AKMA+: Security and Privacy-Enhanced and Standard-Compatible AKMA for 5G Communication

Yang Yang[1], Guomin Yang[1] , Yingjiu Li[2], Minming Huang[1], Zilin Shen[3], Imtiaz Karim[3],
Ralf Sasse[4], David Basin[4], Elisa Bertino[3], Jian Weng[5], Hwee Hwa Pang[1], Robert H. Deng[1]
1. Singapore Management University, Singapore
({yyang, gmyang, mmhuang, hhpang,robertdeng}@smu.edu.sg)
2. University of Oregon, USA (yingjiul@uoregon.edu)
3. Purdue University, USA ({shen624,karim7,bertino}@purdue.edu)
4. ETH Zurich, Switzerland ({ralf.sasse,basin}@inf.ethz.ch)
5. Jinan University, Guangzhou, China (cryptjweng@gmail.com)

Abstract

The Authentication and Key Management for Applications (AKMA) protocol is a fundamental building block for security and privacy of 5G cellular networks. Therefore, it is critical that the protocol is free of vulnerabilities that can be exploited by attackers. Unfortunately, based on a detailed analysis of AKMA, we show that AKMA has several vulnerabilities that may lead to security and privacy breaches.

We define AKMA+, an enhanced protocol for 5G communication that protects against security and privacy breaches while maintaining compatibility with existing standards. AKMA+ includes countermeasures for protecting communication between the user equipment (UE) and application functions (AFs) from attackers, including those within the home public land mobile network. These countermeasures ensure mutual authentication between the UE and the AKMA anchor function without altering the protocol flow. We also address vulnerabilities related to subscriber and AKMA key identifiers that could be exploited in linkability attacks. By obfuscating this data, AKMA+ prevents attackers from associating a target UE with its past application access.

We employ formal verification to demonstrate that AKMA+ achieves key security and privacy objectives. We conduct extensive experiments demonstrating that AKMA+ incurs acceptable computational overhead, bandwidth costs, and UE battery consumption.

1 Introduction

The Authentication and Key Management for Applications (AKMA) protocol, developed by the 3rd Generation Partnership Project (3GPP) in Technical Specification (TS) 33.535 [32], is a framework for managing and distributing keys to the secure application layer data in 5G mobile networks. AKMA provides a standardized method for secure key handling by various application functions (AFs), enhancing security, privacy, interoperability, and scalability in 5G communication. Major network equipment vendors like Ericsson [13], Nokia [18], and Huawei [14] are integrating AKMA into 5G solutions, while telecommunication providers such as AT&T [4] and Verizon [38] are deploying AKMA-enabled networks to offer secure voice, data, and multimedia services. Mobile device manufacturers like Apple [3] and Samsung [23] are embedding AKMA support in their devices to ensure secure interaction with 5G networks and applications.

AKMA is still in its early stages and is not yet fully mature. The 3GPP Technical Report (TR) 33.835 [24] identifies seventeen key issues (#1- #17) that need to be addressed as AKMA evolves. Our focus is on five key issues (#3, #5, #6, #7, and #16) that pose security and privacy threats and significantly undermine the foundation of AKMA from the protocol perspective. The remaining key issues in TR 33.835 pertain to the AKMA architecture framework, architecture interface, API, and regulatory compliance, which are beyond the scope of this work as they cannot be solved from the protocol perspective[1].

Key issues #5 (user privacy) and #7 (protecting subscriber's personal information in control and data traffic) are both related to the leakage of Subscription Permanent Identifier (SUPI) or AKMA key identifier (A-KID) within AKMA. This leakage enables application functions (AFs) and AKMA anchor functions (AAnFs) to link users across the network. We aim to address these issues to achieve user indistinguishability and thus enhance privacy protection in AKMA.

Key issues #6 (secure communication between UE and AF) and #16 (application key freshness) are all related to insufficient protection of communication between user equipment (UE) and application functions (AFs). In AKMA, sensitive data exchanged between UE and AF are safeguarded by session keys derived in the home public land mobile network (HPLMN). However, the use of session keys in AKMA is vulnerable in several ways. First, the session keys can be accessed by the AKMA anchor function (AAnF) within HPLMN, al-

[1] Key issue #9 in TR 33.835 [24] specifies that the AKMA architecture must support key separation for different AFs. This issue has been addressed in the latest version of the AKMA specifications by including the AF identifier in the application key derive function (see Annex A.4 in TS 33.535 V18.4.0 [32] for details).

lowing AAnF to intercept the communication between UE and AF. Second, the session keys may be reused across different sessions between the same pair of UE and AF, leading to various spoofing attacks, such as a malicious UE impersonating another UE towards an AF [40], and a third-party attacker impersonating an AF towards a UE [2]. Third, the session keys between UE and AF are derived from a long-term key and a user's SUPI; if an attacker gains access to both the long-term key and SUPI (either from UE or HPLMN), they could potentially derive all related session keys and thus compromise the corresponding previously recorded communication sessions [2]. We aim to address these issues to achieve session key secrecy and forward secrecy for communication between UEs and AFs in AKMA.

Lastly, key issue #3 (mutual authentication between UE and AAnF) highlights the absence of mutual authentication between UE and AAnF in the AKMA protocol. Without proper authentication, unauthorized UEs may interact with AAnF and gain access to AKMA services. Furthermore, the lack of mutual authentication opens the door for fake AAnFs to communicate with UEs, posing a significant risk of privacy loss and exposure for users. We aim to address this issue to provide mutual authentication between UE and AAnF.

Contributions. We propose AKMA+, a security and privacy-enhanced, standard-compatible revision to the 3GPP AKMA protocol. Our contributions are summarized below.

• We identify the root causes of the existing security and privacy issues from the AKMA protocol perspective and address them systematically in AKMA+ to achieve user indistinguishability, session key secrecy and forward secrecy for communication between UEs and AFs, and mutual authentication between UE and AAnF.

• We make AKMA+ standard-compatible by following all the AKMA commands, message flows, and data formats as defined by 3GPP. While data processing within each entity may be revised to enhance security and privacy, we ensure that AKMA+ can be seamlessly integrated into existing 5G infrastructure by utilizing only the cryptographic functions specified in 5G standards for data processing.

• We formally model AKMA+ using the Tamarin Prover and demonstrate that it achieves the desired security and privacy properties using formal verification. In one case, namely, UE indistinguishability, we combine Tamarin verification and a cryptography proof.

• Our implementations and performance analysis indicate that AKMA+ incurs acceptable overheads compared with AKMA. The additional computation costs range from 1.688 ms to 78.030 ms, and extra bandwidth costs from 0.428 KB to 50.827 KB for a single AKMA+ session based on different parameter values selected for daily communication.

2 AKMA Protocol

We provide an overview of AKMA, covering its system architecture, main entities, key hierarchy, and protocol steps based on 3GPP TS 33.535 [32]. The glossaries and cryptographic notations are listed in Table 4 of Appendix A.

2.1 Architecture of AKMA

Fig. 1 shows the architecture of AKMA, where its main network elements (entities) [32, 36] are described below.

• ***User Equipment (UE)***: represents a subscriber of the AKMA protocol, which comprises Mobile Equipment (ME) and a Universal Integrated Circuit Card (UICC). A Universal subscriber identification module (USIM) resides in a UICC. ME can be any mobile device, such as a smartphone or IoT device, capable of incorporating the UICC.

• ***Home Public Land Mobile Network (HPLMN)*** refers to the network operated by a user's home service provider, where the user's subscription information and authentication credentials are stored and managed. Several functions are provided within HPLMN, including:

- ***United Data Management (UDM)***: stores AKMA subscription data.

- ***Authenticated Server Function (AUSF)***: generates and manages authentication keys, which are then used to derive subsequent keys for protecting communication between UE and AFs.

- ***AKMA Anchor Function (AAnF)***: stores authentication keys and anchor keys for the AKMA protocol, and generates application keys for the Application Functions (AFs).

- ***Network Exposure Function (NEF)***: authorizes external AF assessing the AKMA service and forwards AF's request to the AAnF.

• ***Application Function (AF)***: also known as the application provider, represents any application service which a user intends to access. An AF may be located within or outside HPLMN, as shown in Fig. 1.

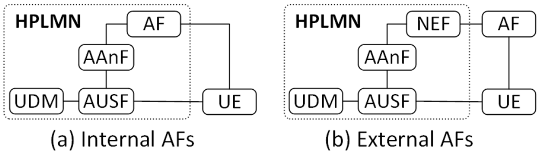


Figure 1: System Architecture of AKMA

2.2 AKMA Procedures

AKMA is designed to provide secure and efficient authentication and key management mechanisms for various applications within the 5G ecosystem. The objective is to establish a secure channel between UE and AF, with the authentication of UE delegated to its HPLMN. The details of the AKMA protocol are described below.

Primary Authentication. AKMA leverages the primary authentication procedures, 5G-AKA or EAP-AKA' [26], established by the 3GPP consortium, to create a secure link between UE and HPLMN. As depicted in Fig. 2, each UE subscribed to the 5G network is assigned a long-term shared key K with UDM. The primary authentication process results in the generation of an authentication key K_{AUSF}, which is derived by UDM and UE from K. This key serves as the foundation for generating further keys to secure communication sessions and services. Once generated, UDM sends K_{AUSF} to AUSF, completing the primary authentication process.

The AKMA framework builds on this security foundation K_{AUSF} by deriving additional keys for application-specific purposes. As depicted in Fig. 2, an AKMA anchor key (K_{AKMA}) and an AKMA key identifier (A-KID) are derived by AUSF and UE from K_{AUSF} using a key derivation function (KDF). The A-KID, which is globally unique, identifies the K_{AKMA} key of UE. AUSF then sends K_{AKMA} and A-KID to AAnF. Subsequently, the AKMA application key (K_{AF}) is derived by AAnF and UE from K_{AKMA}, ensuring the confidentiality of UE's communication with AF. These key derivation procedures are illustrated below.

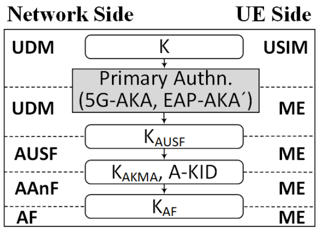


Figure 2: AKMA Key Hierarchy

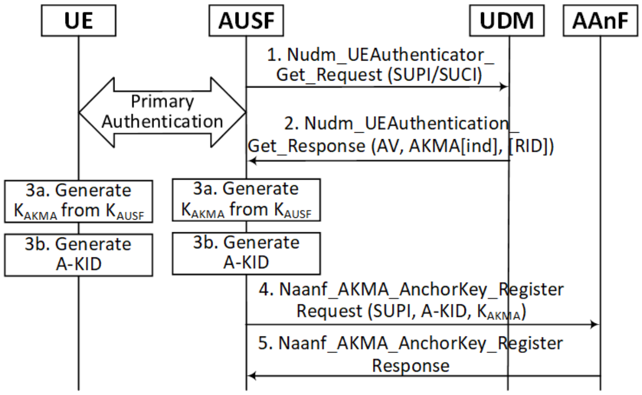

UE
AUSF
UDM
AAnF
1. Nudm_UEAuthenticator_
Get_Request (SUPI/SUCI)
Primary
Authentication
2. Nudm_UEAuthentication_
Get_Response (AV, AKMA[ind], [RID])
3a. Generate
K_{AKMA} from K_{AUSF}
3a. Generate
K_{AKMA} from K_{AUSF}
3b. Generate
A-KID
3b. Generate
A-KID
4. Naanf_AKMA_AnchorKey_Register
Request (SUPI, A-KID, K_{AKMA})
5. Naanf_AKMA_AnchorKey_Register
Response

Figure 3: Deriving K_{AKMA} After Primary Authentication

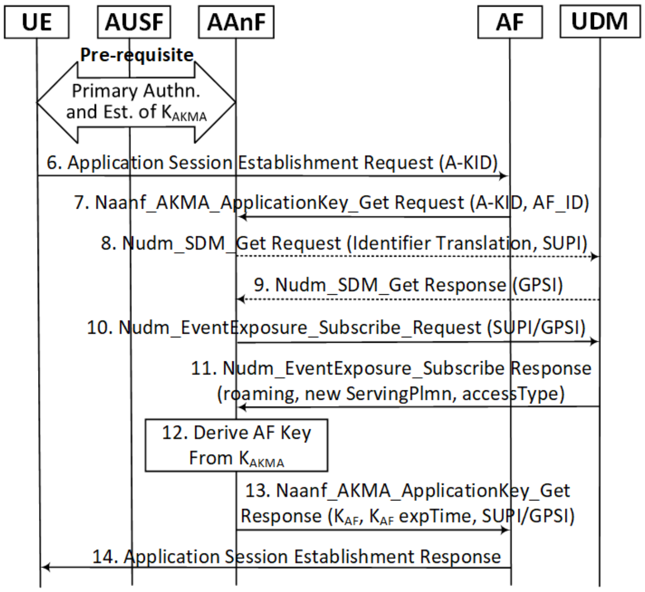

UE
AUSF
AAnF
AF
UDM
Pre-requisite
Primary Authn.
and Est. of K_{AKMA}
6. Application Session Establishment Request (A-KID)
7. Naanf_AKMA_ApplicationKey_Get Request (A-KID, AF_ID)
8. Nudm_SDM_Get Request (Identifier Translation, SUPI)
9. Nudm_SDM_Get Response (GPSI)
10. Nudm_EventExposure_Subscribe_Request (SUPI/GPSI)
11. Nudm_EventExposure_Subscribe Response
(roaming, new ServingPlmn, accessType)
12. Derive AF Key
From K_{AKMA}
13. Naanf_AKMA_ApplicationKey_Get
Response (K_{AF}, K_{AF} expTime, SUPI/GPSI)
14. Application Session Establishment Response

Figure 4: Deriving K_{AF} for AF located within HPLMN

Deriving AKMA Material. The process of deriving K_{AKMA} and A-KID is depicted in Fig. 3.

Steps 1-2. During the primary authentication procedure, AUSF interacts with UDM to fetch the authentication information of UE. Specifically, each UE is assigned a subscription permanent identifier (SUPI) and a subscription concealed identifier (SUCI) during primary authentication [26]. AUSF sends UE's SUPI/SUCI to UDM in a Nudm_UEAuthenticator_Get request. UDM shall then return a 5G authentication vector[2] (AV) to AUSF together with an AKMA indication[3] (AKMA[ind]) and routing indicator[4] (RID) in a Nudm_UEAuthentication_Get response.

Steps 3a-3b. The AKMA anchor key K_{AKMA} and the AKMA key identifier A-KID are derived by UE and AUSF from the shared key K_{AUSF}. A-KID shall be in NAI format [33], i.e. username@realm. The username part shall include RID and AKMA Temporary UE Identifier (A-TID), while the realm part shall include HPLMN Identifier.

Steps 4-5. AUSF transmits (K_{AKMA}, A-KID) along with the SUPI of UE to AAnF, and then gets a response.

Deriving AKMA Application Key for AF. Before communication between UE and AF starts, UE and AF shall derive an AKMA application key K_{AF} to establish a secure session.

Fig. 4 illustrates the procedure by which AF requests K_{AF} from AAnF when the AF is located within HPLMN. For the scenario where AF is located outside HPLMN, refer to §6.3 of TS 33.535 [32]. The only difference in this scenario is the involvement of NEF, which relays the messages transmitted

[2] The 5G AV consists of several elements to provide the necessary information for UE authentication and key derivation, such as RAND (random number), AUTN (authentication token), XRES* (expected response) and K_{SEAF} (serving network authentication key).

[3] AKMA indication is used to notify the network and UE about the support and use of AKMA procedures for securing the communication with AF.

[4] The RID is designed to direct the authentication traffic to the correct AUSF and UDM that manages the subscriber's authentication information and processes [26].

between AF and AAnF.

Step 6. UE initiates a request to access AF by transmitting A-KID to AF. As A-KID contains the identifier of HPLMN, AF needs to establish a connection with HPLMN.

Step 7. AF forwards both A-KID and its identifier (AF_ID) to AAnF.

Steps 8-9. These two steps are optional. Suppose AAnF determines this specific AF needs a Generic Public Subscription Identifier (GPSI) of UE, according to its local policy. In that case, AAnF requests UDM to fetch the GPSI of UE. If AF does not need the GPSI, AAnF shall skip Steps 8-9 and proceed with Step 10.

Steps 10-11. AAnF sends SUPI/GPSI to UDM to request the RoamingStatusReport. Then, UDM responds with UE's roaming status information.

Step 12. Recall that UE and AUSF derived $\mathsf{K}_{\mathsf{AKMA}}$ and A-KID using KDF in Steps 3a-3b, where KDF is the key derivation function (KDF) specified in Annex B.2 of TS 33.220 [31]. The A-KID serves as a globally unique identifier for $\mathsf{K}_{\mathsf{AKMA}}$. AUSF securely stores $\mathsf{K}_{\mathsf{AKMA}}$ and A-KID. Simultaneously, AAnF, which handles application-specific key management, also securely stores these key materials once they are shared by AUSF. When AF includes A-KID in the Naanf_AKMA_ApplicationKey_Get request in Step 7, AAnF uses A-KID to locate and retrieve the corresponding $\mathsf{K}_{\mathsf{AKMA}}$ from its secure storage. Using the $\mathsf{K}_{\mathsf{AKMA}}$, AAnF derives an AKMA application key (K_{AF}). This derivation process involves additional parameters, such as AF's identity (AF_ID), to ensure the uniqueness and security of K_{AF}.

Step 13. AAnF transmits K_{AF} and its expiration time as a response to the AF. Whether to send the SUPI or GPSI is determined by AAnF based on the local policy.

Step 14. Finally, AF sends an application session establishment response to UE.

Subsequently, UE and AF utilize K_{AF} as a secure session key to encrypt the data exchanged between them.

Note that UE and UDM/AUSF establish mutual authentication during the primary authentication protocol. Within the AKMA framework, AUSF selects AAnF as an anchor function. Although AAnF is also located in HPLMN, UE does not establish a direct connection with AAnF, and mutual authentication between UE and AAnF is absent.

The AKMA specification has been updated five times by 3GPP in the past 13 months, from June 2023 (V18.0.0) to July 2024 (V18.4.0). Version V18.3.0 introduced an improvement by adding a roaming status query in Steps 10-11. The difference between V18.4.0 and V18.3.0 is minor, with V18.4.0 requiring AF to disable application services and session encryption for UE when HPLMN detects that UE is roaming (Clause 6.8). AKMA+ is designed to enhance the security and privacy of the AKMA protocol based on the latest version (V18.4.0) published on July 8, 2024.

3 Key Issues, Security Goals, and Assumptions

3.1 Key Issues

3GPP TR 33.835 [24] outlines existing key issues confronting AKMA regarding security and privacy threats and corresponding requirements. This work deals with the following key security and privacy issues specified in TR 33.835.

- **Key Issue #3 (KI#3): Mutual authentication between UE and anchor function**. UE and AAnF must mutually authenticate each other based on 5G credentials using the 5G authentication framework.
- **Key Issue #5 (KI#5): User privacy**. SUPI shall not be revealed to AFs. SUPI should be protected in the data flow exchanged among UE and any other entities in AKMA.
- **Key Issue #6 (KI#6): Secure communication between UE and AF**. UE and AF must derive a session key for end-to-end security using keys derived from the 3GPP network, ensuring that sensitive data transferred between UE and AF remains inaccessible to the 3GPP network.
- **Key Issue #7 (KI#7): Protecting subscriber's personal information in control and data traffic**. The subscriber's personal information should be protected in both control and data traffic utilized within the AKMA architecture.
- **Key Issue #16 (KI#16): Application key freshness of AKMA**. The freshness of keys used between an UE and any AF must be ensured.

3.2 Design Goals

The security and privacy requirements stated in the above key issues can be summarized as the following design goals.

G1. UE Indistinguishability. The adversary cannot distinguish which UE among multiple UEs has accessed an application function. This ensures the preservation of *user privacy* and *protecting subscriber's personal information* as specified in KI#5 and KI#7.

G2. Secrecy of Session Key. Privacy-enhanced AKMA should establish a secret session key between UE and AF while ensuring the confidentiality of the key exchange process. This ensures the preservation of *secure communication between UE and AF* as specified in KI#6.

G3. Forward Secrecy. Even if the long-term secret keys are compromised at some time, the security of previous session keys remains intact to protect the confidentiality of past communication sessions [9]. This ensures the preservation of *secure communication between UE and AF* and *application key freshness of AKMA* as specified in KI#6 and KI#16.

G4. Mutual Authentication between UE and AAnF. UE and AAnF must mutually authenticate each other to verify the legitimacy of each other's identity to prevent impersonation attacks. This ensures the preservation of *mutual authentication between UE and anchor function* as specified in KI#3.

3.3 Security Assumptions

We define security assumptions that cover the requirements outlined in 3GPP TS 33.535 [32] for the security of AKMA and in TS 33.501 [26] for the security of 5G communication. They also encompass the threat model described by key issues in TR 33.835 [24], including KI#3, #5, #6, #7 and #16. We exclude other key issues because they are related to AKMA architecture framework, interface, API, and regulation compliance that cannot be addressed from the protocol perspective, and are therefore out of the scope of this work.

Security Assumptions on Network Entities. The following assumptions are made according to TS 33.501 [26], TS 33. 535 [32] and TR 33.835 [24].

- UDM, AUSF and NEF are *honest* entities since they are core entities residing within the 5G Core Network.

- AAnF is a *covert* entity[5] that honestly adheres to any prescribed protocols within 5G networks, such as provisioning authentic credentials and correctly executing authentication procedures as required. Nevertheless, AAnF may get access to sensitive data transferred between UE and AFs, which is protected by the key derived by AAnF. AAnF may also inject malicious packages, which UE or AF would assess as cryptographically correct (KI#6). An AAnF may communicate with UE to obtain the SUPI or other personal identifiers, leading to the loss and exposure of user privacy (KI#5).

- AFs are *malicious* entities that may compromise UE's privacy by identifying, profiling, and tracking UE's access behaviors (KI#5).

- UE is a *rational* entity[6] that aligns with the prescribed 5G protocols. However, a UE may impersonate another UE to communicate with AAnF and access the AKMA services (KI#3). The SUPI of any UE may be leaked to unauthorized parties for profiling and tracking (KI#5). Identifying information revealed in control or data traffic may enable unauthorized parties to identify subscribers (KI#7).

Security Assumptions on Communication Channels. The wired channel within HPLMN is a secure *end-to-end core network interconnection channel*, as specified in TS 33.501 [26], which encompasses the communication between the entities UDM, AUSF, and AAnF. According to TS 33.501 [26], mutual authentication between AAnF and AF occurs before running AKMA using the TLS protocol [26]. The public communication channel between UE and AF may be eavesdropped on by passive attackers. Active attackers may tamper, intercept, and manipulate messages transferred between UE and AF; they may replay previously intercepted communication and launch attacks by impersonating a fake entity (KI#8).

Security Assumptions on Components. According to TS 33.501 [26], and TS 33. 535 [32], it is assumed that attackers cannot compromise the entities UDM, AUSF, and NEF. This assumption extends to long-term secrets (e.g., UE's root key K in UDM) and session state containing temporary secrets (e.g., $\mathsf{K}_{\mathsf{AUSF}}$). We assume that the long-term key K of honest UE cannot be stolen by attackers.

Discussion of Security Assumptions on Network Entities. Clauses 4.3 and 5.9 of TS 33.501 [26] designate the AUSF, located within the trusted 5G core network, as a fully trusted entity. The 5G core manages key operations such as user authentication and data routing, supported by the security measures outlined in Clauses 5.2 and 6.1 of TS 33.501 and Clauses 4 and 5.1 of TS 33.210 [25]. In contrast, the AAnF, located in the HPLMN but outside the 5G core, lacks the same trust designation, implying potential security risks.

Relevant clauses suggest that the AAnF could be a point of compromise due to its role in key management. Clause 4.2.1 of TS 33.535 [32] defines the AAnF's responsibility for generating key material for secure communication between the UE and AF. Clause 4.6.2 of TR 33.835 [24] acknowledges that sensitive data could be exposed if the AAnF or pre-shared key material is compromised.

Moreover, Clause 4.6.3 of TR 33.835 highlights mitigating measures, proposing independent session key derivation between the UE and AF to prevent unauthorized access by the 3GPP network, including the AAnF. Thus, the AAnF's critical role in key management and potential access to sensitive data warrant considering it as a potential security threat in a comprehensive threat model.

4 AKMA+

We now introduce AKMA+, a security and privacy-enhanced and standard-compatible protocol to address the security and privacy-threatening key issues described in §3.1.

4.1 Design Idea

We initially analyze the root causes of the security and privacy threats posed by KI#5, #7, #6, #16 and #3. Subsequently, we introduce targeted countermeasures to address these causes, providing a comprehensive solution.

Root Causes of KI#5, #7. In the AKMA protocol, the subscription identifiers SUPI/GPSI are included in the Nudm_SDM_Get_Request/Response (Steps 8-9), Nudm_EventExposure_Subscribe_Request (Step 10), and the Naanf_AKMA_ApplicationKey_Get_Response (Step 13). However, the SUPI and GPSI are sensitive information of UE, as they pose a risk of attackers identifying and tracing individual UEs through their identifiers (KI#5). Another significant concern is unprotected identifying information, such as AKMA key identifier (A-KID), that is transmitted in data traffic. Clause

[5] A covert entity will only act maliciously if it believes it can do so without being detected; it may behave maliciously to gain unauthorized access to information or tamper data transmitted on public communication channels.

[6] A rational entity is motivated by personal gain and acts based on a calculated cost-benefit analysis; it makes decisions that maximize its expected utility or profit, considering both the potential gains and the risks or costs associated with its actions.

6.2.2 of TS 33.535 [32] specifies that A-KID serves as a temporary user identifier in cases where AAnF responds with no UE identifier. The exposure of UE's identifiers and A-KID enables UE linking attacks [2,40] (KI#7). External or internal attackers could exploit A-KID and the underlying $\mathsf{K}_{\mathsf{AKMA}}$ to track a user's behaviors of accessing AFs.

Targeted Countermeasure 1. To address KI#5, #7 and meet G1 (UE indistinguishability), AKMA+ avoids transmitting UE's identifiers SUPI and GPSI (see Fig. 6-7). To obfuscate the mapping relationship between A-KID and UE, AUSF generates a set of ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) pairs for each UE, registering them with AAnF, where $i \in \{1,...,n\}$ and n is the number of key-identifier pairs generated for each UE in deriving AKMA keys each time after primary authentication. AUSF shuffles the pairs of ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) among users, preventing AAnF from attributing any received A-KID to a specific UE. These mechanisms ensure that both external and internal attackers cannot track the behaviors of UE for accessing AFs, and AFs cannot distinguish which UE has accessed them.

Root Causes of KI#6, #16. In the AKMA protocol, AF requests the AKMA application key from AAnF. Consequently, AAnF can access all the data transmitted between UE and AF utilizing the application key. This presents a significant risk as it enables AAnF to inject malicious payloads that both UE and AF may accept as legitimate, given that the malicious payloads are encrypted using the application key (KI#6). If the application key lacks freshness, AAnF may impersonate UE when communicating with AF, and vice versa, compromising the security of the communication (KI#16).

Targeted Countermeasure 2. To address KI#6, #16 and meet G2-G3 (secrecy of session key, forward secrecy), we derive a privacy-hardened application key $\mathsf{K}'_{\mathsf{AF},i}$ from the AKMA application key $\mathsf{K}_{\mathsf{AF},i}$ and a Diffie-Hellman (DH) shared key K_{s} using KDF function (see Fig. 5 and 7), where $\mathsf{K}'_{\mathsf{AF},i} \leftarrow \mathsf{KDF}(\mathsf{K}_{\mathsf{AF},i}, \mathsf{K}_{\mathsf{s}})$. To establish the DH shared key K_{s}, UE selects an ephemeral secret key u and computes an ephemeral public key $U = uG$, where G is a generator of group $\mathbb{G}$ (see Step 6 in Fig. 7). AF also chooses an ephemeral secret key v and computes $V = vG$. DH shared key is then calculated as $\mathsf{K}_{\mathsf{s}} = vU$ and $\mathsf{K}_{\mathsf{s}} = uV$ by AF and UE.

To prevent Man-in-the-Middle attacks on its DH key exchange protocol, AKMA+ protects the privacy of U using a public key encryption algorithm PKEnc and the generated ciphertext is $\mathsf{CT}_{\mathsf{UE}\rightarrow\mathsf{AF}} = \mathsf{PKEnc}_{\mathsf{PK}_{\mathsf{AF}}}(U, a_1)$ (Step 6), where $\mathsf{PK}_{\mathsf{AF}}$ is AF's public key and a_1 is a nonce used to ensure session freshness. This ciphertext $\mathsf{CT}_{\mathsf{UE}\rightarrow\mathsf{AF}}$ is decrypted by AF using its secret key $\mathsf{SK}_{\mathsf{AF}}$. To ensure the authenticity of V, AF utilizes its secret key $\mathsf{SK}_{\mathsf{AF}}$ to sign (V, a_1). This signature is verified by UE using AF's public key $\mathsf{PK}_{\mathsf{AF}}$ (Step 14).

The privacy-hardened application key $\mathsf{K}'_{\mathsf{AF},i}$, which provides forward secrecy, safeguards the communication channel between UE and AF against eavesdropping and tampering by AAnF and other attackers.

Note that AKMA does not provide forward secrecy, as all sessions between the UE and AF are secured using the same application key, K_{AF}, until it is refreshed through a new authentication process. If K_{AF} is compromised at any point, the contents of all previous sessions are at risk of exposure.

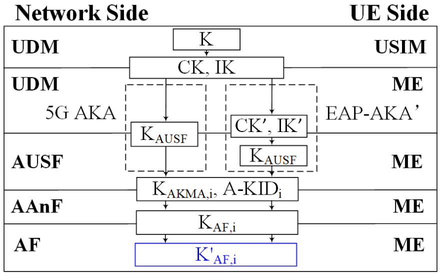


Figure 5: Key Hierarchy Generation in AKMA+

Root Cause of KI#3. To guarantee the privacy of AKMA, mutual authentication between UE and AAnF within the 5G authentication framework must occur initially. However, after primary authentication, the AKMA protocol does not provide a direct communication link between AAnF and UE (see Fig. 4). This implies that the mutual authentication between UE and AAnF should be accomplished using AF as a mediator. The challenge is that AF may tamper with the transferred content without being detected.

Targeted Countermeasure 3. To address KI#3 and meet G4, UE and AAnF should mutually authenticate each other based on the secret key established by the 5G primary authentication protocol (5G-AKA or EAP-AKA') as specified in clause 4.3.3 of TR 33.835. Considering potential threats posed by AF, AKMA+ selects the AKMA anchor key $\mathsf{K}_{\mathsf{AKMA},i}$ shared by AAnF and UE as the root of trust for mutual authentication. Specifically, UE selects a nonce a_2 to ensure session freshness, and encrypts (AF_ID, a_2) to ciphertext $\mathsf{CT}_{\mathsf{UE}\rightarrow\mathsf{AAnF}}$ using $\mathsf{K}_{\mathsf{AKMA},i}$ and an authenticated encryption algorithm AEnc (Step 6). Including AF_ID ensures this ciphertext cannot be replayed in sessions with a different AF. Upon receiving $\mathsf{CT}_{\mathsf{UE}\rightarrow\mathsf{AAnF}}$ relayed by AF (Step 7), AAnF decrypts it using $\mathsf{K}_{\mathsf{AKMA},i}$ and an authenticated decryption algorithm ADec (Step 12). It then verifies if the recovered AF_ID matches the one sent by AF in Step 7. Subsequently, AAnF generates a response by encrypting (AF_ID, $a_2 + 1$) using $\mathsf{K}_{\mathsf{AKMA},i}$ in Step 12. Upon receiving the response, UE verifies that the recovered nonce matches $(a_2 + 1)$ in Step 14, thereby completing mutual authentication between UE and AAnF.

4.2 Construction of AKMA+

In Fig. 6-7, we present the detailed construction of the AKMA+ protocol for the scenario where AF is located within HPLMN. The scenario where AF is located outside HPLMN can be easily derived from Fig. 7. The only difference is NEF's involvement in relaying the messages transmitted between AF and AAnF, which is omitted here. AKMA+ is com-

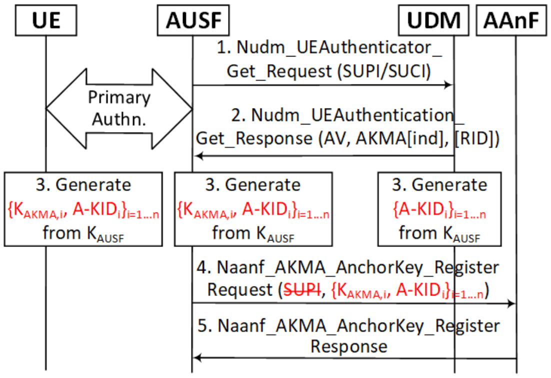

UE
AUSF
UDM
AAnF
1. Nudm_UEAuthenticator_Get_Request (SUPI/SUCI)
Primary Authn.
2. Nudm_UEAuthentication_Get_Response (AV, AKMA[ind], [RID])
3. Generate $\{K_{AKMA,i}, A\text{-}KID_i\}_{i=1...n}$ from K_{AUSF}
3. Generate $\{K_{AKMA,i}, A\text{-}KID_i\}_{i=1...n}$ from K_{AUSF}
3. Generate $\{A\text{-}KID_i\}_{i=1...n}$ from K_{AUSF}
4. Naanf_AKMA_AnchorKey_Register Request (~~SUPI~~, $\{K_{AKMA,i}, A\text{-}KID_i\}_{i=1...n}$)
5. Naanf_AKMA_AnchorKey_Register Response

Figure 6: AKMA+: Deriving AKMA Anchor Keys

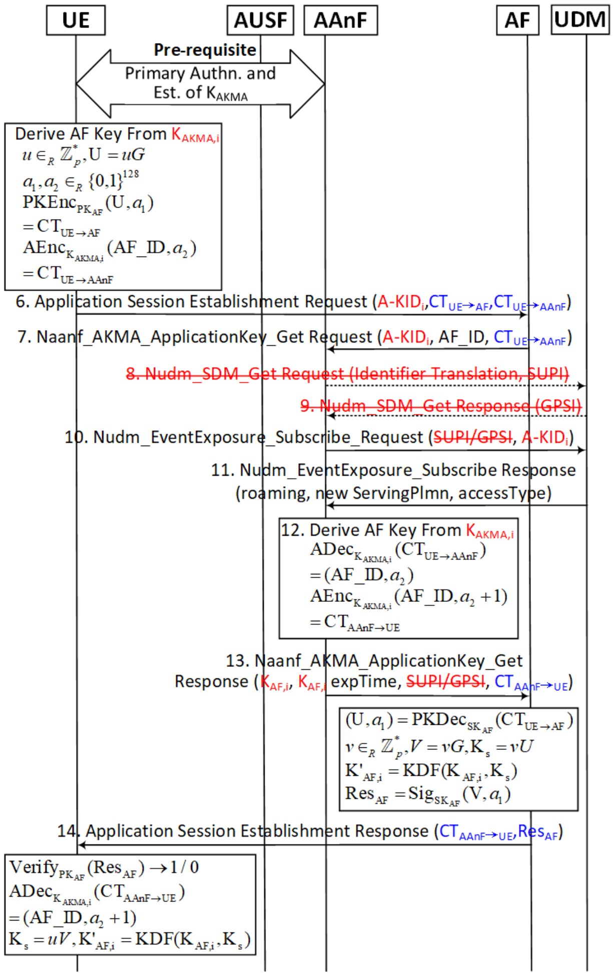

UE
AUSF
AAnF
AF
UDM
Pre-requisite
Primary Authn. and Est. of K_{AKMA}
Derive AF Key From $K_{AKMA,i}$
$u \in_R \mathbb{Z}_p^*, U = uG$
$a_1, a_2 \in_R \{0,1\}^{128}$
$PKEnc_{PK_{AF}}(U, a_1) = CT_{UE \to AF}$
$AEnc_{K_{AKMA,i}}(AF\_ID, a_2) = CT_{UE \to AAnF}$
6. Application Session Establishment Request ($A\text{-}KID_i$, $CT_{UE \to AF}$, $CT_{UE \to AAnF}$)
7. Naanf_AKMA_ApplicationKey_Get Request ($A\text{-}KID_i$, AF_ID, $CT_{UE \to AAnF}$)
~~8. Nudm_SDM_Get Request (Identifier Translation, SUPI)~~
~~9. Nudm_SDM_Get Response (GPSI)~~
10. Nudm_EventExposure_Subscribe_Request (~~SUPI/GPSI~~, $A\text{-}KID_i$)
11. Nudm_EventExposure_Subscribe Response (roaming, new ServingPlmn, accessType)
12. Derive AF Key From $K_{AKMA,i}$
$ADec_{K_{AKMA,i}}(CT_{UE \to AAnF}) = (AF\_ID, a_2)$
$AEnc_{K_{AKMA,i}}(AF\_ID, a_2 + 1) = CT_{AAnF \to UE}$
13. Naanf_AKMA_ApplicationKey_Get Response ($K_{AF,i}$, $K_{AF,i}$ expTime, ~~SUPI/GPSI~~, $CT_{AAnF \to UE}$)
$(U, a_1) = PKDec_{SK_{AF}}(CT_{UE \to AF})$
$v \in_R \mathbb{Z}_p^*, V = vG, K_s = vU$
$K'_{AF,i} = KDF(K_{AF,i}, K_s)$
$Res_{AF} = Sig_{SK_{AF}}(V, a_1)$
14. Application Session Establishment Response ($CT_{AAnF \to UE}$, Res_{AF})
$Verify_{PK_{AF}}(Res_{AF}) \to 1/0$
$ADec_{K_{AKMA,i}}(CT_{AAnF \to UE}) = (AF\_ID, a_2 + 1)$
$K_s = uV, K'_{AF,i} = KDF(K_{AF,i}, K_s)$

Figure 7: AKMA+: Privacy-hardened Application Key Gen.

pliant with AKMA specifications, as *the message flows are consistent*. This means that AKMA+ adheres to the same commands, message flows, and data formats defined by 3GPP, ensuring it operates seamlessly within the existing 5G infrastructure. Although we have enhanced the security and privacy measures through revised data processing within each entity, these improvements are achieved using only the cryptographic functions specified in 5G standards, maintaining full compatibility with the AKMA protocol.

4.2.1. Achieving G1

The modified contents for dealing with KI#5, #7 and achieving G1 are highlighted in red in Fig. 6-7.

Tracking Attacks in AKMA. We first analyze how the attackers track UE's behavior.

(1) UE's identifier SUPI/GPSI is transmitted in Steps 8, 9, 10 and 13, which enables attackers to launch tracking attacks.

(2) UE sends an AKMA key identifier A-KID to AF in the *Application Session Establishment Request* in Step 6. Here, A-KID is sent over a public channel and can be eavesdropped on by attackers. In Step 7, AF sends A-KID and AF_ID to AAnF. By tracking A-KID, attackers can determine that UE with A-KID has accessed an AF with AF_ID. Moreover, UE sends the same A-KID to different AFs to request access to various applications.

(3) AAnF uses A-KID to retrieve K_{AKMA} for deriving the AF key (Step 12), who can track UE's behavior utilizing the pair (K_{AKMA},A-KID).

Resist Tracking Attacks. AKMA+ employs the following mechanisms (shown in Fig. 6-7) to resist tracking attacks and achieve UE indistinguishability (G1).

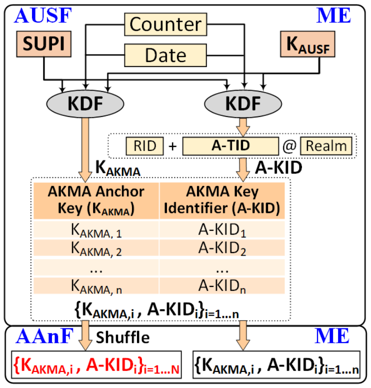
(1) As per clause 6.2 of TS 33.535, Steps 8-9 are optional for retrieving GPSI of UE based on HPLMN's local policy. To conceal UE's identifier, AKMA+ avoids retrieving GPSI from UDM, thereby skipping Steps 8-9. Likewise, the SUPI/GPSI should be excluded from Steps 4, 10 and 13.

(2) To prevent A-KID based tracking attacks, UE and AUSF each generates $\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1,...,n}$, and UDM generates $\{\text{A-KID}_i\}_{i=1,...,n}$ from K_{AUSF} in Step 3, where n is the number of key-identifier pairs generated for each UE in deriving AKMA keys each time after primary authentication. UDM stores $(\text{SUPI}, \{\text{A-KID}_i\}_{i=1,...,n})$ in its database. In Step 4, AUSF transmits $\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1,...,n}$ to AAnF. In Step 6, UE selects one A-KID$_i$ from these pairs and sends it to AF, where the selected A-KID$_i$ must not be reused to ensure the indistinguishability of UE. AAnF sends A-KID$_i$ to UDM to request the RoamingStatusReport of UE (in Step 10). The AKMA key-identifier pair generation algorithm is shown in Algo. 1 and Fig. 8. Following 3GPP TS 33.535 [32], $K_{AKMA,i}$ and A-KID$_i$ are derived from K_{AUSF} using a KDF, where the inputs include K_{AUSF}, a hexadecimal identifier (0x80/0x81), a string identifier ("AKMA"/"A-TID"), and SUPI. To generate n key-identifier pairs, AKMA+ includes Counter and Date as input to the KDF, where the value of Counter changes sequentially from 1 to n. The concrete generation process of ($K_{AKMA,i}$, A-KID$_i$) pairs will be illustrated later.

(3) If AUSF sends $\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1,...,n}$ for a specific UE to AAnF in Step 4, AAnF can track the UE's behaviors by

Algorithm 1 Derive $\mathsf{K_{AKMA}}$ and A-KID pairs

```
Input: $\mathsf{K_{AUSF}}, \mathsf{SUPI}, \mathsf{Date}, n, \mathsf{RID}, \mathsf{Realm}$
Output: $\mathsf{K_{AKMA}}$ and A-KID pairs
for $i = 1, \cdots, n$ do
    $\mathsf{Counter} = i$;
    /* Generate $\mathsf{K}_{\mathsf{AKMA},i}$ */
    $\mathsf{K}_{\mathsf{AKMA},i} = \mathsf{KDF}(\mathsf{K_{AUSF}}, 0x80, \text{“}\mathsf{AKMA}\text{”}, \mathsf{SUPI}, \mathsf{Counter}, \mathsf{Date})$;
    /* Generate A-KID$_i$ */
    /* Step 1: generate A-TID$_i$ */
    $\mathsf{A\text{-}TID}_i = \mathsf{KDF}(\mathsf{K_{AUSF}}, 0x81, \text{“}\mathsf{A\text{-}TID}\text{”}, \mathsf{SUPI}, \mathsf{Counter}, \mathsf{Date})$;
    /* Step 2: concatenate RID, A-TID$_i$, “@" and Realm */
    $\mathsf{A\text{-}KID}_i = \mathsf{RID}||\mathsf{A\text{-}TID}_i||\text{“@”}||\mathsf{Realm}$;    ▷ concatenation notation: ||
end for
return $\{\mathsf{K}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}_i\}_{i=1,\dots n}$
```


AUSF
ME
SUPI
Counter
Date
K_{AUSF}
KDF
KDF
RID + A-TID @ Realm
K_{AKMA}
A-KID
$\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1...n}$
AAnF
Shuffle
ME
$\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1...N}$
$\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1...n}$

AKMA Anchor Key (K_{AKMA})	AKMA Key Identifier (A-KID)
$K_{AKMA,1}$	A-KID$_1$
$K_{AKMA,2}$	A-KID$_2$
...	...
$K_{AKMA,n}$	A-KID$_n$

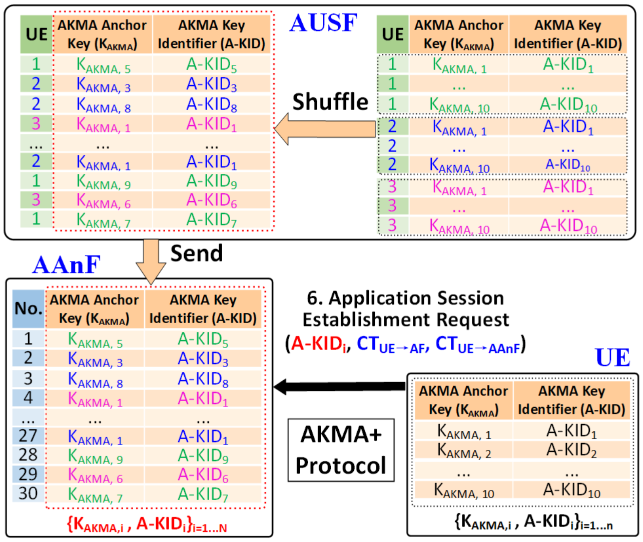
Figure 8: AKMA+: Derive $\mathsf{K_{AKMA}}$ and A-KID pairs

distinguishing the usage of A-KID$_i$ from the same AKMA key-identifier pair set. To resist tracking attacks, AUSF runs Algo. 2 to shuffle the ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) pairs among different UEs (with the same RID and Realm) to disrupt their internal correlation. Suppose AUSF generates $\{\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i\}_{i=n}$ for the k users, where k is the number of users. Then, the total number of ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) pairs for the k users is $N = n \cdot k$. AUSF thoroughly shuffles the N pairs belonging to the k users, renumbers them and forwards the shuffled pairs (without UE identifiers) to AAnF. Even if the same UE sends multiple requests, AAnF cannot distinguish the originating UE based on A-KID$_i$, $\mathsf{K}_{\mathsf{AKMA},i}$ and the shuffled pairs. Fig. 9 provides an example to illustrate this shuffling mechanism, where $n = 10$, $k = 3$, and $N = 30$. Different colors are used to distinguish AKMA key-identifier pairs of other users.

Algorithm 2 Shuffle $\mathsf{K_{AKMA}}$ and A-KID pairs

```
Input: $\{\mathsf{K}^{(1)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(1)}_i\}_{i=1,\dots n}, \cdots, \{\mathsf{K}^{(k)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(k)}_i\}_{i=1,\dots n}$;
    /* Belong to UE$_1$, ..., UE$_k$ (with the same RID and Realm) */
Output: Shuffled $\mathsf{K_{AKMA}}$ and A-KID pairs of $k$ UEs;
Set $N = n \cdot k$;
Randomly shuffle all the input $\mathsf{K_{AKMA}}$ and A-KID pairs;
Renumber the shuffled pairs;
return $\{\mathsf{K}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}_i\}_{i=1,\dots N}$
```

Note that AKMA+ requires AUSF to shuffle the key-identifier pairs among k UEs with the same RID and Realm. Theorem 1 in §5.1 and its proof demonstrate that UE indistinguishability is guaranteed when $k \geqslant 2$.


AUSF
Shuffle
Send
AAnF
6. Application Session Establishment Request
(A-KID$_i$, $CT_{UE\to AF}$, $CT_{UE\to AAnF}$)
UE
AKMA+ Protocol
$\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1...N}$
$\{K_{AKMA,i}, \text{A-KID}_i\}_{i=1...n}$

UE	AKMA Anchor Key (K_{AKMA})	AKMA Key Identifier (A-KID)
1	$K_{AKMA,1}$	A-KID$_1$
1	...	...
1	$K_{AKMA,10}$	A-KID$_{10}$
2	$K_{AKMA,1}$	A-KID$_1$
2	...	...
2	$K_{AKMA,10}$	A-KID$_{10}$
3	$K_{AKMA,1}$	A-KID$_1$
3	...	...
3	$K_{AKMA,10}$	A-KID$_{10}$

UE	AKMA Anchor Key (K_{AKMA})	AKMA Key Identifier (A-KID)
1	$K_{AKMA,5}$	A-KID$_5$
2	$K_{AKMA,3}$	A-KID$_3$
2	$K_{AKMA,8}$	A-KID$_8$
3	$K_{AKMA,1}$	A-KID$_1$
...	...	...
2	$K_{AKMA,1}$	A-KID$_1$
1	$K_{AKMA,9}$	A-KID$_9$
3	$K_{AKMA,6}$	A-KID$_6$
1	$K_{AKMA,7}$	A-KID$_7$

No.	AKMA Anchor Key (K_{AKMA})	AKMA Key Identifier (A-KID)
1	$K_{AKMA,5}$	A-KID$_5$
2	$K_{AKMA,3}$	A-KID$_3$
3	$K_{AKMA,8}$	A-KID$_8$
4	$K_{AKMA,1}$	A-KID$_1$
...	...	...
27	$K_{AKMA,1}$	A-KID$_1$
28	$K_{AKMA,9}$	A-KID$_9$
29	$K_{AKMA,6}$	A-KID$_6$
30	$K_{AKMA,7}$	A-KID$_7$

AKMA Anchor Key (K_{AKMA})	AKMA Key Identifier (A-KID)
$K_{AKMA,1}$	A-KID$_1$
$K_{AKMA,2}$	A-KID$_2$
...	...
$K_{AKMA,10}$	A-KID$_{10}$

Figure 9: AKMA+: Shuffling of $\mathsf{K_{AKMA}}$ and A-KID pairs

Derivation of ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) pairs. The detailed derivation process for the ($\mathsf{K}_{\mathsf{AKMA},i}$, A-KID$_i$) pairs is outlined below.

(1) Derivation of $\mathsf{K}_{\mathsf{AKMA},i}$. According to Annex A.2 of TS 33.535 [32], when deriving a $\mathsf{K_{AKMA}}$ from $\mathsf{K_{AUSF}}$, specific parameters must be utilized to construct the input S for the key derive function (KDF)[7]: S = FC||P0||L0||P1||L1, FC = 0x80, P0 = “AKMA", L0 = length of “AKMA", P1 = SUPI, L1 = length of SUPI. The input key KEY is defined as $\mathsf{K_{AUSF}}$. To generate a set of AKMA key-identifier pairs, AKMA+ incorporates *Counter* and *Date* as additional input parameters, where *Date* specifies the key generation time and *Counter* tracks the number of keys generated each time for a user. This implies that the input S should encompass four additional parameters: P2 = Counter, L2 = length of Counter, P3 = Date, L3 = length of Date, where S = FC||P0||L0||P1||L1||P2||L2||P3||L3.

(2) Derivation of A-KID$_i$. According to clause 6.1 of TS 33.535 [32], A-KID shall be in NAI format [33], i.e. username@realm. The username part shall include RID (Routing InDicator) and A-TID (AKMA Temporary UE Identifier), and a realm part shall consist of HPLMN Identifier.

2.1) RID. As per 3GPP TS 23.003 [35], the home network operator allocates the RID, typically comprising 1 to 4 decimal digits, and is stored in UICC. Along with the Home Network Identifier, it facilitates the routing of network signaling to AUSF and UDM for the subscriber service. If no Routing Indicator is configured on UICC or ME, this field defaults to 0. The RID does not reveal any personal information about UE and does not require modification.

[7] The KDF requires an input S and a key KEY as specified in TS 33.501.

2.2) Derivation of A-TID$_i$. According to Annex A.3 of TS 33.535 [32], when deriving A-TID from $\mathsf{K}_{\mathsf{AUSF}}$, specific parameters must be utilized to construct the input S for the KDF: S = FC||P0||L0||P1||L1, FC = 0x81, P0 = "A-TID", L0 = length of "A-TID", P1 = SUPI; L1 = length of SUPI. Another input key KEY is defined as $\mathsf{K}_{\mathsf{AUSF}}$. Similar to the generation of $\mathsf{K}_{\mathsf{AKMA},i}$, AKMA+ integrates *Counter* and *Date* as supplementary input parameters, necessitating that the input S should include four additional parameters: P2 = Counter, L2 = length of Counter, P3 = Date, L3 = length of Date, where S = FC||P0||L0||P1||L1||P2||L2||P3||L3.

A-KID$_i$ is generated by concatenating RID, A-TID$_i$, "@" and the realm part (see Fig. 8).

4.2.2. Achieving G2-G4

The modified contents in the data flow for addressing KI #6, #16 and #3 and achieving G2-G4 are highlighted in blue in Fig. 7. Let G be a generator of group $\mathbb{G}$. An authenticated encryption scheme contains encryption/decryption algorithms $\mathsf{AEnc}/\mathsf{ADec}$, simultaneously providing confidentiality, integrity, and authenticity assurances. A public key scheme consists of encryption/decryption algorithms $\mathsf{PKEnc}/\mathsf{PKDec}$. A signature scheme consists of signature/verification algorithms $\mathsf{Sig}/\mathsf{Verify}$. Next, we explain the cryptographic mechanisms for generating the privacy-hardened application key.

Step 6. UE derives AF key $\mathsf{K}_{\mathsf{AF},i}$ from $\mathsf{K}_{\mathsf{AKMA},i}$, following Annex A.4 of TS 33.535. UE randomly selects a nonce $a_1 \in \{0,1\}^{128}$ (to ensure session freshness for resisting replay attacks). It then selects an ephemeral DH secret key $u \in_R \mathbb{Z}_p^*$ and calculates $U \leftarrow uG$, where G is a generator of group $\mathbb{G}$. Then, UE encrypts (U, a_1) using AF's public key $\mathsf{PK}_{\mathsf{AF}}$ and public key encryption algorithm PKEnc. The generated ciphertext for AF is denoted as $\mathsf{CT}_{\mathsf{UE}\to\mathsf{AF}} = \mathsf{PKEnc}_{\mathsf{PK}_{\mathsf{AF}}}(U, a_1)$.

For mutual authentication between UE and AAnF (G4), UE randomly selects a nonce a_2 (as a challenge) and encrypts (AF_ID, a_2) using $\mathsf{K}_{\mathsf{AKMA},i}$. The generated ciphertext is denoted as $\mathsf{CT}_{\mathsf{UE}\to\mathsf{AAnF}} = \mathsf{AEnc}_{\mathsf{K}_{\mathsf{AKMA},i}}(\text{AF\_ID}, a_2)$.

When UE initiates communication with an AF, it shall include A-KID$_i$ and ciphertexts $(\mathsf{CT}_{\mathsf{UE}\to\mathsf{AF}}, \mathsf{CT}_{\mathsf{UE}\to\mathsf{AAnF}})$ in an *Application Session Establishment Request*.

Step 7. AF sends to AAnF the received A-KID$_i$, its identity AF_ID, and the received ciphertext $\mathsf{CT}_{\mathsf{UE}\to\mathsf{AAnF}}$ in Naanf_AKMA_ApplicationKey_ Get_Request.

Steps 10-11. AAnF sends $\mathsf{A}\text{-}\mathsf{KID}_i$ to UDM to request the RoamingStatusReport. UDM then searches its database for the SUPI corresponding to $\mathsf{A}\text{-}\mathsf{KID}_i$ and responds with UE's roaming status information.

Step 12. AAnF derives an application key $\mathsf{K}_{\mathsf{AF},i}$ from $\mathsf{K}_{\mathsf{AKMA},i}$. AAnF decrypts ciphertext $\mathsf{CT}_{\mathsf{UE}\to\mathsf{AAnF}}$ using $\mathsf{K}_{\mathsf{AKMA},i}$ and the authenticated decryption algorithm ADec to recover (AF_ID, a_2). It then verifies if the recovered AF_ID matches the one sent by AF in Step 7. AAnF terminates this session if the verification fails. For mutual authentication between UE and AAnF (G4), AAnF encrypts (AF_ID, $a_2 + 1$) using $\mathsf{K}_{\mathsf{AKMA},i}$ (as a response), with the resulting ciphertext denoted as $\mathsf{CT}_{\mathsf{AAnF}\to\mathsf{UE}}$.

Step 13. AAnF sends Naanf_AKMA_ApplicationKey_Get_ Response to AF with $\mathsf{K}_{\mathsf{AF},i}$, the $\mathsf{K}_{\mathsf{AF},i}$ expiration time and $\mathsf{CT}_{\mathsf{AAnF}\to\mathsf{UE}}$.

Step 14. AF decrypts $\mathsf{CT}_{\mathsf{UE}\to\mathsf{AF}}$ using its secret key $\mathsf{SK}_{\mathsf{AF}}$ to recover (U, a_1). AF selects an ephemeral DH secret key $v \in_R \mathbb{Z}_p^*$ and calculates $V = vG$. A DH shared key is computed as $\mathsf{K}_{\mathsf{s}} = vU$, and a privacy-hardened application key as $\mathsf{K}'_{\mathsf{AF},i} = \mathsf{KDF}(\mathsf{K}_{\mathsf{AF},i}, \mathsf{K}_{\mathsf{s}})$. AF generates $\mathsf{Res}_{\mathsf{AF}}$ containing the signed hash result of (V, a_1) using its private key $\mathsf{SK}_{\mathsf{AF}}$, along with the plaintext of these two elements. AF sends $(\mathsf{CT}_{\mathsf{AAnF}\to\mathsf{UE}}, \mathsf{Res}_{\mathsf{AF}})$ to UE in *Application Session Establishment Response*.

Upon receiving the response, UE verifies the signatures of (V, a_1) in $\mathsf{Res}_{\mathsf{AF}}$ and utilizes $\mathsf{K}_{\mathsf{AKMA},i}$ to recover (AF_ID, $a_2 + 1$) from $\mathsf{CT}_{\mathsf{AAnF}\to\mathsf{UE}}$. UE verifies if the recovered nonces (a_1, a_2) match the original nonces. If the verification fails, UE terminates this session; otherwise, UE calculates $\mathsf{K}_{\mathsf{s}} = uV$ and $\mathsf{K}'_{\mathsf{AF},i} = \mathsf{KDF}(\mathsf{K}_{\mathsf{AF},i}, \mathsf{K}_{\mathsf{s}})$.

Afterward, UE and AF establish a secure channel using the privacy-hardened application key $\mathsf{K}'_{\mathsf{AF},i}$.

4.3 Discussions on Key Operational Aspects

There are several key operational aspects involved in the deployment of AKMA+.

Synchronization for Shuffled Key-Identifier Pairs. Synchronization between AUSF and AAnF follows key management protocols outlined in 3GPP TS 33.535 [32]. Clause 6.3.2 defines key distribution procedures, where the AUSF securely provisions key-identifier pairs to the AAnF using authenticated, encrypted signaling channels based on TLS 1.3 as specified in TS 33.210 [25]. Clause 7.2 of TS 33.501 [26] covers mobility-related context transfers using N2/N3 signaling to maintain key-identifier continuity when users switch nodes. To address network latency, Clause 6.4 of TS 29.510 [37] specifies asynchronous HTTP/2-based communication with retries and acknowledgments. Server failure recovery relies on persistent storage and state synchronization procedures described in TS 33.117 [30], ensuring fault tolerance and data integrity. Collectively, these protocols enable secure, scalable, and resilient key-identifier synchronization in AKMA+.

Error Handling and Recovery Mechanisms. AKMA+ follows 3GPP specifications to ensure secure and resilient key management. Clause 6.3.2 of TS 33.535 [32] mandates integrity-protected key updates, minimizing inconsistencies. Clause 6.4 of TS 29.510 [37] outlines HTTP/2-based retries with exponential backoff for network failures, while keyed-hash message authentication codes (HMAC) in TS 33.210 [25] ensure data integrity. Persistent failures are managed through N2/N3 context recovery (Clause 7.2, TS 33.501 [26]) and redundant storage mechanisms (Clause 5.2, TS 33.117 [30]). These mechanisms ensure that AKMA+ can reliably address failover and scalable storage requirements.

Incentivization for AF. The adoption of AKMA by AFs presents a trade-off between operational ease and potential data revenue loss. AKMA simplifies user onboarding by delegating authentication to carriers, but AFs relinquish valuable user-application binding data, which carriers could monetize. To align incentives, future AKMA+ enhancements could introduce revenue-sharing models or provide anonymized analytics to AFs. Additionally, carriers could offer security and compliance guarantees, reducing AFs' legal and operational burdens. These measures could make AKMA+ adoption more mutually beneficial within a competitive service ecosystem.

5 Formal Verification

In this section, we evaluate the security and privacy of AKMA+ using the state-of-the-art symbolic modeling tool Tamarin [20]. Tamarin provides an expressive language for modeling protocols and adversaries based on multiset rewriting [12] and for specifying properties in a first-order logic over protocol execution traces. Tamarin can handle protocols with complex control flow, security properties, and equational theories. Tamarin supports both automated and interactive theorem proving to establish that a protocol, when run in the presence of specified adversaries, satisfies given properties.

We modeled the AKMA+ protocol and meticulously verified its adherence to the requisite properties. Our Tamarin model consists of approximately 580 lines of code (LoC) and is available on Zenodo [1].

5.1 Properties Specification

We formalize and prove the following security guarantees for AKMA+: (1) UE indistinguishability, (2) secrecy of the session key, (3) forward secrecy, and (4) mutual authentication between UE and AAnF.

5.1.1 UE Indistinguishability.

We verify UE indistinguishability through a combination of Tamarin verification and cryptographic proof. The secrecy of SUPI is formalized and verified within Tamarin, while the secrecy of A-KID and K_{AKMA} are proved in Theorem 1.

Property 1 (Secrecy of SUPI). SUPI must be kept secret.

```
lemma secrecy of SUPI:
 all-traces
   All n A #i. Secret(<SUPI, n>, A) @i
   ⇒(not (Ex #j.K(n)@j)) |
  (Ex X data #r. Reveal(X, data) @r &Honest(X) @i)
```

This lemma states that the SUPI value n of a party A that is considered (marked by Secret(<SUPI,n>,A)) to be secret from the adversary (marked by the non-existence of K(n)), unless a party expected to be honest (marked by Honest(X) @i, where i is the timepoint where Secret was noted) was compromised (marked by Reveal(X, data)). This lemma ensures that the SUPI remains confidential unless there is a breach, i.e., a trusted party is compromised and reveals it.

As we have seen with the secrecy lemma, the SUPI itself is secret in AKMA+, thus it cannot be directly used to distinguish UEs. Furthermore, since UE terminates the session after a failed verification instead of sending a failure notification, there are no conditional statements in the AKMA+ protocol that an attacker could exploit to generate (or replay) messages to which different UEs would respond differently [5].

The secrecy of SUPI is necessary but not sufficient to establish indistinguishability, as is well understood. Therefore, we also provide a cryptographic proof to justify our claim about the indistinguishability of UEs through A-KID or K_{AKMA}. Given the size of the model, this is difficult to establish with the current state-of-the-art symbolic verifiers[8]. As a result, we provide a cryptographic proof for Theorem 1 in Appendix C.

Theorem 1 (UE Indistinguishability)**.** *The AKMA+ protocol achieves UE indistinguishability if the KDF is secure.*

Note that resistance to traffic analysis, frequency observations, or side-channel attacks, is out of scope. For example, if only one UE is active after registration, traffic patterns may expose its presence despite cryptographic protections. We acknowledge this limitation and focus solely on UE indistinguishability under the assumption of multiple active UEs.

5.1.2 Secrecy of Session Key

The privacy-hardened application key K'_{AF} serves as the session key for secure communication between UE and AF in AKMA+. We prove that once UE successfully completes the key exchange with AF, an attacker cannot obtain the secret session key (i.e., privacy-hardened application key K'_{AF}).

Property 2 (Secrecy of Session Key). K'_{AF} must be kept secret.

```
lemma secure_K_AF_prime:
 all-traces
   "All n A #i. Secret(<‘K_AF_prime’, n>, A) @i
   ==>(not (Ex #j.K(n)@j)) |
   (Ex X data #r. Reveal(X,data)@r &Honest(X) @i)"
```

This lemma states that for any given nonce (n) and entity (A) at a particular time (@i), the session key K'_{AF} associated with them must remain secret throughout the protocol's execution. The formula shows that the adversary never learns the key, unless a party involved in this run presumed honest (by Honest(X) @i) is compromised and its relevant data (including the key) is learned by the adversary. This ensures that the session key K'_{AF} cannot be compromised unless a party has been compromised and thus there is no hope to communicate securely with them anyway.

[8] This proof was not formulated in terms of observational equivalence in Tamarin due to the complexity and size of the AKMA+ model; such a proof would exceed the capability of current automated tools like Tamarin, which have been used for verification of indistinguishability in small case studies [6] and for finding attacks on indistinguishability for larger protocols [5] but have not provided indistinguishability proofs for any larger protocol.

5.1.3 Forward Secrecy

A stronger secrecy property than session key secrecy is forward secrecy, which requires that messages labeled with a `Secret` action before a compromise remain secret.

Property 3 (Forward Secrecy). We define this property as

```
lemma Forward_Secrecy:
 all-traces
    "All x A #i. Secret(x, A) @i
     ==> (not (Ex #j. KU(A) @j)) |
     (Ex P data #r. Reveal(P,data)@r & r < i
        & Honest(P)@i)"
```

This lemma states that for any session key `x` used by an entity `A` at time point `i`, this key is secret (first disjunct) unless a party `P` involved in this run (marked by `Honest(P)@i`) has been compromised (`Reveal(P,data)@r`) before creation of this key `x` (seen in the `r<i` constraint). This means that unless a participant's long-term key is compromised before a session, then that session remains secure. In other words, previous sessions remain secure even if a long-term key is compromised afterward.

5.1.4 Mutual Authentication between UE and AAnF

In the following, we prove mutual authentication between UE and AAnF, where the two entities verify the authenticity of each other in both directions.

Property 4 (Mutual Authentication). This property is formalized by two lemmas, each addressing one direction of the authentication flow.

```
lemma mutual_authen_UE_AAnF_one_direction:
 all-traces
    "All A B t #i1.
    (Commit(A,B, <‘UE’, ‘AAnF’, <‘K_AKMA’, t>>)
        @i1)
    ==> (Ex #j1. Running(B, A, <‘UE’, ‘AAnF’,
                 <‘K_AKMA’, t>>) @j1) |
     (Ex D m #l. Reveal(D, m) @l &Honest(D) @i1)"


lemma mutual_authen_UE_AAnF_other_direction:
 all-traces
    "All A B t #i2.
    (Commit(B,A, <‘AAnF’, ‘UE’, <‘K_AKMA’, t>>)
        @i2)
    ==> (Ex #j2. Running(A, B, <‘AAnF’, ‘UE’,
                 <‘K_AKMA’, t>>) @j2) |
     (Ex D m #l. Reveal(D, m) @l &Honest(D) @i2)"
```

The first lemma states that if at a specific time (`@i1`), UE (marked by `A`) commits to authenticate with AAnF (marked by `B`) using the key $\mathsf{K}_{\mathsf{AKMA}}$, then one of the two conditions must hold true. 1) There must exist a particular time (`@j1`) where AAnF is running the corresponding protocol session with UE, using the same key $\mathsf{K}_{\mathsf{AKMA}}$. 2) Alternatively, a party involved in this run (`Honest(D)@i1`) has been compromised (`Reveal(D,m)@l`). This lemma ensures that if a UE commits to a session with an AAnF, the AAnF must either be engaged in the corresponding session or a party has been compromised.

The second lemma mirrors the first but in the opposite direction. It states that if an AAnF commits to the authentication process with an UE, the UE must be running the corresponding protocol session with the AAnF, using the same key $\mathsf{K}_{\mathsf{AKMA}}$. Alternatively, this lemma allows for the possibility that a compromise of an honest entity has occurred, which would explain the lack of authentication in this direction.

Together, these two lemmas state that mutual authentication is upheld between UE and AAnF, ensuring that both parties are authentic entities in the secure exchange.

5.2 Verification Results

We run our Tamarin model on a computing server with AMD Ryzen Threadripper PRO 3000WX Series Processor with 64 cores, 3.2 GHz, 64-bit CPU, 128 GB RAM and Ubuntu 20.04.

Property	Result	Runtime
Executability	√	1m 09s
Secrecy of SUPI	√	58s
Secrecy of session key ($\mathsf{K}'_{\mathsf{AF}}$)	√	5m 43s
Forward secrecy	√	9m 58s
Mutual authentication (one direction)	√	59s
Mutual authentication (other direction)	√	4m 45s

Table 1: AKMA+ Formal Analysis Results in Tamarin

In Table 1, we summarize the main privacy and security properties that we proved using Tamarin. The runtime shows the time it takes for Tamarin to prove each of the given properties. The executability of AKMA+ is confirmed[9], indicating that the protocol can be completed and executed correctly. The Tamarin verification results demonstrate that AKMA+ achieves the design goals outlined in §3.2.

6 Implementation and Comparison

We benchmark AF and HPLMN entities (UDM, AUSF, AAnF) on a laptop with an Intel Core i5-10210U CPU (1.60 GHz & 2.11 GHz, 4-core, 64-bit), 16GB RAM and Ubuntu 20.04. To emulate diverse 5G devices for UE, we use a Raspberry Pi 5 with a Broadcom BCM2712 CPU (2.4GHz 64-bit 4-core ARM Cortex-A76), 8GB RAM and 32GB SD card. Our implementation uses the standard OpenSSL library [21] and prime256v1 elliptic curve. Each test case averages over 50 executions. The C-language source code, comprising approximately 11,290 LoC, is available on Zenodo [1].

The public key encryption scheme ($\mathsf{PKEnc}/\mathsf{PKDec}$) is implemented using ECIES, while the signature scheme ($\mathsf{Sig}/\mathsf{Verify}$) is instantiated with ECDSA, and KDF using SHA256. The authenticated encryption scheme ($\mathsf{AEnc}/\mathsf{ADec}$) is instantiated by AES-GCM (Galois/Counter Mode) [7] with 128-bit keys, where GCM is a mode of operation for AES that

[9] The executability of AKMA+ is manually found using Tamarin and the resulting file is available on Zenodo [1].

provides authenticated encryption by combining the counter mode of encryption with the Galois mode of authentication.

Protocols	Computation Costs (ms)					
	UE (RasPi.)	AF (Laptop)	AAnF (Laptop)	AUSF (Laptop)	UDM (Laptop)	Total
AKMA	0.079	0.003	0.014	0.078	0.002	0.176
AKMA+	1.416	0.294	0.031	0.078	0.045	1.864
comp.$^+$	1.337	0.291	0.017	0	0.043	1.688

Protocols	Communication Costs (KB)					
	UE (RasPi.)	AF (Laptop)	AAnF (Laptop)	AUSF (Laptop)	UDM (Laptop)	Total
AKMA	0.091	0.483	0.560	0.297	0.243	0.837
AKMA+	0.429	0.856	0.649	0.289	0.306	1.265
comm.$^+$	0.338	0.373	0.089	-0.008	0.063	0.428

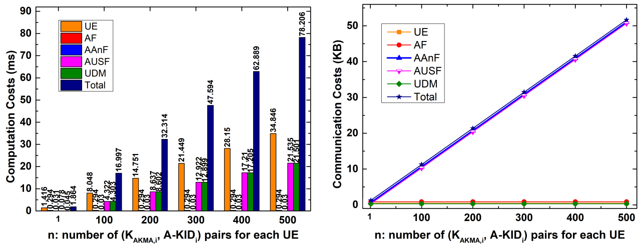
Table 2: Performance Comparison ($n = 1$)

In Table 2, we compare the computation cost (comp.) and communication cost (comm.) of AKMA+ with those of AKMA. The comp$^+$/comm$^+$ line indicates the additional computation/communication costs of AKMA+ compared to AKMA. Here, the number of key-identifier pairs (generated for each UE in deriving AKMA keys each time after primary authentication) in AKMA+ is set to $n = 1$ (i.e., AKMA+ and AKMA use the same number of key-identifier pair) to directly compare the performance impacts of the additional cryptographic mechanisms introduced in AKMA+ relative to AKMA. We will show our scalability result of AKMA+ later.

Table 2 shows that the computation cost for AUSF in two protocols remains unchanged since it performs no additional computations. The communication costs are nearly identical, except for a 0.008 KB reduction in AKMA+ due to AUSF ommiting the SUPI transmission in Step 4. UDM's increased computation cost arises from calculating $\mathsf{A\text{-}KID}_i$ in Step 3, and the higher communication cost is due to replacing SUPI/GPSI with $\mathsf{A\text{-}KID}_i$ in Step 10. For AAnF, the computation cost rises from 0.014 ms to 0.031 ms due to the $\mathsf{ADec}/\mathsf{AEnc}$ calculation in Step 12. Its communications cost increases from 0.560 KB to 0.649 KB due to the transmission of $\mathsf{CT}_{\mathsf{AAnF}\to\mathsf{UE}}$ in Step 7, and $\mathsf{A\text{-}KID}_i$ instead of SUPI/GPSI in Step 10.

For AF, the additional computation time in AKMA+ is 0.291 ms, dominated by the execution of $\mathsf{PKDec}/\mathsf{KDF}/\mathsf{Sig}$ algorithms and the DH exponentiation (expo.) in Step 14. The additional communication cost for AF is 0.373 KB due to the transmission of ciphertexts and signatures. For UE, the communication cost increases by 0.338 KB for ciphertexts and signature transmission in Steps 6 and 14, and the computation time rises by 1.337 ms due to the computations in Steps 3, 6, and 14. Note that Raspberry Pi has lower processing capabilities than a laptop. In summary, the total additional computation cost for a session (among UE, AUSF, UDM, AAnF, and AF to execute Steps 1-14) in AKMA+ is 1.688 ms, and the total increased communication cost is 0.428 KB.

In Fig. 10, we test the scalability of AKMA+ by increasing n from 1 to 500. We set the maximum n to 500 for the following reasons: n represents the number of key-identifier pairs generated daily for each UE, and $n = 500$ allows each UE to access up to 500 AFs per day. As of 2023, global internet users spend an average of 6 hours and 37 minutes online daily, according to the "Digital 2023: Global Overview Report" by DataReportal [10]. With $n = 500$, each UE could switch to an average of 1.26 different AFs per minute during their entire online time (6 hours 37 minutes) daily, which is arguably more than enough for most practical use.



(a) Computation Cost

(b) Communication Cost

Figure 10: Performance of AKMA+

Fig. 10 demonstrates that the computation costs for AAnF, and AF remain unchanged. Meanwhile, the communication costs of UDM, UE, and AF remain consistently low at 0.306 KB, 0.429 KB, and 0.856 KB, respectively. The computation costs for UE, AUSF, and UDM, as well as the communication costs for AUSF and AAnF, increase linearly with n. This growth is due to the processing and transmission of key-identifier pairs in Step 3 and 4, respectively. For $n = 500$, AKMA+ spends only 78.206 ms and consumes merely 51.664 KB bandwidth to complete a session among UE, AUSF, UDM, AAnF, and AF (executing Steps 1-14).

The communication costs presented in Table 2 and Fig. 10(b) represent the payloads in two protocols, referring to the useful data transmitted over a communication channel. In contrast, over-the-air (OTA) overheads include additional data such as headers, control information, and signaling. According to the OTA overheads analyzed in standards like TS 38.300 [27], TS 38.321 [28], and TS 38.331 [29], the approximate percentages at different protocol layers are: 7-10% in the physical layer, 5-10% in MAC layer, 3-5% in IP layer (for IPv6 header) and 5-10% in application layer. Combined, the total OTA overhead in 5G communication can range from 20% to 35%, depending on the service, configuration, and layers. For analysis, we consider the upper limit of 35%.

Next, we examine the bandwidth and power consumption for UE in AKMA+ since users are primarily concerned with these factors when accessing 5G services (AFs). The OTA payload of UE is a constant 0.579 KB [10] per AF access, regardless of n (see Table 2 and Fig. 10(b)). Assuming a UE accesses $n = 500$ different AFs daily, the OTA bandwidth consumed in AKMA+ totals 0.283 MB daily and 8.773 MB

[10] It is computed by multiplying 0.429 KB and 135%.

monthly (assuming 31 days).

The bandwidth costs for AKMA+ in various countries are detailed in Table 5 in Appendix B, based on the latest data plans from major mobile operators. The USA has the highest data rate at 2.333 USD/GB/Mon, resulting in a monthly expense of 0.020 USD. In contrast, Australia and Singapore have the lowest data rates at 0.208 USD/GB/Mon, with a monthly cost of just 0.002 USD.

Regarding power consumption, UE performs Steps 1-5 once to generate 500 key-identifier pairs daily and executes Steps 6-14 up to 500 times to access various AFs with $n = 500$. Based on our experiments, the total daily computation cost for UE is 742.846 ms. The experiments were conducted on a Raspberry Pi 5 with an ARM Cortex-A76 processor, similar to those in smartphones like the Samsung Galaxy A71 and Huawei Mate 30 Pro, which have 4500 mAh batteries. Assuming fully processor loaded, power consumption for 500 AFs accesses is 7.428 mAh daily (measured at 750 mW/core with a 5V input voltage [11, 22]), resulting in only 0.165% daily battery consumption on these devices.

Overall, from our experiments, we conclude that AKMA+ incurs reasonably low overheads in 5G communications.

Limitation of Simulation. While the simulation results presented in this paper demonstrate the feasibility and performance of the AKMA+ system, we acknowledge that relying solely on simulations is a limitation. Simulations provide a controlled environment, allowing us to evaluate the efficiency of key distribution under predefined conditions. However, real-world deployments could introduce unpredictable factors such as network congestion, user mobility patterns, and varying traffic loads, which may affect system performance. Additionally, implementation-specific constraints such as processing delays, data center outages, and inter-operator coordination are challenging to replicate in simulations. Future work will focus on field trials and real-world testing to validate AKMA+ in live network environments, capturing these complexities and enabling a more comprehensive evaluation.

Future Validation Strategy. To validate AKMA+ in more realistic settings, future research should explore several complementary strategies. Collaborating with standardization organizations like 3GPP and GSMA, as well as industry partners such as mobile network operators and infrastructure vendors, would enable pilot deployments in live network environments, providing valuable insights into system performance under real-world conditions. Deploying AKMA+ in real-world, large-scale testbeds for advanced wireless research would also facilitate broader testing and evaluation. Additionally, developing open-source testing frameworks would allow the research community to verify AKMA+ implementations, fostering transparency and innovation. Designing advanced simulations incorporating real-world metrics—such as dynamic user mobility models, network congestion patterns, and multi-operator configurations—could further bridge the gap between theoretical analysis and practical deployment.

7 Related Work

In this section, we examine the related work on AKMA, including technical specifications and reports related to AKMA, its formal verification, and existing efforts on improving security and privacy protection in AKMA services.

Technical Specification/Report on AKMA. 3GPP delineated the 5G system's architecture, procedures, and security measures in TS 23.501 [36], TS 23.502 [34], and TS 33.501 [26]. To safeguard subscribers and application providers communicating over insecure channels, 3GPP standardized the AKMA service in TS 33.535 [32] starting with Release 16 in 2019 through Release 18 in 2024. The AKMA service facilitates authenticated communication between users and AFs. 3GPP also released TR 33.835 [24] detailing key issues, privacy requirements, and potential solutions for enhancing AKMA services. Our work addresses five key issues specified in TR 33.835 [24] (see §3.1) and offers verified privacy-enhanced solutions compatible with 3GPP standards.

Formal verification of AKMA. In a recent study [40], Yang et al. used the Tamarin verification tool [20] to formally analyze AKMA. They modeled the desired properties of AKMA, including authentication (between UE and HPLMN, and between AF and HPLMN), secrecy of the application key, and privacy properties according to the 3GPP Technical Specifications [26, 32]. Their analysis revealed that some of these properties were not satisfied by AKMA. Yang et al.'s work [40] motivates us to address these weaknesses and develop security and privacy-enhanced solutions. Our work extends [40] by verifying forward secrecy and mutual authentication between UE and AAnF using Tamarin.

Improving security and privacy protection in AKMA services. Several solutions have been proposed to improve security and/or privacy protection in AKMA services [2, 15, 19, 24]. Akman et al. [2] discovered several vulnerabilities in AKMA, including a spoofing attack where a malicious AF impersonates another AF towards a UE. They developed a privacy-enhanced version of AKMA and verified that it is secure against these vulnerabilities. Khan et al. [15] developed a privacy mode for AKMA to protect UE privacy against insider attackers in the home network. Li et al. [19] tackled the risks associated with the long-term use of the same application key and proposed an enhanced AKMA protocol. Additionally, 3GPP TR 33.835 outlines candidate solutions to mitigate certain privacy-threatening key issues, presented as proof-of-concept sketches without concrete algorithms [24].

To highlight the advantages of AKMA+, we compare it with the related security and privacy-enhanced solutions mentioned above along three dimensions: privacy issues, formal verification, and standard compatibility. For security and privacy issues, we focus on five key issues specified in TR 33.835 [24] that significantly undermine the security and privacy of AKMA related to the protocol layer. Table 3 summarizes our comparison results, which are detailed below.

Key Issues (KI) and Features	**Security and Privacy-Enhanced AKMA Protocols**				**Related Candidate Solutions in TR 33.835 [24]**								
	AKMA+	AGDN [2]	KGN [15]	LHW [19]	#4	#6	#13	#15	#17	#18	#19	#22	#23
KI#3: Mutual authentication between UE and anchor function	√	√	√	×	√	√	√	√	√	×	√	×	√
KI#5: User privacy	√	×	×	×	×	×	×	×	√	×	√	×	√
KI#6: Secure communication between UE and application server	√	×	×	×	×	×	×	×	×	×	×	×	×
KI#7: Protecting subscriber's personal information in control and data traffic	√	×	√	×	×	×	×	×	×	×	×	×	×
KI#16: Application key freshness of AKMA	√	×	×	√	×	×	×	×	×	×	×	√	×
Formal verification	√	√	×	×	×	×	×	×	×	×	×	×	×
Standard compatibility	√	×	×	×	×	×	×	√	×	√	√	×	×

Table 3: Comparison of Privacy-Enhanced Solutions for Improving AKMA

• *Key Issues*. AKMA+ tackles all five security and privacy issues specified in TR 33.835 and provides comprehensive solutions for improving the security and privacy protection in AKMA services. In comparison, the other security and privacy-enhanced solutions listed in Table 3 address at most two of the five key issues.

• *Formal Verification*. We provide formal verifications for AKMA+ to meet the security and privacy requirements using the state-of-the-art symbolic verification tool Tamarin Prover [20]. With the exception of the AGDN scheme [2], which has a formal verification based on ProVerif [8], none of the other solutions listed in Table 3 offer any formal verification, undermining the credibility of their claimed properties.

• *Standard Compatibility*. AKMA+ maintains its compatibility with the AKMA specifications [32] as it alters neither message flows nor data formats in the original AKMA protocol. All the algorithms used for deriving new message terms in AKMA+ are readily available in the existing 5G specifications [26, 34, 36]. By maintaining compatibility, AKMA+ can be seamlessly integrated into 5G infrastructures without requiring time-consuming modifications to the network or devices. This approach not only preserves interoperability with existing systems but also accelerates deployment, making it easier for operators to adopt enhanced security and privacy measures without disrupting their established operations.

With the exception of the three candidate solutions outlined in TR 33.835 [24], none of the other solutions listed in Table 3 are compatible with the standard AKMA specifications [32]. In particular, the AGDN scheme [2] deviates significantly from the 5G primary authentication protocol and AKMA protocol, comprising a distinct set of protocols for user sign-up, sign-in, and mutual authentication between UE and AF. Hence, it should be regarded as being independent of AKMA rather than merely an improved AKMA protocol.

The KGN scheme [15] is not compatible with AKMA either since it changes the anchor key derivation function, application key derivation function, protocol flows, and message terms in AKMA. Similarly, the LHW scheme [19] alters the AKMA system architecture and key derivation process to enable application key refreshment without triggering 5G primary authentication.

Next, we consider the nine candidate solutions for enhancing security and privacy outlined in TR 33.835, including solutions #4, #6, #13, #15, #17, #18, #19, #22, and #23, that address certain key issues as shown in Table 3. While solutions #15, #18, and #19 are compatible with the standard AKMA specifications [32], the other related solutions are not. Specifically, solutions #4 and #17 completely overhaul the primary authentication protocols (5G-AKA and EAP-AKA') upon which AKMA builds. Solutions #6 and #13 significantly modify the authentication procedure and message terms in AKMA. And solutions #22 and #23 alter the procedure for application key derivation in AKMA to meet various security and privacy requirements.

Lastly, several privacy-preserving protocols were proposed to enhance privacy protection in 5G-AKA, which provide authentication and key agreement functions between the User Equipment (UE) and the Home Public Land Mobile Network (HPLMN) in 5G [16, 39, 41]. The 5G-AKA can be used as a primary authentication protocol in AKMA, extending its security features to the application level. However, these privacy-preserving protocols fail to address the key security and privacy issues of AKMA, as specified in TR 33.835 [24]. This is because all the key security and privacy issues of AKMA are beyond the scope of 5G-AKA, involving new entities such as the application function and anchor function.

8 Conclusion

The AKMA protocol has security and privacy vulnerabilities, including susceptibility to linkability, tampering, spoofing, and impersonation attacks. Our proposed AKMA+ protocol addresses these critical security and privacy issues at the protocol layer while remaining compliant with existing 5G network specifications. The security and privacy of AKMA+ is proven using a combination of Tamarin verification and cryptographic formal proof. Extensive experiments conducted on Raspberry Pi and laptop platforms demonstrate that AKMA+ is efficient and well-suited for 5G communication.

Acknowledgments

The authors thank the shepherd and anonymous reviewers for their valuable comments and insightful suggestions.

This research is supported by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Trust Tech Funding Initiative, AXA Research Fund, National Natural Science Foundation of China (No. 62372110, 62332007, U22B2028), Fujian Provincial Natural Science of Foundation (No. 2023J02008), Lee Kong Chian Chair Professorship, Ripple University Blockchain Research Initiative, National Science Foundation (No. 2112471), Werner Siemens-Stiftung (WSS) as part of the Centre for Cyber Trust (CEYT), Science and Technology Major Project of Tibetan Autonomous Region of China (No. XZ202201ZD0006G), Open Research Fund of Machine Learning and Cyber Security Interdiscipline Research Engineering Center of Jiangsu Province (No. SDGC2131), National Joint Engineering Research Center of Network Security Detection and Protection Technology, Guangdong Key Laboratory of Data Security and Privacy Preserving, Guangdong Hong Kong Joint Laboratory for Data Security and Privacy Protection, and Engineering Research Center of Trustworthy AI, Ministry of Education. Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore and Infocomm Media Development Authority.

Ethics Considerations and Compliance with the Open Science Policy

A. Ethics Considerations

All the experiments in this paper followed the ethics consideration policies. The experiments were done in a lab setup where both the attack and victim devices were ours. Furthermore, it was ensured that no other adjacent devices were present and affected by the attacks. The discussed attacks have been responsibly disclosed to the standardization body—WiFi Alliance—and to all the other affected vendors. Most vendors have acknowledged and said they will modify the implementation following the guidelines of WiFi Alliance. We are coordinating with others to provide the necessary details and code to help through the responsible disclosure process.

B. Compliance with the Open Science Policy

In compliance with the open science policy, we have released the formal models, associated codes, and testbed setup to the community to foster further research. Furthermore, we will participate in the artifact evaluation. The experimental codes of AKMA+ are available on https://doi.org/10.5281/zenodo.14649847.

References

[1] Souce code of akma+. https://doi.org/10.5281/zenodo.14649847, JAN. 2025.

[2] Gizem Akman, Philip Ginzboorg, Mohamed Taoufiq Damir, and Valtteri Niemi. Privacy-enhanced akma for multi-access edge computing mobility. *Computers*, 12(1):2, 2022.

[3] Apple. Apple device support for private 5g and lte networks. https://support.apple.com/en-sg/guide/deployment/depac6747317/web, May. 2024. (Accessed 14 July 2024).

[4] AT&T. Att launches 5g managed advanced security capabilities to further protect enterprise network infrastructure. https://about.att.com/story/2021/5G-managed-security-capabilites.html, Oct. 2021. (Accessed 14 July 2024).

[5] David Basin, Jannik Dreier, Lucca Hirschi, Saša Radomirovic, Ralf Sasse, and Vincent Stettler. A formal analysis of 5g authentication. In *Proceedings of the 2018 ACM SIGSAC conference on computer and communications security*, pages 1383–1396, 2018.

[6] David Basin, Jannik Dreier, and Ralf Sasse. Automated symbolic proofs of observational equivalence. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1144–1155, 2015.

[7] Mihir Bellare and Björn Tackmann. The multi-user security of authenticated encryption: Aes-gcm in tls 1.3. In *Advances in Cryptology–CRYPTO 2016: 36th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 14-18, 2016, Proceedings, Part I 36*, pages 247–276. Springer, 2016.

[8] B Blanchet, B Smyth, V Cheval, and M ProVerif Sylvestre. Proverify 2.04: automatic cryptographic protocol verifier. *User Manual and Tutorial, INRIA Paris-Rocquencourt*, 2021.

[9] Colin Boyd, Anish Mathuria, and Douglas Stebila. *Protocols for authentication and key establishment*, volume 1. Springer, 2003.

[10] DataReportal. Digital 2023: Global overview report. https://datareportal.com/reports/digital-2023-global-overview-report, Jan. 2023. (Accessed 14 July 2024).

[11] Arm Developer. Arm cortex-a76 core technical reference manual. https://developer.arm.com/documentation/100798/latest, Jan. 2023. (Accessed 14 July 2024).

[12] Nancy Durgin, Patrick Lincoln, John Mitchell, and Andre Scedrov. Multiset rewriting and the complexity of bounded security protocols. *Journal of Computer Security*, 12(2):247–311, 2004.

[13] Ericsson. Ericsson and xyz partner to enhance 5g security with akma. https://www.ericsson.com/en/news/2021/3/ericsson-xyz-partner-5g-security-akma, Mar. 2021. (Accessed 14 July 2024).

[14] Huawei. Huawei's 5g security assurance above stds for the core network. https://consumer.huawei.com/za/community/details/Huawei-s-5G-security-assurance-above-stds-for-the-core-network/topicId_195543, Jun. 2022. (Accessed 14 July 2024).

[15] 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.

[16] Adrien Koutsos. The 5g-aka authentication protocol privacy. In *2019 IEEE European symposium on security and privacy (EuroS&P)*, pages 464–479. IEEE, 2019.

[17] Hugo Krawczyk. Cryptographic extraction and key derivation: The hkdf scheme. In *Annual Cryptology Conference*, pages 631–648. Springer, 2010.

[18] Nokia Bell Labs. Realizing a zero-trust architecture for 5g networks. https://www.bell-labs.com/institute/articles/realizing-zero-trust-architecture-for-5g-networks/#gref, Sep. 2023. (Accessed 14 July 2024).

[19] Jinhui Li, Chengbin Huang, and Jinhua Wang. An enhanced application authentication and key management in 5g. In *Journal of Physics: Conference Series*, volume 2625, page 012071. IOP Publishing, 2023.

[20] Simon Meier, Benedikt Schmidt, Cas Cremers, and David Basin. The tamarin prover for the symbolic analysis of security protocols. In *Computer Aided Verification: 25th International Conference, CAV 2013, Saint Petersburg, Russia, July 13-19, 2013. Proceedings 25*, pages 696–701. Springer, 2013.

[21] OpenSSL. Openssl: The open source toolkit for ssl/tls, http://www. openssl.org, 2024. [Online: accessed 24-June-2024].

[22] Raspberry. Raspberry pi 5 product. https://datasheets.raspberrypi.com/rpi5/raspberry-pi-5-product-brief.pdf, Apr. 2024. (Accessed 14 July 2024).

[23] Samsung. 5g security — improving user and data protection. https://images.samsung.com/is/content/samsung/assets/global/business/networks/insights/brochures/5g-security-improving-user-and-data-protection/Samsung-5G-Security-Brief_FINAL.pdf, Mar. 2021. (Accessed 14 July 2024).

[24] 3GPP TR 33.835 (V16.1.0). Study on authentication and key management for applications based on 3gpp credential in 5g, Technical report, July, 2020.

[25] 3GPP TS 33.210 (V18.1.0). Network domain security (nds); ip network layer security, Technical specification, July, 2024.

[26] 3GPP TS 33.501 (V18.2.0). Security architecture and procedures for 5g system, Technical specification, July, 2024.

[27] 3GPP TS 38.300 (V18.2.0). Nr; nr and ng-ran overall description; stage-2, Technical specification, July, 2024.

[28] 3GPP TS 38.321 (V18.2.0). Nr; medium access control (mac) protocol specification, Technical specification, July, 2024.

[29] 3GPP TS 38.331 (V18.2.0). Nr; radio resource control (rrc); protocol specification, Technical specification, July, 2024.

[30] 3GPP TS 33.117 (V18.3.0). Catalogue of general security assurance requirements, Technical specification, March, 2024.

[31] 3GPP TS 33.220 (V18.3.0). Generic authentication architecture (gaa); generic bootstrapping architecture (gba), Technical specification, March, 2024.

[32] 3GPP TS 33.535 (V18.4.0). Authentication and key management for applications (akma) based on 3gpp credentials in the 5g system (5gs), Technical specification, July, 2024.

[33] IETF RFC 7542 (V18.4.0). The network access identifier, Technical specification, May, 2015.

[34] 3GPP TS 23.502 (V18.5.0). Procedures for the 5g system (5gs), Technical specification, 2024.

[35] 3GPP TS 23.003 (V18.6.0). Numbering, addressing and identification, Technical specification, June, 2024.

[36] 3GPP TS 23.501 (V19.0.0). System architecture for the 5g system (5gs), Technical specification, June, 2024.

[37] 3GPP TS 29.510 (V19.0.0). 5g system; network function repository services, Technical specification, September, 2024.

[38] Verizon. First principles for securing 5g. https://www.verizon.com/business/en-sg/resources/whitepapers/first-principles-for-securing-5g/, Dec. 2019. (Accessed 14 July 2024).

[39] Yuchen Wang, Zhenfeng Zhang, and Yongquan Xie. Privacy-preserving and standard-compatible aka protocol for 5g. In *USENIX Security*, pages 3595–3612, 2021.

[40] 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.

[41] Hexuan Yu, Changlai Du, Yang Xiao, Angelos Keromytis, Chonggang Wang, Robert Gazda, Y Thomas Hou, and Wenjing Lou. Aaka: An anti-tracking cellular authentication scheme leveraging anonymous credentials. In *NDSS*, 2024.

Appendix

A. Abbreviations and Notations

Glossaries and cryptographic notations used throughout this paper are listed below.

Table 4: Abbreviations and Notations

Abbreviation & Notations	Meaning
AAnF	AKMA Anchor Function
AF	Application Function
AF_ID	AF Identifier
AKA	Authentication and Key Agreement
AKMA	Authentication and Key Management for Applications
AMF	Access and Mobility Management for Applications
AUSF	Authentication Server Function
AV	authentication vector
A-KID	AKMA Key IDentifier
A-TID	AKMA Temporary UE IDentifier
FQDN	Fully Qualified Domain Name
GPSI	Generic Public Subscription Identifier
HPLMN	Home Public Land Mobile Network
$\mathsf{K}_{\mathsf{AUSF}}$	Authentication key shared between AUSF and UE
$\mathsf{K}_{\mathsf{AKMA}}$	AKMA Anchor Key
K_{AF}	AKMA Application Key
$\mathsf{K}'_{\mathsf{AF}}$	AKMA Privacy-hardened Application Key
K_{s}	Diffie-Hellman Shared Key
KDF	Key Derivation Function
KI	key issue in TR 33.835 [24]
ME	Mobile Equipment
NEF	Network Exposure Function
RID	Routing InDicator
SN	Serving Network
SUPI	Subscription Permanent Identifier
SUCI	Subscription Concealed Identifier
UE	User Equipment
UICC	Universal Integrated Circuit Card

B. Expense for AKMA+

The bandwidth expenses for AKMA+ in different countries are listed below. Unite data prices are expressed in USD/GB/-Mon based on the data plans and rates as of Aug 12, 2024.

Country	Operator	Data Plan	Rate (2024-08-12)	Unit Price	AKMA+ (MB/Mon)	Expense (USD/Mon)
USA	Verizon	35 USD (15GB/Mon)	1.00 (USD/USD)	2.333	8.773	**0.020**
GBR	Vodafone	21 GBP (50GB/Mon)	0.78 (USD/GBP)	0.538	8.773	**0.005**
DEU	Deutsche Telekom	23.81 EUR (80GB/Mon)	0.92 (USD/EUR)	0.324	8.773	**0.003**
CHN	China Mobile	500 CNY (300GB/Mon)	7.18 (USD/CNY)	0.232	8.773	**0.002**
AUS	Telstra	95 AUD (300GB/Mon)	1.52 (USD/AUD)	0.208	8.773	**0.002**
SGP	Singtel	55 SGD (200GB/Mon)	1.32 (USD/SGD)	0.208	8.773	**0.002**

Table 5: Monthly Expense for AKMA+ Protocol

C. UE Indistinguishability Model and Proof

We provide a cryptographic formal model and proof to demonstrate that an attacker cannot distinguish UE through $\mathsf{A\text{-}KID}$ and $\mathsf{K}_{\mathsf{AKMA}}$.

Intuitive Idea of Formal Model. This formal model introduces an interactive game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$ to prove that: given two UE entities denoted by UE_0^* and UE_1^*, and a challenge key-identifier pair $(\mathsf{K}_{\mathsf{AKMA},i}^{(b)}, \mathsf{A\text{-}KID}_i^{(b)})$, no active attacker can determine whether it belongs to UE_0^* or UE_1^*.

(1) In *Init* phase, $\mathcal{A}$ selects two UE entities denoted by UE_0^* and UE_1^* with common parameters (i.e., RID, Realm, and roaming status), which are sent to $\mathcal{C}$.

(2) In *Setup* phase, $\mathcal{C}$ generates the public/secret keys for AFs, which are returned to $\mathcal{A}$. $\mathcal{C}$ generates n key-identifier pairs for UE_0^* and UE_1^*, respectively. $\mathcal{C}$ shuffled the key-identifier pairs of UE_0^* and UE_1^*, and the shuffled result is returned to $\mathcal{A}$.

(3) In *Query* phase, $\mathcal{A}$ can adaptively query for UE_0^* and UE_1^*. $\mathcal{A}$ selects an AF for the query, and $\mathcal{C}$ simulates the privacy-hardened application key generation process (see Fig. 7) between UE_0^* (or UE_1^*) and AF in the AKMA+ protocol. $\mathcal{A}$ acquires the $\mathsf{A\text{-}KID}$ for UE_0^* (or UE_1^*) during the simulation.

(4) In *Challenge* phase, $\mathcal{C}$ flips a coin $b \in_R \{0,1\}$ and sends a challenge key-identifier pair $(\mathsf{K}_{\mathsf{AKMA},i}^{(b)}, \mathsf{A\text{-}KID}_i^{(b)})$ for UE_b^* to $\mathcal{A}$, where the challenge pair is not used in the query phase.

(5) In *Guess* phase, $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$ indicating that $\mathcal{A}$ guesses that the challenge key-identifier pair belongs to $\mathsf{UE}_{b'}^*$. $\mathcal{A}$ wins the game if $b' = b$.

Definition 1 (UE Indistinguishability)**.** *The AKMA+ protocol achieves UE indistinguishable if no probabilistic polynomial time attacker $\mathcal{A}$ can win the following game with a non-negligible advantage.*

We define the following interactive game between an attacker $\mathcal{A}$ and a challenger $\mathcal{C}$ for UE indistinguishability proof. According to the security assumptions defined in §3.3, the attacker $\mathcal{A}$ may be an external attacker or an internal attacker

(AAnF or AFs). In this game, we suppose that $\mathcal{A}$ has compromised AAnF and the AFs. Therefore, $\mathcal{A}$ obtains all the long-term and short-term keys of AAnF and AFs.

- ***Init***. An attacker $\mathcal{A}$ specifies $(\mathsf{UE}_0^*, \mathsf{UE}_1^*)$ to be distinguished with the restriction that they share the same RID and realm for A-KID[11] and the same roaming status information. Then, $\mathcal{A}$ sends $(\mathsf{UE}_0^*, \mathsf{UE}_1^*)$, RID, realm, and roaming status information to a challenger $\mathcal{C}$.
- ***Setup***. Denote the set of AFs as $\mathcal{AF}$. $\mathcal{C}$ generates the public/secret key pairs $(\mathcal{PK}_{\mathsf{AF}}, \mathcal{SK}_{\mathsf{AF}})$ for AFs, which are sent to $\mathcal{A}$. Denote the key-identifier pair number for each UE as n. $\mathcal{C}$ runs the $\mathsf{K_{AKMA}}$ derivation procedure for UE_0^* and UE_1^*, respectively. The generated key-identifier set for UE_0^* is denoted by $\mathcal{S}_0 = \{\mathsf{K}^{(0)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(0)}_i\}_{i=1,\ldots,n}$, and that for UE_1^* is denoted by $\mathcal{S}_1 = \{\mathsf{K}^{(1)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(1)}_i\}_{i=1,\ldots,n}$. $\mathcal{C}$ shuffles the elements in $\mathcal{S}_0$ and $\mathcal{S}_1$ and the shuffled result is denoted by $\mathcal{S}_{0,1} = \{\mathsf{K}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}_i\}_{i=1,\ldots,2n}$, which is sent $\mathcal{A}$.
- ***Phase 1***. $\mathcal{A}$ can adaptively make the following queries.

- **Send query for UE_0^***. $\mathcal{A}$ can adaptively make this query for q_0' times, where $q_0' \leqslant n-1$. $\mathcal{A}$ selects an $\mathsf{AF} \in \mathcal{AF}$ for the query indicating that $\mathcal{C}$ should simulate the privacy-hardened application key generation process (see Fig. 7) between UE_0^* and AF in the AKMA+ protocol.

To simulate Step 6, $\mathcal{C}$ randomly selects a key-identifier pair $(\mathsf{K}^{(0)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(0)}_i)$ from $\mathcal{S}_0$. Then, $\mathcal{C}$ deletes this pair from $\mathcal{S}_0$, which ensures that each pair in $\mathcal{S}_0$ is used only once. $\mathcal{C}$ looks for the public key $\mathsf{PK_{AF}}$ of AF from $\mathcal{PK}_{\mathcal{AF}}$. $\mathcal{C}$ generates the ciphertexts $(\mathsf{CT_{UE\to AF}}, \mathsf{CT_{UE\to AAnF}})$ as defined in Step 6. Then, $\mathcal{C}$ sends $(\mathsf{A\text{-}KID}^{(0)}_i, \mathsf{CT_{UE\to AF}}, \mathsf{CT_{UE\to AAnF}})$ to $\mathcal{A}$ as *Application Session Establishment Request*.

Since $\mathcal{A}$ has compromised AAnF and AF, $\mathcal{A}$ can get all the data sent to them, manipulate all the messages they sent, and execute all the inner computations. $\mathcal{A}$ can simulate Steps 11-14 by itself, where the roaming status of UE in response from UDM (Step 11) is already known to $\mathcal{A}$. This completes the send query for UE_0^*.

- **Send query for UE_1^***. $\mathcal{A}$ can adaptively make this query for q_1' times, where $q_1' \leqslant n-1$. This query is similar to the send query for UE_0^*, except that the UE entity is substituted with UE_1^*. $\mathcal{A}$ selects an $\mathsf{AF} \in \mathcal{AF}$ for the query. To simulate Step 6, $\mathcal{C}$ randomly selects a key-identifier pair $(\mathsf{K}^{(1)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(1)}_i)$ from $\mathcal{S}_1$. Then, $\mathcal{C}$ deletes this pair from $\mathcal{S}_1$. $\mathcal{C}$ looks for the public key $\mathsf{PK_{AF}}$ of AF from $\mathcal{PK}_{\mathcal{AF}}$. $\mathcal{C}$ generates the ciphertexts $(\mathsf{CT_{UE\to AF}}, \mathsf{CT_{UE\to AAnF}})$ as defined in Step 6. Then, $\mathcal{C}$ sends $(\mathsf{A\text{-}KID}^{(1)}_i, \mathsf{CT_{UE\to AF}}, \mathsf{CT_{UE\to AAnF}})$ to $\mathcal{A}$ as *Application Session Establishment Request*.

- ***Challenge***. The challenger $\mathcal{C}$ flips a coin $b \in_R \{0,1\}$. $\mathcal{C}$ randomly selects a key-identifier pair $(\mathsf{K}^{(b)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(b)}_i)$ from $\mathcal{S}_b$ of UE_b^* as the challenge, which is sent to $\mathcal{A}$. Then, this challenge pair is deleted from $\mathcal{S}_b$.
- ***Phase 2***. The send query for UE_0^* in Phase 1 is repeated for $q_0 - q_0'$ times, and the send query for UE_1^* in Phase 1 is repeated for $q_1 - q_1'$ times, where $q_0, q_1 \leqslant n-1$.
- ***Guess***. The attacker $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

The advantage of an attacker $\mathcal{A}$ to win the game is defined as $\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{AKMA+},\mathcal{A}}(\lambda) = \Pr[b' = b] - 1/2$, where λ is the security parameter of the AKMA+ protocol.

Theorem 1 (UE Indistinguishability). *The AKMA+ protocol achieves UE indistinguishability if the KDF is secure.*

Proof. Refer to Annex B.2 of TS 33.220 [31] and §3 of [17] for the definition and the security game of KDF. The advantage of an attacker to win the KDF security game is denoted by $\varepsilon_{\mathsf{KDF}}$.

We define a series of hybrid games to prove UE indistinguishability.

- Game_0: This game is the same as a real interaction with the AKMA+ protocol, described in Definition 1. Hence, we have

$$\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{AKMA+},\mathcal{A}}(\lambda) = \mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_0,\mathcal{A}}(\lambda).$$

- Game_1: It is the same as Game_0, except that all the AKMA keys, i.e., $\mathsf{K}_{\mathsf{AKMA},i}$, in $\mathcal{S}_0 = \{\mathsf{K}^{(0)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(0)}_i\}_{i=1,\ldots,n}$ and $\mathcal{S}_1 = \{\mathsf{K}^{(1)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(1)}_i\}_{i=1,\ldots,n}$ are replaced by random values with ℓ-bit. As per Annex A.2 of TS 33.535 [32], $\mathsf{K}_{\mathsf{AKMA},i}$ is derived using KDF. Hence, we have

$$\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_0,\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_1,\mathcal{A}}(\lambda) \leqslant 2n \cdot \varepsilon_{\mathsf{KDF}}.$$

- Game_2: It is the same as Game_1, except that all the key identifiers, i.e., $\mathsf{A\text{-}KID}_i$, in $\mathcal{S}_0 = \{\mathsf{K}^{(0)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(0)}_i\}_{i=1,\ldots,n}$ and $\mathcal{S}_1 = \{\mathsf{K}^{(1)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(1)}_i\}_{i=1,\ldots,n}$ are replaced by random values, where $\mathsf{A\text{-}TID}_i$ in $\mathsf{A\text{-}KID}_i$ is substituted with random value with ℓ-bit and the RID, realm are kept the same. As per Annex A.3 of TS 33.535 [32], $\mathsf{A\text{-}TID}_i$ is derived using KDF. Hence, we have

$$\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_1,\mathcal{A}}(\lambda) - \mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_2,\mathcal{A}}(\lambda) \leqslant 2n \cdot \varepsilon_{\mathsf{KDF}}.$$

In Game_2, the attacker $\mathcal{A}$ guesses $b' \in \{0,1\}$ according to the challenge $(\mathsf{K}^{(b)}_{\mathsf{AKMA},i}, \mathsf{A\text{-}KID}^{(b)}_i)$, where $\mathsf{K}^{(b)}_{\mathsf{AKMA},i}$ and $\mathsf{A\text{-}KID}^{(b)}_i$ are all random numbers. Hence, we have $\Pr[b' = b] = 1/2$ and

$$\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{Game}_2,\mathcal{A}}(\lambda) = \Pr[b' = b] - 1/2 = 0.$$

Combining the above results, we have

$$\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{AKMA+},\mathcal{A}}(\lambda) \leqslant 4n \cdot \varepsilon_{\mathsf{KDF}}.$$

In the above equation, the term $4n \cdot \varepsilon_{\mathsf{KDF}}$ is negligible for the following reasons. Typically, n does not exceed 500, as this is considered high for most practical applications, as discussed in §6. Since the advantage $\varepsilon_{\mathsf{KDF}}$ is negligible due to the

[11] According to clause 6.1 of TS 33.535, A-KID shall be in the NAI format as specified in clause 2.2 of IETF RFC 7542 [33], i.e. username@realm. The username part shall include RID (Routing InDicator) and A-TID, and a realm part shall consist of HPLMN Identifier.

security of the KDF, the product $4n \cdot \varepsilon_{\mathsf{KDF}}$ is also negligible. Consequently, the advantage $\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{AKMA+},\mathcal{A}}(\lambda)$ in winning the AKMA+ game is negligible as well.

This completes the proof for Theorem 1. □

Hybrid game-based proof is a powerful tool in cryptography, allowing us to prove security in small, understandable steps. A hybrid game is a clever strategy to show security. Instead of proving security in one big step, the Prover breaks it down into a series of smaller, simpler games. Each game is slightly different from the last one, but only in a small way. The first game is the real protocol. The last game is one where the attacker clearly cannot win (because it is either impossible or very difficult). In between, there are several "hybrid" games that slowly transform the real game into the final game. The Prover shows that if an attacker cannot win one game, they also cannot win the next one by analyzing the advantages between the games. By proving that each small step is secure, the Prover shows that the real protocol is secure. Hybrid games help make complex proofs simpler and more understandable.

In the proof of Theorem 1, we define Game_0 as the original one to execute the AKMA+ protocol. In Game_1, we substitute all the $\mathsf{K}_{\mathsf{AKMA},i}$ in the key-identifier pairs with random values. In Game_2, we continuously substitute all the $\mathsf{A\text{-}KID}_i$ in the key-identifier pairs with random values. Therefore, the two elements in the challenge key-identifier pair are random values and the attacker has no advantage in distinguishing it. A series of advantage analyses are provided for these hybrid games to derive $\mathsf{Adv}^{\mathsf{UE\text{-}IND}}_{\mathsf{AKMA+},\mathcal{A}}(\lambda) \leqslant 4n \cdot \varepsilon_{\mathsf{KDF}}$.