E2E-AKMA: An End-to-End Secure and Privacy-Enhancing AKMA Protocol Against the Anchor Function Compromise

Yueming Li[1, 2], Long Chen[1, 3] ✉, Qianwen Gao[1, 2], Zhenfeng Zhang[1, 3] ✉
1. Institute of Software Chinese Academy of Sciences, Beijing, China
2. University of Chinese Academy of Sciences
3. Key Laboratory of System Software, Chinese Academy of Sciences
Email: {yueming2021,chenlong,qianwen2021,zhenfeng}@iscas.ac.cn

Abstract

The Authentication and Key Management for Applications (AKMA) system represents a recently developed protocol established by 3GPP, which is anticipated to become a pivotal component of the 5G standards. AKMA enables application service providers to delegate user authentication processes to mobile network operators, thereby eliminating the need for these providers to store and manage authentication-related data themselves. This delegation enhances the efficiency of authentication procedures but simultaneously introduces certain security and privacy challenges that warrant thorough analysis and mitigation.

The 5G AKMA service is facilitated by the AKMA Anchor Function (AAnF), which may operate outside the boundaries of the 5G core network. A compromise of the AAnF could potentially allow malicious actors to exploit vulnerabilities, enabling them to monitor user login activities or gain unauthorized access to sensitive communication content. Furthermore, the exposure of the Subscription Permanent Identifier (SUPI) to external Application Functions poses substantial privacy risks, as the SUPI could be utilized to correlate a user's real-world identity with their online activities, thereby undermining user privacy.

To mitigate these vulnerabilities, we propose a novel protocol named E2E-AKMA, which facilitates the establishment of a session key between the User Equipment (UE) and the Application Function (AF) with end-to-end security, even in scenarios where the AAnF has been compromised. Furthermore, the protocol ensures that no entity, aside from the 5G core network, can link account activities to the user's actual identity. This architecture preserves the advantages of the existing AKMA scheme, such as eliminating the need for complex dynamic secret data management and avoiding reliance on specialized hardware (apart from standard SIM cards). Experimental evaluations reveal that the E2E-AKMA framework incurs an overhead of approximately 9.4% in comparison to the original 5G AKMA scheme, which indicates its potential efficiency and practicality for deployment.

✉ Corresponding authors.

1 Introduction

In mobile communication networks, authentication via the cellular network provides a higher level of security, greater convenience of use, and cost-effective deployment for application servers. One example of a delegated authentication system is AKMA, which has been proposed by the 3rd Generation Partnership Project (3GPP) and is specified in TS 33.535 [3]. This system is planned to be integrated into the 5G standards. AKMA enables application service providers to delegate user authentication tasks to mobile network operators, thereby eliminating the need for maintaining a separate confidential database.

As discussed in Section 2, the system involves three key participants: the Home Network (HN), User Equipment (UE), and Application Function (AF). The HN is established and managed by telecommunications operators such as AT&T, T-Mobile, and Verizon. UE refers to the user's mobile device, which includes a SIM card issued by the corresponding HN. The AF represents application service providers, such as social networking platforms and online shopping websites. The AF delegates the task of authenticating the UE to the respective HN to which the UE is subscribed. To facilitate this process, the HN deploys a specialized network element known as the AKMA Anchor Function (AAnF), which is designed to provide this specific service. This mechanism allows the AF to verify the identity of the UE through the HN without requiring access to additional information or knowledge about the user's actual identifier.

AKMA offers a promising solution for enhancing security and convenience in user authentication. Unlike password-based authentication methods, it does not require AF to store a private dynamic database, thereby reducing the risk of database breaches [11,25,27]. By leveraging cellular network-based authentication, these systems enable seamless logins for users and offload the burden of authentication-related data storage and maintenance from AFs. Furthermore, integrating AKMA into 5G makes it more secure than SMS OTP, as it does not involve the personal participant of the user, which

decreases the risks of phishing attacks [10]. Additionally, the convenience of AKMA is further enhanced by not requiring users to prepare additional hardware, such as FIDO WebAuthn [26], since cell phones are ubiquitous.

AKMA is currently in its early stages of development and remains an active area of research. 3GPP Technical Report (TR) 33.835 [2] outlines seventeen critical issues (#1–#17) that must be resolved to enable the progression of AKMA. Among these, five key issues (#3, #5, #6, #7, and #16) are specifically related to the protocol layer and present substantial security and privacy challenges, thereby threatening the fundamental integrity of the protocol. While some of these issues can be addressed through minor adjustments to existing protocols, others appear to require significant modifications to the current architecture in order to be effectively resolved.

Communication Security. Addressing Key Issue #6, which pertains to secure communication between UE and AFs, remains one of the most challenging problems in modern network security. Specifically, session keys used for UE-AF communication are accessible to the AAnF. This accessibility introduces potential vulnerabilities, as the AAnF could intercept communications between UEs and AFs. Notably, the AAnF resides within the Home Network but operates outside the 5G core network, lacking the same trust designation as core network components. This distinction further exacerbates security concerns, suggesting heightened risks [29].

Ensuring the confidentiality of session keys for UE-AF communication is therefore paramount, especially in scenarios where the AAnF may be compromised. However, this requirement presents significant challenges, as the trust foundation for UE-AF communication authentication is inherently tied to the AAnF. *If the AAnF is compromised, adversaries could exploit this vulnerability to execute man-in-the-middle attacks, thereby undermining the security of the entire communication scheme.*

Yang et al. [29] proposed the development of AKMA+, which aims to protect communication between UEs and AFs from attackers, including those within the HN. Nevertheless, recent studies [17] have demonstrated that if an adversary is able to compromise the AAnF—as assumed in the AKMA+ threat model—they can impersonate legitimate UEs to AFs. This vulnerability arises as a direct consequence of the centralized trust architecture inherent in AKMA+.

User Privacy. The other critical issues that warrant attention are Key Issue #5 (user privacy) and Key Issue #7 (protection of subscriber personal information in control and data traffic), both of which are intricately linked to the potential exposure of a user's identity and behavioral privacy. Notably, AFs can track the user's SUPI. The SUPI is directly associated with the user's actual identity, which poses a significant risk of privacy leakage. Furthermore, the AAnF has the authority to identify the user's intended application functions, as well as monitor the frequency and timing of their logins. This ability to track login activities could potentially reveal sensitive insights into the user's daily behavioral patterns.

On the other hand, Key Issue #5 simultaneously stipulates that when a network operator seeks to provide authentication and key management services to an application server, it must possess the capability to exchange subscriber-related information. This may enable the application server to identify its users effectively. Consequently, there arises a need for an alternative type of identifier (either permanent or temporary) to facilitate user identification between the 3GPP network and the application server. Additionally, the Mobile Network Operators(MNO) must be able to map this alternative identifier to the permanent identifier within its domain. This dual requirement creates a fundamental contradiction in protocol design: *while there is a necessity to safeguard user privacy, there is also a need to retain the ability to trace a user's true identity.*

In conclusion, it is of paramount importance to enhance the existing AKMA protocol to address critical security and privacy concerns. Firstly, the protocol must be fortified to guarantee secure and confidential communication between the UE and the AF, even in scenarios where a temporary security breach occurs within the HN. Secondly, the AKMA protocol should emphasize the protection of user privacy, ensuring that neither the AF nor a compromised AAnF can correlate a user's account activity with their real identity. At the same time, the HN must possess the capability to trace a user's real identity when necessary.

The preservation of the key advantages inherent in the existing AKMA protocol is of paramount importance. One key advantage of its design is eliminating the need for a private database. This significantly reduces the risk of database breaches, thereby enhancing overall security. Additionally, the protocol does not necessitate the acquisition of additional hardware, as it operates seamlessly on existing infrastructure, such as SIM cards. These features collectively contribute to improving security while also reducing the overhead associated with the dynamic storage requirements of AF.

1.1 Our Contributions

This paper presents an innovative framework for the AKMA scheme, referred to as End-to-End AKMA (E2E-AKMA). The primary objective of E2E-AKMA is to establish session keys for UE-AF communication with robust end-to-end security while simultaneously strengthening privacy protection mechanisms. In particular, this study focuses on scenarios where the AAnF may be compromised, addressing concerns raised by previous works, including [2, 29].

End-to-End Security. The E2E-AKMA framework incor-

The remaining issues outlined in TR 33.835 focus on aspects such as the architectural framework, interface design, API functionality, and regulatory compliance of AKMA. These issues fall outside the scope of this study, as they cannot be effectively resolved through a protocol-centric approach.

porates robust security mechanisms aimed at preventing adversaries from impersonating users or gaining unauthorized access to their accounts to retrieve sensitive information. Importantly, these security measures remain effective even under circumstances where the AAnF on the HN is temporarily compromised. For an adversary to carry out malicious activities successfully, it would be necessary to simultaneously compromise both the UE and the AAnF, or compromise the UE while obtaining physical possession of the SIM card. This framework effectively addresses key issues 6 and 16, while implicitly resolving issue 3 as outlined in 3GPP TR 33.835 [2].

Privacy Enhancement. The E2E-AKMA framework addresses two critical privacy issues (Key Issues #5 and #7 in [2]) through strategic design improvements.

- **Protection against compromised AAnF.** When the AAnF is compromised, it remains unaware of the specific AF or account being accessed by the UE, minimizing exposure of sensitive account information and user behavioral patterns.
- **Prevention of cross-AF tracking.** The framework prevents UEs from being globally identifiable across different AFs, providing robust protection against re-identification. When a UE maintain accounts across multiple AFs, these AFs cannot associate or link the accounts to the same user during registration or login processes, even under AF collusion. Similarly, multiple accounts within a single AF cannot be correlated or traced back to the same UE.
- **Preservation of MNO Functionality**
  The proposed privacy enhancements are designed to ensure that the Mobile Network Operators (MNOs) retain their ability to identify users across the 3GPP network and associated application servers. Specifically, the 5G core network continues to support the mapping of user accounts to their SUPI. This capability enables MNOs to exchange subscriber-related information seamlessly while preserving critical auditing and management functionalities. Such functionalities are essential for maintaining operational efficiency and adhering to regulatory compliance requirements.

Deployability and Usability. Our protocol retains the established advantages of AKMA, particularly in terms of its deployment and operational procedures.

- **Elimination of Confidential Data Repositories**. The E2E-AKMA system eliminates the need for any party to manage large-scale repositories containing sensitive or confidential data. This approach not only mitigates the risk of data breaches but also significantly reduces deployment costs.
- **No Additional Secure Hardware Required.** The implementation of our E2E-AKMA system eliminates the need for additional secure hardware or any modifications to the existing SIM card infrastructure. This design facilitates effortless integration with current systems, offering a practical and cost-efficient solution for strengthening the security of the AKMA scheme.

Our experimental findings, conducted using a real mobile phone and cloud servers, demonstrate that the login time for the E2E-AKMA protocol is 464 milliseconds, representing only a 9.4% overhead compared to the 5G AKMA protocol. Furthermore, in real-world applications, the performance of the protocol can be enhanced further by leveraging hardware optimizations.

1.2 Technique Overview

Our E2E-AKMA protocol is designed based on several key observations and security considerations:

First, leveraging AF certificates for unidirectional authentication. We observe that AFs possess certificates in the 5G infrastructure, enabling UEs to authenticate AFs through TLS. Therefore, we only need to address the challenge of AF authenticating UE. We adopt a *trust on first use* approach where UEs register their public keys with AFs while keeping their private keys protected through distributed storage.

Second, mitigating single point of failure through distributed key management. We recognize that relying solely on AAnF for authentication creates a critical vulnerability - if AAnF is compromised, the entire system's security is at risk. To address this, we split the private key into two parts: one stored at AAnF and another at UE. Crucially, accessing the AAnF's key share requires SIM-based authentication through the 5G AKA channel. This design ensures that:1). If UE is compromised but the SIM card secrets remain secure, the adversary cannot invoke AAnF's private key for signing. 2). If AAnF is compromised, the adversary still lacks the UE's key share, preventing successful signature generation

Third, ensuring privacy against AAnF through cryptographic indistinguishability. To protect user privacy from AAnF, we ensure that for each account, AAnF uses the same public key for the same SUPI. This design prevents AAnF from correlating different public keys to distinct services. To achieve this goal, our solution employs a single signature generated interactively by both the AAnF and UE through our novel *Two-Party Oblivious Signature* (2POS) protocol. Each account is associated with a single public key PK_S, with the UE holding private key SK_1 and the AAnF holding SK_2. The UE authenticates himself to HN via 5G AKA and uses 2POS to generate signatures for AF challenges. The AAnF remains unaware of the generated signature, while the UE does not know SK_2 during this approach. Crucially, each SK_1 is unique to different accounts, whereas SK_2 remains consistent across all accounts for a UE. This ensures the AAnF cannot identify which AF the UE is authenticating to.

The 2POS protocol facilitates the generation of standard ECDSA signatures by distributing the secret keys between the AAnF and the UE. A central design principle of the protocol is that messages originating from the UE are encrypted using homomorphic encryption techniques. This ensures that the AAnF is unable to differentiate between various SK_1 values. Furthermore, since the AAnF does not possess the decryption key, it can not independently generating final signatures.

This mechanism effectively prevents unauthorized signature creation while simultaneously preserving user privacy.

Fourth, achieving unlinkability across AFs. For privacy against AFs, while AAnF uses the same public key for identical SUPIs, UE generates different public keys for each account. From AF's perspective, each account's public key appears independently and randomly generated, preventing cross-service user linking even if multiple AFs collude. More specifically, due to the design of the 2POS protocol, AFs cannot link accounts to the same UE since they cannot derive SK_1 and SK_2 from PK_S and signatures.

Fifth, enabling lawful traceability while preserving privacy. To support regulatory compliance, we have AAnF encrypt the SUPI using the core network's public key and include this ciphertext as part of the account's public identifier forwarded to AF. This design ensures that: 1).AFs cannot decrypt the SUPI, preserving user privacy at the service level. 2). The core network can decrypt and recover the SUPI for lawful interception when necessary. 3) Even if AAnF is malicious and provides incorrect SUPI encryption (e.g., using its own public key), AF's re-randomization process acts as a second layer of encryption, making malicious ciphertexts indistinguishable from random, preventing AAnF from tracking users through this field.

Additionally, to thwart phishing attacks, we hash the identity of AF together with the challenge as the message to be signed. This binds the identity of AF to the challenge, ensuring that the adversary cannot create a malicious AF$'$ to impersonate the legitimate AF, forwarding the challenge obtained from the legitimate AF to UE for signing, thereby launching a phishing attack against UE. This protection is particularly crucial in scenarios where a phishing website AF$'$ mimics a legitimate website AF, and the user mistakenly fails to verify the correct TLS certificate, inadvertently connecting to the malicious service.

1.3 Related Work

Comparison with AKMA+. The AKMA+ protocol proposed by Yang et al. [29] addresses several security and privacy vulnerabilities in the original 5G AKMA protocol. However, compared to our E2E-AKMA approach, AKMA+ exhibits critical limitations in both security and usability aspects.

The primary vulnerability of AKMA+ lies in its persistent dependence on centralized key storage within the AAnF. Specifically, all AKMA anchor keys ($K_{\text{AKMA,i}}$) are stored in the AAnF, which introduces a critical single point of failure. In the event that the AAnF is compromised—a scenario explicitly considered within the threat model—adversaries can gain access to all stored keys. This access enables them to derive application keys (K_{AF}), decrypt previously recorded sessions, and impersonate legitimate users. Furthermore, even if privacy-hardened application keys are employed through Diffie-Hellman key exchange, the compromise of anchor keys would still allow adversaries to successfully impersonate the UE.

AKMA+ requires the AAnF to dynamically generate and manage multiple key-identifier pairs $(K_{AKMA,i}, \text{A-KID}_i)$ for each user to achieve indistinguishability. For k users with n pairs each, the AAnF must store and manage $N = n \times k$ shuffled pairs, significantly increasing storage requirements and operational complexity. Their experimental results show computation costs scaling linearly with n, reaching 78.206 ms for $n = 500$, which becomes problematic at scale.

In contrast, E2E-AKMA eliminates centralized key storage through a distributed trust model using two-party oblivious signatures. Neither the UE nor HN can independently generate valid signatures, removing single points of failure. Our approach achieves true end-to-end security even under the AAnF compromise, while maintaining only 9.4% overhead compared to standard AKMA and requiring no complex key management infrastructure.

	5G AKMA	**AKMA+**	**Ours**
Compromise UE storage to impersonate user (SIM card safe)	○	◑	●
Compromise AAnF to impersonate user	○	◑	●
AAnF without maintaining a secure dynamic database	○	○	●

○: Not satisfied; ●: Satisfied.
◑: Claimed to satisfy this property, but has known attacks.

Table 1: Comparison of three protocols.

Other Related Works. Several studies have addressed privacy concerns in AKMA protocols. Khan et al. [14] analyzed the AKMA protocol's working mechanism and identified linking issues between HN and AF. Their subsequent work [13] introduced a privacy mode for AKMA that achieves unlinkability between HN and AF by routing communication through the UE instead of direct HN-AF connection. However, their scheme requires group operations on SIM card secret keys, preventing deployment on existing devices. Yang et al. [28] conducted a formal analysis of 5G AKMA focusing on authentication and key management, acknowledging the HN-AF privacy issues we address but without providing solutions. Akman et al. [4] proposed Privacy-Enhanced AKMA considering malicious AF scenarios where adversarial AFs intercept UE communications to request keys from HN. They also addressed AF collusion for user account linking by introducing new identifiers to replace Generic Public Subscription Identifiers, achieving AF-unlinkability. However, their scheme still allows HN to determine which AF users are accessing, leaving the HN-side privacy issue unresolved.

Structurally, E2E-AKMA functions as a delegated authentication system, analogous to Single Sign-On (SSO) protocols like OIDC [21] and SAML [20]. In this model, the UE, AF, and HN correspond to the User, Relying Party (RP), and Identity Provider (IdP), respectively. However, unlike typical web-based SSO, E2E-AKMA is rooted in the mobile network trust infrastructure. It leverages the existing 5G AKA protocol and SIM-based credentials to bootstrap trust, enabling the HN (IdP) to assist in user authentication while ensuring end-to-end security and privacy.

2 AKMA Protocol Overview

2.1 Original AKMA Design Philosophy

The 5G AKMA protocol, as specified in 3GPP TS 33.535 [3], adopts a delegated authentication architecture that enables AFs to leverage mobile network operator authentication capabilities without requiring direct credential management.

Core Design Principles. As illustrated in Figure 1, the AKMA protocol follows a three-party model involving UE, HN, and AF. The design leverages the existing 5G authentication infrastructure through a centralized AAnF that serves as the trust anchor for key derivation and distribution.

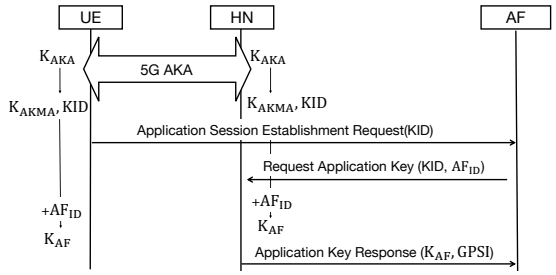

Figure 1: 5G AKMA Protocol Overview

The protocol workflow proceeds as follows: (1) After successful 5G AKA primary authentication, both UE and AAnF derive the AKMA anchor key K_{AKMA} and its identifier (A-KID); (2) When accessing an AF, the UE presents the A-KID to initiate session establishment; (3) The AF requests the corresponding application key K_{AF} from the AAnF via the Network Exposure Function (NEF); (4) The AAnF derives K_{AF} using K_{AKMA} and AF identifier (AF_ID), then provides it to the AF along with user identity information (SUPI/GPSI); (5) Both UE and AF independently derive the same K_{AF} for secure communication.

Architectural Benefits. This centralized approach offers several advantages: elimination of complex credential databases at AFs, seamless integration with existing 5G infrastructure, and reduced authentication overhead for resource-constrained IoT devices. The protocol enables "single sign-on" functionality where one primary authentication supports multiple application sessions.

2.2 Security and Privacy Challenges

However, the centralized trust model introduces critical vulnerabilities that 3GPP TR 33.835 [2] identifies through seventeen key issues. Five protocol-layer issues pose fundamental threats to AKMA's security foundation.

Security Objectives

Mutual authentication between UE and anchor function (Key Issue #3). The original AKMA lacks robust mutual authentication between UE and AAnF. While 5G AKA provides UE-network authentication, the AAnF operates outside the 5G core network boundary, creating trust gaps. Without proper authentication, unauthorized UEs may access AKMA services, while fake anchor functions can communicate with legitimate UEs, leading to privacy exposure and potential attacks.

Secure communication between UE and application server (Key Issue #6). The AAnF's central role in key derivation creates a single point of failure. All application keys K_{AF} are derived from K_{AKMA} stored at the AAnF, enabling potential interception of UE-AF communications when the AAnF is compromised. The current design provides no forward secrecy guarantees, as compromised anchor keys expose all past and future sessions.

Application key freshness of AKMA (Key Issue #16). AKMA's key derivation mechanism lacks proper freshness guarantees. Application keys derived using static parameters (K_{AKMA}, AF_{ID}) remain constant across multiple sessions between the same UE-AF pair, enabling replay attacks where attackers who obtain a previously used K_{AF} can masquerade as legitimate UEs towards Application Functions.

Privacy Objectives

User privacy (Key Issue #5). The protocol directly exposes Subscription Permanent Identifiers (SUPI) to Application Functions, creating significant privacy risks. The SUPI represents sensitive information that can be exploited to identify and track individual subscribers, while operators need SUPI access for service provision, creating a fundamental tension between privacy and functionality.

Protecting subscriber's personal information in control and data traffic (Key Issue #7). The A-KID serves as a persistent identifier that enables both AAnFs and AFs to track user authentication behaviors. Privacy-sensitive content in control and data traffic messages can be exploited by attackers to identify subscribers through correlation attacks, while the leakage of AKMA key identifiers enables linking users across different network sessions.

3 Modeling Security and Privacy

3.1 Security Assumptions and Threat Model

The design of robust AKMA systems requires careful consideration of which participants may be compromised in practical

deployments. Drawing from AKMA+ [29], we identify the primary threat vectors and establish our security assumptions:

Threat Vectors. We consider three main categories of threats: (1) *UE threats* including malware infiltration and malicious UEs attacking other entities. Note that while UEs operate in general-purpose software environments that may not provide secure key storage, the SIM card is typically regarded as trusted hardware capable of securely storing AKA keys and establishing authenticated relationships with the HN; (2) *AAnF compromise* where AAnFs may be malicious due to deployment outside secure 5G core networks; and (3) *AF threats* including data leakage and collusion between multiple AFs to correlate user behaviors.

HN as a Communication Facilitator. In accordance with 3GPP TS 33.501 [1] and AKMA+ [29], we assume that core 5G network functions (UDM, AUSF, NEF) remain secure within the protected core network perimeter and reliably provide authentic SUPI information. To simplify our security analysis, we further assume that the HN does not directly participate in application-layer key derivation. Instead, the HN functions as a secure intermediary that facilitates communication while preserving end-to-end security properties between UEs and AAnFs.

Untrusted AAnF. In contrast to core network functions, we assume that AAnFs are potentially malicious. This assumption arises from the architectural reality that AAnFs may be deployed outside the secure perimeter of the 5G core network, rendering them more vulnerable to compromise [29]. Consequently, malicious AAnFs could inject messages that appear cryptographically valid to both UEs and AFs. Additionally, they may collect and expose sensitive user privacy information, such as SUPIs and access patterns, exploit their privileged position in key management to perform man-in-the-middle attacks, and collaborate with other network entities to undermine user privacy and security. Unlike the approach taken in [29], we allow for the possibility that an AAnF may not adhere to established protocols. This consideration is based on the assumption that an adversary capable of compromising an AAnF might gain access to its authentication secrets, effectively enabling them to impersonate a legitimate AAnF within the system. As a result, the attacks we consider are more realistic and reflective of potential real-world threats.

Secure Communication Channels. The proposed protocol operates under the assumption that authenticated and confidential communication channels exist between the participating entities. Specifically:

(1) Before the protocol is executed, UEs are provisioned with SIM cards issued by their respective HNs. These HNs possess comprehensive knowledge of the SIM card contents, including the SUPI, which is integral to AKMA computations. Utilizing the 5G AKA protocols, UEs and HNs establish bidirectionally authenticated secure channels that ensure both mutual authentication and confidentiality.

(2) It is assumed that secure communication channels exist between the HN core network and the AAnF. Based on the assumptions outlined in points (1) and (2), it is further presumed that UEs and AAnFs can securely communicate in the absence of compromise. Additionally, AAnFs are assumed to be capable of identifying the SUPI of the UEs they interact with, as this identifier is forwarded by the HN.

(3) UEs authenticate AFs by verifying their digital certificates, subsequently establishing secure communication channels using TLS. This process achieves unilateral authentication, wherein the UE authenticates the AF, while also fulfilling the security properties of Authenticated and Confidential Channel Establishment (ACCE) [12, 15].

These assumptions are substantiated by the widespread deployment and rigorous security analysis of 5G AKA protocols [1] and TLS protocols within existing 5G infrastructure.

3.2 Syntax

Building on this threat and communication model, we define the AKMA protocol as a three-party system involving UE, AAnF (facilitated by HN), and AF. The protocol enables UEs and AFs to establish session keys without revealing these keys to the AAnF, while ensuring the AAnF cannot determine which AFs users are accessing.

The AKMA protocol consists of four algorithms: initialization, registration, login, and tracing, denoted by Init, Reg, Login, and Trace. In practical deployments, AAnFs maintain public-private key pairs (PK_{AAnF}, SK_{AAnF}), all participants possess the core network's public key PK_{core}, AFs possess public keys of all AAnFs $\mathcal{PK}_{AAnF}$, and AAnFs securely store master keys K_{AAnF} for key derivation. The Init algorithm generates long-term public parameters including algebraic group parameters, while Reg supports multiple account creation, Login enables multiple authentication sessions per account, and Trace allows the core network to recover SUPIs from AF databases.

Definition 1 (AKMA). *An Authentication and Key Management for Applications (AKMA) scheme is a tuple of interactive protocols,* $\mathsf{AKMA} = (\mathsf{Init}, \mathsf{Reg}, \mathsf{Login}, \mathsf{Trace})$*:*

- $PP_{AKMA} \leftarrow \mathsf{Init}\big(UE(\emptyset), AAnF(SUPI, SK_{AAnF}, K_{AAnF}, PK_{core}), AF(PK_{AAnF}, PK_{core})\big)$: *The participants input their respective existing parameters, including the core network's public key, and the three parties jointly generate the public parameter* PP_{AKMA}*.*
- $(K_{UE}, \perp, DB'_{AF}) \leftarrow \mathsf{Reg}\big(UE(\emptyset), AAnF(SUPI, SK_{AAnF}, K_{AAnF}, PK_{core}), AF(PK_{AAnF}, PK_{core}, DB_{AF})\big)$: *UE outputs* K_{UE} *for the* Login *phase. AAnF takes SUPI of the UE,* K_{AAnF}*, and* PK_{core} *as input and outputs* $\perp$*. AF stores UE registration data in* DB_{AF}*.*
- $(ssk, \perp, ssk) \leftarrow \mathsf{Login}\big(UE(K_{UE}), AAnF(SUPI, K_{AAnF}, PK_{core}), AF(DB_{AF})\big)$: *UE inputs* K_{UE}*, AAnF inputs SUPI,* K_{AAnF}*, and* PK_{core}*, AF inputs* DB_{AF}*. UE and AF output a shared session key ssk,*

while AAnF outputs $\perp$.

- *(SUPI)*$\leftarrow$ Trace*(DB_{AF}, SK_{core}): The core network inputs the AF's database DB_{AF} and its private key SK_{core}, and outputs the corresponding SUPI to enable user traceability for legitimate purposes.*

3.3 Security Model

Building upon the security assumptions and threat model established earlier, we now present a formal cryptographic security model that captures the unique challenges of AKMA protocols under malicious AAnF assumptions. Our model extends traditional authenticated key exchange frameworks to address the specific architectural constraints and privacy requirements of 5G networks.

Modeling Rationale. The formal model must reconcile several competing requirements: (1) *Architectural fidelity* - accurately reflecting 5G network structure where core functions (HN) are trusted but edge functions (AAnF) may be compromised; (2) *Cryptographic rigor* - providing precise security definitions amenable to formal analysis; and (3) *Privacy preservation* - capturing unlinkability requirements against both malicious AAnFs and colluding AFs.

Protocol Participants and Computational Model. We conceptualize the AKMA ecosystem as a network comprising three distinct types of entities, each represented by multiple parties.

- *User Equipment (UE):* Possesses SIM card with long-term key shared with its home network. Executes registration and authentication protocols.
- *AKMA Anchor Function (AAnF):* Potentially malicious entity responsible for key derivation and user-AF authentication. May be deployed outside secure network perimeter.
- *Application Function (AF):* Service provider maintaining user accounts and session states. May collude with other AFs for user tracking.

Each party may have multiple instances, modeled as separate Turing machines $\pi_{Role}^{i,j}$ where $Role \in \{\mathcal{UE}, \mathcal{AANF}, \mathcal{AF}\}$ (note: HN only relays messages), i denotes the account/registration index, and j denotes the session index ($j = 0$ for registration, $j > 0$ for authentication sessions).

To optimize our model, we have designed the 5G Core Network to handle user credentials and SUPI information effectively. Within the framework of our protocol, the 5G Core Network primarily serves as a secure message relay. Except in specific scenarios where tracking user activity is explicitly required, it does not perform computational tasks related to the protocol in routine operations. Consequently, the 5G Core Network is not categorized as a Turing machine.

Communication Model. The model reflects 5G network communication patterns:

UE - AAnF Channel: The 5G AKA protocol ensures mutual authentication and confidentiality between communicating parties. However, it does not provide forward secrecy. This means that adversaries are unable to intercept encrypted communication or impersonate any of the involved entities unless they compromise one of the parties. Nonetheless, the absence of forward secrecy in 5G AKA implies that if the long-term keys of the UE are compromised, it becomes possible to decrypt historical communication transcripts. Consequently, these transcripts can be recovered in the event of long-term key compromise.

UE-AF Channel: The TLS protocol ensures server authentication and ACCE security, allowing the UE to verify the certificates provided by the AF. This guarantees that adversaries cannot intercept encrypted communications or impersonate the AF to the UE. However, adversaries may still have the ability to impersonate the UE to the AF. Furthermore, the forward security property of TLS with ECDHE handshake ensures that the compromise of long-term cryptographic keys does not lead to the exposure of previously transmitted data or historical.

AAnF-AF Channel: Although AAnF-AF communications typically utilize secure network protocols, in our protocol, this particular aspect of communication is not applicable. Therefore, we do not make any assumptions regarding its usage.

Adversarial Capabilities. Adversary $\mathcal{A}$ is a PPT algorithm with access to oracles reflecting realistic attack capabilities:

- *Protocol Initiation:* Adversaries can initiate registration and authentication sessions between UE instances and AFs of their choice.
- *UE Corruption:* Full corruption returns UE's long-term keys and all historical transcripts, while compromise returns only derived keys without long-term secrets.
- *AAnF Access:* Since AAnFs are assumed inherently malicious, adversaries have complete access to all AAnF state including key derivation materials and transcripts.
- *AF Corruption:* Full corruption returns AF's complete database and session states, while breach returns only user databases without control.
- *Session Key Revelation:* Adversaries can obtain session keys for specific protocol instances.
- *Message Injection:* Adversaries can send arbitrary messages to honest instances on behalf of corrupted parties.

Security Properties. We define security properties to formalize the requirements identified in our threat analysis.

Authentication. This property ensures that honest UE and AF instances can only accept protocol execution with their intended partners. We use a "Trust on First Use" model where initial registration establishes authentic binding between UE and AF, and subsequent authentications must preserve this binding even against malicious AAnFs. The adversary wins if they can cause honest instances to accept with incorrect partners or without unique partnerships.

Key-Indistinguishability. This property guarantees that session keys derived by honest UE-AF pairs are computationally indistinguishable from random keys, even when adversaries

have access to malicious AAnFs and can corrupt non-target parties. The adversary is challenged to distinguish between real and random session keys under carefully defined freshness conditions that ensure the target session remains secure despite AAnF compromise.

AF-Unlinkability. This property ensures that malicious AFs cannot link user accounts across different services. Within this framework, even if two honest UE devices are chosen, and their previously associated registration account information along with their SUPI are already known, a malicious AF is still unable to determine which honest UE is engaged in the registration process.

*AAnF-Unlinkability and AAnF-Indistinguishability.*These properties are essential for safeguarding user privacy in the context of 5G deployments, particularly against one of the most plausible threats: malicious AAnFs attempting to construct user profiles across multiple services. The principle of *AAnF Unlinkability* ensures that a single UE can register multiple accounts through its HN without the HN being able to determine which account corresponds to which AF. In the experimental setup, this property is characterized by allowing an adversary to select two honest UE instances associated with the same UE and two honest AF entities, while ensuring that the adversary is unable to discern which UE instance interacts with which AF entity. Furthermore, the principle of *AAnF Indistinguishability* ensures that even if a malicious AAnF has access to the registration information stored on the AF, it cannot associate or link the SUPI corresponding to the account. These two properties guarantees a robust level of privacy by preventing adversaries from inferring sensitive identity information.

Traceability. This property ensures that the HN is capable of mapping public account identifiers used by AFs to user SUPIs for legitimate purposes, such as lawful interception, billing, and emergency services. Our model for traceability operates under the assumption that all participating entities (e.g., AAnF, AFs, etc.) adhere to the specified protocols. This assumption is reasonable because, unlike other properties such as privacy, a breach of traceability does not immediately result in information leakage. Instead, it is sufficient to retain the capability to hold entities accountable retrospectively when failures in traceability are identified. In such cases, regulatory authorities can initiate out-of-band investigative procedures, including equipment audits, compliance verification, and the imposition of operational restrictions.

This framework enables rigorous security proofs while capturing the unique challenges of 5G network architectures. The formal oracle definitions and security experiments are provided in Appendix B, and the subsequent protocol design will demonstrate how cryptographic techniques can achieve these security properties under our threat model.

3.4 Correspondence to 3GPP Key Issues.

Our security models directly address the key issues identified in 3GPP's AKMA security analysis [2]:

- **Key Issue #3 - Mutual authentication between UE and anchor function:** In 5G AKMA, UE does not communicate directly with AAnF; messages pass through the AF, causing no explicit mutual authentication between UE and AAnF. In our framework, UE and AAnF communicate directly via the HN. UE–HN authentication uses the AKA protocol, and secure channels exist between AAnF and HN. Thus, mutual authentication between UE and AAnF is naturally achieved without extra mechanisms.
- **Key Issue #6 - Secure Communication Between UE and Application Server:** Our *Authentication* property ensures mutual authentication between legitimate UE and AF instances with their intended partners, resisting man-in-the-middle attacks even if the AAnF is compromised. Additionally, our *Key-Indistinguishability* property guarantees session keys generated by legitimate UE-AF pairs are computationally indistinguishable from random keys, ensuring compromised anchor keys do not jeopardize session confidentiality.
- **Key Issue #16 - Application Key Freshness in AKMA:** Our proposed *Key-Indistinguishability* properties provide forward security. In contrast to the current AKMA design, where application keys derived from static parameters (K_{AKMA}, AF_{ID}) remain constant across sessions, our security framework ensures enhanced protection. Specifically, even if an adversary compromises a participant after previous sessions have concluded, the session keys from those earlier sessions remain indistinguishable, thereby preserving their confidentiality and integrity.
- **Key Issue #5 - User privacy:** Our proposed properties, *AF-Unlinkability*, *AAnF-Unlinkability*, and *AAnF-Indistinguishability*, address key user privacy concerns. These ensure SUPIs are not exposed to AFs and prevent colluding AFs from identifying, tracking, or profiling subscribers across services. This resolves the conflict between user privacy and system functionality. Meanwhile, our *Traceability* property allows the trusted core network to link public account identifiers used by AFs back to user SUPIs. This balance preserves anonymity while enabling essential network operations.
- **Key Issue #7 - Protecting subscriber's personal information in control and data traffic:** Our *AF-Unlinkability* properties specifically prevent the exploitation of persistent identifiers like A-KID for tracking user authentication behaviors. These properties ensure that privacy-sensitive content in control and data traffic cannot be used for correlation attacks, and prevent the leakage of AKMA key identifiers that could enable linking users across different network sessions.

4 Two-Party Oblivious Signature

In this section, we introduce a new and innovative concept called the two-party oblivious signature, abbreviated as 2POS. 2POS is similar to the two-party threshold signature, but it differs in that only one party has access to the final signature. A more important difference is that P_1 possesses multiple secret keys while P_2 only has one in our scenario. Hence 2POS additionally ensures that P_2 cannot distinguish which secret key P_1 is used to generate a signature, a property we refer to as P_2-indistinguishability. This new feature is essential for the identity unlinkability of the E2E-AKMA protocol.

4.1 Definition of 2POS

Syntax. In the 2POS scheme, there are two participants called P_1 and P_2, the protocol is divided into four phases: Setup, Key Generation, Signing, and Verification. In the Setup phase, P_1 and P_2 jointly generate the public parameters of 2POS, and P_2 generates its long-term SK_2 and PK_2. During the Key Generation, P_1 combines its key with the long-term PK_2 of P_2 to produce the 2POS public key. In the Signing phase, P_1 interacts with P_2 to sign a message chosen by P_1, and outputs the signature, while P_2 produces no output. The verification process follows standard signature verification.

Definition 2. *A Two-Party Oblivious Signature (2POS) scheme denoted as* 2POS=*(*Setup*,* KeyGen*,* Sign*,* Ver*) consists of three interactive protocols and a* PPT *algorithm.*

- $((PP_{2POS}, PK_2/\perp), (PP_{2POS}, PK_2, SK_2)) \leftarrow$ Setup $(P_1(\emptyset), P_2(\emptyset))$*: Generate public parameter and long-term key.*
- $(SK_1, PK_S) \leftarrow \mathsf{KeyGen}(PK_2)$*: The key generation phase is locally computed by P_1, taking PK_2 as input and outputting the private key share of P_1 and the 2POS signature public key PK_S.*
- $(\sigma_{2POS}, \perp) \leftarrow \mathsf{Sign}(P_1(m, SK_1), P_2(m, SK_2))$*: P_1 and P_2 input their respective secret shares, P_1 outputs the signature.*
- $0/1 \leftarrow \mathsf{Ver}(m, \sigma_{2POS}, PK_S)$*: Identical to the standard signature verification. Input m, σ_{2POS}, PK_S, and output the verification result.*

4.2 Security

Now we define the security of 2POS. Since the security requirements of P_1 and P_2 are inconsistent in the 2POS protocol, we need to consider the cases of P_1 being malicious and P_2 being malicious, respectively. The detailed definition of the 2POS security model and the security proof are provided in our full version [16].

P_1-Unforgeability. Informally, P_1-Unforgeability ensures that even with access to the private key share of P_1, the adversary cannot forge a signature without interacting with the honest P_2. This definition corresponds to the UE being honest during the registration phase of E2E-AKMA. This approach is as follows: the challenger generates keys honestly and gives the adversary $\mathcal{A}$ the private key share of P_1, the long-term public key of P_2, and the public key of 2POS. The adversary interacts with P_2, selects a message m to produce the signature, and must produce a signature for a new message m^* not previously queried.

P_2-Unforgeability. Informally, P_2-Unforgeability ensures that even if the adversary $\mathcal{A}$ acts as P_2, as long as P_1 is honest, no signature can be obtained. Unlike P_1-Unforgeability, P_2-Unforgeability allows $\mathcal{A}$ to act maliciously during the Setup phase of 2POS, capturing that even if HN misbehaves during the registration phase of E2E-AKMA, valid signatures cannot be forged. The approach is as follows: $\mathcal{A}$ interacts with the challenger $\mathcal{C}$ to complete the Setup phase. $\mathcal{C}$ generates the public key for $2POS$ and provides it to $\mathcal{A}$. $\mathcal{A}$ may continue interacting with the $\mathcal{C}$ to perform the signing phase, but it cannot obtain any valid signature.

P_2-Indistinguishability. Informally, P_2-Indistinguishability ensures that a malicious P_2 cannot distinguish which P_1 party it is interacting with. This approach is as follows, the challenger selects two honest P_1 party, Π_2^0 and Π_2^1, and chooses $b \leftarrow \{0,1\}$. P_2 performs Setup and Sign with these two parties, and guesses the value of b chosen by the adversary.

Untraceable. Informally, Untraceable ensures that the 2POS public key appears random to an external adversary, preventing identification of the P_1-P_2 pair that generated it. This approach is as follows: $\mathcal{A}$ selects two groups of honest P_1 and P_2 parties for Setup, multiple KeyGen, and Sign operations. In the test session, the challenger selects $b \leftarrow \{0,1\}$ to decide which group performs KeyGen and Sign. Finally, $\mathcal{A}$ succeeds if it correctly guesses the value of b.

4.3 Construction of 2POS

The 2POS signature scheme represents an advancement over traditional threshold ECDSA, incorporating two enhancements. First, P_2 cannot independently generate a valid signature, even after arbitrary interactions with P_1. Second, P_2's computational perspective can be distinguished based on P_1's distinct secret keys. Building upon these principles, we design the 2POS scheme using Paillier encryption [22] as a cryptographic component. The roles of P_1 and P_2 are deliberately designed to be asymmetric. Specifically, the messages associated with sk_1, which are transmitted by P_1, are encrypted using a homomorphic encryption scheme, such as the Paillier cryptosystem. The confidentiality ensured by this encryption guarantees that the information visible to P_2 remains indistinguishable, even when different secret keys sk_1 are utilized. Moreover, since P_2's functionality is restricted to performing homomorphic operations on the encrypted data, it is inherently incapable of deriving the final signature. This limitation arises from P_2's lack of access to the decryption key, which is essential for deriving the final signature.

The ideal zero-knowledge functionality is denoted by $\mathcal{F}_{\text{zk}}$,

while the committed non-interactive zero-knowledge functionality is represented as $\mathcal{F}_{\text{com-zk}}$. The encryption and decryption processes of the Paillier cryptosystem are symbolized by Enc and Dec, respectively. Detailed explanations of the ideal functionalities, as well as the Paillier encryption scheme [22], can be found in Appendix. The construction of the 2POS protocol is outlined as follows.

Setup. P_2 independently selects $x_2 \leftarrow\$ \mathbb{Z}_q$ as their long-term private key, denoted as SK_2, and computes their corresponding long-term public key PK_2 as $Q_2 = x_2 \cdot G$.

Subsequently, P_2 transmits the public key PK_2 to P_1, accompanied by a zero-knowledge proof π that demonstrates possession of the associated private key SK_2. Upon receiving this information, P_1 verifies the validity of the proof and securely stores PK_2 locally.

Key Generation. P_1 locally selects $x_1 \leftarrow\$ \mathbb{Z}_q$ as the private key share SK_1, and calculates the 2POS public key $Q = x_1 \cdot Q_2$ based on the long-term public key Q_2 of P_2.

Sign. At the beginning of the signing phase, party P_1 has x_1,Q and the message m, P_2 has x_2.

- For the first message, P_1 chooses $k_1 \leftarrow\$ \mathbb{Z}_q$, computes $R_1 = k_1 \cdot G$ and sends (com-prove, sid||1, R_1, k_1) to $\mathcal{F}_{\text{com-zk}}^{R_{DL}}$.
- For the second message, P_2 receives (proof-receipt, sid||1) from $\mathcal{F}_{\text{com-zk}}^{R_{DL}}$, chooses a random $k_2 \leftarrow \mathbb{Z}_q$, computes $R_2 = k_2 \cdot G$ and sends (prove, sid||2, R_2, k_2) to $\mathcal{F}_{\text{zk}}^{R_{DL}}$.
- For the third message, P_1 receives (proof, sid||2, R_2) form $\mathcal{F}_{\text{zk}}^{R_{DL}}$, generates a new Paillier key-pair (pk, sk) and computes $C_{HE} = \text{Enc}_{pk}(x_1)$. P_1 sends C_{HE}, pk to the P_2 and sends the (decom-proof, sid||1) to $\mathcal{F}_{\text{com-zk}}^{R_{DL}}$ for decommit.
- For the fourth message, P_2 receives (decom-proof, sid||1, R_1) from $\mathcal{F}_{\text{com-zk}}^{R_{DL}}$ and computes $R = k_2 \cdot R_1$ and $m' = H(m)$ for the hash function $H(\cdot)$. Denote $R = (r_x, r_y)$, $r = r_x$ mod q. P_2 chooses $\rho \leftarrow\$ \mathbb{Z}_{q^2}$, $\tilde{r} \in \mathbb{Z}_N^*$ (verifying that gcd($\tilde{r}$,N)=1), and computes $C_1 = \text{Enc}_{pk}(\rho \cdot q + [k_2^{-1} \cdot m' \bmod q; \tilde{r}])$. Then P_2 computes $v = k_2^{-1} \cdot r \cdot x_2 \bmod q$, $C_2 = v \odot C_{HE}$ and $C_3 = C_1 \oplus C_2$ and sends C_3 to P_1.
- Finally, P_1 computes r in the same way as P_2 and $s' = \text{Dec}_{sk}(C_3)$, $s'' = k_1^{-1} \cdot s' \bmod q$. P_1 sets $s = min\{s'', q - s''\}$, ensuring that the signature is always the smaller of the two possible values. P_1 verifies (r, s) is a valid signature with public key Q. If yes it outputs the signature (r, s). In this way, P_1 and P_2 can jointly produce a standard ECDSA signature of Q.

Verification. The verification algorithm is the same as the standard ECDSA verification algorithm.

We included the theorems on signature security, along with the corresponding security analysis, in our full version [16].

5 E2E-AKMA

Following the high-level construction outlined in Section 1.2, we now present the detailed cryptographic specification of E2E-AKMA.

5.1 Construction Details

Before the commencement of the protocol, each AAnF must independently generate a unique public-private key pair, denoted as (PK_{AAnF}, SK_{AAnF}), which is specifically designated for signature operations. This cryptographic key pair is integral to guaranteeing the authenticity of the registration process for a UE, as it ensures that the process has been performed using the services provided by the corresponding AAnF. each AF is provisioned with the public keys of all AAnFs, collectively represented as PK_{AAnF}. Moreover, AAnFs securely store master keys, denoted as K_{AAnF}, which are employed for cryptographic key derivation processes.

Building upon this foundational framework, we propose the design of the E2E-AKMA as follows.

The Init Phase. Generate the parameters of E2E-AKMA, including those for signature and encryption algorithms.

The Registration Phase.

1. In the registration phase, after interacting with the $AAnF$, the UE registers an account with the AF and logs in. After UE requests registration with AF, AF sends a challenge C to UE. AAnF uses the master key K_{AAnF} with $SUPI$ to generate SK_2 for 2POS and calculates PK_2, while providing a zero-knowledge proof π of knowing the SK_2 corresponding to PK_2. After verifying π, UE executes the key generation phase of 2POS, $(SK_1, PK_S) \leftarrow 2POS.\text{KeyGen}(PK_2)$. SK_1 serves as K_{UE} in E2E-AKMA, PK_S serves as the public key of the account registered by the UE.
2. UE generates an account ID UID for this registration, which serves as a part of the username. UE computes the hash value h_1 for id_{AF}, C, PK_S, and UID, requests AAnF to use its private key to sign h_1, obtaining σ_{AAnF}. This illustrates that the key generation process is carried out through interaction with the AAnF. Additionally, AAnF encrypts the SUPI using the core network's public key via a Re-randomizable encryption scheme (described in Appendix A): $tag \leftarrow \text{Enc}(PK_{core}, SUPI)$, and includes this encrypted tag in the signature: $\sigma_{AAnF} \leftarrow \text{Sig}(SK_{AAnF}, h_1||tag)$.
3. The UE computes the hash h_2 to id_{AF} and C, UE and AAnF execute the signing phase of 2POS to generate a signature σ_{2POS}, which can be verified by PK_S. In the implementation, steps 2 and 3 can be combined to reduce the communication rounds.
4. The UE sends the two signatures σ_{AAnF}, σ_{2POS}, UID, and PK_S to the AF. AF verifies the signatures by computing $h_1 \leftarrow H(id_{AF}||C||PK_S||UID)$ and $h_2 \leftarrow H(id_{AF}||C)$, then checking $\text{Ver}(h_1||tag, \sigma_{AAnF}, PK_{AAnF}) = 1$ and $\text{Ver}(h_2, \sigma_{2POS}, PK_S) = 1$. Upon successful verification, AF performs a critical security step: it re-randomizes the encrypted SUPI tag to obtain $tag' \leftarrow \text{Re-randomize}(tag)$. This re-randomization acts as a second layer of encryption, ensuring that even if AAnF provides malicious ciphertexts, they become indistinguishable from random, preventing AAnF from tracking users through this field.

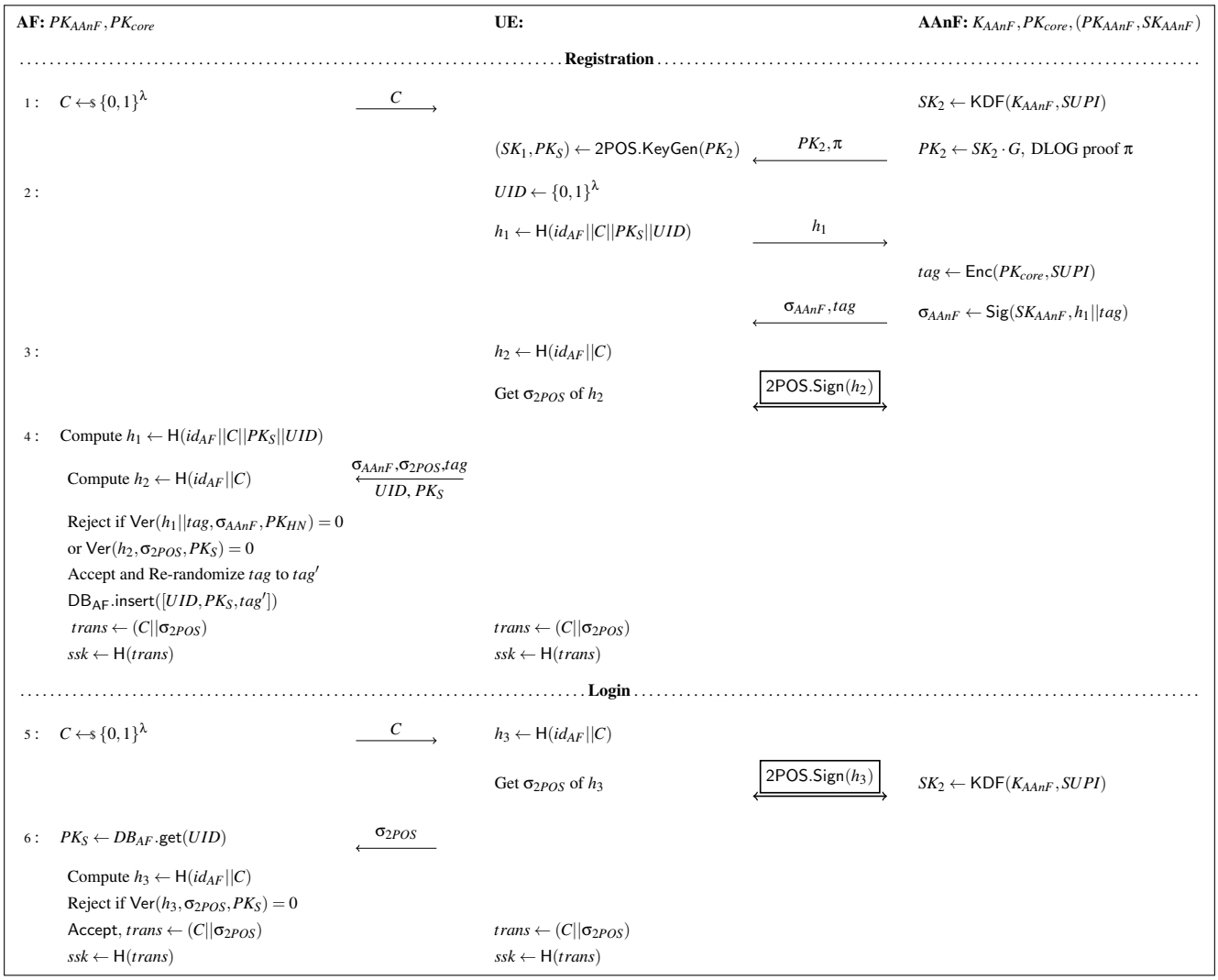

Figure 2: The E2E-AKMA Construction. The communication between the UE and the AAnF is facilitated through the 5G AKA protocol, as well as being relayed by the HN. The interaction between the UE and the AF is secured using the TLS protocol.

AF then stores the UID, PK_S, and re-randomized tag' in its database for the login phase, and computes the session key using the transcript.

The Login Phase.

5. The login phase is similar to the registration phase. First, the UE submits a login request to the AF on behalf of UID, and the AF sends a challenge C to the UE. UE generates $h_3 \leftarrow H(id_{AF}||C)$ and performs the 2POS signing phase with AAnF as in the registration step 3 to obtain σ_{2POS}.
6. UE sends the σ_{2POS} to the AF. AF retrieves PK_S using the UID and computes the hash $h_3 \leftarrow H(id_{AF}||C)$ and verifies the signature $\mathrm{Ver}(h_3, \sigma_{2POS}, PK_S) = 1$. Upon successful verification, the UE and AF utilize the hash value of their transcript as the session key $ssk \leftarrow H(\mathrm{trans})$ where $\mathrm{trans} \leftarrow (C||\sigma_{2POS})$.

The Trace Algorithm. The trace algorithm enables the core network to retrieve the true SUPI of a registered user from the AF's database for legitimate purposes, such as law enforcement compliance. Using its private key SK_{core} and the re-randomized encrypted tag tag' from the AF's database, the core network decrypts the tag as: $SUPI \leftarrow \mathrm{Dec}(SK_{\mathrm{core}}, tag')$ recovering the user's true identity.

5.2 Security Analysis.

We present the underlying intuition of security. A more detailed security proof is in our full version [16].

Theorem 1 (Authentication)**.** *The E2E-AKMA satisfies the Authentication as defined in Definition 3, assuming 2POS is correct, AFs avoid identical challenges, and 2POS is P_1-UNF and P_2-UNF.*

Proof sketch: The protocol ensures mutual authentication between UE and AF through the 2POS signature scheme. Honest execution is guaranteed by the correctness of 2POS, while adversarial attempts to impersonate either party would require forging signatures, which contradicts the unforgeability properties (P_1-UNF and P_2-UNF) of the 2POS scheme. The binding of AF identity to challenges prevents replay attacks across different services.

Theorem 2 (Key-indistinguishability)**.** *Consider the hash function as a random oracle. The E2E-AKMA satisfies the Key-indistinguishability as defined in Definition 4, assuming 2POS is correct, AF avoids identical challenges, 2POS is P_1-UNF and P_2-UNF, RO outputs differ for distinct inputs, and the guessing the challenge is negligible.*

Proof sketch: Session keys are derived from protocol transcripts using a random oracle (hash function). Since the adversary cannot predict the random challenge C generated by honest AFs, and the 2POS signature σ_{2POS} appears random without knowledge of the signing keys, the resulting session key $ssk = H(C||\sigma_{2POS})$ is indistinguishable from random to any external observer.

Theorem 3 (AAnF-unlinkable)**.** *The E2E-AKMA satisfies the AAnF-Unlinkable as defined in Definition 6 assuming the 2POS satisfies P_2-IND, the hash function H is modeled as a random oracle, and the re-randomizable encryption scheme satisfies malformed ciphertext indistinguishability.*

Proof sketch: The AAnF (acting as P_2 in 2POS) cannot distinguish which public key PK_S a UE is using across different authentication sessions. This follows directly from the P_2-IND property of 2POS, which ensures that P_2's view during the signing protocol reveals no information about the specific key being used by P_1 (the UE).

Theorem 4 (AAnF-indistinguishability)**.** *The E2E-AKMA satisfies the AAnF-Indistinguishability property as defined in Definition 7, assuming the 2POS scheme satisfies the Untraceable property and the re-randomizable encryption scheme satisfies malformed ciphertext indistinguishability.*

Proof sketch: The AAnF is unable to associate the SUPI with the accounts on AF due to the limitations of the information accessible through AF's public data. Specifically, the *Account UIDs* and *Public Keys (PK_S)* appear as random values from the perspective of AAnF, thereby preventing any direct correlation with specific UE identities. Moreover, the *Re-randomized Encrypted Tags (tag')* are re-randomized by AF in such a manner that they remain unlinkable to their original encrypted counterparts, ensuring enhanced privacy and security.

Theorem 5 (AF-unlinkable)**.** *Consider the KDF as a random oracle. The E2E-AKMA satisfies the AF-Unlinkable as defined in Definition 5 assuming the 2POS is Untraceable and the re-randomizable encryption scheme is IND-CPA secure.*

Proof sketch: Different AFs cannot link accounts belonging to the same UE, even through collusion. This property stems from the Untraceable property of 2POS, which ensures that multiple public keys PK_S generated by the same UE for different AFs appear completely independent and unlinkable to any external observer, including the AFs themselves.

Theorem 6 (Traceability)**.** *The E2E-AKMA satisfies the Traceability property as defined in Definition 8, assuming the signature scheme used by AAnF is existentially unforgeable under chosen message attacks (EUF-CMA) and the re-randomizable encryption scheme is correct.*

*Proof sketch:*The key insight is that traceability is guaranteed by the unforgeability of the AAnF signature σ_{AAnF} which commits to the encrypted SUPI. Since the AF only accepts registrations with valid AAnF signatures, and honest AAnFs always include the correct encrypted SUPI in their signatures, malicious UEs cannot prevent the storage of their true SUPI in the AF's database.

6 Implementation and Evaluation

In this section, we assess the performance of E2E-AKMA in real-world deployment scenarios.

Device Configuration. The experiments used real SIM cards to closely reflect real-world scenarios, ensuring our evaluation considers the actual computational and communication overhead of the E2E-AKMA protocol on the UE. This validates the protocol's compatibility with existing SIM card infrastructure and its feasibility for deployment without hardware changes. Performance testing was conducted on an iPhone 15 (A16 processor, real SIM card, iOS 18.1) . The Golang implementation is built as a linkable library for Swift on iOS.

Due to lack of mobile network operator privileges, physical base stations were inaccessible. Instead, the HN and AF components were implemented on cloud servers and communicated with the UE via a real 5G network. Both HN and AF demo servers were deployed as cloud services with 8-core Intel Xeon E5-2620 v4 CPUs, 32GB RAM, and located in the same data center. While this setup does not replicate physical base station infrastructure, it effectively models the protocol's computational and communication processes. Similar approaches are common in academic research when operator infrastructure access is restricted [29].

5G AKMA and AKMA+ involve interactions between the AF and HN, so we measured the latency between the three parties. UE connects to the server through the 5G cellular network (i.e. 5G AKA), while the maximum network bandwidth of the HN and AF is 100 Mbps. The latency is approximately 20 ms between the UE and HN, 20 ms between the UE and AF,

Our solution also supports eSIMs, which securely store sensitive user information within the eUICC module, ensuring secure communication with the HN like traditional SIM cards [24].

and 10 ms between the HN and AF. We generate self-signed certificates for the HN and AF using RSA 4096 signatures, which are enough to simulate the TLS channel.

Execution Time. In Table 2, the execution times for individual registration and login operations of the E2E-AKMA protocol are presented. The table provides a detailed breakdown of the communication time and local computation time, both of which have been optimized to a reasonable extent. The parameters and performance metrics of 2POS, along with the communication size of E2E-AKMA, are detailed in Appendix 6. To enhance clarity, a double-headed arrow is used to represent the exchange of interactive messages between two participants, the number inside the circle indicating the total number of message rounds exchanged during the operation.

To evaluate the performance overhead introduced by our E2E-AKMA protocol, we conducted an experimental comparison with the 5G AKMA and AKMA+ protocols under identical experimental conditions. The results of our empirical analysis indicate that the E2E-AKMA protocol incurs a moderate overhead of 9.4% (+40.11 ms) compared to the 5G AKMA protocol, while compared with AKMA+, it increased by 9.3% (+39.69ms).

Research shows humans typically do not perceive latency below 100 ms [19]. The 40.11 ms introduced by E2E-AKMA is well within this threshold. Studies in HCI [7] and industry practices [6] confirm such minor latency increases are imperceptible, making E2E-AKMA's impact negligible.

Table 2: The Runtime of the Register and Login phase.

	Registration	Runtime	Total (ms)
Communicate with HN and AF	AF ↔ UE ①	180.87	702.80
	HN ↔ UE ①	198.97	
	HN ↔ UE ②	91.14	
	HN ↔ UE ③	143.69	
	AF ↔ UE ②	88.13	
Local computation time of UE	HN ↔ UE ① to HN ↔ UE ②	1.18	90.15
	HN ↔ UE ② to HN ↔ UE ③	10.85	
	HN ↔ UE ③ to AF ↔ UE ②	78.12	
Total			792.95
	Login	Runtime	Total (ms)
Communicate with HN and AF	AF ↔ UE ①	55.86	355.04
	HN ↔ UE ①	148.18	
	HN ↔ UE ②	148.18	
	AF ↔ UE ②	74.85	
Local computation time of UE	HN ↔ UE ① to HN ↔ UE ②	2.43(μs)	109.08
	HN ↔ UE ② to HN ↔ UE ③	30.66	
	HN ↔ UE ③ to AF ↔ UE ②	78.41	
Total			464.12
			5G AKMA runtime: 424.01
			AKMA+ runtime: 424.43

The latency is approximately 20 ms between the UE and HN, 20 ms between the UE and AF, and 10 ms between the HN and AF.

Computation cost on the server side. In Table 3, we present an evaluation of the capability of HN and AF to handle high-user traffic. The table provides the number of requests that HN and AF can process per second in multithreaded mode. The test focuses on the efficiency of request processing on the server side, excluding network transmission time.

Although the throughput of the second message exchanged between HN and UE during the login phase is 84.27, in practical deployments, performance can be significantly improved through high-performance servers and dedicated hardware accelerators, which are commonly used in hardware security modules to optimize public key computations and greatly enhance throughput. Therefore, this will not become a performance bottleneck for E2E-AKMA.

Table 3: The Throughput of the HN and AF.

Registation	**Throughput (Requests/s)**	**Login**	**Throughput (Requests/s)**
AF ↔ UE ①	4329.07	AF ↔ UE ①	4776.02
HN ↔ UE ①	290.66	HN ↔ UE ①	319.08
HN ↔ UE ②	286.69	HN ↔ UE ②	84.27
HN ↔ UE ③	82.70	AF ↔ UE ②	581.65
AF ↔ UE ②	481.05		

In practice, side-channel countermeasures may introduce additional overheads. AAnF is at risk of DoS attacks, a situation also present in the original AKMA protocol, which can be mitigated by measures such as load balancing, hardware accelerators, and rate-limiting against malicious SUPIs via the 5G-AKA protocol.

7 Conclusion

The E2E-AKMA protocol proposed in this work effectively addresses the security and privacy challenges identified by 3GPP [2]. In comparison to existing solutions, the proposed scheme guarantees end-to-end communication security, even in scenarios where the AAnF is compromised. Moreover, the E2E-AKMA protocol offers notable advantages in terms of deployment feasibility when compared to the scheme presented in [29]. Specifically, it eliminates the requirement for maintaining a secure data repository, thereby simplifying implementation and reducing associated costs.

Acknowledgments

The authors express their profound appreciation to the shepherd and the anonymous reviewers for their constructive feedback and insightful suggestions, which have played a pivotal role in improving the quality and rigor of this work. Additionally, the authors extend their heartfelt thanks to Meihui Chen and Yanfei Guo from Huawei for their invaluable guidance and support in navigating this research domain. This work was supported by the National Key R&D Program of China (Grant No. 2022YFB3102500), the CAS Project for Young Scientists in Basic Research (Grant No. YSBR-035), the National Natural Science Foundation of China Key Program

(Grant No. 92270204) and the ISCAS Basic Research Project (Grant No. ISCAS-JCZD-202404).

Ethical Considerations

Stakeholders. This research was conducted in strict adherence to ethical research principles for cybersecurity and mobile network protocol analysis. All experimental evaluations were performed in a controlled, isolated environment using equipment owned by the research authors. The UE used in experiments was the authors' personal device, while AF and AAnF servers were legally obtained through legitimate cloud service providers. No real user data, SUPIs, or personal information were accessed during the research. All experimental data consisted of synthetic test parameters generated specifically for research purposes, and no interference with operational 5G networks or real user services occurred.

While our research identifies theoretical limitations in existing AKMA protocols, we have avoided publishing specific exploit details that could enable malicious attacks. Our focus remains on constructive protocol improvements rather than exposing actionable vulnerabilities. All research activities were conducted within applicable legal frameworks and industry ethical guidelines. The proposed E2E-AKMA protocol enhancements are designed to improve user privacy protection and authentication security for all mobile network users, contributing to the development of more secure mobile authentication standards.

Broader Societal Impact and Mitigations. Beyond the research context, we address the potential societal implications:

Equity and Accessibility: Regarding concerns about *resource-constrained environments* and the *vulnerability landscape*, E2E-AKMA is designed as an optional enhancement compatible with standard SIM cards (unlike hardware-dependent solutions like FIDO). This ensures that users with legacy devices are not excluded from security upgrades. Deploying E2E-AKMA will increase power consumption and data usage overhead. Since AFs typically support multiple authentication methods, users or operators sensitive to the marginal overhead can opt for standard authentication, preventing any forced burden on underserved populations.

Dual-Use and Regulation: To address *dual-use* concerns where strong privacy might be exploited to evade oversight, our protocol explicitly incorporates a *Lawful Interception (LI)* mechanism. While E2E-AKMA prevents unauthorized tracking by commercial entities or compromised intermediaries, the Core Network retains the capability to decrypt and recover the SUPI. This design ensures that the protocol enhances user privacy without circumventing necessary legal and regulatory frameworks. However, we acknowledge the ongoing debate regarding law enforcement access [18]; while essential for public safety in some frameworks, such mechanisms can introduce security risks or potential for misuse.

Open Science

We have made our experimental test code publicly available, which includes the test code for 5G-AKMA and E2E-AKMA. Furthermore, we will participate in artifact evaluation. The experimental code is available at https://doi.org/10.5281/zenodo.17896393.

References

[1] 3GPP. Security architecture and procedures for 5g system. Technical Report TS 33.501, 3rd Generation Partnership Project, 2023.

[2] 3GPP Technical Specification. Study on authentication and key management for applications based on 3GPP credential in 5G, 2020.

[3] 3GPP Technical Specification. Authentication and key management for applications based on 3GPP credentials in the 5G system, 2024. Version 18.5.0.

[4] Gizem Akman, Philip Ginzboorg, Mohamed Taoufiq Damir, and Valtteri Niemi. Privacy-Enhanced AKMA for Multi-Access Edge Computing Mobility. *Computers*, 12(1):2, 2022.

[5] Mihir Bellare, Zvika Brakerski, Moni Naor, Thomas Ristenpart, Gil Segev, Hovav Shacham, and Scott Yilek. Rerandomizable encryption. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 515–534. Springer, 2012.

[6] Jake Brutlag. Speed matters for google web search. *Google. June*, 2(9), 2009.

[7] Stuart K Card. *The psychology of human-computer interaction*. Crc Press, 2018.

[8] Taher ElGamal. A public key cryptosystem and a signature scheme based on discrete logarithms. Technical report, IEEE transactions on information theory, 1985.

[9] Philippe Golle, Markus Jakobsson, Ari Juels, and Paul Syverson. Universal re-encryption for mixnets. In *Cryptographers' Track at the RSA Conference*, pages 163–178. Springer, 2004.

[10] Paul A Grassi, James L Fenton, Elaine M Newton, Ray A Perlner, Andrew R Regenscheid, William E Burr, Justin P Richer, Naomi B Lefkovitz, Jamie M Danker, YeeYin Choong, et al. Draft NIST special publication 800-63b digital identity guidelines. *National Institute of Standards and Technology (NIST)*, 27, 2016.

[11] IDStrong. MyFitnessPal Data Breach. https://www.idstrong.com/sentinel/myfitnesspal-data-breach/, 2022.

[12] Tibor Jager, Florian Kohlar, Sven Schäge, and Jörg Schwenk. On the security of TLS-DHE in the standard model. In *Annual Cryptology Conference*, 2012.

[13] Mohsin Khan, Philip Ginzboorg, and Valtteri Niemi. Privacy preserving AKMA in 5G. In *Proceedings of the 5th ACM Workshop on Security Standardisation Research Workshop*, pages 45–56, 2019.

[14] Mohsin Khan, Philip Ginzboorg, and Valtteri Niemi. AKMA: Delegated authentication system of 5G. *IEEE Communications Standards Magazine*, 2021.

[15] Hugo Krawczyk, Kenneth G Paterson, and Hoeteck Wee. On the security of the TLS protocol: A systematic analysis. In *Annual Cryptology Conference*. Springer, 2013.

[16] Yueming Li, Long Chen, Qianwen Gao, and Zhenfeng Zhang. E2E-AKMA: An End-to-End Secure and Privacy-Enhancing AKMA Protocol Against the Anchor Function Compromise. Cryptology ePrint Archive, Paper 2025/2276, 2025.

[17] Yueming Li, Long Chen, and Zhenfeng Zhang. A Comprehensive Analysis of the AKMA+ Protocol. In *2025 IEEE 24th International Conference on Trust, Security and Privacy in Computing and Communications*, 2025.

[18] National Academies of Sciences, Engineering, and Medicine. *Decrypting the Encryption Debate: A Framework for Decision Makers*. The National Academies Press, Washington, DC, 2018.

[19] Jakob Nielsen. *Usability engineering*. Morgan Kaufmann, 1994.

[20] OASIS Security Services Technical Committee. Security assertion markup language (SAML) V2.0 technical overview. Working Draft sstc-saml-tech-overview-2.0, OASIS Open, March 2008. Accessed: 2025-12-09.

[21] OpenID Foundation. OpenID connect core 1.0. https://openid.net/specs/openid-connect-core-1_0-final.html, 2014. Accessed: 2025-12-09.

[22] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In *International conference on the theory and applications of cryptographic techniques*, pages 223–238. Springer, 1999.

[23] Manoj Prabhakaran and Mike Rosulek. Rerandomizable RCCA encryption. In *Annual International Cryptology Conference*, pages 517–534. Springer, 2007.

[24] RSIM Provisioning. eSIM Whitepaper the what and how of remote SIM provisioning. *GSMA, London, UK, Tech. Rep., Mar*, 2018.

[25] Terena Bell. Adobe's CSO talks security, the 2013 breach, and how he sets priorities. https://www.csoonline.com/article/3268035/adobe-s-cso-talks-security-the-2013-breach-and-how-he-sets-priorities.html, 2021.

[26] W3C. Web Authentication: An API for accessing Public Key Credentials Level 3. https://www.w3.org/TR/webauthn-3/, 2021.

[27] Wikipedia Contributors. 2012 LinkedIn hack. https://en.wikipedia.org/wiki/2012_LinkedIn_hack, 2021.

[28] Tengshun Yang, Shuling Wang, Bohua Zhan, Naijun Zhan, Jinghui Li, Shuangqing Xiang, Zhan Xiang, and Bifei Mao. Formal Analysis of 5G Authentication and Key Management for Applications (AKMA). *Journal of Systems Architecture*, 126:102478, 2022.

[29] Yang Yang, Guomin Yang, Yingjiu Li, Minming Huang, Zilin Shen, Imtiaz Karim, Ralf Sasse, David Basin, Elisa Bertino, Jian Weng, et al. AKMA+: Security and privacy-enhanced and standard-compatible AKMA for 5G communication. 2025.

A Preliminary

In this section, we will present an overview of the cryptographic primitives utilized in our protocol.

5G Authentication and Key Agreement. The 5G AKA protocol establishes mutual authentication and secure key agreement between UE and the HN [1]. The protocol relies on a symmetric key stored both in the SIM card and the HN, along with the Subscription Permanent Identifier (SUPI). Upon successful authentication, both parties derive shared session keys for secure 5G communication. For our purposes, we abstract the 5G AKA protocol as enabling the establishment of a unique and private secret key between the UE and HN, which serves as the foundation for subsequent AKMA operations.

Re-randomizable Public Key Encryption. Re-randomizable encryption schemes allow anyone to transform a given ciphertext into a new ciphertext that encrypts the same plaintext but appears statistically independent from the original [9, 23]. This property is crucial for ensuring privacy in our protocol.

The ElGamal encryption scheme [8] is a classic example of re-randomizable encryption. Given a ciphertext $(g^r, m \cdot h^r)$ where $h = g^x$ is the public key, one can re-randomize it by computing $(g^r \cdot g^{r'}, m \cdot h^r \cdot h^{r'}) = (g^{r+r'}, m \cdot h^{r+r'})$ for fresh randomness r'. The re-randomized ciphertext is computationally indistinguishable from a fresh encryption of the same message [5]. Importantly, ElGamal encryption satisfies the malformed ciphertext indistinguishability property. For any

given ciphertext pair (c_0, c_1) in the group $\mathbb{G} \times \mathbb{G}$ (regardless of whether they are valid encryptions), after re-randomization we obtain $(c_0', c_1') = (c_0 \cdot g^{r'}, c_1 \cdot h^{r'})$ for fresh randomness r'. Since $g^{r'}$ and $h^{r'} = g^{x \cdot r'}$ are uniformly distributed group elements (assuming the DDH assumption), the re-randomized pair (c_0', c_1') is computationally indistinguishable from two random group elements, regardless of the validity of the original ciphertext (c_0, c_1).

The ideal zero-knowledge functionality $\mathcal{F}_{\mathbf{zk}}$. We use the standard ideal zero-knowledge functionality defined by $((x,w),\lambda) \to (\lambda,(x,R(x,w)))$, where λ denotes the empty string. For a relation R, the functionality is denoted by $\mathcal{F}_{\text{zk}}^{R}$. Note that any zero-knowledge proof of knowledge fulfills the $\mathcal{F}_{\text{zk}}$ functionality; non-interactive versions can be achieved in the random-oracle model via the Fiat-Shamir paradigm with security under sequential composition. Here is the zero-knowledge functionality $\mathcal{F}_{\text{zk}}^{R}$ for Relation R:

- Receiving $(\text{prove}, \text{sid}, x, w)$ from party P_i (for $i \in \{1,2\}$): if $(x,w) \notin R$ or sid has been previously used then ignore the message. Otherwise, send $(\text{proof}, \text{sid}, x)$ to party P_{3-i}.

The committed non-interactive zero-knowledge functionality $\mathcal{F}_{\mathbf{com\text{-}zk}}$. In our protocol, parties will send commitments to non-interactive zero-knowledge proofs. We model this via commit-zk functionality, denoted $\mathcal{F}_{\text{com-zk}}$. Given non-interactive zero-knowledge proofs of knowledge, this functionality is securely realized by having the prover commit to such a proof using the ideal commitment functionality $\mathcal{F}_{\text{com}}$. Here is a committed functionality $\mathcal{F}_{\text{com-zk}}^{R}$ for Relation R, it works with parties P_1 and P_2:

- Upon receiving $(\text{com-prove}, \text{sid}, x, w)$ from a party P_i (for $i \in \{1,2\}$): if sid has been previously used then ignore. Otherwise, store (sid, i, x, w) and send $(\text{proof-receipt}, \text{sid})$ to P_{3-i}.
- Upon receiving $(\text{decom-proof}, \text{sid})$ from P_i (for $i \in \{1,2\}$): if (sid, i, x, w) has been stored – if $(x,w) \in R$ then send $(\text{decom-proof}, \text{sid}, x, 1)$ to P_{3-i}; otherwise send $(\text{decom-proof}, \text{sid}, x, 0)$ to P_{3-i}.

Paillier encryption. The Paillier encryption scheme, introduced by Pascal Paillier in 1999 [22], is a probabilistic public key encryption scheme that is notable for its homomorphic properties, allowing specific algebraic operations to be performed on ciphertexts which correspond to operations on the underlying plaintexts. Denote the public/private key pair by (pk, sk), and denote encryption and decryption under these keys by $\mathsf{Enc}_{\text{pk}}(\cdot)$ and $\mathsf{Dec}_{\text{sk}}(\cdot)$, respectively. We denote by $C_1 \oplus C_2$ the "addition" of the plaintexts in C_1, C_2, and by $a \odot C$ the multiplication of the plaintext in C by scalar a.

- If $C_1 = \mathsf{Enc}_{\text{pk}}(m_1)$ and $C_2 = \mathsf{Enc}_{\text{pk}}(m_2)$ are two ciphertexts, then the product $C_1 \oplus C_2$ is a valid encryption of $m_1 + m_2$.
- For a plaintext m and a ciphertext $C = \mathsf{Enc}_{\text{pk}}(m)$, the operation $a \odot C$ results in a ciphertext that encrypts $a \cdot m$.

B Detailed Security Models

A party is composed of many instances. For example, in the case of a UE party UE, it may register multiple accounts with different AFs through the $AAnF$ that issues its SIM card. Among these, the registration or login processes corresponding to different accounts are represented by various instances. The same applies to an AAnF or AF party. We regard a Turing machine as an instance, denoted by the symbol π. Given specific input, it produces specific output. We use $\pi_{Role}^{i,j}$ to represent an instance, where the subscript $Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF}$ indicates the party to which the instance belongs.

For an instance $\pi_{UE}^{i,j}$ of a UE party, $i > 0, j = 0$ indicates the registration of the i-th account, while $j > 0$ represents the j-th login of that account. For an instance $\pi_{AAnF}^{i',j'}$ of an AAnF party, $i' > 0, j' = 0$ indicates the i'-th registration performed with a certain UE, while $j' > 0$ represents the j'-th login corresponding to that registration. For an instance $\pi_{AF}^{i'',j''}$ of an AF party, $i'' > 0, j'' = 0$ indicates the i''-th registration performed with a certain UE, while $j'' > 0$ represents the j''-th login corresponding to that registration.

Since communication between instances utilizes the 5G AKA channel and the TLS channel, the adversary cannot obtain the transcript between the participants by eavesdropping on the channel. However, if the adversary corrupts a certain party, the adversary can act as that party to communicate with other parties. In this case, the adversary can observe the transcript within the channel of the corrupted party. The adversary can only observe the communication content between the parties it corrupts and the honest parties. For simplicity and better understanding, we did not explicitly illustrate the instances within the parties in the picture. Actually, communication occurs between the instances within the parties.

The use of the 5G AKA channel and the TLS channel results in the matching relationships between instances being determined during the execution of the protocol. For the definition of partner relationships, an instance needs to record its communication partner in the variable $Role_{par}$ ($Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF}$).

The 5G AKA lacks forward secrecy, so it is necessary to consider the capability of the adversary to subsequently compromise UE/AAnF and utilize the symmetric key K_{SIM} stored in SIM cards to decrypt prior sessions. Therefore, in the model, the transcript under the 5G AKA channel is recorded as a variable "trans" of an instance. When the adversary corrupts the UE or the $AAnF$, these prior transcripts need to be provided to the adversary.

Each instance possesses a set of internal states, denoted as $\pi_{Role}^{i,j}.\mathsf{st}(Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF})$. These internal states play a pivotal role in the functioning of the instance. When a message is received, the instance processes it by integrating the message with its existing parameters to produce an output.

Furthermore, the instance updates its parameters to reflect the changes induced by this interaction. The specific components of the internal state include the following categories:

- $\pi^{i,j}_{Role}.\mathsf{exe}(Role \in \mathcal{UE} \cup \mathcal{AANF} \cup \mathcal{AF}) \in \{accepted/rejected, running, \perp, send\_to\_AF, send\_to\_AAnF\}$: the execution state of each instance is initially $\perp$. When executing, it is *running*. After execution, it is *accepted* if the protocol succeeds or *rejected* if it fails. For UE instances, additional states *send_to_AF* and *send_to_AAnF* indicate that the UE needs to interact with AF or AAnF respectively.
- $\pi^{i,j}_{UE}$.pending_msg: stores the message that UE needs to send to the third party (AF or AAnF) when $\pi^{i,j}_{UE}.\mathsf{exe} \in \{\text{send\_to\_AF}, \text{send\_to\_AAnF}\}$.
- $\pi^{i,j}_{Role}.AKA(Role \in \mathcal{UE} \cup \mathcal{AANF})$: the transcript in the AKA channel.
- $\pi^{i,j}_{Role}.trans(Role \in \mathcal{UE} \cup \mathcal{AF})$: the transcript between the UE and the AF, in the TLS channel.
- $\pi^{i,j}_{UE}.K_{UE}$: the K_{UE} of this UE instance. For the same i, that is, the same account, the K_{UE} remains identical.
- $\pi^{i,j}_{UE}.AF_{par}$: this value is initially empty $\emptyset$ and is set to the partner the instance believes it is communicating with after protocol execution, e.g., this variable is the AF partner of a UE instance that refers to a specific AF AF.
- $\pi^{i,j}_{AF}.UE_{par}$: the UE partner of the AF instance.
- $\pi^{i,j}_{AAnF}.SK_{AAnF}$: the private key of this $AAnF$ instance. This variable is the same for instances of the same AAnF party.
- $\pi^{i,j}_{AAnF}.SUPI$: the $SUPI$ of this $AAnF$ instance. This variable is the same for instances of the same AAnF party.
- $\pi^{i,j}_{AF}.PK_{AAnF}$: the public key of all the AAnFs. This variable is the same for all the AF parties.
- $\pi^{i,j}_{AF}.DB_{AF}$: the database of this AF.
- $\pi^{i,j}_{Role}.ssk(Role \in \mathcal{UE} \cup \mathcal{AF}, j > 0)$: session key of this instance.

Oracles. The oracles defined in Figure 3 serve as the interface between the adversary and the protocol execution environment, modeling various adversarial capabilities and attack scenarios. Each oracle is carefully designed to capture specific types of threats and interactions that may occur in real-world deployments. At the beginning of the game, the challenger $\mathcal{C}$ establishes the set of all parties involved in the protocol, $\mathcal{UE}$, $\mathcal{AANF}$, and $\mathcal{AF}$.

Global Lists. During the process, the challenger maintains the following global lists.

- $\mathcal{L}_{crp}$: The set of corrupted entities (UEs, AAnFs, or AFs) whose internal states or secrets have been exposed to the adversary.
- $\mathcal{L}_{rvl}$: The set of oracle instances (protocol sessions) for which the session keys have been revealed to the adversary via the Reveal oracle.
- $\mathcal{L}_{cpm}$: The set of UEs that have been compromised through the Compromise oracle, meaning all their secret keys have been exposed.

Partner. The partner relationship between UE and AF is established through their interaction, represented by the transcript parameter, denoted as *trans*, which is maintained by the respective instance.

A UE instance $\pi^{i,j}_{UE}$ and an AF instance $\pi^{i'',j''}_{AF}$ are considered to be in a partner relationship if the following conditions are satisfied:

1. Both instances have successfully completed the protocol, meaning their execution states, exe, is *accepted*. Formally, this is expressed as $\pi^{i,j}_{UE}.\mathsf{exe} = \pi^{i'',j''}_{AF}.\mathsf{exe} = accepted$.
2. The transcripts of the two instances are identical, i.e., $\pi^{i,j}_{UE}.trans = \pi^{i'',j''}_{AF}.trans$.
3. The corresponding registration instances for each entity have an execution state of *accepted*, and their transcripts are also identical. Specifically, $\pi^{i,0}_{UE}.\mathsf{exe} = \pi^{i'',0}_{AF}.\mathsf{exe} = accepted$ and $\pi^{i,0}_{UE}.trans = \pi^{i'',0}_{AF}.trans$.

Authentication.

EXPERIMENT $\mathsf{Expt\text{-}Auth}^{\mathcal{A}}_{AKMA}(1^n)$

For an $\mathsf{AKMA} = (\mathsf{Init}, \mathsf{Reg}, \mathsf{Login})$, the following security experiment is run between the challenger and adversary $\mathcal{A}$.

- **Setup.** For each $Role \in \mathcal{UE} \cup \mathcal{AANF} \cup \mathcal{AF}$, the challenger executes the Init algorithm and provides the output PP_{AKMA} to $\mathcal{A}$.
- **Query.** $\mathcal{A}$ interact with oracles defined in Figure 3.
- **Output.** Finally, $\mathcal{A}$ terminates with no output, and the experiment outputs 1 if and only if there exists an honest UE instance $\pi^{i,j}_{UE}$ and an honest AF instance $\pi^{i'',j''}_{AF}$ such that $\mathcal{A}$ has neither compromised the UE nor corrupted the $AAnF$ to which the UE belongs simultaneously, i.e., $(UE \in \mathcal{L}_{crp}) \cup (AF \in \mathcal{L}_{crp}) \cup (UE \in \mathcal{L}_{cpm} \cap AAnF \in \mathcal{L}_{crp})$=0, and at least one of the specified conditions is satisfied:
  1. $\pi^{i,j}_{UE}$ and $\pi^{i'',j''}_{AF}$ form a partner relationship and are intentional partners of each other, $\pi^{i,j}_{UE}.AF_{par} = AF$, $\pi^{i'',j''}_{AF}.UE_{par} = UE$. And without adversary involvement, $UE \notin \mathcal{L}_{cpm}$, $AAnF \notin \mathcal{L}_{crp}$. But the execution state of at least one of them is not accepted, $\pi^{i,j}_{UE}.\mathsf{exe} \neq accepted$ or $\pi^{i'',j''}_{AF}.\mathsf{exe} \neq accepted$.
  2. $\pi^{i,j}_{UE}$ accepted without a partner, or with multiple partners, and its intentional partner was uncorrupted.
  3. The intentional partner of $\pi^{i'',j''}_{AF}$ is not $\emptyset$ and $\pi^{i'',j''}_{AF}$ accepted with no partner, or accepted with more than one partner, and its intentional partner was not corrupted.

Definition 3 (Authentication)**.** *An AKMA scheme is Authentication, if for any* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that* $\Pr[\mathsf{Expt\text{-}Auth}^{\mathcal{A}}_{\mathsf{AKMA}}(1^n) = 1] \leq \mathsf{negl}(n)$.

Key-Indistinguishability.

EXPERIMENT $\mathsf{Expt\text{-}KeyIND}^{\mathcal{A}}_{AKMA}(1^n)$

For an $\mathsf{AKMA} = (\mathsf{Init}, \mathsf{Reg}, \mathsf{Login})$, the following security experiment is run between the challenger and the adversary.

Oralce Start$((UE,i),AF)$

/ Check the registration status of the UE.

1 : **if** $\pi_{UE}^{i,0}.\mathsf{exe} = rejected$:

2 : **return** $\perp$

/ Determine the AAnF associated with the UE,

/ as well as the specific instance of the AAnF and AF that aim to be registered.

/ Execute the registration process and subsequently update the associated instance variables to reflect the changes.

3 : **elseif** $\pi_{UE}^{i,0}.\mathsf{exe} = \perp$:

4 : $AAnF \leftarrow Identify(UE)$, find $\pi_{AAnF}^{i',0}.\mathsf{exe} = \pi_{AF}^{i'',0}.\mathsf{exe} = \perp$

5 : $(K_{UE}, \perp, DB'_{AF}) \leftarrow \mathsf{Reg}\left(\pi_{UE}^{i,0}(\emptyset), \pi_{AAnF}^{i',0}(SUPI, SK_{AAnF}, K_{AAnF}, PK_{core}), \pi_{AF}^{i'',0}(\mathcal{PK}_{AAnF}, DB_{AF})\right)$

6 : $\pi_{UE}^{i,0}.K_{UE} := K_{UE}, \pi_{UE}^{i,0}.AF_{par} := AF, \pi_{AF}^{i'',0}.UE_{par} := UE,$
Update $\pi_{UE}^{i,0}.AKA, \pi_{UE}^{i,0}.trans, \pi_{AAnF}^{i',0}.AKA, \pi_{AF}^{i'',0}.trans, \pi_{AF}^{i'',0}.DB_{AF}$

7 : **return** $\perp$

/ Determine the AAnF associated with the UE,

/ as well as the specific instance of the UE, AAnF, and AF that are intended to initiate the login process.

/ Execute the login process and subsequently update the instance variables accordingly.

8 : **elseif** $\pi_{UE}^{i,0}.\mathsf{exe} = accepted$:

9 : $AAnF \leftarrow Identify(UE)$, find $\pi_{UE}^{i,j}.\mathsf{exe} = \pi_{AAnF}^{i',j'}.\mathsf{exe} = \pi_{AF}^{i'',j''}.\mathsf{exe} = \perp$

10 : $(ssk, \perp, ssk) \leftarrow \mathsf{Login}\left(\pi_{UE}^{i,j}(K_{UE}), \pi_{AAnF}^{i',j'}(SUPI, K_{AAnF}), \pi_{AF}^{i'',j''}(DB_{AF})\right)$

11 : $\pi_{UE}^{i,j}.ssk := \pi_{AF}^{i'',j''}.ssk := ssk, \pi_{UE}^{i,j}.AF_{par} := AF, \pi_{AF}^{i'',j''}.UE_{par} := UE,$
Update $\pi_{UE}^{i,j}.AKA, \pi_{UE}^{i,j}.trans, \pi_{AAnF}^{i',j'}.AKA, \pi_{AF}^{i'',j''}.trans$

12 : **return** $\perp$

Oracle CorruptUE(UE)

/ Return transcripts and K_{UE}s of the UE

1 : $\mathcal{L}_{AKA} \leftarrow \emptyset, \mathcal{L}_{K_{UE}} \leftarrow \emptyset$

2 : $AAnF \leftarrow Identify(UE)$

3 : **for** $i > 0, j \geq 0$ **do** :

4 : $\mathcal{L}_{AKA} \leftarrow \mathcal{L}_{AKA} \cup \pi_{UE}^{i,j}.AKA$

5 : $\mathcal{L}_{K_{UE}} \leftarrow \mathcal{L}_{K_{UE}} \cup \pi_{UE}^{i,0}.K_{UE}$

6 : $\mathcal{L}_{crp} \leftarrow \mathcal{L}_{crp} \cup UE$

7 : **return** $(\mathcal{L}_{AKA}, \mathcal{L}_{K_{UE}})$

Oracle CorruptAAnF$(AAnF)$

/ Return transcripts, K_{AAnF} and $SUPI$s of the $AAnF$.

1 : $\mathcal{L}_{AKA} \leftarrow \emptyset, \mathcal{L}_{SUPI} \leftarrow \emptyset$

2 : **for** $i > 0, j \geq 0$ **do** :

3 : $\mathcal{L}_{AKA} \leftarrow \mathcal{L}_{AKA} \cup \pi_{AAnF}^{i,j}.AKA$

4 : $\mathcal{L}_{SUPI} \leftarrow \mathcal{L}_{SUPI} \cup \pi_{AAnF}^{i,j}.SUPI$

5 : $\mathcal{L}_{crp} \leftarrow \mathcal{L}_{crp} \cup AAnF$

6 : **return** $(\mathcal{L}_{AKA}, \mathcal{L}_{SUPI}, K_{AAnF})$

Oracle Breach(AF)

/ Return local storage of AF.

1 : **return** DB_{AF}

Oracle UE$((i,j), \pi_{Role}^{i',j'}, m)$

/ Send m to the UE on behalf of Role.

/ $m = \emptyset$ means UE sends the first msg.

1 : **if** $Role \notin \mathcal{L}_{crp}$:

2 : **return** $\perp$

3 : $(\mathsf{st}', \mathsf{exe}', m') \leftarrow \pi_{UE}^{i,j}(Role, m)$

4 : / Handle potential UE-to-AF interaction when Role is AAnF

5 : **if** $Role = AAnF \wedge \pi_{UE}^{i,j}.\mathsf{exe} =$ "send_to_AF" $\wedge AF \notin \mathcal{L}_{crp}$:

6 : $m_{AF} \leftarrow \pi_{UE}^{i,j}.\mathsf{pending\_msg}$

7 : $(\mathsf{st}_{AF}, \mathsf{exe}_{AF}, m'_{AF}) \leftarrow \pi_{AF}^{i'',j''}(UE, m_{AF})$

8 : $(\mathsf{st}'', \mathsf{exe}'', m'') \leftarrow \pi_{UE}^{i,j}(AF, m'_{AF})$

9 : **return** m''

10 : / Handle potential UE-to-AAnF interaction when Role is AF

11 : **if** $Role = AF \wedge \pi_{UE}^{i,j}.\mathsf{exe} =$ "send_to_AAnF" $\wedge AAnF \notin \mathcal{L}_{crp}$:

12 : $m_{AAnF} \leftarrow \pi_{UE}^{i,j}.\mathsf{pending\_msg}$

13 : $(\mathsf{st}_{AAnF}, \mathsf{exe}_{AAnF}, m'_{AAnF}) \leftarrow \pi_{AAnF}^{i',j'}(UE, m_{AAnF})$

14 : $(\mathsf{st}'', \mathsf{exe}'', m'') \leftarrow \pi_{UE}^{i,j}(AAnF, m'_{AAnF})$

15 : **return** m''

16 : **return** m'

Oracle Reveal$(Role,i,j)$

/ Return the session key.

1 : **if** $Role \notin \mathcal{UE} \cup \mathcal{AF}$:

2 : **return** $\perp$

3 : **elseif** $\pi_{Role}^{i,j}.\mathsf{st} = \perp / reject$

4 : **return** $\perp$

5 : **elseif** $\pi_{Role}^{i,j}.\mathsf{st} = accept$:

6 : $\mathcal{L}_{rvl} \leftarrow \mathcal{L}_{rvl} \cup \pi_{Role}^{i,j}$

7 : **return** $\pi_{Role}^{i,j}.ssk$

Oracle Compromise(UE)

/ Return all K_{UE}s owned by the UE.

1 : $\mathcal{L}_{K_{UE}} \leftarrow \emptyset$

2 : **while** $i > 0$:

3 : $\mathcal{L}_{K_{UE}} \leftarrow \mathcal{L}_{K_{UE}} \cup \pi_{UE}^{i,0}.K_{UE}$

4 : $\mathcal{L}_{cpm} \leftarrow \mathcal{L}_{cpm} \cup UE$

5 : **return** $\mathcal{L}_{K_{UE}}$

Oracle CorruptAF(AF)

/ Return local storage of AF.

1 : $\mathcal{L}_{crp} \leftarrow \mathcal{L}_{crp} \cup AF$

2 : **return** DB_{AF}

Oracle AAnF$((i,j), \pi_{Role}^{i',j'}, m)$

/ Send m to the $AAnF$ on behalf of Role.

1 : **if** $Role \notin \mathcal{L}_{crp}$:

2 : **return** $\perp$

3 : $(\mathsf{st}', \mathsf{exe}', m') \leftarrow \pi_{AAnF}^{i',j'}(Role, m)$

4 : **return** m'

Oracle AF$((i,j), \pi_{Role}^{i',j'}, m)$

/ Send m to the AF on behalf of Role.

1 : $(\mathsf{st}', \mathsf{exe}', m') \leftarrow \pi_{AF}^{i'',j''}(Role, m)$

2 : **return** m'

Figure 3: Oracles.

- **Setup.** For each $Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF}$, the challenger executes the Init algorithm and provides the output PP_{AKMA} to $\mathcal{A}$.
- **Query.** $\mathcal{A}$ interact with oracles defined in Figure 3. The challenger randomly selects a bit $b \leftarrow\$ \{0,1\}$.
- **Test.** $\mathcal{A}$ selects a UE instance $\pi_{UE}^{i,j}$ or an AF instance $\pi_{AF}^{i'',j''}$ with the exe variable set to *accepted* as the challenge session. If $b=0$, the challenger randomly samples $k \leftarrow\$ \{0,1\}^{\lambda}$ for the adversary; if $b=1$, the challenger provides the adversary with the real session key *ssk* of the chosen instance. If the adversary selects a UE instance $\pi_{UE}^{i,j}$, all the following conditions must be satisfied:
  1. $\pi_{UE}^{i,j}$ has neither been revealed $\pi_{UE}^{i,j} \notin \mathcal{L}_{rvl}$, and has not been queried for corrupt before test $UE \notin \mathcal{L}_{crp}$.
  2. If $\pi_{UE}^{i,j}$ has partner $\pi_{AF}^{k,h}$, then $\pi_{AF}^{k,h}$ has neither been revealed session key $\pi_{AF}^{k,h} \notin \mathcal{L}_{rvl}$, and has not been queried for corrupt before test $AF \notin \mathcal{L}_{crp}$.
  3. $\mathcal{A}$ has not both compromise UE and corrupt the $AAnF$ to which the UE belongs simultaneously, $UE \notin \mathcal{L}_{cpm} \cap AAnF \notin \mathcal{L}_{crp}$=1.

  If $\mathcal{A}$ selects an AF instance $\pi_{AF}^{i'',j''}$, it must satisfy all conditions. As in the Authentication experiment, we use a non-empty intentional partner requirement to exclude cases where the adversary creates a UE to interact with the AF.
  1. $\pi_{AF}^{i'',j''}$ has neither been revealed $\pi_{AF}^{i'',j''} \notin \mathcal{L}_{rvl}$, and has not been queried for corrupt before test $AF \notin \mathcal{L}_{crp}$.
  2. The intentional partner of $\pi_{AF}^{i'',j''}$ is not $\emptyset$, if it has partner $\pi_{UE}^{k'',h''}$, $\mathcal{A}$ has not been revealed session key $\pi_{UE}^{k'',h''} \notin \mathcal{L}_{rvl}$, and has not been queried for corrupt before test $UE \notin \mathcal{L}_{crp}$.
  3. The intentional partner of $\pi_{AF}^{i'',j''}$ is not $\emptyset$, if it has partner $\pi_{UE}^{k'',h''}$, $\mathcal{A}$ has not both compromise UE and corrupt the $AAnF$ to which the UE belongs simultaneously, $UE \notin \mathcal{L}_{cpm} \cap AAnF \notin \mathcal{L}_{crp}$=1.
- **Query.** $\mathcal{A}$ can continue querying the oracles in Figure 3, but the chosen instance must satisfy the aforementioned conditions.
- **Output.** Finally, $\mathcal{A}$ terminates with output b', and the experiment outputs 1 if $b'=b$.

Definition 4 (Key-Indistinguishability)**.** *An AKMA protocol is Key-Ind, if for any* PPT *adversary $\mathcal{A}$, there exists a negligible function* negl *such that* $\Pr[\text{Expt-KeyIND}^{\mathcal{A}}_{\text{AKMA}}(1^n)=1] \leq \frac{1}{2} + \text{negl}(n)$.

AF Unlinkability.
EXPERIMENT $\text{Expt-AF-UNL}^{\mathcal{A}}_{AKMA}(1^n)$
This security experiment is conducted between a challenger and the adversary $\mathcal{A}$ for an $\text{AKMA} = (\text{Init}, \text{Reg}, \text{Login})$. The procedure is as follows:

- **Setup.** For each $Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF}$, the challenger executes the Init algorithm and provides the resulting public parameters PP_{AKMA} to the adversary $\mathcal{A}$.
- **Query.** The adversary $\mathcal{A}$ interacts with oracles defined in Figure 3. During this phase, $\mathcal{A}$ is allowed to select multiple AFs, denoted as AF_1 to AF_n, and query them using the CorruptAF oracle. Among these, one AF is chosen as the challenge target, denoted by AF^*.
- **Test.** The adversary $\mathcal{A}$ selects two honest UEs, denoted as UE_0 and UE_1, both corresponding to the *same* AAnF, denoted $AAnF^*$. These UEs must not have been queried by $\mathcal{A}$ using the CorruptAAnF oracle. The adversary can utilize the UE oracle to send messages to these UEs for registration and login with the previously corrupted AFs, namely AF_1 to AF_n.
  Subsequently, $\mathcal{A}$ selects two unregistered instances, denoted as $\pi_{UE_0}^{i_0,0}$ and $\pi_{UE_1}^{i_1,0}$, from these UEs, where their execution status is set to $\perp$. As part of the challenge session, the challenger selects a random bit $b \leftarrow \{0,1\}$ and proceeds as follows:
  - The instance $\pi_{UE_b}^{i_b,0}$ registers with the challenge target AF^*, facilitated by the corresponding $AAnF^*$.
- **Query.** The adversary $\mathcal{A}$ may continue interacting with the oracles defined in Figure 3. Additionally, $\mathcal{A}$ may request the selected UE instance from the Test phase, denoted as UE_b, to perform a login operation. This interaction is subject to the following three conditions:
  - Neither of the UEs (UE_0, UE_1) belong to the sets $\mathcal{L}_{crp}$ or $\mathcal{L}_{cpm}$: $(UE_0 \in \mathcal{L}_{crp} \cup UE_0 \in \mathcal{L}_{cpm} \cup UE_1 \in \mathcal{L}_{crp} \cup UE_1 \in \mathcal{L}_{cpm}) = 0$.
  - The AAnF $AAnF^*$ does not belong to the corruption set $\mathcal{L}_{crp}$: $AAnF^* \notin \mathcal{L}_{crp}$.
  - The session keys of both UE instances $(\pi_{UE_0}^{i_0,j_0}, \pi_{UE_1}^{i_1,j_1})$ have not been queried via the $\mathcal{L}_{rvl}$ set: $(\pi_{UE_0}^{i_0,j_0} \in \mathcal{L}_{rvl} \cup \pi_{UE_1}^{i_1,j_1} \in \mathcal{L}_{rvl}) = 0, \quad (j_0 > 0, j_1 > 0)$.
- **Output.** Finally, the adversary $\mathcal{A}$ outputs a guess bit b'. The experiment outputs 1 if and only if $b'=b$.

Definition 5 (AF-Unlinkability)**.** *An AKMA scheme is AF-Unlinkable if for any* PPT *adversary $\mathcal{A}$, there exists a negligible function* negl *such that* $\Pr\left[\text{Expt-AF-UNL}^{\mathcal{A}}_{AKMA}(1^n) = 1\right] \leq \frac{1}{2} + \text{negl}(n)$, *where the two honest UEs UE_0 and UE_1 correspond to the* same *AAnF.*

AAnF-Unlinkability.
EXPERIMENT $\text{Expt-AAnF-UNL}^{\mathcal{A}}_{AKMA}(1^n)$
For an $\text{AKMA} = (\text{Init}, \text{Reg}, \text{Login})$, the following security experiment is run between the challenger and the adversary.

- **Setup.** For each $Role \in \mathcal{UE} \cup \mathcal{AAN}\mathcal{F} \cup \mathcal{AF}$, the challenger executes the Init algorithm and provides the output PP_{AKMA} to $\mathcal{A}$.
- **Query.** $\mathcal{A}$ interact with oracles defined in Figure 3 where $\mathcal{A}$ can select an AAnF to query CorruptAAnF.
- **Test.** $\mathcal{A}$ selects one honest UE, corresponding to the corrupted AAnF, which has not been queried by $\mathcal{A}$ via CorruptUE or Compromise oracle. $\mathcal{A}$ also selects two honest AFs, AF_L and AF_R, which have not been queried by $\mathcal{A}$

via CorruptAF oracle. $\mathcal{A}$ selects two unregistered instances $\pi_{UE}^{i_0,0}$ and $\pi_{UE}^{i_1,0}$ (exe $= \perp$) from the UE. As a challenge session, the challenger selects $b \leftarrow \{0,1\}$ and acts as follows:
  - If $b = 0$: $\pi_{UE}^{i_0,0}$ registers with AF_L and $\pi_{UE}^{i_1,0}$ registers with AF_R through the corrupted AAnF.
  - If $b = 1$: $\pi_{UE}^{i_0,0}$ registers with AF_R and $\pi_{UE}^{i_1,0}$ registers with AF_L through the corrupted AAnF.
- **Query.** $\mathcal{A}$ can continue querying the oracles in Figure 3, and can require the selected UE in **Test** phase to perform the login, subject to three conditions: the UE is honest, the AFs are honest and the session keys of two UE instances are not queried.
  - $(UE \in \mathcal{L}_{crp} \cup UE \in \mathcal{L}_{cpm})$=0.
  - $(AF_L \in \mathcal{L}_{crp} \cup AF_R \in \mathcal{L}_{crp})$=0.
  - $(\pi_{UE}^{i_0,j_0} \in \mathcal{L}_{rvl} \cup \pi_{UE}^{i_1,j_1} \in \mathcal{L}_{rvl})$=0, $(j_0 > 0, j_1 > 0)$.
- **Output.** Finally, $\mathcal{A}$ terminates with output b', and the experiment outputs 1 if $b' = b$.

Definition 6 (AAnF-unlinkability)**.** *An AKMA scheme is AAnF-Unlinkable if for any* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that* $\mathsf{Pr}[\text{Expt-AAnF-UNL}^{\mathcal{A}}_{\mathsf{AKMA}}(1^n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$.

AAnF-Indistinguishability.
EXPERIMENT $\text{Expt-AAnF-IND}^{\mathcal{A}}_{AKMA}(1^n)$
For an $\mathsf{AKMA} = (\mathsf{Init}, \mathsf{Reg}, \mathsf{Login})$, the following security experiment is run between the challenger and the adversary $\mathcal{A}$.

- **Setup.** For each $Role \in \mathcal{UE} \cup \mathcal{AANF} \cup \mathcal{AF}$, the challenger executes the Init algorithm and provides the output PP_{AKMA} to $\mathcal{A}$.
- **Query.** $\mathcal{A}$ interacts with oracles defined in Figure 3. The adversary is required to select an AAnF as the challenge target and issue a CorruptAAnF query to obtain full control over this AAnF, including access to all SUPI values and the master key K_{AAnF}.
- **Challenge Setup.** $\mathcal{A}$ selects:
  - Two honest UEs: UE_0 and UE_1, both associated with the corrupted AAnF. The adversary knows their respective SUPIs: $SUPI_0$ and $SUPI_1$.
  - Two honest AFs: AF_L and AF_R (not necessarily distinct), which have not been queried via CorruptAF.

  The challenger facilitates the registration of accounts:
  - UE_0 registers an account with AF_L through the corrupted AAnF
  - UE_1 registers an account with AF_R through the corrupted AAnF

  The adversary observes all registration interactions and obtains the public keys and account information generated during these registrations.
- **Test.** The challenger selects a random bit $b \leftarrow \{0,1\}$ and proceeds as follows:
  - If $b = 0$: UE_0 performs a login to its account on AF_L
  - If $b = 1$: UE_1 performs a login to its account on AF_R

  The adversary observes this login interaction through its control of the AAnF but must determine which UE performed the login.
- **Query.** $\mathcal{A}$ can continue querying the oracles in Figure 3, subject to the following restrictions:
  - Neither UE has been corrupted or compromised: $(UE_0 \in \mathcal{L}_{crp} \cup UE_0 \in \mathcal{L}_{cpm} \cup UE_1 \in \mathcal{L}_{crp} \cup UE_1 \in \mathcal{L}_{cpm}) = 0$
  - Neither AF has been corrupted: $(AF_L \in \mathcal{L}_{crp} \cup AF_R \in \mathcal{L}_{crp}) = 0$
  - The session keys of the challenge login sessions have not been revealed
- **Output.** Finally, $\mathcal{A}$ terminates with output b', and the experiment outputs 1 if and only if $b' = b$.

Definition 7 (AAnF-Indistinguishability)**.** *An AKMA protocol satisfies AAnF-IND if for any* PPT *adversary* $\mathcal{A}$*, there exists a negligible function* negl *such that* $\mathsf{Pr}[\text{Expt-AAnF-IND}^{\mathcal{A}}_{\mathsf{AKMA}}(1^n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$.

Traceability.
EXPERIMENT $\text{Expt-Traceability}(1^n)$
For an $\mathsf{AKMA} = (\mathsf{Init}, \mathsf{Reg}, \mathsf{Login}, \mathsf{Trace})$, the following security experiment is run between the challenger and the adversary $\mathcal{A}$.

- **Setup.** The challenger generates $(PK_{core}, SK_{core}) \leftarrow \mathsf{KeyGen}(1^n)$ and executes the Init algorithm for each $Role \in \mathcal{UE} \cup \mathcal{AANF} \cup \mathcal{AF}$, providing the output PP_{AKMA} and PK_{core} to $\mathcal{A}$.
- **Query.** $\mathcal{A}$ can interact with oracles defined in Figure 3.
- **Challenge.** The adversary $\mathcal{A}$ selects a target $SUPI^*$ and chooses honest AAnF and AF instances $\pi_{AAnF}^{i',0}$ and $\pi_{AF}^{i'',0}$ with $\pi_{AAnF}^{i',0}.\mathsf{exe} = \pi_{AF}^{i'',0}.\mathsf{exe} = \perp$. Here the $AAnF$ and the AF must remain honest throughout the experiment, i.e., $\{AAnF, AF\} \notin \mathcal{L}_{crp}$.
- **Registration.** $\mathcal{A}$ then acts as a malicious UE and interacts with the chosen AAnF and AF instances through the AAnF and AF oracles to perform the registration protocol. The registration is considered successful if the corresponding AF instance outputs $\pi_{AF}^{i'',0}.\mathsf{exe} = accepted$. During this procedure, $\mathcal{A}$ can still interact with oracles defined in 3 but cannot query CorruptAAnF and CorruptAF on the $AAnF$ and the AF respectively.
- **Trace.** If the registration succeeds (i.e., $\pi_{AF}^{i'',0}.\mathsf{exe} = accepted$), the challenger executes the trace algorithm on the AF's database: $SUPI' \leftarrow \mathsf{Trace}(\pi_{AF}^{i'',0}.DB_{AF}, SK_{core})$.
- **Output.** The experiment returns 1 if $SUPI' = SUPI^*$, i.e., the traced SUPI matches the SUPI that the honest AAnF used during the registration process (stored in $\pi_{AAnF}^{i',0}.SUPI$), regardless of the malicious UE's behavior.

Definition 8 (Traceability)**.** *An AKMA scheme satisfies Traceability if for any* PPT *adversary* $\mathcal{A}$ *that can corrupt UE instances, there exists a negligible function negl such that* $\mathsf{Pr}[\text{Expt-Traceability}(1^n) = 1] \geq 1 - \mathsf{negl}(n)$.