67                          signResult1.signature.length);
68
69                  // Second signature
70                  SignatureResult signResult2 = signAndMeasure(keyPair2
                        , nestedMessage);
71                  secondSignatureTimes[i] = signResult2.time;
72
73                  // Verify first signature
74                  VerificationResult verifyResult1 = verifyAndMeasure(
                        keyPair1, message256Bits, signResult1.signature);
75                  firstVerificationTimes[i] = verifyResult1.time;
76
77                  // Verify nested signatures
78                  NestedVerificationResult nestedVerifyResult =
                        verifyNestedSignatures(keyPair1, keyPair2,
                        message256Bits,
79                          nestedMessage, signResult2.signature);
80                  outerVerificationTimes[i] = nestedVerifyResult.
                        outerVerificationTime;
81                  innerVerificationTimes[i] = nestedVerifyResult.
                        innerVerificationTime;
82                  totalNestedVerificationTimes[i] = nestedVerifyResult.
                        totalTime;
83              }
84
85              // Calculate and print average times
86              printAverageTime("First signature", firstSignatureTimes);
87              printAverageTime("Second signature", secondSignatureTimes
                    );
88              printAverageTime("First verification",
                    firstVerificationTimes);
89              printAverageTime("Outer verification",
                    outerVerificationTimes);
90              printAverageTime("Inner verification",
                    innerVerificationTimes);
91              printAverageTime("Total nested verification",
                    totalNestedVerificationTimes);
92
93              // Print individual round times for comparison
94              System.out.println("\nIndividual round times (ms):");
95              System.out.println("Round\tFirst Sig\tSecond Sig\tFirst
                    Verify\tOuter Verify\tInner Verify\tTotal Nested");
96              for (int i = 0; i < ROUNDS; i++) {
97                  System.out.printf("%d\t%.3f\t\t%.3f\t\t%.3f\t\t%.3f\t
                        \t%.3f\t\t%.3f%n",
98                          i + 1,
99                          firstSignatureTimes[i] / 1_000_000.0,
100                         secondSignatureTimes[i] / 1_000_000.0,
```

```
                        firstVerificationTimes[i] / 1_000_000.0,
                        outerVerificationTimes[i] / 1_000_000.0,
                        innerVerificationTimes[i] / 1_000_000.0,
                        totalNestedVerificationTimes[i] / 1_000_000.0);
            }

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    private static SignatureResult signAndMeasure(KeyPair keyPair, byte[] message) throws Exception {
        long[] time = new long[1];
        byte[][] signature = new byte[1][];
        time[0] = measureTime(() -> {
            try {
                signature[0] = sign(keyPair.getPrivate(), message);
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
        return new SignatureResult(signature[0], time[0]);
    }

    private static VerificationResult verifyAndMeasure(KeyPair keyPair, byte[] message, byte[] signature)
            throws Exception {
        long[] time = new long[1];
        boolean[] isValid = new boolean[1];
        time[0] = measureTime(() -> {
            try {
                isValid[0] = verify(keyPair.getPublic(), message, signature);
            } catch (Exception e) {
                e.printStackTrace();
            }
        });
        return new VerificationResult(isValid[0], time[0]);
    }

    private static NestedVerificationResult verifyNestedSignatures(KeyPair keyPair1, KeyPair keyPair2,
            byte[] originalMessage, byte[] nestedMessage,
            byte[] outerSignature) throws Exception {
        long startTime = System.nanoTime();

        // Verify outer signature
        long outerStartTime = System.nanoTime();
        boolean outerValid = verify(keyPair2.getPublic(),
```

```
            nestedMessage, outerSignature);
147         long outerEndTime = System.nanoTime();
148         long outerVerificationTime = outerEndTime - outerStartTime;
149
150         boolean innerValid = false;
151         long innerVerificationTime = 0;
152
153         // If outer signature is valid, verify inner signature
154         if (outerValid) {
155             // Extract inner signature from nested message
156             byte[] innerSignature = Arrays.copyOfRange(nestedMessage,
                    originalMessage.length + 1, nestedMessage.length);
157             long innerStartTime = System.nanoTime();
158             innerValid = verify(keyPair1.getPublic(), originalMessage
                    , innerSignature);
159             long innerEndTime = System.nanoTime();
160             innerVerificationTime = innerEndTime - innerStartTime;
161         }
162
163         long endTime = System.nanoTime();
164         long totalTime = endTime - startTime;
165
166         return new NestedVerificationResult(outerValid, innerValid,
                outerVerificationTime, innerVerificationTime,
167                 totalTime);
168     }
169
170     private static void printAverageTime(String operation, long[]
            times) {
171         double avgTime = Arrays.stream(times).average().orElse(0) / 1
                _000_000.0;
172         System.out.printf("Average time for %s: %.3f ms%n", operation
                , avgTime);
173     }
174
175     private static class SignatureResult {
176         byte[] signature;
177         long time;
178
179         SignatureResult(byte[] signature, long time) {
180             this.signature = signature;
181             this.time = time;
182         }
183     }
184
185     private static class VerificationResult {
186         boolean isValid;
187         long time;
188
189         VerificationResult(boolean isValid, long time) {
190             this.isValid = isValid;
```

```
191             this.time = time;
192         }
193     }
194
195     private static class NestedVerificationResult {
196         boolean outerValid;
197         boolean innerValid;
198         long outerVerificationTime;
199         long innerVerificationTime;
200         long totalTime;
201
202         NestedVerificationResult(boolean outerValid, boolean
                innerValid, long outerTime, long innerTime,
203                     long totalTime) {
204             this.outerValid = outerValid;
205             this.innerValid = innerValid;
206             this.outerVerificationTime = outerTime;
207             this.innerVerificationTime = innerTime;
208             this.totalTime = totalTime;
209         }
210     }
211 }
```

A.1: SanSig

Output:

```
Average time for First signature: 0.725 ms
Average time for Second signature: 0.705 ms
Average time for First verification: 0.049 ms
Average time for Outer verification: 0.041 ms
Average time for Inner verification: 0.043 ms
Average time for Total nested verification: 0.084 ms

Individual round times (ms):
Round   First Sig       Second Sig      First Verify    Outer Verify    Inner Verify    Total Nested
1       4.950           2.598           0.119           0.092           0.058           0.152
2       0.889           0.807           0.095           0.072           0.053           0.125
3       0.768           0.785           0.062           0.069           0.072           0.141
4       0.763           0.766           0.064           0.056           0.052           0.109
5       0.761           0.790           0.076           0.057           0.059           0.117
6       0.787           0.956           0.142           0.093           0.053           0.147
7       0.769           0.763           0.055           0.054           0.051           0.105
8       0.749           0.744           0.053           0.063           0.052           0.115
9       0.753           0.764           0.059           0.074           0.062           0.136
10      0.785           0.762           0.054           0.056           0.051           0.107
11      0.754           0.757           0.053           0.054           0.058           0.113
12      0.774           0.756           0.068           0.059           0.055           0.114
13      0.785           0.767           0.059           0.060           0.051           0.111
14      0.766           0.776           0.057           0.065           0.056           0.121
15      0.780           0.748           0.100           0.054           0.049           0.103
16      0.680           0.666           0.050           0.056           0.048           0.105
17      0.687           0.701           0.057           0.056           0.049           0.105
18      0.688           0.690           0.049           0.049           0.044           0.094
19      0.681           0.667           0.072           0.049           0.045           0.094
20      0.668           0.670           0.047           0.056           0.049           0.105
21      0.679           0.703           0.054           0.049           0.052           0.108
22      0.686           0.680           0.046           0.048           0.051           0.100
23      0.690           0.681           0.047           0.047           0.043           0.091
24      0.681           0.681           0.052           0.071           0.049           0.120
```

Figure A.1: *SanSig output*

A.2 Universal Accumulator Implementation

This appendix presents a Java implementation of a universal accumulator, a cryptographic scheme that enables efficient membership and non-membership proofs without revealing the entire set of elements. This implementation serves a dual purpose: it provides a simplified, concrete example of how universal accumulators can be realised, and more importantly, it offers a foundation for performance analysis critical to this thesis. The code measures and reports average execution times, as shown in Figure A.2, for key operations such as element addition, deletion, and witness generation, providing valuable insights into the computational efficiency of universal accumulators. By examining this simplified implementation and its output, it offers a practical understanding of the performance characteristics of universal accumulator schemes, particularly in the context of managing revocation lists.

```
1  /**
2   * This is the Accumulator code
3   */
4
5  import java.math.BigInteger;
6  import java.security.MessageDigest;
7  import java.security.NoSuchAlgorithmException;
8  import java.security.SecureRandom;
9  import java.util.HashSet;
10 import java.util.Set;
11 import java.util.ArrayList;
12 import java.util.List;
13 import java.util.function.Supplier;
14
15 public class DynamicUniversalAccumulator2 {
16     private BigInteger N; // RSA modulus
17     private BigInteger g; // generator
18     private BigInteger acc; // current accumulator value
19     private Set<BigInteger> elements;
20     private BigInteger product; // product of all elements
21     private BigInteger phi; // Euler's totient function value for N
22
23     public DynamicUniversalAccumulator2(int bitLength) {
24         SecureRandom random = new SecureRandom();
25         BigInteger p, q;
26         do {
27             p = BigInteger.probablePrime(bitLength / 2, random);
28             q = BigInteger.probablePrime(bitLength / 2, random);
29             N = p.multiply(q);
30             phi = p.subtract(BigInteger.ONE).multiply(q.subtract(
                   BigInteger.ONE));
31         } while (N.bitLength() != bitLength);
32         g = new BigInteger("2"); // Simple choice for generator
33         acc = g;
34         elements = new HashSet<>();
35         product = BigInteger.ONE;
36     }
```

```
37
38      private BigInteger generateCoprimeElement() {
39          SecureRandom random = new SecureRandom();
40          BigInteger element;
41          do {
42              element = new BigInteger(N.bitLength(), random);
43          } while (element.compareTo(BigInteger.ONE) <= 0 || element.
                compareTo(N) >= 0
44                  || !element.gcd(phi).equals(BigInteger.ONE));
45          return element;
46      }
47
48      public void add(BigInteger element) {
49          if (!element.gcd(phi).equals(BigInteger.ONE)) {
50              throw new IllegalArgumentException("Element must be
                    coprime with \phi(N)");
51          }
52          acc = acc.modPow(element, N);
53          elements.add(element);
54          product = product.multiply(element).mod(phi);
55      }
56
57      public boolean delete(BigInteger element) {
58          if (!elements.contains(element)) {
59              return false;
60          }
61          BigInteger inverse = element.modInverse(phi);
62          acc = acc.modPow(inverse, N);
63          elements.remove(element);
64          product = product.multiply(inverse).mod(phi);
65          return true;
66      }
67
68      public byte[] membershipWitness(BigInteger element) {
69          if (!elements.contains(element)) {
70              throw new IllegalArgumentException("Element is not in the
                    accumulator");
71          }
72          BigInteger witness = g.modPow(product.divide(element), N);
73          return hashBigInteger(witness);
74      }
75
76      public byte[] optimizedNonMembershipWitness(BigInteger element) {
77          if (elements.contains(element)) {
78              throw new IllegalArgumentException("Element is already in
                    the accumulator");
79          }
80          BigInteger d = product.gcd(element);
81          BigInteger a = product.divide(d);
82          BigInteger b = element.divide(d);
83          BigInteger x = a.modInverse(b);
```

```
84          BigInteger y = acc.modPow(x, N);
85          return hashBigInteger(y);
86      }
87
88      public boolean verifyMembership(BigInteger element, byte[]
            witnessHash) {
89          BigInteger computedWitness = g.modPow(product.divide(element)
                , N);
90          return MessageDigest.isEqual(witnessHash, hashBigInteger(
                computedWitness));
91      }
92
93      public boolean verifyOptimizedNonMembership(BigInteger element,
            byte[] witnessHash) {
94          BigInteger d = product.gcd(element);
95          BigInteger b = element.divide(d);
96          BigInteger computedWitness = acc.modPow(b, N);
97          return MessageDigest.isEqual(witnessHash, hashBigInteger(
                computedWitness));
98      }
99
100     private byte[] hashBigInteger(BigInteger value) {
101         try {
102             MessageDigest md = MessageDigest.getInstance("SHA-256");
103             return md.digest(value.toByteArray());
104         } catch (NoSuchAlgorithmException e) {
105             throw new RuntimeException("SHA-256 not available", e);
106         }
107     }
108
109     public static void main(String[] args) {
110         int numTrials = 100; // Number of trials for averaging
111         DynamicUniversalAccumulator2 acc = new
                DynamicUniversalAccumulator2(2048);
112
113         List<BigInteger> elements = new ArrayList<>();
114         for (int i = 0; i < 10; i++) {
115             elements.add(acc.generateCoprimeElement());
116         }
117         BigInteger nonMember = acc.generateCoprimeElement();
118
119         // Add elements to the accumulator
120         for (BigInteger elem : elements) {
121             acc.add(elem);
122         }
123
124         // Measure average times
125         double avgAddTime = measureAverageTime(numTrials, () -> {
126             acc.add(acc.generateCoprimeElement());
127             return null;
128         });
```

```
129         double avgDeleteTime = measureAverageTime(numTrials, () ->
                acc.delete(elements.get(0)));
130         double avgMembershipWitnessTime = measureAverageTime(
                numTrials, () -> acc.membershipWitness(elements.get(1)));
131         double avgNonMembershipWitnessTime = measureAverageTime(
                numTrials,
132                 () -> acc.optimizedNonMembershipWitness(nonMember));
133         double avgVerifyMembershipTime = measureAverageTime(numTrials
                , () -> {
134             byte[] witness = acc.membershipWitness(elements.get(1));
135             return acc.verifyMembership(elements.get(1), witness);
136         });
137         double avgVerifyNonMembershipTime = measureAverageTime(
                numTrials, () -> {
138             byte[] witness = acc.optimizedNonMembershipWitness(
                    nonMember);
139             return acc.verifyOptimizedNonMembership(nonMember,
                    witness);
140         });
141
142         // Print results
143         System.out.printf("Average add time: %.2f ms%n", avgAddTime);
144         System.out.printf("Average delete time: %.2f ms%n",
                avgDeleteTime);
145         System.out.printf("Average membership witness generation time
                : %.2f ms%n", avgMembershipWitnessTime);
146         System.out.printf("Average non-membership witness generation
                time: %.2f ms%n", avgNonMembershipWitnessTime);
147         System.out.printf("Average membership verification time: %.2f
                 ms%n", avgVerifyMembershipTime);
148         System.out.printf("Average non-membership verification time:
                %.2f ms%n", avgVerifyNonMembershipTime);
149
150         // Measure witness sizes
151         byte[] memberWitness = acc.membershipWitness(elements.get(1))
                ;
152         byte[] nonMemberWitness = acc.optimizedNonMembershipWitness(
                nonMember);
153         System.out.println("Membership witness size: " +
                memberWitness.length + " bytes");
154         System.out.println("Optimized non-membership witness size: "
                + nonMemberWitness.length + " bytes");
155     }
156
157     private static <T> double measureAverageTime(int numTrials,
            Supplier<T> operation) {
158         long totalTime = 0;
159         for (int i = 0; i < numTrials; i++) {
160             long startTime = System.nanoTime();
161             operation.get();
162             long endTime = System.nanoTime();
```

```
163                 totalTime += (endTime - startTime);
164             }
165             return (totalTime / (double) numTrials) / 1e6; // Convert to
                    milliseconds
166         }
167     }
```

A.2: Dynamic Accumulator

Output:

```
PS C:\Users\Rabiah>  & 'C:\Program Files\Eclipse Adoptium\jd
tailsInExceptionMessages' '-cp' 'C:\Users\Rabiah\AppData\Loc
Average add time: 2.24 ms
Average delete time: 0.04 ms
Average membership witness generation time: 0.03 ms
Average non-membership witness generation time: 2.29 ms
Average membership verification time: 0.01 ms
Average non-membership verification time: 4.10 ms
Membership witness size: 32 bytes
Optimized non-membership witness size: 32 bytes
```

Figure A.2: *Accumulator output*