This paper has been accepted by IEEE Computer Security Foundations Symposium (CSF) 2024

UniHand: Privacy-preserving Universal Handover for Small-Cell Networks in 5G-enabled Mobile Communication with KCI Resilience

Rabiah Alnashwan[1,2], Prosanta Gope[1], and Benjamin Dowling[1]

[1]Department of Computer Science, The University of Sheffield, Sheffield, UK
{*ralnashwan1,p.gope, b.dowling,* }*@sheffield.ac.uk*
[2]Department of Computer Science, Imam Mohammad Ibn Saud Islamic University, Riyadh, Saudi Arabia
ralnashwan@imamu.edu.sa

***Abstract*—Introducing Small Cell Networks (SCN) has significantly improved wireless link quality, spectrum efficiency and network capacity, which has been viewed as one of the key technologies in the fifth-generation (5G) mobile network. However, this technology increases the frequency of handover (HO) procedures caused by the dense deployment of cells in the network with reduced cell coverage, bringing new security and privacy issues. The current 5G-AKA and HO protocols are vulnerable to security weaknesses, such as the lack of forward secrecy and identity confusion attacks. The high HO frequency of HOs might magnify these security and privacy concerns in the 5G mobile network. This work addresses these issues by proposing a secure privacy-preserving universal HO scheme (UniHand) for SCNs in 5G mobile communication. UniHand can achieve mutual authentication, strong anonymity, perfect forward secrecy, key-escrow-free and key compromise impersonation (KCI) resilience. To the best of our knowledge, this is the *first* scheme to achieve secure, privacy-preserving universal HO with *KCI* resilience for roaming users in 5G environment. We demonstrate that our proposed scheme is resilient against all the essential security threats by performing a comprehensive formal security analysis and conducting relevant experiments to show the cost-effectiveness of the proposed scheme.**

***Index Terms*—Universal Handover, 5G, SCN, Authentication, Privacy, KCI Resilience.**

I. INTRODUCTION

The generations of cellular network standards have evolved continuously, and each generation offers new improvements over its successor to cope with the market needs. The newly deployed cellular network (the fifth generation) provides its users with a better realisation of continuous access to networks worldwide. Increased network capacity guarantees faster speed, higher broadband, and lower latency. One promising way to achieve these network improvements is through increasing network density, i.e. increasing the number of base stations (cells) in the network, known as a Small-Cell Network (SCN). However, SCNs reduce the serving spectrum of each cell, causing the handover process to be triggered more frequently than in the previous mobile cellular generations (like GSM and 3G), aggravating handover privacy and security issues in 5G. For example, [4], [8], [9], [17] have analysed the 5G-AKA and HO protocols and identified security, privacy and efficiency weaknesses, such as identity confusion attacks, confidentiality attacks on sequence numbers (breaking untraceability) and confusion attacks. It follows that current solutions for 5G-AKA and HO security are insufficient. To overcome most of these weaknesses in 5G-AKA and HO, the following security and privacy requirements must be adequately addressed: mutual authentication (MA), Strong anonymity (SA) (user anonymity and unlinkability) and perfect forward secrecy (PFS). As important as these security features, key-escrow freeness (KEF) and Key Compromise Impersonation (KCI) resilience should also be supported in 5G networks. Both security features are essential to improve the security of the 5G network and avoid catastrophic security degradation in case of compromising a single private key in the network. KEF ensures that secret keys are jointly computed between users and communicating partners and are not fully controlled by any third party. KCI vulnerabilities allow impersonation attacks that trigger if the private key of a network participant/entity (such as UE, gNB or AuC) is revealed, which enables the adversary to impersonate others to the compromised participant [16]. The consequence of this attack will be catastrophic, especially if the adversary compromises a base station/gNB in a 5G network, which enables the adversary to impersonate other users to the compromised gNB, or perform a MITM attack with all connected users in that cell. This consequence may be aggravated by the significant increase of small cells in the 5G environment, as it linearly increases the probability of compromising gNB. Therefore, providing resilience against KCI attacks in 5G networks is essential. This attack represents a subtle, often underestimated threat and is difficult to counter. Unfortunately, existing and conventional 5G AKA and HO protocols cannot deal with this type of attack. Therefore, in this article, we introduce UniHand scheme, an Authentication and Key Agreement (AKA) protocol and handover protocol that achieves all the desirable security requirements (such as PFS, SA, MA); it also supports KEF and provides resistance against KCI attacks. To realise these feature UniHand utilises sanitisable signatures [1] and universal accumulators [26]. Sanitisable signatures provide UniHand with a modifiable signature, where two parties (i.e. signer and sanitiser) can generate a valid signature of a certificate. Hence, the certificate initiator issues certificates with their signatures for network users. Users can modify a *part* of their certificate and generate a valid signature of the modified certificate. By utilising SanSig, UniHand preserves the following signature properties: immutability, privacy, accountability, transparency and unforgeability [1], [10]. On the other hand, allowing device revocation within UniHand is required to manage the legitimate users in the network. UniHand uses dynamic accumulators to provide efficient membership revocation while maintaining user privacy. However, the accumulator's efficiency depends on the frequency of updating the accumulator, which is itself correlated to the number of joining users. Usually, in a typical setting of mobile networks, the number of joining users exceeds the number of revoked users. Therefore, non-membership witnesses are significantly more efficient, as updating the revoked membership occurs less frequently than updating the membership accumulator. Thus, we utilise an accumulator that supports *dynamic* and *non-membership* witnesses, proposed by [26], to achieve privacy and increase the efficiency of UniHand.

To the best of our knowledge, this is the *first work* that provides KCI resilience in the authenticated key agreement and handover protocols in 5G settings.

A. Related Works

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. Peltonen, Sasse and Basin [29] provide a comprehensive formal security analysis for the conventional 5G handover scheme. Basin et al. [4], and Cremers and Dehnel-Wild [17] also provide a comprehensive formal security analysis for the standard 5G AKA protocol. These studies highlighted under-specified security requirements and identified security and privacy weaknesses in the current version of 5G-AKA and HO protocols [21], such as traceability attacks from active adversaries, identity confusion attacks, lack of perfect forward secrecy and confidentiality attacks on sequence numbers. Therefore, several related authentication schemes such as ReHand [20], RUSH [32], LSHA [31] CPPHA [15] and Braeken [9] have been proposed in the literature to improve the security and privacy of AKA and HO protocols. These schemes can be divided into two categories based on the underlying cryptography: symmetric-based and asymmetric-based schemes. Braeken [9] proposed an efficient symmetric-based AKA protocol that improves the 5G-AKA and overcomes identity replay attacks discovered in 5G-AKA. Nevertheless, the proposed protocol suffers from an unlinkability attack, which only supports in-session unlinkability in case of successful authentication. However, if authentication fails, user identities can be linked due to the reused Globally Unique Temporary User Equipment Identity (GUTI) [21]. Additionally, Fan et al. [20], proposed ReHand, a symmetric-based protocol that supports a secure region-based handover scheme and fast revocation management for SCNs. This protocol provides a seamless handover for roaming inside a region only, where users roaming to a different region cannot perform the fast HO; instead, they must execute another initial authentication. Yan and Ma [31] propose LSHA, a symmetric-based handover and authentication protocol exploiting neighbouring base stations in the 5G network. This protocol depends on two keys to secure HOs: a gNB secret key and a session key between gNBs. However, LSHA relies on 5G-AKA, which is susceptible to several attacks, undermining the security of LSHA.

Some proposed schemes rely on asymmetric cryptography, such as RUSH [32] and CPPHA [15]. Zhang et al. [32] propose RUSH, a universal handover authentication protocol that preserves user anonymity, achieving perfect forward secrecy and key-escrow freeness. The RUSH protocol provides universal handover in 5G HetNets through blockchains and leverages chameleon hashes (CH) to achieve user anonymity. However, their analysis omits the use of blockchain, and thus security and performance issues due to blockchain use may have been overlooked. Additionally, RUSH does not address CH's linkability issue. Cao et al. [15] propose CPPHA, an asymmetric-based scheme which utilises software-defined networks (SDN) in 5G. CPPHA relies on an authentication handover module (AHM) that resides in the SDN controller, which is responsible for monitoring and predicting users' future paths. The AHM pre-distributes (before the actual HO) user-related information to the predicted gNBs, including the user's ID and first session key. Thus, it only supports weak anonymity against eavesdroppers, where all prospective gNBs maintain user IDs. Finally, CPPHA is susceptible to sequence number desynchronisation attacks.

B. Motivation and Contributions

Authenticated key agreement is the initial and most crucial step for validating joining users before providing any services to them. Similarly, the handover procedure re-authenticates roaming users and ensures the continuity of network services. So guaranteeing a high level of security and privacy of AKA and HO is vital for a 5G network. Nevertheless, the existing 5G-AKA and HO protocols are susceptible to security and privacy issues, yet several existing works have attempted to address some of these issues. However, to the best of our knowledge, the existing works fail to address *all* security and privacy issues in the 5G AKA and HO protocols, such as traceability attacks, identity confusion attacks, lack of PFS and confidentiality attacks on sequence numbers. Most previous solutions use the existing 5G-AKA to build their HO protocol, inheriting all 5G-AKA weaknesses. In addition, no existing protocols have provably achieved KCI resilience and KEF. This creates a critical security vulnerability in the 5G network, especially when considering an active attacker who can impersonate an honest party to the compromised party. Both properties are essential to achieve higher security and fairness in key generation and reduce the probability of protocol failure due to a single key compromise. Therefore, we propose UniHand, a

Universal Handover scheme achieving seamless user mobility and all required security and privacy properties (explained in Section III-C), including KCI resilience and KEF. UniHand is the first to achieve secure, privacy-preserving universal HO with KCI resilience for roaming users in 5G without relying on additional infrastructural support (such as blockchain). Our contributions are as follows:

- The *first* standalone solution to achieve KCI resilience for roaming users in 5G SCNs, providing privacy-preserving and secure authentication and universal HO protocols;
- An effective user membership revocation scheme for efficiently managing a large number of users in 5G using dynamic universal accumulators [26];
- A rigorous formal security analysis of our proposed scheme;
- A comparative analysis of UniHand with previously existing literature, demonstrating UniHand achieves all required security properties.

The rest of this paper is organised as follows: Section II introduces sanitisable signatures and accumulators. Section III introduces the system architecture, adversary model and targeted security properties. Next, we present the secure Universal Handover scheme UniHand, with a detailed description of all phases, including registration, initial authentication and universal HO. Section VI conducts a formal security analysis of UniHand. Section VII compares the achieved security properties of UniHand with previous literature and evaluates UniHand's computational and communication cost, concluding with Section VIII.

II. Preliminaries

Here we introduce sanitisable signatures and accumulators, two cryptographic primitives underlying UniHand. While both primitives (sanitisable signatures and accumulators) have more complex functionality and properties compared to conventional cryptographic primitives, they can be constructed using conventional cryptographic primitives.

A. Sanitisable signatures

Sanitisable signature schemes (SanSig), introduced by Ateniese et al. [1] provide similar features to standard digital signatures while allowing the signing authority to delegate signing to another party, the so-called sanitiser. The sanitiser can modify specific blocks of the signed message and generate another valid signature over the modified (sanitised) message. The sanitiser can sanitise a signature without the original signer's participation, providing signature flexibility between the signer and sanitiser. In particular, let a message m consisting of n blocks $m = (m_1, ..., m_n)$, where $n \in \mathbb{Z}$ and $m_i \in 0, 1^*$. The signer allows the sanitiser to modify specific j blocks of the message, i.e., $(m_j, ...m_{j+i})$ that the signer defined as admissible blocks of the message. These admissible blocks are the only blocks that a sanitiser can modify in the message. In order to achieve that, the signer has to specify sections of a signed message that the sanitiser can modify using deterministic functions MOD and ADM. The signer uses ADM to specify the modifiable blocks of a message, and the sanitiser uses MOD to specify the modified blocks of the messages, i.e., $m^* = MOD(m)$ and $ADM(m^*, m) \rightarrow \{0, 1\}$. We require SanSig to provide existential unforgeability under chosen message attack (EUFCMA) security. SanSig can be constructed using chameleon hashes associated with a standard digital signature as in [1], a digital signature and a group signature [12], or even from two standard digital signatures [11], [13], [14]. In general, SanSig consists of a tuple of algorithms SanSig = {KGen, Sign, Sanit, Verify, Proof, Judge}. We omit Proof and Judge since they are not required in UniHand.

- **KGen**$(\mathbf{1^n}) \rightarrow (\boldsymbol{pk}, \boldsymbol{sk})$ is a pair of key generation algorithms for the signer and the sanitiser respectively: $(pk_{sig}, sk_{sig}) \leftarrow_\$ \mathsf{KGen}_{sig}(1^n)$, $(pk_{san}, sk_{san}) \leftarrow_\$ \mathsf{KGen}_{san}(1^n)$.
- **Sign**$(\boldsymbol{m}, \boldsymbol{sk_{sig}}, \boldsymbol{pk_{san}}, \boldsymbol{ADM}) \rightarrow_\$ (\boldsymbol{\sigma})$: Sign algorithm takes as input four parameters: a message $m \in \{0, 1\}^*$, a signer private key sk_{sig}, sanitiser public key pk_{san} and the admissible modifiable message blocks (ADM).
- **Sanit**$(\boldsymbol{m}, \boldsymbol{MOD}, \boldsymbol{\sigma}, \boldsymbol{pk_{sig}}, \boldsymbol{sk_{san}}) \rightarrow_\$ (\boldsymbol{m^*}, \boldsymbol{\sigma^*})$: Sanit algorithm takes as input five parameters: original message m, a modification of the original message MOD, a signature σ, signer public key pk_{sig} and sanitiser private key sk_{san}.
- **Verify**$(\boldsymbol{m}, \boldsymbol{\sigma}, \boldsymbol{pk_{sig}}, \boldsymbol{pk_{san}}) \rightarrow_\$ \boldsymbol{b}$: Verify algorithm takes as input five parameters: a message m, a signature σ and the public keys of the signer pk_{sig} and sanitiser pk_{san}. Next it 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. Accumulators

Cryptographic accumulators was first introduced by Benaloh and de Mare [7] in 1994. The proposed one-way accumulator was constructed using a hash function with quasi-commutativeness and one-way property. The generic concept of accumulators is to accumulate a number of elements from a finite set $X = x_1, ...x_n$ into one accumulated value acc of constant size. Since the accumulator depends on a quasi-commutativeness property, the order of the accumulated elements is not important. Cryptographic accumulators can be categorised based on the construction (symmetric and asymmetric), characteristics (i.e. static and dynamic) and security assumptions (i.e. RSA, Strong Diffie-Hellman and Bilinear Diffie-Hellman). Hence, accumulators can be constructed distinctly depending on the application and the required security features. In UniHand, a universal accumulator that supports negative and positive membership witnesses can be built from RSA and relying on the strong RSA assumption [3], [26]. In general, accumulator acc consists of a tuple of algorithms acc = {$\mathsf{KGen}_{\ddot{C}}$, $\mathsf{Gen}_{\ddot{C}}$, $\mathsf{Update}_{\ddot{C}}$, NonWitCreate, $\mathsf{Verify}_{\ddot{C}}$, NonWitUpdate}.

- $\mathbf{KGen}_{\ddot{C}}(\mathbf{1^n}) \rightarrow_\$ (\boldsymbol{sk_{\ddot{C}}})$: $\mathsf{KGen}_{\ddot{C}}$ generates a secret key $sk_{\ddot{C}}$.
- $\mathbf{Gen}_{\ddot{C}}(\boldsymbol{sk_{\ddot{C}}}, \boldsymbol{X}) \rightarrow_\$ (\boldsymbol{\ddot{C}})$: $\mathsf{Gen}_{\ddot{C}}$ algorithm takes as input an accumulator secret key $sk_{\ddot{C}}$, and all values to be

accumulated $X = \{x_i, \ldots x_n\}$ (where $X \leftarrow \phi$ when initialised), returning an accumulator $\ddot{C}$.

- **Update$_{\ddot{C}}$($sk_{\ddot{C}}, \ddot{C}, x^*$) $\rightarrow_\$$ ($\ddot{C}^*$)**: Update$_{\ddot{C}}$ takes as input an accumulator secret key $sk_{\ddot{C}}$, an accumulator $\ddot{C}$ and all new values to be accumulated x^*, returning the updated accumulator $\ddot{C}^*$.
- **NonWitCreate($sk_{\ddot{C}}, \ddot{C}, X, x^*$) $\rightarrow_\$$ (ω_x)**: NonWitCreate takes as input an accumulator secret key $sk_{\ddot{C}}$, an accumulator $\ddot{C}$, previously accumulated values X and value x^* (where $x^* \notin X$) returning a non-membership witness ω_x for x^*.
- **Verify$_{\ddot{C}}$($\ddot{C}, \omega_x, x$) $\rightarrow_\$$ b**: Verify$_{\ddot{C}}$ takes as input an accumulator $\ddot{C}$, a non-membership witness ω_x and queried value x, returning a bit $b \in \{0, 1\}$, where $b = 1$ if $x \notin X$, and $b = 0$ otherwise.
- **NonWitUpdate($\ddot{C}, \ddot{C}^*, x^*, x, \omega_x$) $\rightarrow_\$$ (c_x^*)**: NonWitUpdate takes as input an accumulator $\ddot{C}$, an updated accumulator $\ddot{C}^*$, a (new) accumulated value x^*, an non-accumulated value x and the non-membership witness ω_x, returning a new non-membership witness ω_x^*.

III. System Architecture, Design Goals and Adversary Model

In this section, we first describe the UniHand system architecture, then describe the adversary model and design security and privacy goals.

A. System Architecture

This section introduces the System Architecture of our proposed UniHand scheme. There are three major components of the system architecture: Authentication Center (AuC), 5G radio base station (Next Generation NodeB gNB) and User Equipment (UE), as shown in Figure 1. AuC encapsulates the 5G core network entities, i.e. access and mobility management function, user plane function, session management function, and the authentication server function. AuC is responsible for configuring all parties in the network, including the gNB and UE and generating certificates for authentication purposes. Meanwhile, gNB connects UE to the AuC. UniHand consists of two main protocols: An initial authentication and a universal handover. During the initial authentication, the AuC authenticates UEs, generates certificates for UEs for future handovers, and generates session keys for future communication. In the universal handover, the gNB authenticates the UEs by validating their certificates and generating session keys.

B. Adversary Model

Our security model is intended to capture security notions recommended by the 3GPP group [21] and [4], [17], [29] for 5G authentication and handover protocols, described in Section III-C. 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.

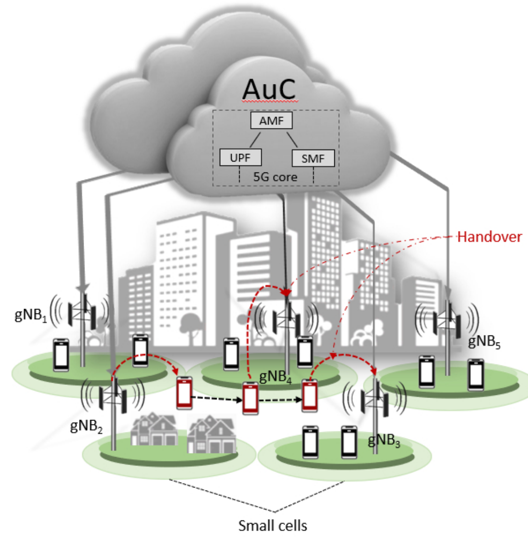

Fig. 1: System Architecture

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 AuC.
2) Asymmetric secret keys (ASK), signing and sanitising keys of the protocol participants, i.e. UE, gNB and AuC
3) Session keys established between protocol participants.

An adversary may also try to launch several other attacks on UniHand, including impersonation, replay, man-in-the-middle attacks, etc. However, the proposed UniHand scheme will be able to resist these attacks through mutual authentication, user anonymity, unlinkability, PFS, and KCI resilience.

C. Design Goals

The newly deployed 5G mobile network supports increasingly dense connections while providing optimised network efficiency. However, several requirements must be fulfilled to ensure security and privacy in 5G mobile communication. In this regard, [4] and [29] have reviewed the 5G-AKA and handover protocols and identified the following security: privacy requirements for authentication and handover protocols that need to be adequately addressed. Our security analysis (in Section VI) and the discussion (in Section VII) ensure that UniHand can successfully achieve all security goals.

1) ***Mutual authentication:*** Ensuring the authenticity of every party in the network is essential to reduce the risk of many security threats, such as MITM and impersonation attacks. More details about the MA experiment can be found in [V-C].
2) ***Strong anonymity:*** Maintaining users' privacy is one of the main goals to be achieved in the 5G network. Most authentication and handover protocols address privacy by anonymising user identities (using pseudo-identities). However, providing user anonymity alone is insufficient,

as the adversary may break privacy by linking users' protocol executions. In this regard, we introduce strong anonymity properties, capturing user anonymity and unlinkability. Additional details about this security feature can be found in V-E.

3) ***Perfect forward secrecy:*** PFS is essential in 5G to ensure the security of the previously-computed session keys after long-term secrets are compromised. This security feature is especially essential in 5G networks due to the high frequency of handovers caused by the increased deployment of cells, which generates many session keys with each handover. Thus, 5G protocols should ensure that any compromise does not affect the security of past session keys.
4) ***Key-escrow free:*** There has been a lot of debate among researchers about the key escrow problem of using a trustworthy escrow to maintain a copy of all users' secret keys. Even though providing escrow is essential in some domains, it brings issues regarding the trustworthiness of a specific third party controlling the escrow. In today's information security, it isn't easy to fully trust one entity, as it might take advantage of this trust and reveal some of the encrypted information in the network. Another case is if this entity is compromised, it will cause a single point of failure to the entire network. Moreover, the idea of key escrow contradicts the end-to-end encryption and the strong security requirement of the 5G network. Therefore it is not recommended to give the authority to a single party to generate all secret keys or maintain any users' keys in the network. Nevertheless, providing a key escrow-free property provide higher security, and it is essential to protect users in case of a compromised third party.
5) ***Key compromise impersonation resilience:*** Although providing a PFS protects previous sessions' information, it doesn't prevent an attacker from leaking new information by masquerading as a legitimate party to the compromised one, forming a KCI attack. Therefore providing a KCI resilience scheme is essential in 5G, primarily to ensure the security and authenticity of communicating partners after a long-term secret is compromised.
6) ***Effective revocation management:*** It is essential to provide effective revocation management to accommodate the extended capacity of connections in the 5G network. Therefore, our proposed protocols use a revocation list (RL) to manage the revoked users from the network. In general, the number of joined users supersedes the revoked ones. For this reason, our RL utilises a dynamic accumulator that supports non-membership witnesses for better efficiency.

IV. The Proposed Uni-Hand Scheme

In this section, we introduce UniHand, which consists of a *System Initialisation phase*, an *initial authentication* protocol and *universal HO* protocol. The first phase in UniHand is *System Initialisation*, responsible for registering gNBs and new users into the network, generating the initialisation parameters for UE and gNB. During this phase, the AuC generates certificates for all gNBs in the network. Additionally, AuC generates a long-term secret key and shares it with the user along with pseudo-identities (pid and T_{ID}) to preserve users' anonymity. Next, AuC creates an accumulator to manage user revocation. In the *Initial Authentication* protocol, the AuC generates certificates for joining users, which are created using SanSig algorithm using their pid and users' sanitising public keys. Finally, the *Universal HO* protocol provides roaming users with network access while maintaining network and users' security and privacy via assuring security features mentioned in Section III-C.

A. System Initialisation

In this phase, we assume that all communication channels are secure. During this phase, all required parameters for UniHand are produced, such as the registration of UEs and gNBs in the network and initiating the accumulator. Hence this phase can be divided into three main parts:

1) **gNB Registration**: Every gNB in the network has to do registration and authentication with the AuC. For that, first, gNB has to generate asymmetric sanitising key pair $(pk_{san}^{\text{gNB}}, sk_{san}^{\text{gNB}}) \leftarrow_{\$} \mathsf{SanSig.KGen}_{san}(1^n)$. Next, the gNB will send a registration message to AuC, which includes his/her sanitising public key (pk_{san}^{gNB}). Upon receiving the registration message, the AuC will authenticate the gNB and generate a certificate $(\mathbb{C}_G, \sigma_G)$ via SanSig algorithm, $\sigma_G \leftarrow_{\$} \mathsf{SanSig.Sign}(\mathbb{C}_G, sk_{sig}^{\text{AuC}}, pk_{san}^{\text{gNB}}, ADM(\mathbb{C}_{mod}^{G}))$. To expedite this process, AuC may execute this step offline.
2) **UE Registration**: Similarly, every user in the network needs to register into the network to utilise the provided network services. A UE begins this phase by generating asymmetric sanitising key pair $(pk_{san}^{\text{UE}}, sk_{san}^{\text{UE}}) \leftarrow_{\$} \mathsf{SanSig.KGen}_{san}(1^n)$. Next, the UE sends a registration request to the AuC, which includes all UE credentials, including their sanitising public key pk_{san}^{UE} via a secure channel. Upon receiving the registration message, the AuC generates two independent pseudo identities pid and T_{ID} (i.e. there is no direct relationship between the aliases), in addition to a symmetric long-term key (k_i) for the intended user and shares them with the UE, where $pid_i, T_{IDi} \leftarrow_{\$} \{0,1\}^n$.
3) **Accumulator Initialisation**: To initiate the revocation list (RL) using an accumulator ($\ddot{C}$), first the AuC has to generate a secret key for the accumulator via $\mathsf{KGen}_{\ddot{C}}(1^n) \rightarrow_{\$} (sk_{\ddot{C}})$, then generate the accumulator using $\mathsf{Gen}_{\ddot{C}}(sk_{\ddot{C}}, X) \rightarrow_{\$} RL$, where X is initially empty.

B. Initial authentication

Each registered user needs to execute this protocol to join the network securely. During this protocol, each communicating party mutually authenticates their partner, and then the AuC generates a new certificate for the intended UE, as illustrated in Figure 2:

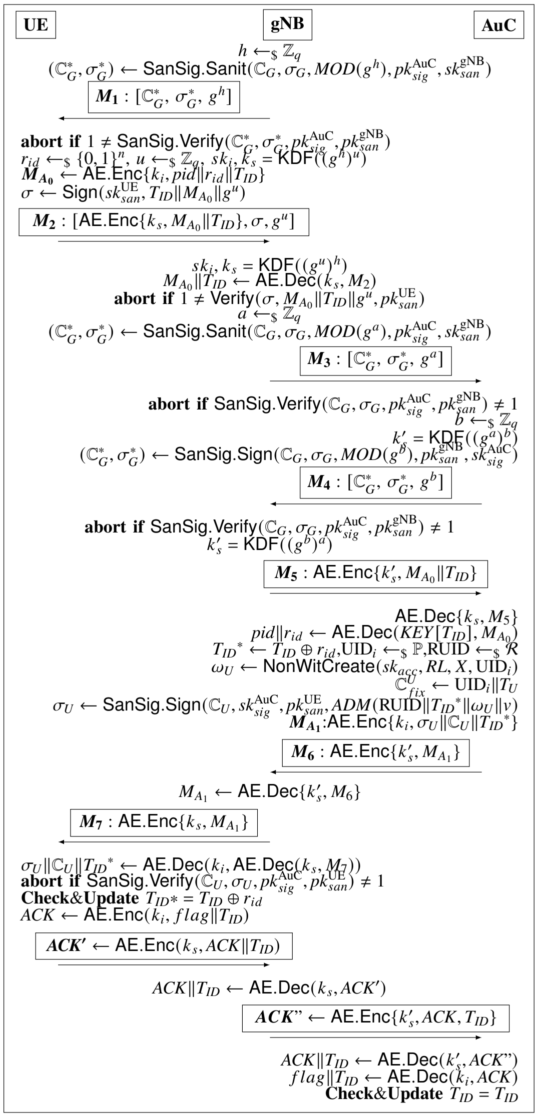

Fig. 2: The Initial Authentication protocol of UniHand Scheme.

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 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. 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 signature using (SanSig.Verify()) and checks the message integrity (i.e. $g^h \stackrel{?}{=} g^h$). If both verification hold then UE samples randomly (r_{id}&u) and computes session keys (sk_i, k_s). Next UE encrypts r_{id}, pid and T_{ID} using k_i, where pid&T_{ID} is user pseudo identities and k_i is the symmetric long-term key shared between the UE and AuC 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 → AuC. $\mathbf{M_3} : [\mathbb{C}^*_G,\ \sigma^*_G,\ g^a]$
After receiving the message M_2, gNB first computes session keys (sk_i, k_s) and decrypt M_2. Subsequently, the UE verifies the signature σ and checks the message integrity (i.e. $g^u \stackrel{?}{=} g^u$). If both verification hold, then gNB randomly samples a temporary key (a) and compute (g^a). Then the gNB utilises the 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 AuC via M_3.
Step 4: AuC → gNB. $\mathbf{M_4} : [\mathbb{C}^*_G,\ \sigma^*_G,\ g^b]$
Upon receiving M_3, AuC first verifies the signature using (SanSig.Verify) and checks the message integrity (i.e. $g^a \stackrel{?}{=} g^a$). If both verification hold, then AuC samples randomly (b) and computes the session key (k'_s). Next the AuC re-sign $\mathbb{C}^*_G$ using SanSig.Sign, which includes the generated DH ephemeral ($MOD(g^b)$). Then the AuC constructs a message M_4 and sends it to gNB.
Step 5: gNB → AuC. $\mathbf{M_5} : \mathsf{AE.Enc}\{k'_s, M_{A_0} \| T_{ID}\}$
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 AuC.
Step 6: AuC → gNB. $\mathbf{M_6} : \mathsf{AE.Enc}\{k'_s, M_{A_1}\}$
Upon receiving M_5, the AuC 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 AuC 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, AuC generates a non-membership witness ($\omega_U \leftarrow$ NonWitCreate(.), and specifies the version number (v) of RL. AuC creates and signs the ($\mathbb{C}_U$) by generating "fixed" part ($\mathbb{C}^U_{fix}$) and "modifiable" part $\mathbb{C}^U_{mod}$, where $\mathbb{C}^U_{fix} = \mathrm{UID}_i \| T_U$ (T_U is a user subscription validity period) and $\mathbb{C}^U_{mod} = \mathrm{RUID} \| T_{ID}{}^* \| \omega_U \| v$ (RUID a random-user ID). Then AuC signs the entire certificate using SanSig.Sign(.) algorithm. AuC then stores UID_i, T_{IDi} and $T_{ID}{}^*_i$ (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 AuC encrypts M_6 using the session key and sends it to the gNB.
Step 7: gNB → UE. $\mathbf{M_7} : \mathsf{AE.Enc}\{k_s, M_{A_1}\}$
Upon the receipt of the message from AuC, 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 $\xrightarrow{ACK"}$ AuC.
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 verification hold, then UE sends an acknowledgement message to the AuC to confirm the new temporary identity and delete the old one.

UE | **gNB**

$h \leftarrow_\$ \mathbb{Z}_q$
$\mathbb{C}^G_{mod} = \mathrm{EID} \| g^h$
$(\mathbb{C}^*_G, \sigma^*_G) \leftarrow \mathsf{SanSig.Sanit}(\mathbb{C}^G_{fix} \| \mathbb{C}^G_{mod}, \sigma_G, pk^{\mathrm{AuC}}_{sig}, sk^{\mathrm{gNB}}_{san})$
$\boxed{\boldsymbol{M_1} : [\mathbb{C}^*_G,\ \sigma^*_G,\ g^h]}$ $\longleftarrow$

abort if $1 \neq \mathsf{SanSig.Verify}(\mathbb{C}^*_G, \sigma^*_G, pk^{\mathrm{AuC}}_{sig}, pk^{\mathrm{gNB}}_{san})$
$u \leftarrow_\$ \mathbb{Z}_q$, $\mathrm{RUID} \leftarrow_\$ \mathcal{R}$
$sk_i, k_s = \mathsf{KDF}((g^h)^u)$
$\mathbb{C}^U_{mod} = \mathrm{RUID} \| \omega_U \| v \| g^u$
$(\mathbb{C}^*_U, \sigma^*_U) \leftarrow \mathsf{SanSig.Sanit}(\mathbb{C}^U_{fix} \| \mathbb{C}^U_{mod}, \sigma_U, pk^{\mathrm{AuC}}_{sig}, sk^{\mathrm{UE}}_{san})$
$\boxed{\boldsymbol{M_2} : [\mathsf{AE.Enc}\{k_s, \mathbb{C}^*_U, \sigma^*_U\}, g^u]}$ $\longrightarrow$

$sk_i, k_s = \mathsf{KDF}((g^u)^h)$
abort if $1 \neq \mathsf{SanSig.Verify}(\mathbb{C}^*_U, \sigma^*_U, pk^{\mathrm{AuC}}_{sig}, pk^{\mathrm{UE}}_{san})$
abort if $1 \neq \mathsf{Verify}_C(RL_v, \omega_U, \mathrm{UID}_i)$
Update $[(\omega^*_U) \leftarrow_\$ \mathsf{NonWitUpdate}(RL, RL^*, x^*, \mathrm{UID}_i, \omega_U)]$
$\boxed{\boldsymbol{M_3} \leftarrow \mathsf{AE.Enc}\{k_s, \omega^*_U \| v*\}}$ $\longleftarrow$

$\omega^*_U \| v* \leftarrow \mathsf{AE.Dec}(k_s, M_3)$
Store $(\omega^*_U \| v*)$

Fig. 3: UniHand's Universal Handover phase.

C. Universal HO

Each roaming user must execute this protocol to secure transit between small cells in 5G. During this protocol, each communicating party mutually authenticates their partner. The Universal HO protocol is described below and illustrated in Figure 3.

Step 1: gNB → UE. $\mathbf{M_1}$: $[\mathbb{C}^*_G, \sigma^*_G, g^h]$.
This step is similar to the initial authentication protocol's first step (step1).
Step 2: UE → 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 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, then computes session keys (sk_i, k_s). Next, UE updates their certificate (the modifiable part), i.e. $MOD(\mathrm{RUID} \| \omega \| v \| g^u)$ (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.
Step 3: gNB → UE $\mathbf{M_3}$:← $\mathsf{AE.Enc}\{k_s, \omega^*_U \| v*\}$
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 ($\mathsf{Verify}_{\ddot{C}}(\mathrm{UID}_i, ..)$) is in the revocation list. 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. 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. Details of this protocol are depicted in Figure 3.

V. Security Framework

Here we formalise the security properties of our UniHand scheme, which follows Bellare-Rogaway [6] 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 C. The adversary wins the game if it either causes a winning event (i.e. breaking authentication or anonymity) or terminates and guesses a challenge bit b (i.e. breaking key indistinguishability). We utilise the Khan et al. framework [23] to capture notions of *user unlinkability*, and eCK framework [25] to capture key indistinguishability (KIND).

A. 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: mutual authentication, key indistinguishability and unlinkability. In our games, the challenger C maintains a single AuC, 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 authentication centre AuC), and systems $\mathrm{gNB}_1, \ldots, \mathrm{gNB}_{n_P}$ (representing gNBs communicating with the authentication centre), each potentially running up to n_S executions of protocol Π. We abuse notation and use π^s_i 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.
- $\rho \in \{\mathrm{UE}, \mathrm{gNB}, \mathrm{AuC}\}$: 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 AuC/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 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 $\rho \in \{\text{AuC}, \text{gNB}, \text{UE}\}$ (for instance, **CorruptASK**$(\text{AuC}, 0)$ or **CorruptASK**(gNB, i)). C checks if $\mathcal{A}$ previously corrupted these secrets, returning $\perp$ if so, otherwise the 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 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 C returns $\perp$. Otherwise, C sets $k_0 \leftarrow \pi_i^s.k$, and $k_1 \leftarrow_\$ \{0, 1\}^\lambda$, and returns k_b to $\mathcal{A}$ (where b was sampled by 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 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 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 $\pi.id = i$. 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**. C returns a $\perp$ if $\pi_b.\alpha \neq \texttt{in-progress}$.

B. 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*. On a high level, partnering ensure 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 [5], 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 AuC. Thus, two problems occur: we need to capture the messages that UE authenticates to the AuC, 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 AuC, and we introduce *matching subsets* to capture the subset of messages authenticated between the gNB and the AuC and UE respectively.

***Definition** 1 (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** 2 (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 $\pi_i^s.sid \subset \pi_j^t.sid$.

We now turn to define each security game for MA, KIND and Unlink.

C. Mutual Authentication

In this section, we describe the overall goal of $\mathcal{A}$ in the MA security game and the queries that $\mathcal{A}$ has access to. The experiment $\mathbf{Exp}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger C and an adversary $\mathcal{A}$. At the beginning of the experiment, C generates long-term asymmetric keys for the AuC 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 **Create**, **Send**, **CorruptLTK**, **CorruptASK**, **StateReveal** and **Reveal** queries. $\mathcal{A}$ 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.

We now turn to describe our cleanness predicates. In the initial authentication protocol, if a session (owned by AuC or UE) accepts without either a matching subset or a matching session, then $\mathcal{A}$ only wins if they have not compromised: (a) the state of any (at the point of compromise) matches, or (b) both the long-term shared key of the UE partner and the long-term asymmetric key of the intended partner, or (c) the long-term asymmetric secrets of the gNB.

***Definition** 3 (Initial authentication cleanness):* A session π_i^s in the MA experiment described above is $\mathbf{clean}_{IA}$ 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 *and* for all

sessions π_j^t such that π_j^t is a matching session of π_i^s **StateReveal**$(j, t, \pi_j^t.\rho)$ has not been issued.
2) If $\pi_i^s.\rho \neq$ gNB, 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**$(\pi_i^s.pid, (\text{AuC}, \text{UE}) \backslash \pi_i^s.\rho)$ have not both been issued before $\pi_i^s.\alpha = \texttt{accepted}$.
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**$(, \text{AuC})$ have not been issued before $\pi_i^s.\alpha = \texttt{accepted}$.

***Definition** 4 (Universal Handover cleanness):* A session π_i^s in the MA experiment described above 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 $\pi_i^s.\alpha = \texttt{accepted}$.

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 *AuC* 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** 5 (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.

D. Key Indistinguishability

Here we describe the overall goal of $\mathcal{A}$ in the key indistinguishability KIND security game and the queries that $\mathcal{A}$ has access to. The experiment $\mathbf{Exp}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger C and an adversary $\mathcal{A}$. At the beginning of the experiment, C generates long-term symmetric keys for the AuC and each user UE_i and each gNB gNB_i (where $i \in [n_P]$), samples a random bit $b \leftarrow_\$ \{0,1\}$ and then interacts with $\mathcal{A}$ via queries mentioned below. At some point, $\mathcal{A}$ issues a **Test**(i, s, ρ) 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** 6 (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.

***Definition** 7 (cleanness predicate):* A session π_i^s in the KIND experiment described above is $\mathbf{clean}_{IA}$& $\mathbf{clean}_{UH}$ 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 $\pi_i^s.\alpha = \texttt{accepted}$.

E. Unlinkability

In this section, we describe the overall goal of $\mathcal{A}$ in the Unlink security game and the queries that $\mathcal{A}$ has access to. The experiment $\mathbf{Exp}^{\mathsf{Unlink},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ is played between a challenger C and an adversary $\mathcal{A}$. At the beginning of the experiment, C generates long-term symmetric keys for the AuC and each user UE_i and each gNB gNB_i (where $i \in [n_P]$), samples a random bit $b \leftarrow_\$ \{0,1\}$ and then interacts with $\mathcal{A}$ via queries mentioned below. $\mathcal{A}$ 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 $b' = b$.

Adversary Queries In the KIND 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. We now turn to define our cleanness predicate for the unlinkability game.

***Definition** 8 (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 $\pi_i^s.\alpha = \texttt{accepted}$.

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 non-negligible advantage, which we formalise below.

***Definition** 9 (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.

VI. Security Analysis

In this section, we formally prove the security of our protocols. We begin by demonstrating that our protocols achieve mutual authentication, then show that the session keys output from the secure handover scheme are indistinguishable from random, and finally show that external attackers cannot link sessions owned by the same party.

A. Mutual Authentication security

Here we formally analyse the MA-security of the UniHand scheme.

1) MA-security of Initial Authentication: We begin by showing that the Initial Authentication protocol achieves mutual authentication.

Theorem 1: **MA-security of Initial Authentication**. Initial Authentication depicted in Figure 2 is MA-secure under the cleanness predicate $\mathbf{clean}_{IA}$ in Definition 3. 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 $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$, $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ 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 $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}_{IA}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_3}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$. Due to space constraints, we give only a proof sketch and point readers to [2][1] for the full proof details.

Case 1: We assume the first **clean** session π_i^s to accept without a matching session or subset sets $\pi_i^s.\rho$ = 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 AuC). In this subcase $\mathcal{A}$ cannot corrupt the long-term UE symmetric secret or the $\pi_i^s.pid_{\mathrm{gNB}}$ asymmetric key.

On a high level, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the UE due to the gNB signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the UE are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the UE due to the AE security of the Enc scheme. Similarly, since all messages sent between the AuC and the UE are encrypted under the UE long-term symmetric key (and the UE long-term key cannot be compromised by $\mathcal{A}$, $\mathcal{A}$ cannot modify messages sent between the gNB and the UE due to the AE security of the Enc scheme.

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 AuC). In this subcase, $\mathcal{A}$ cannot corrupt the long-term AuC asymmetric key or the $\pi_i^s.pid_{\mathrm{gNB}}$ asymmetric key.

On a high-level, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the UE due to the gNB signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the UE are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the UE due to the AE security of the Enc scheme. Similarly, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the AuC due to the gNB signatures and the AuC signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the AuC are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the AuC due to the AE security of the Enc scheme. As the gNB proxies ciphertexts between the UE and the AuC, this means that $\mathcal{A}$ cannot modify any messages between UE and AuC.

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 $\pi_i^s.\rho$ = AuC.

Case 2.1: According to the definition of this case, AuC 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 $\pi_i^s.pid_{\mathrm{gNB}}$ asymmetric key.

On a high level, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the UE due to the gNB signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the UE are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the UE due to the AE security of the Enc scheme. Similarly, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the AuC due to the gNB signatures and the AuC signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the AuC are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the AuC due to the AE security of the Enc scheme. As the gNB proxies ciphertexts between the UE and the AuC, this means that $\mathcal{A}$ cannot modify any messages between UE and AuC.

[1]The full version of the security analysis and the security framework is available in the Supplementary Material.

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

Case 2.2: According to the definition of this case, AuC 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 asymmetric keys of UE or the $\pi_i^s.pid_{\text{gNB}}$ asymmetric key.

On a high level, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the UE due to the gNB and UE signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the UE are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the UE due to the AE security of the Enc scheme. Similarly, we show that $\mathcal{A}$ cannot inject DH public keys between the gNB and the AuC due to the gNB signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged between the gNB and the AuC are encrypted, $\mathcal{A}$ cannot modify messages sent between the gNB and the AuC due to the AE security of the Enc scheme. As the gNB proxies ciphertexts between the UE and the AuC, this means that $\mathcal{A}$ cannot modify any messages between UE and AuC.

We turn to bound the advantage of $\mathcal{A}$ in **Case 3**, where assume that the first **clean** session π_i^s to accept without a matching session or a matching subset is a gNB (i.e. $\pi_i^s.\rho$ = gNB).

Case 3: According to the definition of 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 AuC partners). Note that in this case, $\mathcal{A}$ cannot corrupt the asymmetric long-term keys of either the AuC or the $\pi_i^s.pid$.

On a high level, we show that $\mathcal{A}$ cannot inject DH public keys from the UE to the gNB due to the UE signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged from the UE to the gNB are encrypted, $\mathcal{A}$ cannot modify these messages due to the AE security of the Enc scheme. Similarly, we show that $\mathcal{A}$ cannot inject DH public keys from the AuC to the gNB due to the AuC signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged from the UE to the gNB are encrypted, $\mathcal{A}$ cannot modify these messages due to the AE security of the Enc scheme. this means that $\mathcal{A}$ cannot modify any messages between UE and AuC.

2) MA- security of Universal Handover: This section formally analyses the MA-security of the Universal Handover protocol.

Theorem 2: **MA-security of the Universal Handover**. Universal Handover depicted in Figure 3 is MA-secure under the cleanness predicate in Definition 4. 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, Auth security of AE, the KDF security of KDF and the DDH assumption.

Proof: Our proof is divided into two cases $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ and $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

1) **Case 1:** The test session π_i^s, where $\pi_i^s.\rho$ = UE accepts messages $\mathbf{M_1}$ and $\mathbf{M_3}$ without an honest matching gNB partner (no matching subset). The analysis of **Case 1** proceeds similarly to **Case 1.1** of the initial authentication protocol and thus, for brevity, we omit the proof of this case, which can be found in [2][1].
2) **Case 2:** The test session π_i^s, where $\pi_i^s.\rho$ = gNB accepts message $\mathbf{M_2}$ without an honest matching UE partner (no matching subset). The analysis of **Case 2** proceeds similarly to **Case 3** of the initial authentication protocol, and thus, for brevity, here we omit the proof of this case, which can be found in [2][1].

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to $\mathbf{Adv}^{\mathsf{MA},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{MA},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{MA},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda))$.

B. Key Indistinguishability

Here we prove the key indistinguishability of our protocols. Due to space constraints, we encapsulate both protocols into Theorem 3.

Theorem 3: **KIND-security of UniHand**. UniHand scheme depicted in Figure 2 and Figure 3 are KIND-secure under the cleanness predicate (Definition 7). 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, Auth security of AE, the KDF security of KDF and the DDH assumption.

Our proof is divided into two cases $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},c_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$ and $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},c_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$:

1. **Case 1:** The test session π_i^s accepts messages without an honest matching session (or matching subset).
2. **Case 2:** The test session π_i^s accepts messages with a matching session (and subset).

We then bound the advantage of $\mathcal{A}$ winning the game under certain assumptions to $\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean}}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) \leq (\mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_1}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda) + \mathbf{Adv}^{\mathsf{KIND},\mathbf{clean},C_2}_{\Pi,n_P,n_S,\mathcal{A}}(\lambda)$.

On a high-level, in **Case 1** we show that $\mathcal{A}$'s advantage in causing a session to accept without a matching session (or subset) to be equal to $\mathcal{A}$ breaking the MA-security of UniHand. In **Case 2** we show that since the Diffie-Hellman values are exchanged honestly, the DH outputs can be replaced with uniformly random values in π_i^s, which are in turn used to derive the session key. Hence, the session key is indistinguishable from a uniformly random value from the same distribution (i.e. indistinguishable from the output of a **Test** query), and $\mathcal{A}$ has no advantage in guessing b.

[1]The full version of the security analysis and the security framework is available in the Supplementary Material.

C. Unlinkability

Here we discuss the unlinkability of UniHand. We consider a strong notion of anonymity where $\mathcal{A}$ wins simply by linking a so-called "test" session to another protocol execution - since it knows the identities of all other protocol executions, this allows de-anonymisation. Thus, our framework captures user anonymity, user confidentiality and untraceability.

Due to space constraints, we include the proof sketch of Unlink-security for the initial authentication protocol since the universal handover analysis follows similarly (and more simply). For full details of each proof, refer to [2][1].

Theorem 4: **Unlink-security of UniHand.** UniHand depicted in Figure 2 and Figure3 is unlinkable under the cleanness predicate in Definition 8. 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, Conf security of AE, the KDF security of KDF and the DDH assumption.

On a high level, we show that $\mathcal{A}$ cannot inject DH public keys from the UE to the gNB due to the UE signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged from the UE to the gNB are encrypted, $\mathcal{A}$ cannot break the confidentiality of the messages and hence cannot guess the identity of the intended user. Similarly, we show that $\mathcal{A}$ cannot inject DH public keys from the AuC to the gNB due to the AuC signatures over these values, based on the EUFCMA security of the SanSig scheme. As the secret DH output is used to derive keys for the authenticated encryption scheme securely, and all other messages exchanged from the UE to the gNB are encrypted, $\mathcal{A}$ cannot learn plaintext messages and hence cannot guess the identity of the intended user.

VII. Discussion

In this section, we first compare our proposed scheme UniHand with existing state-of-the-art AKA protocols proposed for 5G [21], [15], [20] and [24] in terms of security features and then describe the computational and communication cost of the proposed UniHand scheme. Table I show that the existing schemes cannot simultaneously ensure all essential security features (described in Section III-C). In particular, no previous scheme is KCI-resilient, which is important in preventing impersonation attacks due to the leakage of long-term keys. Similarly, managing the significant number of subscribers in the 5G network is always essential. However, apart from UniHand, only [20] manages subscription revocation. The revocation mechanism used in [20] is based on Nyberg's one-way accumulator [27] that is a static base accumulator (i.e. regeneration of the accumulator is required for every addition/deletion to the revocation list), hence introducing computation overhead and negatively affecting the network's efficiency. From Table I, we can also see that none of the previous schemes has considered *fully* protecting previous communication in the presence of an adversary with a compromised long-term key (PFS) or supported KEF for fair

TABLE I: Features comparison.

Features / Schemes	**MA**	**SA**	**PFS**	**KEF**	**KCI**	**SRM**	**UHO**
5G [21]	YES	NO	NO	NO	NO	NO	NO
CPPHA [15]	YES	NO	NO	NO	NO	NO	YES
ReHand [20]	YES	YES	NO	NO	NO	YES	NO
RUSH [32]	YES	Partial	Partial	YES	NO	NO	YES
Protocol of [24]	YES	YES	NO	NO	NO	NO	NO
UniHand	YES	YES	YES	YES	YES	YES	YES

MA:Mutual Authentication, **SA**:Strong Anonymity, **PFS**: Perfect Forward Secrecy, **KEF**: Key-escrow Free, **KCI**: Key Compromise Impersonation, **SRM**: Secure Revocation Management, **UHO**: Universal HO

secret key generation other than RUSH protocol [32]. Both properties are essential to maintain the security of the 5G mobile network in case of long-term key corruption or trusted third-party corruption. For example, the protocol presented in [32] utilises the authentication protocol of the conventional 5G AKA and extends it to propose a new handover protocol with the assistance of Blockchain. The proposed handover protocol in [32] supports PFS but their initial authentication, which is based on the conventional 5G AKA, can not support PFS; thus, we can say that the protocol presented in [32] can partially address the PFS in the overall scheme. Similarly, the protocol of [32] partially supports strong anonymity, as it achieves anonymity but does not support unlinkability. On the contrary, UniHand supports all the crucial security properties such as mutual authentication, strong anonymity, perfect forward secrecy, key-escrow-free, KCI resilience, and secure revocation management while achieving secure handover.

Now, to show the effectiveness of our scheme, we present and compare the computation cost of UniHand with the state-of-art handover protocols. To ensure fairness of comparison, we compare UniHand with [15], [20], [21], [32], since like UniHand these schemes can also support handover. Here we conduct simulations of the cryptographic operations used in UniHand. In this context, we implemented the required cryptographic operations at the server level (as the aggregated network entities, i.e. AuC, gNB) on a Dell Inspiron machine with an i7 core, 2.30GHz CPU, and 16.0 GB RAM. As for measuring the computational cost at UE, we use Samsung Galaxy Note9, which runs the Android-10 mobile operating system and is equipped with octa-core 1.8GHz Quad-Core ARM Cortex-A55, 2.7GHz Quad-Core Mongoose M3 processors, and 6GB RAM. For implementation specifications, we use Java pairing-based cryptography (JPBC) [19] and Java Cryptography Extension (JCE) [30].

Table II compares the performance based on the computations cost. It shows the used primitives in each protocol to measure the total required time to perform the protocol at the UE level (T_{UE}) and system level (T_{Sys}). UniHand's

TABLE II: Performance comparison based on computational cost

Protocol	**Entity**	**Initial Authentication**	**Total (ms)**	**Entity**	**Universal HO**	**Total (ms)**
Conventional 5G [21]	T_{UE}	$4T_{PRG} + 2T_{MAC} + T_{AES}$	$\approx$ 2.817	T_{UE}	$4T_{AES} + T_{PRG}$	$\approx$ 2.703
	T_{Sys}	$3T_{PRG} + 1T_{MAC} + 2T_H + T_{ELG}$	$\approx$ 1.278	T_{Sys}	$4T_{AES} + T_{PGR}$	$\approx$ 1.559
CPPHA [15]	T_{UE}	$5T_{PRG} + 2T_{MAC} + T_{AES} + T_H$	$\approx$ 3.687	T_{UE}	$2T_{AES} + T_{PRG} + 4T_H$	$\approx$ 3.19
	T_{Sys}	$4T_{PRG} + 1T_{MAC} + 3T_H + 2T_{ELG} + T_{AES}$	$\approx$ 2.536	T_{Sys}	$T_{AES} + T_{PGR} + 6T_H + 3T_{ELG}$	$\approx$ 3.04
ReHand [20]	T_{UE}	$2T_{AES} + 4T_H + T_{PRG}$	$\approx$ 3.193	T_{UE}	$T_{PGR} + 3T_H + T_{AES}$	$\approx$ 2.232
	T_{Sys}	$3T_{AES} + 5T_H + T_{PRG}$	$\approx$ 1.727	T_{Sys}	$T_{PRG} + 5T_H + 2T_{AES}$	$\approx$ 1.342
RUSH [32]	T_{UE}	$7T_{PRG} + 2T_{MAC} + T_{AES} + 3T_H + T_E + 5T_{Mod} + 3T_{SM}$	$\approx$ 9.737	T_{UE}	$3T_{PRG} + T_{SM} + 5T_H + T_E + T_{Mod}$	$\approx$ 5.42
	T_{Sys}	$4T_{PRG} + 1T_{MAC} + 3T_H + T_{ELG} + 2T_{SM} + 3T_{Mod}$	$\approx$ 1.967	T_{Sys}	$2T_{PRG} + T_{SM} + 6T_H + T_E + T_{Mod}$	$\approx$ 1.365
UniHand	T_{UE}	$3T_{PRG} + 6T_{AES} + T_H + 5T_E + 5T_{Mod}$	$\approx$ 9.48	T_{UE}	$3T_{PRG} + 2T_{AES} + T_H + 3T_E + 3T_{Mod}$	$\approx$ 5.516
	T_{Sys}	$8T_{PRG} + 11T_{AES} + 4T_H + 9T_E + 9T_{Mod}$	$\approx$ 8.759	T_{Sys}	$2T_{PGR} + 2T_{AES} + T_H + 3T_E + 3T_{Mod}$	$\approx$ 2.153

$\boldsymbol{T_{PRG}}$: Random number generators[$T_{\text{UE}} = 0.47$],[$T_{Sys} = 0.13$], $\boldsymbol{T_{AES}}$:Symmetric encryption/decryption[$T_{\text{UE}} = 0.56$],[$T_{Sys} = 0.39$], $\boldsymbol{T_{MAC}}$: Message authentication (Hmac-SHA256) [$T_{\text{UE}} = 0.195$],[$T_{Sys} = 0.071$] $\boldsymbol{T_H}$: Hash operations(SHA-256)[$T_{\text{UE}} = 0.40$],[$T_{Sys} = 0.09$], $\boldsymbol{T_E}$: exponentiation operation[$T_{\text{UE}} = 0.86$],[$T_{Sys} = 0.34$] $\boldsymbol{T_{Mod}}$: Modular operations[$T_{\text{UE}} = 0.002$],[$T_{Sys} = 0.001$], $\boldsymbol{T_{ELG}}$: Elgamal Asymmetric encryption/decryption operations[$T_{\text{UE}} = 1.176$],[$T_{Sys} = 0.648$] , $\boldsymbol{T_{SM}}$: scalar multiplication [$T_{\text{UE}} = 1.148$],[$T_{Sys} = 0.235$]

TABLE III: Performance comparison based on communication cost.

Protocol	**Link**	**Total message size (bits)**	**Transmission time (μs)**	**Propagation time (μs)**	**Total time (μs)**
Conventional 5G [21]	UP	640	25.6	2.01	27.64
	Down	256	5.12	1.33	6.46
CPPHA [15]	UP	1728	69.12	2.68	71.8
	Down	1056	21.12	1.33	22.45
ReHand [20]	UP	832	33.28	1.34	34.62
	Down	768	15.36	0.67	39.8
RUSH [32]	UP	896	35.84	1.33	37.17
	Down	896	17.92	0.67	18.59
UniHand	UP	2109	84.36	0.67	85.03
	Down	3901	78	1.33	79.33

initial authentication protocol requires approximately 9.48ms and 8.759 ms on UE and system level, respectively. On the other hand, the universal HO protocol requires approximately 5.561 ms and 2.153 ms on UE and system level, respectively.

The AuC (Authentication Center) plays a crucial role in various important functions within the mobile network, including connectivity and mobility management, authentication and authorization, subscriber data management, and policy management. As the number of connected and roaming users increases in 5G handovers, the communication to the AuC and its computation also grows. Due to the higher number of small cells, handovers will occur more frequently compared to 4G. If the AuC is incorporated in each handover, it could potentially be overwhelmed [20], [28]. Our proposed UniHand scheme can resolve this issue by eliminating the need for the AuC's involvement during the execution of the handover protocol. This reduction in AuC's responsibilities effectively decreases or stabilizes the overall computation overhead on the AuC compared to the conventional 5G AKA and handover protocols (as shown in Figure 4). Figure 4 presents a comparative analysis between UniHand and the conventional 5G HO scheme [21] concerning the required computational cost at the AuC for authenticating a single user, including one execution of 5G-AKA and one execution of UniHand-AKA, while the user roams between several small cells (involving 35 HO protocol executions). In the conventional 5G scheme, where AuC assistance is obligatory for each handover, the computational cost at the AuC escalates proportionally with the number of handovers. Conversely, our proposed scheme (UniHand) maintains a consistent computational cost at the AuC, independent of the number of handovers, as it doesn't necessitate AuC intervention. As a result, our proposed scheme does not impose any additional computation cost on the AuC during the HO process. Additionally, when the AuC is not accessible or offline, our proposed HO protocol remains unaffected, representing a significant improvement from the current solution.

Next, we analyse and compare the communication cost of the proposed UniHand universal HO protocol with respect to the existing handover protocols, including the conventional 5G [21], ReHand [20], RUSH [32] and CPPHA [15]. In this context, we measure the transmitted message's propagation and transmission time. We consider the size and the network's data rate for accurate computation of transmission delay. Based on the 3GPP specification [21], in a wide-area scenario, the data rate of uplink data rate is 25 Mbps, and the downlink is 50 Mbps. Accordingly, here we use the above-specified measurements to compute the transmission delay of transmitted messages in all protocols. For the message size, we use the recommended message size in each state-of-the-art protocol for a head-to-head comparison. In UniHand the certificate $\mathbb{C}$ is of size 192 bits, the signature is 1533 bits, ECDH key size

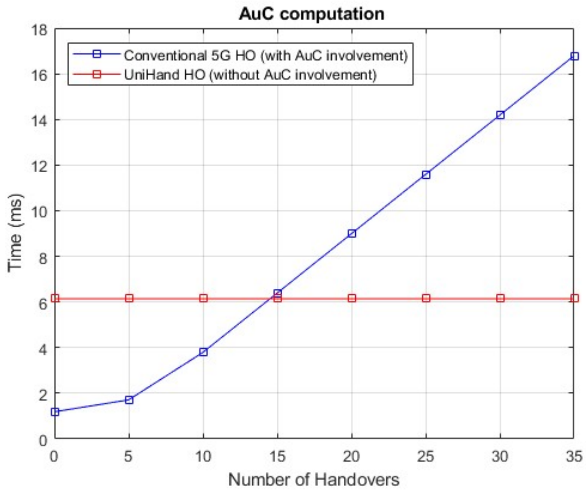

Fig. 4: Computation overhead at AuC during HO for conventional 5G and UniHand

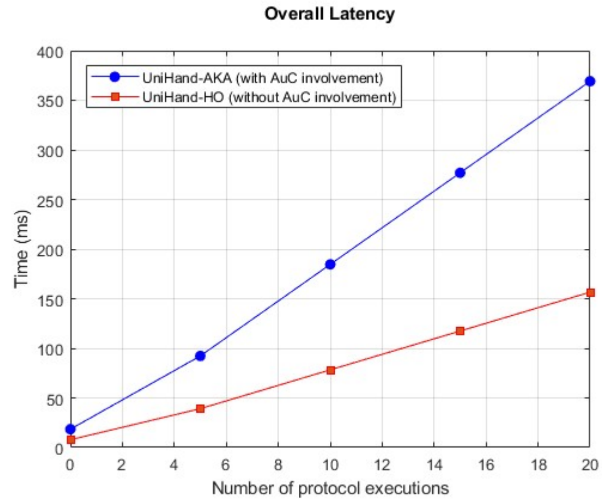

Fig. 5: Overall latency at AuC during the executions of UniHand protocols

is 256 bits, and finally, the non-membership witness is 2048 bits. On the other hand, we consider the spectrum wave speed and the distance between the user and the nearest connected gNB to measure the propagation delay. In this context, we follow the specified propagation speed and distance in 3GPP specification [21], which is $3 \times 108 m/s$ and 200 m for the propagation speed and the approximate maximum cell size in 5G, respectively. Hence the propagation delay in sending one message in 5G is approximately 0.67 μs. Table III shows the communication cost of UniHand and the related schemes. From the Table III we can see that UniHand has a little more communication overhead compared to the existing state-of-the-art protocols [21], [20], [32], [15].

***Remark** 1:* The rise in the communication cost is directly associated with the size of the used certificate alongside its signature and the non-membership witness for user revocation. Nevertheless, the combination of signatures and certificates enables our scheme to achieve a secure universal handover without the assistance of the AuC. While the addition of a non-membership witness enables UniHand to manage revoked users in the network. It is also worth noting that our comparison is not quite fair, as only UniHand counts the communication cost of the revocation scheme, which adds more cost on UniHand.

Although both the proposed protocols (Initial Authentication and Handover) ensure all the specified security properties, there is a notable distinction between them. Unlike the Initial Authentication protocol, the execution of our proposed Handover protocol does not involve the AuC. Figure 5 provides an overview of the overall latency cost for each protocol in a handover-based scenario, including both communication and computation costs. To simulate this scenario, where a user moves between cells (0-20 cells), we execute both UniHand protocols (Initial Authentication and Handover) and measure the overall latency required for 0-20 executions of each protocol. From the observations in Figure 5, it is evident that the execution time of our proposed Handover protocol (which does not require the AuC) is considerably less than our Initial Authentication protocol (which does). This reduction in execution time demonstrates the efficiency and benefits of our Handover protocol, which eliminates the need for AuC involvement. Despite this advantage, our proposed scheme ensures several key security properties (such as PFS, KEF, and KCI) that the conventional 5G scheme fails to achieve. By striking this balance between efficient performance and enhanced security features, UniHand represents a promising approach to address the limitations of state-of-the-art protocols.

***Remark** 2:* In general, the 5G networks have an acceptable latency range defined by ETSI [18] of 1 ms to 100 ms, with further discussion of the ETSI standards in [22], [28]. This latency is particularly relevant for scenarios that require low latency, such as VR-assisted tele-surgery, Intelligent Transportation Systems, and factory automation. The latency rates are influenced by both computational and communication costs. In the case of the proposed UniHand handover, the total computational cost including UE and Sys is 7.66 ms, while the total communication cost for upload and download is 164.36 μs. Consequently, the overall latency cost incurred by the proposed scheme is only 7.83 ms, which perfectly falls within the acceptable range of latency requirement for 5G networks and remarkably close to the absolute minimum range of 1 ms. Therefore, our handover achieves accepted latency rates as defined by the ETSI standards.

VIII. Conclusion

This article proposed a new AKA handover scheme (UniHand) to achieve secure, privacy-preserving universal authentication for small cell networks in 5G mobile networks. Our proposal UniHand can guarantee all the essential security properties (as mentioned in Section III-C). It can tackle all the security vulnerabilities of the existing schemes and weaknesses in the conventional 5G-AKA. Our proposed scheme has been designed based on sanitisable signatures, ephemeral Diffie-Hellman key exchange, key derivation functions, authenticated encryption, and dynamic accumulator. We conducted a formal security and privacy analysis of the proposed scheme and compared the security features of the proposed UniHand scheme with the existing state-of-the-art 5G-AKA protocols. Finally, We evaluated the performance of the UniHand scheme, which shows a reasonable computation and communication cost while achieving all required security and privacy properties.

REFERENCES

[1] Ateniese, G., Chou, D.H., de Medeiros, B., Tsudik, G.: Sanitizable Signatures. In: di Vimercati, S.d.C., Syverson, P., Gollmann, D. (eds.) Computer Security – ESORICS 2005. pp. 159–177. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2005). https://doi.org/10.1007/11555827-10

[2] Authors, A.: UniHand supplementary material (2023), https://drive.google.com/file/d/1UZtLpNMOW21ptTNMBAK34818gtRaasFT/view?usp=share_link

[3] Baldimtsi, F., Camenisch, J., Dubovitskaya, M., Lysyanskaya, A., Reyzin, L., Samelin, K., Yakoubov, S.: Accumulators with Applications to Anonymity-Preserving Revocation. In: 2017 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 301–315 (Apr 2017). https://doi.org/10.1109/EuroSP.2017.13

[4] Basin, D., Dreier, J., Hirschi, L., Radomirovic, S., Sasse, R., Stettler, V.: A Formal Analysis of 5G Authentication. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security. pp. 1383–1396. CCS '18, Association for Computing Machinery, New York, NY, USA (Oct 2018). https://doi.org/10.1145/3243734.3243846, http://doi.org/10.1145/3243734.3243846

[5] Bellare, M., Namprempre, C.: Authenticated Encryption: Relations among Notions and Analysis of the Generic Composition Paradigm. Journal of Cryptology **21**(4), 469–491 (Oct 2008). https://doi.org/10.1007/s00145-008-9026-x, https://doi.org/10.1007/s00145-008-9026-x

[6] Bellare, M., Rogaway, P.: The Security of Triple Encryption and a Framework for Code-Based Game-Playing Proofs. In: Vaudenay, S. (ed.) Advances in Cryptology - EUROCRYPT 2006. pp. 409–426. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2006). https://doi.org/10.1007/11761679-25

[7] Benaloh, J., de Mare, M.: One-Way Accumulators: A Decentralized Alternative to Digital Signatures. In: Helleseth, T. (ed.) Advances in Cryptology — EUROCRYPT '93. pp. 274–285. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (1994). https://doi.org/10.1007/3-540-48285-7-24

[8] Borgaonkar, R., Hirschi, L., Park, S., Shaik, A.: New Privacy Threat on 3G, 4G, and Upcoming 5G AKA Protocols. Tech. Rep. 1175 (2018), http://eprint.iacr.org/2018/1175

[9] Braeken, A.: Symmetric key based 5G AKA authentication protocol satisfying anonymity and unlinkability. Computer Networks **181**, 107424 (Nov 2020). https://doi.org/10.1016/j.comnet.2020.107424, https://www.sciencedirect.com/science/article/pii/S1389128620311130

[10] Brzuska, C., Fischlin, M., Freudenreich, T., Lehmann, A., Page, M., Schelbert, J., Schröder, D., Volk, F.: Security of Sanitizable Signatures Revisited. In: Jarecki, S., Tsudik, G. (eds.) Public Key Cryptography – PKC 2009. pp. 317–336. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00468-1-18

[11] Brzuska, C., Fischlin, M., Lehmann, A., Schröder, D.: Santizable Signatures: How to Partially Delegate Control for Authenticated Data. pp. 117–128 (Jan 2009)

[12] Brzuska, C., Fischlin, M., Lehmann, A., Schröder, D.: Unlinkability of Sanitizable Signatures. pp. 444–461 (May 2010). https://doi.org/10.1007/978-3-642-13013-7-26

[13] Brzuska, C., Pöhls, H.C., Samelin, K.: Non-interactive Public Accountability for Sanitizable Signatures. In: De Capitani di Vimercati, S., Mitchell, C. (eds.) Public Key Infrastructures, Services and Applications. pp. 178–193. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2013). https://doi.org/10.1007/978-3-642-40012-4-12

[14] Brzuska, C., Pöhls, H.C., Samelin, K.: Efficient and Perfectly Unlinkable Sanitizable Signatures without Group Signatures. In: Katsikas, S., Agudo, I. (eds.) Public Key Infrastructures, Services and Applications. pp. 12–30. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2014). https://doi.org/10.1007/978-3-642-53997-8-2

[15] Cao, J., Ma, M., Fu, Y., Li, H., Zhang, Y.: 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). https://doi.org/10.1109/TDSC.2019.2916593, conference Name: IEEE Transactions on Dependable and Secure Computing

[16] Chalkias, K., Baldimtsi, F., Hristu-Varsakelis, D., Stephanides, G.: Two Types of Key-Compromise Impersonation Attacks against One-Pass Key Establishment Protocols. In: Filipe, J., Obaidat, M.S. (eds.) E-business and Telecommunications. pp. 227–238. Communications in Computer and Information Science, Springer, Berlin, Heidelberg (2009). https://doi.org/10.1007/978-3-540-88653-2-17

[17] Cremers, C., Dehnel-Wild, M.: Component-Based Formal Analysis of 5G-AKA: Channel Assumptions and Session Confusion. San Diego, CA, USA (Feb 2019), https://publications.cispa.saarland/2758/

[18] Dahmen-Lhuissier, S.: 5G, https://www.etsi.org/technologies/5G

[19] De Caro, A., Iovino, V.: jPBC: Java pairing based cryptography. Proceedings of the 16th IEEE Symposium on Computers and Communications, ISCC 2011, IEEE (2011), http://gas.dia.unisa.it/projects/jpbc/

[20] Fan, C.I., Huang, J.J., Zhong, M.Z., Hsu, R.H., Chen, W.T., Lee, J.: 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). https://doi.org/10.1109/TIFS.2019.2931076

[21] 3rd Generation Partnership Project (3GPP), E.: Security architecture and procedures for 5G System. Technical Specification version 16.3.0 Release 16, 3GPP (Aug 2020), https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/16.03.00_60/ts_133501v160300p.pdf

[22] thales group: 5G technology and networks (speed, use cases, rollout) (Feb 2023), https://www.thalesgroup.com/en/markets/digital-identity-and-security/mobile/inspired/5G

[23] Khan, H., Dowling, B., Martin, K.M.: Identity Confidentiality in 5G Mobile Telephony Systems. Tech. Rep. 876 (2018), http://eprint.iacr.org/2018/876

[24] Koutsos, A.: The 5G-AKA Authentication Protocol Privacy. In: 2019 IEEE European Symposium on Security and Privacy (EuroS&P). pp. 464–479 (Jun 2019). https://doi.org/10.1109/EuroSP.2019.00041

[25] LaMacchia, B., Lauter, K., Mityagin, A.: Stronger Security of Authenticated Key Exchange. In: Susilo, W., Liu, J.K., Mu, Y. (eds.) Provable Security. pp. 1–16. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2007). https://doi.org/10.1007/978-3-540-75670-5

[26] Li, J., Li, N., Xue, R.: Universal Accumulators with Efficient Nonmembership Proofs. In: Katz, J., Yung, M. (eds.) Applied Cryptography and Network Security. pp. 253–269. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72738-5-17

[27] Nyberg, K.: Fast accumulated hashing. In: Gollmann, D. (ed.) Fast Software Encryption. pp. 83–87. Lecture Notes in Computer Science, Springer, Berlin, Heidelberg (1996). https://doi.org/10.1007/3-540-60865-6-45

[28] Parvez, I., Rahmati, A., Guvenc, I., Sarwat, A., Dai, H.: A Survey on Low Latency Towards 5G: RAN, Core Network and Caching Solutions. IEEE Communications Surveys & Tutorials **PP** (Aug 2017). https://doi.org/10.1109/COMST.2018.2841349

[29] Peltonen, A., Sasse, R., Basin, D.: A comprehensive formal analysis of 5G handover. In: Proceedings of the 14th ACM Conference on Security and Privacy in Wireless and Mobile Networks. pp. 1–12. WiSec '21, Association for Computing Machinery, New York, NY, USA (Jun 2021). https://doi.org/10.1145/3448300.3467823, http://doi.org/10.1145/3448300.3467823

[30] Technology Network., O.: Java Cryptography Architecture (JCA) Reference Guide (Sep 2022), https://docs.oracle.com/javase/8/docs/technotes/guides/security/crypto/CryptoSpec.html

[31] Yan, X., Ma, M.: A lightweight and secure handover authentication scheme for 5G network using neighbour base stations. Journal of Network and Computer Applications **193**, 103204

(Nov 2021). https://doi.org/10.1016/j.jnca.2021.103204, https://www.sciencedirect.com/science/article/pii/S1084804521002095

[32] Zhang, Y., Deng, R.H., Bertino, E., Zheng, D.: Robust and Universal Seamless Handover Authentication in 5G HetNets. IEEE Transactions on Dependable and Secure Computing **18**(2), 858–874 (Mar 2021). https://doi.org/10.1109/TDSC.2019.2927664