A Systematic Study of Practical & Formal Privacy in the 5G AKMA Procedure

Ioana Boureanu, Stephan Wesemeyer, Fortunat Rajaona, Steve Schneider, Helen Treharne
Surrey Centre for Cyber Security, University of Surrey

***Abstract*—We systematically scrutinise all the facets of privacy in the 5G delegated-authentication procedure called AKMA (Authentication and Key Management for Applications based on 3GPP credentials in the 5G Systems). We define, in general terms, a privacy-threat model and privacy requirements for this protocol. Using these definitions, we find numerous privacy failings in the AKMA protocol. We propose a patch, called AKMA^P, which imposes minimal changes on AKMA, yet it attains all our privacy requirements. We also formalise and analyse all of this in terms of formal privacy-verification in the Dolev-Yao model; to this end, we use the Tamarin prover to systematically carried out our formal analyses of AKMA and AKMA^P.**

1. Introduction

Modern access-control platforms, drawing inspiration from Federated Identity Management (FIM) paradigms, such as OAuth and Single Sign On [14], expand users' authentication across several third parties.. One such system is the new *Authentication and Key Management for Applications based on 3GPP credentials in the 5G Systems*, *AKMA* for short [6]. Indeed, in the 5G (5th Generation) mobile networks, this AKMA procedure was added, to allow for delegated authentication from the network to third-party providers called *application functions* (AFs). For instance, using AKMA, a driver inside a connected car *securely* employs a proprietary system (i.e., an AKMA AF) to pay for road services and tolls automatically, without needing to authenticate in any other way to any of these third-party systems/AFs but by sheer virtue of their car including a SIM-card (Subscriber Identification Module) registered onto a mobile network. For this, the third-parties/AFs will have provisioned AKMA with the mobile operator, and it is the AKMA protocol that the SIM will use when accessing different third-party services. Moreover, as we said, the owners of AKMA-capable devices, be it phones or cars, may not even know that their device is connecting to such a third-party service/AF using their SIM/mobile-network credentials. Importantly, the user's privacy can be severely impacted by these connections: e.g., their connection to a given third-party service or the time of it, or their location, their mobile-network provider, all may leak. Finally, the privacy aspects in AKMA also go the other way: a third-party, AKMA-functionality provider/AF may not wish to be linked to a given set of users or connected to other providers/AFs who also serve AKMA traffic to those users.

Importantly, AKMA is a procedure undergoing standardisation by 3GPP, with the technical specifications (TS) [6] changing constantly in the last two years, and even very recently, i.e., September 2024, with high likelihood for full adoption in future iterations of the mobile-networks' revisions and generations. Privacy is not yet included as a requirement in these TS, but it is being discussed and addressed, as even the non-specialist reader can observe from the last two years' iterations of the specifications [6].

Modified Versions of AKMA. Recently, there has been also an academic endeavour [7] into privacy-centric enhancements of AKMA. Firstly, whilst the proposal in [7] does indeed improve the privacy standing of the AKMA protocol, the work takes certain standpoints that are in conflict with 3GPP's views on AKMA, and more widely they violate certain principles of (the 5G) mobile networks. For instance, the privacy enhancements in [7] break the following core design principle of AKMA: "the AKMA AF shall be able to identify the AAnF serving the UE from the A-KID" (see the AKMA specification [6]). Similarly, in their new design, the UEs employ public keys of the core, when the AKMA specification only allows symmetric cryptography therein. Secondly, the line in [7] considers that the so-called $Ua*$-channels between the UE and the AF is always to be insecure; not only does this leads to trivial privacy failing, but it is again at odds with the AKMA specification [6] which clearly stipulates that $Ua*$-channels are often secure. To this end, one natural question left to the study is: "If a UE has an insecure $Ua*$-channel with one AF and a secure one with another AF, is privacy compromised in relation with the latter AF, or just the former AF?" Lastly, whilst the work in [7] contains formal verification, it does not clearly formulate the privacy properties demanded of the UE, of the AF, or of the two, not outside of the verification tool.

In the context of formal verification, there are a couple of works, i.e., [8], [25], that formally verify AKMA in tools like ProVerif and Tamarin respectively, but focus on security properties, not on privacy; so, their results (be it in new designs, or in verification) are therefore orthogonal to ours[1].

Our Pursuit & Contributions. In contrast with prior lines [7], our work is centred in and around formally proven privacy-enhancements of AKMA, such that:
(a) it is standardisation-friendly and, in any new propositions of AKMA enhancement, does not deviate from the design principles of AKMA or those of 5G networks, and we have been working with 3GPP towards this;
(b) for the first time, we give formal definitions of the privacy and unlinkability requirements of AKMA which are generic, such that any verification tool, be it symbolic [15] or computational [22], can take these definitions and cast them in a specific language;

1. We discuss all these works, on privacy and on security, in more detail in our "Related Work" section.

(c) we prove privacy lacks in AKMA and prove privacy preservation in our AKMA-patch using known results in formal verification [11] and translating our generic unlinkability definitions into symbolic verification models and tools, in systematic and tractable ways.

Whilst our contribution (c) is only of theoretical interest, contributions (a) and (b) walk the tight rope drawn of improving existing designs, systematically, but doing so with practical constraints and backwards compatibility in mind all the while giving provable guarantees of the improvement.

2. Background

2.1. A Glance on (5G) Mobile Networks

In Figure 1 and below, we first give a simplified overview of the relevant 5G network entities for AKMA:

1) the *User Equipment (UE)* – a device (e.g., phone, car with a SIM onboard) subscribed to a mobile service;
2) the *Radio Access Network (RAN)* – the 5G radio "towers" providing network connectivity to the UEs;
3) the *5G core* [3], [4] – servers implementing the operator's logic, split into services: e.g., the *Authentication Server Function (AUSF)* and the *Access and Mobility Management Function (AMF)* authenticate the UE via a protocol called the *5G Registration / AKA (Authentication and Key Agreement)*; the *Network Exposure Function (NEF)* is an API-based proxy for 5G to allow third-party applications' queries; e.g., some of these calls go to the core via the *Applications (AKMA) Anchor Function* (*AANF*);
4) the *Data Network (DN)* – largely, the Internet;
5) the *User Plane Function (UPF)* – largely, gateways routing the UE onto the radio and data networks;
6) *Application Functions (AFs)* – 3rd-party application-servers leveraging, e.g., the network's authentication.

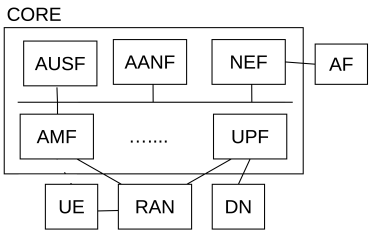

Figure 1: Simplistic Overview of 5G for Non-specialists (see [3], [4] for details)

2.2. AKMA & Its Privacy-Relevant Aspects

We now present the AKMA protocol. We give it diagrammatically in Figure 2, where we emphasise privacy-centric aspects in blue, as well as via *Notes 1-4* to follow.

2.2.1. AKMA – An Overview. AKMA [6] is a delegated authentication service in 5G mobile networks. It aims to extend a subscriber's authentication onto the 5G network further into applications outside the network. In this way, an *application-function (AF) server* identifies a subscriber's UE indirectly, mediated by the core – who, in fact, authenticates the UE. After this proxied authentication, the UE and the application function (AF) server will (re)establish a channel, secured with a key called K_{AF} *(application function key)*.

2.2.2. The AKMA Protocol. The AKMA protocol is succinctly represented here, in Figure 2, and explained further now. The AKMA protocol can be seen as executing in two phases, which we denote as follows: the "initial phase" of AKMA shown on the left of Figure 2, and the "K_{AF}-key generation" phase of AKMA given on the right of Figure 2. The two phases are generally not executed in immediate sequence; also, there is not a one-to-one mapping between them: i.e., for the same UE, there can be several runs of the "initial phase" of AKMA and just one run of the "K_{AF}-key generation" phase, and vice versa.

For the purposes of this work, we can consider that the AKMA protocol is executed between a UE, an AF, and simply the core[2], e.g., the *Application Anchor Function (AAnF)*.

We proceed in describing these phases.

The "Initial Phase" of AKMA. As per points 3.a and 3.b on Figure 2, the end of 5G authentication (i.e., Registration/AKA [2]), an *AKMA-ready UE*[3] and the AAnF will both hold a new K_{AKMA} key for all AFs that the UE may connect to, and index this key under a so-called *AKMA Key Identifier (AKID)* (or *A-KID*, as both terms are used interchangeably in the specifications [6]).

Note 1: In the current AKMA specifications [6], a UE has one single AKID to identify itself in front of *all* AFs serving it. I.e., one UE uses *same* AKID to connect to any number of *different* AFs.

The Derivations for AKID, K_{AKMA} and K_{AF}. The AKID is an identifier formed of a part containing routing information about the user-equipment (denoted below as "ue_routing_info") and a cryptographically derived identifier called "A-TID (AKMA Temporary UE Identifier)" in AKMA's 3GPP specifications [6]. Such an AKID is a pointer to an associated K_{AKMA} key in the core's database. These derivations are described in the Equations (1) to (4) below.

Based on the K_{AKMA} key, the UE and *each* AF associated with the newly derived AKID can derive a new K_{AF} key. In more detail, the key derivations relevant to the AKMA protocol are as follows:

$$A\text{-}TID = KDF(const, K_{AUSF}, \text{“}ATID\text{”}, \text{SUPI}); \quad (1)$$

$$AKID = ue\_routing\_info \,||\, A\text{-}TID \quad (2)$$

$$K_{AKMA} = KDF(const, K_{AUSF}, \text{“}AKMA\text{”}, \text{SUPI}) \quad (3)$$

$$K_{AF} = KDF(const, K_{AKMA}, AF\_ID), \quad (4)$$

whereby: KDF is a hash, $const$ symbolises constants, *SUPI (Subscription Permanent Identifier (SUPI)* is a long-term identifier of the UE described in [1], and AF_ID is constructed as $AF\_ID = AF\_qualified\_name||Ua^*$ with Ua^* being the identifier[4] of the protocol used at the application level between the UE and an application server associated with an *AKMA Application Function (AF)*.

2. It is run between the AF and the AAnF – if the AF is internal to the operator, and between the AF, the AAnF (Application Anchor Function) and the NEF (network exposure function) – if the AF is external to the operator.

3. This readiness is something users buy as part of their mobile-phone contracts, or when they purchase a modern car with a SIM onboard, etc.

4. This is specified in Annex H of 3GPP 33.220 TS [5].

The "K_{AF}-key Generation" of AKMA, with focus on exchanges of privacy-relevant identifiers. We now describe the second phase of AKMA, shown in Figure 2, and we focus on the privacy-relevant identifiers exchanges at various points in the protocol.

In line with AKMA specifications [6], we refer to the channel on which UEs communicate with AFs as the Ua^* *protocol*, the Ua^* *channel*, or the Ua^* *connection*. This is set up by the AF, when the AKMA service is provisioned.

Note 2: According to Annex H of 3GPP 33.220 TS [5] where the Ua^* protocol is specified, the Ua^* protocol can be a secure (e.g., HTTPS) or an insecure protocol (e.g., HTTP).

1) In step 1, the UE sends its current, global AKID to one AF with whom it intends to communicate; this is done over the Ua^* protocol that the UE and this AF are set up to use.
2) In step 2, since the AKID identifies the network associated with it, then the AF at hand sends the received AKID to the right AAnF, alongside with its own identifier AF_ID.
   In line with the specifications [6], this is done over a secure channel between the AF and the AAnF.
3) In steps 3-6, the AAnF gets the long-term network identifiers (SUPI, GPSI) from the part of the core holding this (i.e., the UDM). Also, the AAnF checks that the contacting AF can provide the service to the UE linked to the AKID, based on the subscriber's information.
4) If the above steps are successful, then a new K_{AF} key is generated in step 7.
5) Then, in step 8, this new K_{AF} key, its time-to-live (*TTL*), and a long-term identifier of the UE (i.e., the SUPI or the more generic GPSI) are sent[5] by the AAnF to the AF.
   The UE does the same K_{AF} key-computation, on its side.

   Note 3: The SUPI/GPSI, unlike the AKID, is a long-term identifier. Since the AKID is ephemeral (i.e., it changes with every re-Registration of the UE), the AFs need to receive the SUPI/GPSI, e.g., to book-keep data across sessions of the same UE. However, the SUPI/GPSI is also a network-wide identifier, not specific to AKMA; nonetheless, *all, third-party* AFs serving one UE will receive their SUPI/GPSI. So, in AKMA, multiple AFs hold long-term, network-wide identification information of the same UE.

6) In step 9, the AF replies to the UE, to say if the AKMA-session establishment requested in step 1 was successful or not, including the potential reason of failure; reasons for failures are transmitted by the core to the AF in step 8, should it occur.

   Note 4: The last Ua* message in AKMA key-establishment (i.e., step 9 above/in Figure 2) contains discriminating details, such as the reasons of success/failure of an UE/AKID attempting to connect to an AF/AF_ID.

7) After step 9, provided the UE and the AF deem the process as successful at both ends, all communication between the UE and this AF is encrypted from then on with this K_{AF} inside the Ua^* protocol.
8) If, for whichever reason, the AF requires a new K_{AF} key (i.e., because its TTL has expired, or the Ua^* protocol demands it via its tickets, etc.), then the AF will contact the AAnF, and this phase is re-run from step 2 onwards.

5. The SUPI is sent to the AF if the AF is found inside the *Core* network, while the GPSI is sent when the AF is outside it.

3. Execution & Threat Models

Our execution and threat models follow a commonplace description of protocol executions, for sake of being generic. That is, for now, we omit any protocol measures be it computational, or Dolev-Yao [15].

Our Execution Model & Environment $\mathcal{E}$.. We assume the following settings. The AKMA protocol has several concurrent executions, successful as well as unsuccessful over various AF, UE, and *core* entities (e.g., $AF_1, AF_2, \ldots, UE_1, UE_2, \ldots, core_1, core_2, \ldots,$). The latter entities are denoted *parties* and have been *enrolled* in the system such as to have all the cryptographic material and knowledge to execute the AKMA protocol.

A party engaging once in an AKMA execution describes an *instance* of that party. In the enrolment or provisioning[6] phase, *channels* between parties are created, emulating that the instances of different party[7] *types* (AF, UE, and *core*) can communicate with one other, with incoming and outgoing messages, as per the AKMA protocol; we speak of the channels on which the Ua^* protocol is run as the Ua^* *channels*.

Each party can have several concurrent instances running at any given point. The interleaving of at least three instances of an AF, a UE and a *core* party respectively, creating an execution of the AKMA protocol is denoted as an AKMA-protocol *session*. A session can be *partial* – if not all the instances involved in it have reached the final step in AKMA, or –otherwise, it is said to be *complete*.

We consider an *AKMA execution environment* $\mathcal{E}$ in which several parties of each type (AF, UE, and *core*) are involved, several instances of each party are present, and several partial and complete sessions are under way.

Our Threat Model $\mathcal{T}$. This is defined as follows:

T1) an AKMA execution environment $\mathcal{E}$, as described above;
T2) the presence of an active adversary $\mathcal{A}$ who can corrupt AF and UE parties from the enrolment phase or during the protocol execution;
T3) a party compromised at enrolment phase will be totally controlled by the adversary, including the attacker controlling all its long-term cryptographic

6. This is when a third-party such as a car manufacturer partners with an operator to provide AKMA to some of its subscribers.

7. Any AKMA-capable device on the UE-side, be it phone, car, etc., is a party of type UE. Operators such as Orange, Vodafone, Telefonica, are parties of type core. Third-party AKMA providers, say BMW, YouTube, etc., are parties of type AF.

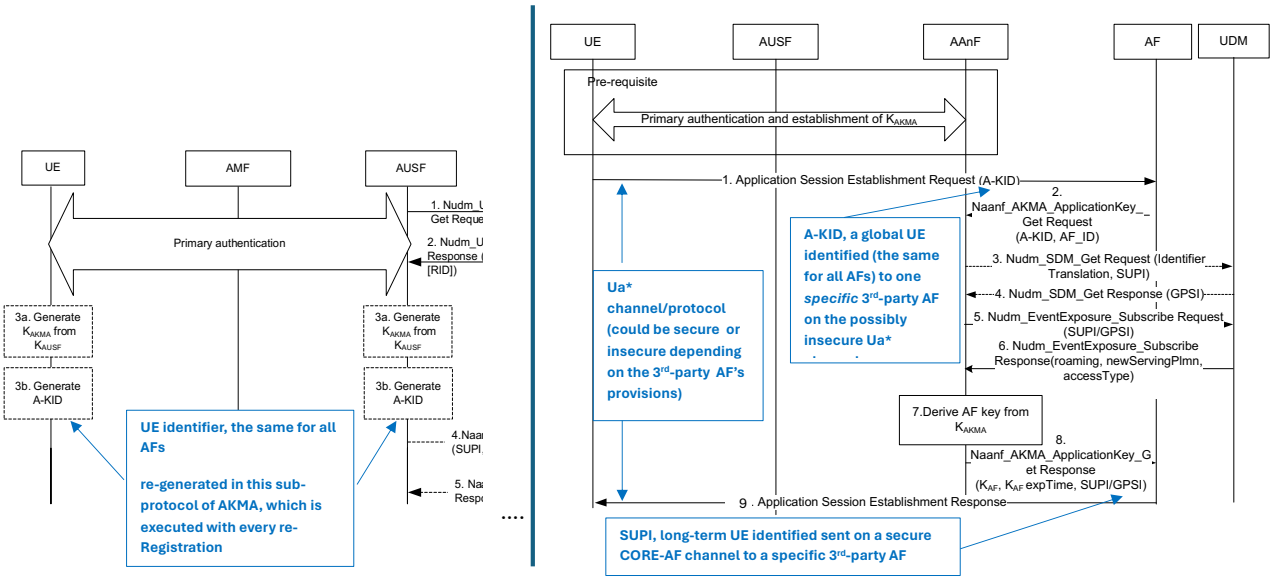

Figure 2: AKMA. Left – $AKID$ generation as per Fig.6.1-1 [6]. Right – K_{AF} generation as per Fig.6.2-1 [6]); messages detailed by us, above. Privacy aspects added by us, in blue

material, which need not be the case for a party compromised during the protocol execution;

T4) trusted *core*s, meaning that the attacker cannot corrupt any party of type *core*;

T5) corrupted parties will not follow the AKMA protocol;

T6) honest parties follow the AKMA protocol;

T7) upon corruption, the attacker is subsumed by itself together with all the parties it has corrupted, and excluding any honest parties;

T8) as per usual, if AKMA is studied against a privacy notion underpinning one party P_1, then this party P_1 cannot be corrupt;

T9) as per usual, if AKMA is studied against a privacy notion underpinning collectively a certain UE_i and a certain AF_j, then not both parties UE_i and AF_j can be corrupt;

T10) the channels between the core and the AFs remain secure, ensuring authenticity, confidentiality and integrity. I.e., our corruption of an AF is not at the level where it can defeat the authentication on the channels with the core[8];

T11) the active adversary can listen on all channels, at the security-level of the channel (as per specified in AKMA) compound with his corruptions. I.e., if the UE-to-AF channel is secure and the end-points of the channel are not corrupt, the attacker cannot defeat the security of the channel; the UE-to-core communication can be compromised, if the UE is compromised; the AF-core channels remain secure.

Security Settings $\mathcal{S}$. We consider the following security settings denoted, as a whole, by $\mathcal{S}$:

S1) All the privacy properties we study are from the perspective of our attacker, i.e., from the perspective of corrupted parties and not honest parties.

S2) For any privacy property that we study with respect to the UE, we consider the Ua^*-channel between UE_i and a party AF_j be secure (i.e., ensuring authenticity, integrity, and confidentiality), since the privacy of the UE is trivially broken when this channel is insecure. Whilst we are interested in studying privacy primarily on a secure Ua^*-connection say between UE_j and AF_i, we do allow that other Ua^*-connections, say, e.g., between UE_i and AF_k, be insecure. This is exactly in line with the possibilities implied by the 3GPP specifications. Notably, (i) there may be privacy attacks which occur because the insecurity of one Ua^*-channel leads to a privacy-attack on another Ua^*-channel even if the latter was secure; and, (ii) there may be privacy attacks which occur even if all Ua^*-channel are secure.

8. Arguably, this is realistic. If corrupted AFs can create fake certificates to connect to the core impersonating other, honest AFs and thus breaking the security of these channels, then this is a security flaw w.r.t. the (public-key) infrastructure in general, defeating entire sets of security procedures relying on it, not just AKMA. If this fails, then all security assumptions and procedures by 3GPP [1], are futile ab initio.

4. Privacy Notions for AKMA

There are two identifiable AKMA parties: the users and the application servers/functions. From the viewpoint of all their AKMA identifiers we proceed to define privacy notions, in our threat model.

4.1. UE-focused, AKID-based Privacy in AKMA

Firstly, recall from Section 2 that the UE is identified in AKMA via the so-called AKID, as Section 6.2.2 of the AKMA specifications [6] say:

> "The AKID functions as a temporary user identifier." [6], p. 16

In Section 2.2.2, in more detail, we saw that the AKID contains the so-called ATID and some "$ue\_routing\_info$". The latter contains static information, which can be inverted, i.e., linked backed to a UE by any party that can corrupt the core; however, in our threat

model, the core is trusted (see (T4) in Section 3). Also, this $ue\_routing\_info$ is a 4-digit number, which is not necessarily UE-specific. Finally, one given $ue\_routing\_info$ will be the same outside of the AKMA protocol, so this is not an AKMA-centric identifier; as such, looking at privacy via the lens of the $ue\_routing\_info$ is not of interest. The latter part of the AKID (i.e., the ATID), however, is cryptographically re-generated per UE, with every Registration and it is used to identify the UE by parts of the network (i.e., the AF) which we do consider corruptible (see (T2) in Section 3). So , w.r.t. any privacy linked to AKID, we will in fact be interested in privacy analysis w.r.t. the A-TID, and henceforth, we use "AKID" to mean "A-TID" for purposes of privacy analysis.

4.1.1. What privacy notions. AKIDs, as AKMA-specific identifiers, are ephemeral: i.e., as Section 2.2.2 explained, each of these are re-generated with every Registration (see left-hand side of Figure 2), as Section 6.1 of the AKMA specifications [6] say:

"AKID can only be refreshed by a new ... authentication", [6], p. 14

So, we are interested in strong forms of privacy, such as tracking UEs based on ephemeral identifiers:

Definition 1. **Strong Secrecy of the UE's AKID in AKMA.** In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *strong secrecy of UE's AKID* (*SS_UE-AKID*) holds if:

- for any attacker $\mathcal{A}$ in the settings $\mathcal{T},\mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$ such that $\mathcal{A}$ does not know[9] (now) the active AKID of an honest UE, the attacker $\mathcal{A}$ cannot follow (future) presences of this AKID within secure Ua^* connections within $\mathcal{E}$.

In the spirit of our earlier setting (S2) and/or Note 2 in Section 3, Definition 1 focuses on secure Ua^* connections (as opposed to any Ua^* connections); it can be seen as asking: if AKIDs remain secret, do secure channels imply that one cannot track, observe, link-together the executions of an AKID (maybe based on a some protocol data linking to specific AKIDs)?

Now, we move to a different notion of privacy, which we call "post-compromise privacy"; see Definition 2.

Definition 2. **Post-Compromise Privacy of UE's AKID in AKMA.** In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *post-compromise privacy of UE's AKID* (*PP_UE-AKID*) holds if:

- for any attacker $\mathcal{A}$ in the settings of $\mathcal{T}$ and $\mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$ such that $\mathcal{A}$ knows that an AKID of an honest UE was present in the execution environment $\mathcal{E}$, the attacker $\mathcal{A}$ cannot follow future presences of this AKID within secure Ua^* connections within $\mathcal{E}$.

Definition 2 is also in the spirit of our earlier setting (S2) and/or Note 2 in Section 3, i.e., AKIDs' privacy studied on secure Ua^* channels yet bearing in mind that insecure Ua^* channels exist, but also with a strong focus on Note 1 (i.e., the fact that the AKID is not AF-specific). And, more intuitively, *PP_UE-AKID* asks whether secure Ua^* channels in AKMA can help re-gain or repair AKID-based privacy potentially lost due to certain insecure Ua^* channels. Since one AKID is the same for all AFs (see Note 1 in Section 2.2), each AF having with their own $Ua*$-security provision, the question in Def. 2 is pertinent. In other words, is the co-existence of secure and insecure Ua^* channels across various AFs in AKMA fatal to the privacy of "global" AKIDs? Once an AKID leaks (e.g., on an insecure channel), can we track this AKID forever, for instance due to (revealing) message in AKMA?

More formally, the AKMA protocol would provide PP_UE if no attacker were able to tell that a leaked AKID will be present on secure Ua^* channels/protocols, after the leak. This leakage could occur, e.g., as follows: (a) due to the fact that the Ua^* protocol can be insecure between one honest UE and one honest AF_1; (b) due to the attacker having corrupted one single AF and thus getting honest AKIDs anyway, etc., which may connect to other AFs (since AKIDs are AF-agnostic – see Note 1).

4.2. Privacy of AKMA's Application-Functions

We now move to privacy properties pertaining to the application functions (AFs) and their identifiers – the AF_IDs. We propose one privacy notion, in Definition 3:

Definition 3. **Weak Privacy of the AF in AKMA.** In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *weak privacy of the AF* (*WP_AF*) holds if:

- any attacker $\mathcal{A}$ in the settings of $\mathcal{T},\mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$, cannot detect the presence of an AF_ID pertaining to an honest AF, within the execution environment $\mathcal{E}$.

Definition 3 says that weak privacy of the AF (*WP_AF*) holds if no attacker, in our model, is able to learn, by executing AKMA, if one AF or another is present onto a network.

4.3. Unlinkability of UE-AKIDs and AFs

We move to describing the notions of whether an attacker can find a link between a given UE and a given AF, or vice versa, in the case where the two are honest and are communicating securely. Intuitively, asking around such linking makes sense due to a number of reasons. For instance, corruptions of the UEs vs. the AFs lead to different observability levels, since the AKID is global to all AFs, and a legitimate AF may serve one AKID but not another. Also, recall our Note 4 in Section 2, whereby various codes for successes or failures of the (re-)establishment of K_{AF} are sent on the Ua^* channel to every UE, thus leaking their status w.r.t. connections to AFs.

Definition 4. **Unlinkability of UE-AKIDs and AFs in AKMA (on the Ua* Channel).** In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *unlinkability of UE-AKIDs and AFs* (*L_AKID-AF*) holds if

- any attacker $\mathcal{A}$ in the settings of $\mathcal{T},\mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$, cannot detect

9. Here, knowing an AKID is equivalent to knowing that this AKID was/is present in the execution environment $\mathcal{E}$.

Table 1: Our Privacy Notions for AKMA

Property	About	Meaning	Setting
strong secrecy of UE's AKID (*SS_UE-AKID* Def. 1)	AKID	Can $\mathcal{A}$ track AKIDs, if AKIDs' confidentiality holds?	secure Ua^* channels, secure core-AF channels
post-compromise privacy of UE's AKID (*PP_UE-AKID*, Def. 2)	AKID	Can $\mathcal{A}$ track an AKID even when only sent encrypted, once this AKID's confidentiality was breached?	secure & insecure Ua^* channels, secure core-AF channels
weak privacy of the AF (*WP_AF*, Def. 3)	AF_ID	Can $\mathcal{A}$ track AF_IDs even if all channels are secure?	secure & insecure Ua^* channels, secure core-AF channels
unlinkability of UE-AKIDs and AFs (*L_AKID-AF*, Def. 4)	AKID and AF_ID	A mix of PP_UE and SP_AF: i.e., Can $\mathcal{A}$ link together an AKID and an AF_ID as communicating?	secure & insecure Ua^* channels, secure core-AF channels
unlinkability of UE-SUPIs and AFs (*L_SUPI-AF*, Def. 5)	SUPI/GPSI and AF_ID	Can $\mathcal{A}$ link together a SUPI/GPSI and an AF_ID as having been in contact?	no relation to Ua* channels secure core-AF channels

if an honest UE identified via their current AKID is communicating with an honest AF, within secure Ua^* connections within $\mathcal{E}$.

Specifically, this linkability can occur in two "directions", from the UE to the AF, or vice versa:

(a) In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *unlinkability of UE-AKIDs to AFs* ($\exists UE-AKID \rightarrow \forall AF$) holds if:
any attacker $\mathcal{A}$ in the settings of $\mathcal{T}, \mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$, targeting a given honest UE cannot tell this UE is communicating with a specific honest AF, within secure Ua^* connections within $\mathcal{E}$.

(b) In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *unlinkability of AFs to UE-AKIDs* ($\exists AF \rightarrow \forall UE-AKID$) holds if:
any attacker $\mathcal{A}$ in the settings of $\mathcal{T}, \mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$, targeting a given honest AF cannot tell this AF is communicating with a specific honest UE, within secure Ua^* connections within $\mathcal{E}$.

Part (a) of Definition 4 ask this: given an AKID, is there a specific AF (i.e., AF_ID) we can link them to? Part (b) then asks: given an AF (i.e., an AF_ID), is there a specific AKID we can link them to? And these links as per security setting (S2) have to be made in the strongest sense possible: i.e., when the Ua* channel is secure.

4.4. UE-Focused, SUPI-based Privacy in AKMA

Now, we shift attention from the ephemeral AKMA-specific UE identifiers that are the AKIDs to the AKMA-nonspecific, network-wide, long-term identifiers of UEs used in step 8 of the protocol – SUPIs/GPSIs. This is linked to our Note 3 in Section 2.2.

Via Note 3 in Section 2.2, we underline that in AKMA, the SUPIs/GPSIs are sent in step 8 of the protocol from the core to the intended, honest AFs on an authenticated channel. So, unlike in the case of the AKIDs which could be sent even on insecure channels, there is little point in us looking at the "leakage-like" privacy of the SUPIs/GPSIs (as we did for the AKIDs in Definition 3).

However, what Note 3 alludes to is that corrupt AFs can go beyond what each honest AF knows on SUPIs/GPSI. That is, intuitively, several corrupt (or "honest but curious") AFs can collude and find out which SUPIs/GPSIs are served by each. And – since the SUPIs/GPSIs are long-term – if this SUPI/GPSI–tracking were possible, then it is worse than tracking AKIDs (which we formalised in Definition 4). So, we generalise this notion of adversarial tracking of SUPIs/GPSIs that goes beyond the honest receipt of SUPIs/GPSIs in step 8 of AKMA.

Definition 5. **Unlinkability of UE-SUPIs and AFs in AKMA.** In the threat model $\mathcal{T}$, under the security assumptions $\mathcal{S}$, *unlinkability of UE-SUPIs and AFs* (*L_SUPI-AF*) holds if

- any attacker $\mathcal{A}$ in the settings of $\mathcal{T}, \mathcal{S}$, spanning any AKMA execution environment $\mathcal{E}$, cannot detect if a SUPI/GPSI of an honest UE can connect to parties other than themselves.

Notes & Summary of Our Definitions. To summarise, all notions we introduced thus far are recounted in Table 1.

Our first four properties (Def. 1 – Def. 4) reason differently about AKIDs, AFIDs, and their inter-relation, also modulo the (in)security of the Ua* channel. Indeed, our unlinkability notion in Def. 4 does not necessarily follow from the previous two notions on privacy:

Definition 4 $\not\Rightarrow$ Definition 3;

Definition 4 $\not\Rightarrow$ Definition 2.

This is because of the following three aspects. Firstly, an attack w.r.t. Definition 3 does not imply an attack w.r.t. Definition 4. That is, an attacker may be able to break post-compromise privacy of UE's AKID on AKIDs, by observing just something on the first message on the Ua^* channel between the UE and the AF, but meanwhile being unable to say/see anything of the AF_ID sent on the channel between the AF and the core. Second, an attack w.r.t. Definition 2 does not imply an attack w.r.t. Definition 4. That is, one can break weak privacy of the AF by corrupting a UE in the enrolment phase and getting an AF_ID from the K_{AF} derivation function, but not knowing which other AKIDs this AF/AF_ID serves. So, investigating all notions 1–4 is valid pursuit at this stage, as we do not know whether they imply one another in AKMA, or not[10].

Definition 5 is incomparable to the first four definitions, as the first refer to AKIDs and the latter to SUPIs/GPSIs, and the AKID and SUPIs/GPSIs are not inferrable from one another.

5. Our Formal Analysis of Privacy in AKMA

5.1. Our Symbolic Models for AKMA's Privacy

We took our generic threat model and properties and under-approximated them into Dolev-Yao (DY) mod-

10. Perhaps, there is enough information in AKMA executions that once, e.g., an AKID is trackable, it also follows which is the AF_IDs that this AKID connects to, or vice versa, or not all, or it may depend on the security of the Ua^* channels, or on the corruptions' settings.

els [10], [19], [12], from the viewpoint of disproving them (rather than proving them), in the case of AKMA.

We implemented this in various models, specifications and we carried out the analysis in the Tamarin prover [19].

- All our Tamarin files are at [24].
- We do our Tamarin proofs in Tamarin 1.6.1 and in Tamarin 1.8 (the latest). There is no difference in meaning between our models/files in the two Tamarin versions. We use Tamarin 1.6.1 for the `diff`-equivalence proofs, purely since they perform badly[11] in Tamarin 1.8. Our repository [24] makes clear which files/proofs have been run with which Tamarin version.
- Apart from disproving our privacy notion on various models for AKMA (sometimes with restrictions in place for tractability), we also prove agreement properties for AKMA, in a separate and un-restricted model but consistent with our privacy modelling.

Now, we discuss concretely our various (family of) models for AKMA, resulted from varying all security assumptions underpinning our privacy notions in Section 4.

DY Models Varying the Security of the Ua^* Channels. Our privacy properties –in line with our threat model in Section 3– vary the security assumptions: e.g., strong secrecy of UE's AKID in Definition 1 requires secure Ua^* channels, whereas post-compromise privacy of UE's AKID in Definition 2 allows for the possibility for the Ua^* channels to be both secure and insecure (i.e., one way for AKIDs to leak is insecure Ua^* channels). Thus, we start by yielding two classes of models:

- $\mathcal{M}^{\text{Sec\_Insec-}Ua^*}$ – models where Ua^* channels can be both secure or insecure.

  The above models are suited to modelling *PP_UE-AKID* (Def. 2), *WP_AF* (Def. 3), unlinkability of UE-AKIDs to AFs as well as unlinkability of AFs to UE-AKIDs (Def. 4).

- $\mathcal{M}^{\text{Sec-}Ua^*}$ – models where Ua^* channels can only be secure.

  The above models are suited to modelling *SS_UE-AKID* (Def. 1).

Main Characteristics of Models $\mathcal{M}^{\textbf{Sec\_Insec-}Ua^*}$ and $\mathcal{M}^{\textbf{Sec-}Ua^*}$. In these models, we have followed the following settings:

- There are no restrictions on the number of AFs, *core*s or UEs.
- Each UE is associated with one and only one *core*, which is standard for mobile-networks' subscribers.
- Each UE is assigned 2 AFs, at random, from all the AFs "on-boarded" in the setup of the model (i.e., point 1 above). These AFs remain the same throughout the protocol's multiple execution: i.e., we do not model full subscription or re-subscription by UEs to AFs, as this is not part of the AKMA specifications.
- In the $\mathcal{M}^{\text{Sec-}Ua^*}$ models, the Ua^* channels between the UEs and its AFs are always secure.
- In the $\mathcal{M}^{\text{Sec\_Insec-}Ua^*}$ models, the Ua^* channels between the UEs and its AFs are chosen to be secure or insecure, non-deterministically, in the setup of the UEs and AFs in the model. We ensure that in each model there is at least an UE with an insecure Ua^* channel and one with a secure channel; this is to ensure we are satisfying the premises of our definitions.
- The channel between AF and the *core* is always secure.
- UEs can re-Register with the *core* (i.e., as such, change of their AKID, and renew their K_{AKMA}) unboundedly many times, and all parties can re-run the AKMA protocol, in line with specifications [6] and Fig. 2.

DY Models Varying the Privacy Encoding. We use two ways of encoding and disproving our privacy properties in DY tools.

Firstly, we disprove some of our privacy properties in AKMA, by using diff-equivalences [21]. A diff-equivalence is a strong reachability condition. Intuitively, when proving diff-equivalence between processes $P|Q$ and $P'|Q'$, diff-equivalence requires that P is equivalent (observationally by an attacker but sometimes even w.r.t. internal reductions) to P' and Q is equivalent (in the same sense) to Q'. We put the diff-equivalence-based statements in models that we refer to as $\mathcal{M}^{\textbf{\textit{diff}}}$**:**

- $\mathcal{M}^{\textbf{\textit{diff}}}$**:** — models where we use diff-equivalence to (dis)prove privacy properties.

We use $\mathcal{M}^{\textit{diff}}$ models with unlinkability of UE-AKIDs to AFs and unlinkability of AFs to UE-AKIDs in Def. 4.

Secondly, we exhibit some privacy failures in AKMA via trace-based lemmas that explicitly show the attacker tracking a UE, or an AF. We call the models where we use for this: $\mathcal{M}^{\textbf{noDiff}}$:

- $\mathcal{M}^{\textbf{noDiff}}$ — models where we use trace properties not holding as witnesses disproving our privacy properties.

We use $\mathcal{M}^{\textbf{noDiff}}$ models with, e.g., *PP_UE-AKID* (Def. 2).

We use the $\mathcal{M}^{\textbf{noDiff}}$ models, separately from the $\mathcal{M}^{\textbf{diff}}$ models for two reasons: (a) due to inefficiency with Tamarin 1.8 and diff-equivalences; (b) in the former, we are able to show failures of our privacy properties on AKMA in very clear, explicit ways, via attack traces that are a formal witness to the causes of these attacks as well.

Main Characteristics of Models $\mathcal{M}^{\textbf{diff}}$ and $\mathcal{M}^{\textbf{noDiff}}$. In these models, we have followed the following settings:

- Both these models can be of the $\mathcal{M}^{\text{Sec\_Insec-}Ua^*}$ type or of the $\mathcal{M}^{\text{Sec-}Ua^*}$ type.
- The $\mathcal{M}^{\text{diff}}$ models are simplified models of AKMA in that they contain at most three UEs and two AFs; these simplifications come from the fact that we aim to prove specific *diff*-equivalences driven by our notions unlinkability of UE-AKIDs to AFs and unlinkability of AFs to UE-AKIDs, and this is a safe abstraction to undertake: if the attacker can distinguish protocol aspects in this simplified setting, then the attacker can also distinguish these in richer

11. This issue is known; see here https://github.com/tamarin-prover/tamarin-prover/issues/615 that the developers are investigating but it remains an issue even in Tamarin 1.10 .

settings. To this end, the models focus on proving *diff*-equivalence primarily based on AKIDs (driven by the unlinkability of UE-AKIDs to AFs notion) or proving *diff*-equivalence primarily based on AFIDs (driven by the unlinkability of AFs to UE-AKIDs notion); we call the former models $\mathcal{M}^{diff}_{AKID}$ and the latter models $\mathcal{M}^{diff}_{AFID}$.

To see at a glance the relationship between our classes of models mentioned thus far, please refer to Figure 3.

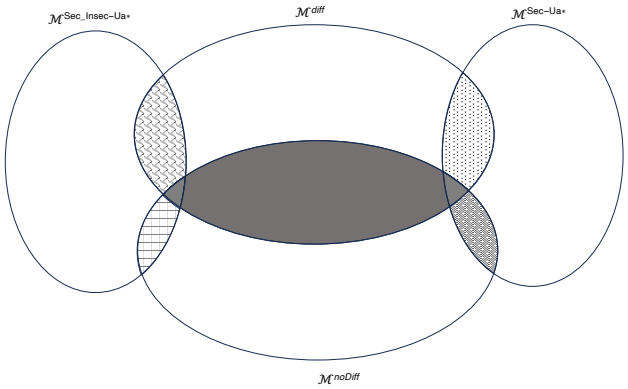

Figure 3: Our Classes of Privacy Models for AKMA (of most interest are checkered areas: $\mathcal{M}^{\text{noDiff,Sec\_Insec-Ua*}}$, $\mathcal{M}^{\text{diff,Sec\_Insec-Ua*}}$; of interest for one property are dotted areas: $\mathcal{M}^{\text{noDiff,Sec-Ua*}}$, $\mathcal{M}^{\text{diff,Sec-Ua*}}$; dark area – not used)

The grey area on Figure 3 denotes *diff*-equivalence properties as well as trace-based lemmas, present in the same model, and both being used to disprove privacy. We do not use models in this set. Actually, Table 1 shows which class of model we actually use for which privacy property. Thus, this table also shows that the models of most interest are those in the checkered areas in Figure 3. The models within the dotted areas are also of interest: i.e., these are required by *SS_UE-AKID* in Def. 1.

From here on, we express the intersection of the classes of models (i.e., the checkered and the dotted areas in Figure 3) in a natural way: e.g., $\mathcal{M}^{\textbf{\textit{Sec\_Insec-Ua*,diff}}}_{AKID}$ is a model in which we (dis)prove *diff*-equivalence based on AKIDs, and the Ua^* channels can be both secure and insecure.

5.2. Our Verification of AKMA's Privacy

We now explain how we modelled the various privacy properties from Section 4.

We recount the results of our privacy analyses and findings discussed below in Table 2.

5.2.1. Analysing Post-compromise Privacy of the UE. In line with the secure-channels' setting required by *PP_UE-AKID*, we encoded *PP_UE-AKID* in a model in the subclass $\mathcal{M}^{\text{noDiff,Sec\_InSec-}Ua^*}$ and in a model in the $\mathcal{M}^{\text{noDiff,Sec-}Ua^*}$. We wrote a "no-desynchronization (ND)" lemma a la [17], [11]: it checks whether all honest UE and honest AF are synchronised: i.e., whether an arbitrary UEs with a given AKID and an arbitrary AF that this AKID has contacted to have a synchronous view of their runs of AKMA, over repeated executions. To be able to show this, we modelled a crude version of sessions' management across executions of the AKMA protocol (i.e., an UE's view to have contacted a given AF once, twice, and keeping states in between); both for the UE and the AF, we did this via counters[12]. At the high-level, the ND lemma would hold of these counters are in sync at the UE and at the AF's end.

This ND lemma fails and this refutes our *PP_UE-AKID*.

The exact attack trace, in our $\mathcal{M}^{\text{noDiff,Sec\_InSec-}Ua^*}$ model, in Tamarin, is as follows: (1) An honest UE1 established an AKMA session with an honest AF1, successfully; (2) UE1 then contacts AF2 with the same AKID, for a session; (3) AF2 is corrupted and leaks the AKID; (3) The DY attacker uses the AKID through a corrupt UE2 to contact AF1 again, thus de-syncing the counters in the honest AF1 from those in the honest UE1. An immediate consequence of this no-desynchronisation attack is that the attacker learns of the state of UE1 on the network. That is, in the Ua^*-connection opened via the corrupt UE2, the attacker also receives information from AF1 on UE1. The information leaked is whether the request for a new the Ua^*-connection has gone through, and if not – why not: e.g., because another UE1 connection (to AF1) is live.

In Figure 4, we explain the attack-trace above via an image, and –primarily via lower, left-hand-side rectangle– we also summarise again why this entails that our *PP_UE-AKID* fails. As Figure 4 also depicts, note that this *PP_UE-AKID* fails both on $\mathcal{M}^{\text{Sec\_InSec-}Ua^*}$ models and on $\mathcal{M}^{\text{Sec-}Ua^*}$ models (i.e., irrespective if the Ua^* channel is secure or not), as long as the attacker – as per our threat model – can corrupt UEs and AFs.

In Appendix A, in Figure 7, we show the code of our no-desynchronisation (ND) lemma.

5.2.2. Analysing Weak Privacy of the AF. In line with the secure-channels' setting required by *WP_AF*, we encoded *WP_AF* in a model in the subclass $\mathcal{M}^{\text{noDiff,Sec\_InSec-}Ua^*}$ and in a model in the $\mathcal{M}^{\text{noDiff,Sec-}Ua^*}$. And, to check property *WP_AF*, we actually looked at a "well-authentication (WA)" lemma a la [17], [11]. Our WA lemma checks if every time the *core* evaluates positively the start of an AKMA session for a UE, this UE is honest and has requested the session per se.

Our WA lemma fails for AKMA and this shows *WP_AF* failing.

The Tamarin attack-trace, on our $\mathcal{M}^{\text{noDiff,Sec\_InSec-}Ua^*}$ model, is as follows: (1) An honest UE1 tries to establish an AKMA session with an honest AF1, but the attacker blocks this getting the UE1's AKID; (2) The attacker uses the AKID through a corrupt UE2 to contact AF1 again, thus making the core allow an AKMA-session for UE1 but via the corrupt UE2. This means that the attacker gets information about AF1 illicitly (e.g., which UEs they serve, when); this refutes our weak privacy of the AF.

12. To make things more tractable, in the models submitted for review, the counters in the AF go up to a maximum of 3 requests for each UE.

Table 2: Our Systematic Privacy Verification of AKMA in Tamarin (Tamarin files at [24])

-	Property	About	Model Class	Verification Method	Status	Filename for Model(s)	Time
1	strong secrecy of UE's AKID (Def. 1)	AKID	$\mathcal{M}^{\text{diff,Sec-Ua*}}$	diff-equiv on two unknown AKIDs	holds	diff/SS_AKMA_Sec	cca. 2min
2	post-compromise privacy of UE's AKID (Def. 2)	AKID	$\mathcal{M}^{\text{noDiff,Sec(_Insec)-Ua*}}$	trace-based desychronisation/tracking	ND_UE_AF fails	indist/PP_AKMA	manual (with oracle cca. 10 mins)
3	weak privacy of the AF (Def. 3)	AF_ID	$\mathcal{M}^{\text{noDiff,Sec(_Insec)-Ua*}}$	trace-based desychronisation/tracking	ND_UE_CORE fails WA fails	indist/WP_AKMA	cca. 60 mins
4	unlinkability of UE-AKIDs to AFs (Def. 4)	AKID, AFID	$\mathcal{M}^{\text{noDiff,Sec(_Insec)-Ua*}}$	trace-based desychronisation/tracking	ND_UE_AF fails ($\neg$ Def. 2 $\Rightarrow \neg$ Def. 4)	indist/PP_AKMA	manual (with oracle cca. 10 mins)
5	unlinkability of UE-AKIDs to AFs (Def. 4)	AKID & AF_ID	$\mathcal{M}^{\text{Sec\_Insec-Ua*}}$	– diff-equiv between AKID1 speaking to AFID i and AKID2 speaking to AFID i	fails	diff/Unlink1_AKMA	cca. 3 mins
6	unlinkability of AFs to UE-AKIDs (Def. 4)	AKID AF_ID	$\mathcal{M}^{\text{Sec\_Insec-Ua*}}$	–**6.1.** diff-equiv over AKID-AFID1 on Ua* secure, when no known AKID, one known AFID1 –**6.2.** diff-equiv over AKID-AFID1 on Ua* secure, when no known AKID, one known AFID1	holds fails	diff/ Unlink2(oneUE)_AKMA diff/Unlink2(twoUEs)_AKMA	cca. 3 mins cca. 3 mins
7	unlinkability of UE-SUPIs and AFs (Def. 5)	SUPI/GPSI AF_ID	any/not about Ua*	diff-equiv, i.e., $\mathcal{A}$ cannot associate SUPI/GPSI i to AF_ID j	fails	diff/Unlink_AKMA_GPSI	cca. 2 mins

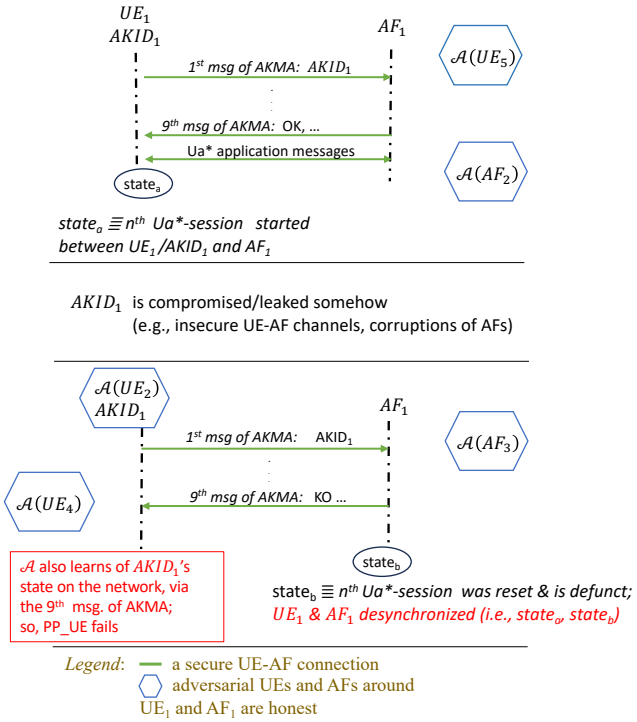

Figure 4: *PP_UE-AKID* Failing (via a desynchronisation attack between UEs and AFs)

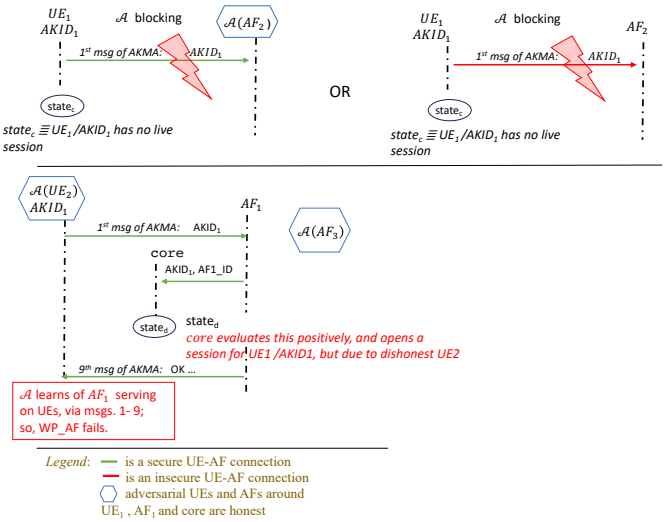

Figure 5: *WP_AF* Failing via a MiM attack between UEs and *core*, "visible" as trace

In Figure 5, we explain the attack-trace above, and –primarily via lower, left-hand-side rectangle– we also summarise again why this entails that our *WP_AF* fails. As Figure 5 depicts, note that *WP_AF* fails both on $\mathcal{M}^{\text{Sec\_InSec-}Ua^*}$ models and on $\mathcal{M}^{\text{Sec-}Ua^*}$ models (i.e., irrespective if the Ua^* channel is secure or not), if the attacker – as per our threat model – can corrupt UEs and AFs.

In Appendix A, in Figure 8, we show the code of our UE-to-*core* well-authentication lemma.

We note that property *WP_AF* can also be shown to fail as via an no-desynchronisation (ND)-style lemma failing between the *core* and UE1 above; we have done this too, modelling session-booking via counters as in the case of the model dis-proving *PP_UE-AKID*, only that this time the counters used and quantified over are, of course, inside the *core*[13] and UEs. Using Figure 5, we can see that $state_c$ of honest UE1 and $state_d$ of the (honest) *core* are also de-synchronised.

13. To make things more tractable, in the models submitted for review, the counters in the *core* go up to a maximum of 3 requests for each UE.

Our the code of our ND lemma for this is given in Appendix A, in Figure 9.

5.2.3. Analysing Strong Secrecy of the UE. In line with the secure-channels' setting required by *SS_UE-AKID*, we encoded *SS_UE-AKID* in a model in the subclass $\mathcal{M}^{\text{diff,Sec-}Ua^*}$. And, to check property *SS_UE-AKID*, we actually looked at analysing a diff-equivalence quantifying over two AKIDs (the same or different, belonging the same UE or not).

In fact, in the case of strong secrecy of UE's AKID, as per Def. 1, we need that the attacker necessarily does not learn the AKIDs. Since we operate in a DY setting, we therefore need the Ua^* to be secure, hence building a model in $\mathcal{M}^{\text{Sec-}Ua^*}$. In this case, for simplicity and clarity of the diff proof, we restrict[14] the model to 2 UE and 2 AFs. W.r.t. the AKIDs to distinguish, we take the following approach. We use three types of AKIDs:
(a) for UE1, we single out AKID1 is for AF1 (and AF2), which becomes AKID2 after the re-Registration;
(b) for UE2, we single out AKID3 for AF1 (and AF2).
Then, we prove `diff`-equivalence of AKID1 to all the other AKIDs, in fact w.r.t. also to other AKID-indexed information, i.e., long-terms keys, K_{AF}s, K_{AKMA}s, etc.: e.g.,

14. Note that if the attacker distinguishes AKIDs in this case, it will also distinguish them in the ampler models, without restrictions on the numbers of parties.

```
diff(<A_KID3, K_AF3, K_AKMA3, ˜K_sh3, ˜K_sh3>,
       <A_KID2,K_AF2, K_AKMA2,˜K_sh2, ˜K_sh4>).
```

This `diff` is clearly a faithful representation of Def. 1: that is, " for the active AKID of an honest UE" (i.e., AKID1) , "the attacker $\mathcal{A}$ cannot follow (future) presences of this AKID within secure Ua^*-connections" (i.e., AKID2), compared to other presences in the AKMA executions (i.e., AKID3).

The strong secrecy of UE's AKID property holds in AKMA.

5.2.4. Analysing Unlinkability Users and AFs. As per Def. 4, there are two types of unlinkability: one from the direction of linking "known" UEs to AFs (i.e., unlinkability of UE-AKIDs to AFs), and one from the direction of linking "known" AFs to UEs (i.e., unlinkability of AFs to UE-AKIDs). We discern between the analysis of the two in the below.

5.2.4.1 **Analysing unlinkability of UE-AKIDs to AFs.**. Looking at the way post-compromise privacy of UE's AKID fails (see Section 5.2.1 and/or Figure 4), it should become clear that, in the case of the AKMA protocol, the failure of post-compromise privacy of UE's AKID also leads to a failure of unlinkability of UE-AKIDs to AFs a.k.a. $\exists UE - AKID \rightarrow \forall AF$ (Def. 4).

I.e., in AKMA: $\neg$ Def. 2 $\Rightarrow \neg$ Def. 4

That is, the attacker in the run in Figure 4 not only tracks an AKID, but via the details in msg. 1 and msg. 9 of the AKMA protocol, it links this AKID (i.e., AKID1) to an AF (i.e., AF1). In Tamarin, we can easily quantify over an extra variable (i.e., the AF) in the lemma (i.e., ND) that shows post-compromise privacy of UE's AKID failing, and thus show $\exists UE - AKID \rightarrow \forall AF$ failing. This is why in Table 2 where we summarise our results, row 2 and row 4 report the same models/results.

However, for completeness, we do the above not only in $\mathcal{M}^{\text{noDiff}}$ models (row 4 in Table 2), but also in $\mathcal{M}^{\text{diff}}$ models. This is reported in row 5 of Table 2.

To understand our *diff*-based modelling, note that *diff* distinguishing power comes from the observation presented in our Note 4, in Section 2. I.e., in step 9 of the AKMA protocol, the status of a AKID-AF session (e.g., ongoing, successful establishment, failure for reason 1, reason 2, ...,) is leaked. Intuitively, using this, an attacker can distinguish the "status" of AKID1 vs that of AKID2. We detail below how this can be encoded via *diff*-equivalences, leveraging details on step 9 of AKMA.

For instance, consider the following setting not entirely known to the attacker: (i) UE1/AKID1 is communicating to AF1 using a secure Ua^* connection, and to AF2 using an insecure Ua^*; (ii) UE2/AKID2 can communicate to AF1 and AF2, but is for now communicating just with AF2 (not to AF1); it uses an insecure Ua^* connection.

Due to the insecure channels towards AF2, clearly what the attacker knows if the above setting is AKID1 and AKID2, and their connection to AF2.

To mount a distinguishing attack against AF1 (i.e., to discern if one is connecting to AF1), the attacker will use step 1 on AKMA and send AKID1 and AKID2 to AF1 (on whichever type of Ua^* connection). As per Note 4 in Section 2, in step 9 of AKMA, AF1 will send the attacker an error-response saying that AKID1 is already connected, and a success-response saying that AKID2 can connect. We show this attack via a *diff*-equivalence failing in Tamarin.

5.2.4.2 **Analysing unlinkability of AFs to UE-AKIDs w.r.t. AKIDs**.. The privacy attacks shown thus far (e.g., weak privacy of the AF) do not imply an attack on unlinkability of AFs to UE-AKIDs; this is because the attacks before stem from knowing/learning an AKID, whereas unlinkability of AFs to UE-AKIDs is about starting from knowing an AFID and linking it to (potentially as-of-then unknown) AKIDs.

We do this in a $\mathcal{M}^{\text{diff,Sec\_InSec-}Ua^*}$ model. The results are shown in row 6 of Table 2.

To analyse unlinkability of AFs to UE-AKIDs, as per its definition, we leak to the attacker one AFID, say AF1. Consider that some UE1 connects to AF1, over secure channels, and to some AF2, unknown to the attacker, over insecure channels. Then, we check a `diff` similar to the one described for row 5, above.

Interestingly, it turns out that the `diff` above holds if there is only one UE (e.g., UE1 as named above) present, but it fails if there are at least two UEs present (e.g., UE1 as named above, and a different UE2). This is because one UE will show up with the same AKID, and that AKID is global (see Note 1 in Section 2), i.e., the same for all AFs.

5.2.5. Analysing Weak Privacy of UE's SUPI and Unlinkability of UE-SUPIs and AFs. We also formalise and analyse Definition 5 (unlinkability of UE-SUPIs and AFs); see row 7 on Table 2. Since the SUPI/GPSI is a fixed, unique identifier for a UE, colluding AFs[15] can know which SUPI/GPSIs connect to their entire clique. So, Definition 5 clearly fails in AKMA.

We recall from Note 3 that the SUPI/GPSI are unique, long-term identifiers sent to the AFs in step 8 of AKMA. To enhance our security results, we consider primarily AFs outside the network, so we consider the GPSI is sent, but this is purely a name w.r.t. the modelling that follows.

We create a model in the class $\mathcal{M}^{\text{diff}}$: i.e., we use *diff*s for verification.

The model allows for two UEs/GPSIs a and b and two AFs i and j, and we encode that the first UE/GPSI a will contact one of the AF i.

To show that Definition 5 holds/fails, the model then tries to distinguish (via a *diff* using GPSIs sent on step 8 of the protocol) between:
– (*case1*) the first UE a contacts the second AF j;
– (*case2*) the second UE b contacts the second AF j.

This *diff* fails provided that the AFs are corrupt and leaks the GPSI they receive, at some point. Then, concretely, in *case1* the attacker will observe the same GPSI, while in the *case2* – the attacker will see two different GPSIs. Thus, *case1* and *case2* (i.e., which GPSIs contact which AFs) are distinguished via the *diff* above. This shows that Definition 5 fails in AKMA.

5.2.6. Our Verification of AKMA Beyond Privacy. All our models are also checked w.r.t. standard security requirements, such as Lowe's hierarchy [18], and the results are as expected: e.g., weak agreement holds between

15. Collusion of AFs is a possibility, not a requirement in our threat model and definitions.

parties two-by-two, except for between the UE and the AF when the two communicate on an insecure channel; most secrecy and key-agreement properties hold (e.g., on AKID, and K_{AF}, respectively), except for when, again, the UE and the AF communicate on an insecure channel. Such findings are recounted inside our models as comments.

5.3. On Our Tamarin Modelling

In this work, we developed new models in Tamarin, taking the existing ones in [25] as a starting point. In Section 7.1, Section 7.2 and in Table 4, we discuss at length why we needed to develop these models almost anew: i.e., other models in [7] were in ProVerif, the ones and all models, included the ones in [25], have a weaker threat model, with simpler assumptions on the $Ua*$ channels; this also makes each of these older model simpler and therefore more tractable. To deal with our specific privacy properties, not only did we need to strengthen the threat model, but we also needed to consider multiple classes of models based on this $Ua*$-security dichotomy (e.g., $\mathcal{M}^{\text{Sec\_Insec-}Ua^*}$, $\mathcal{M}^{\text{Sec-}Ua^*}$). Finally, even our simplest class of models cannot just reuse the models from [25], since those were for authentication, and our privacy/unlinkability properties require not only new tagging and proving around `diffs`, but also new trace-based lemmas, with different predicates and different rules all over the model, compared to [25]. To this end, as we mentioned in Section 5.2.1, our privacy/unlinkability attacks on AKMA are sometimes found via `diffs`, and other times via failure of trace properties that we carefully encode into specific lemmas encoding lack of well-authentication or non-desynchronisation[16]; so, we also make a link between our formal definitions in Section 3, weak unlinkability by Arapinis [9] and its verification via WA, ND by [17], [11] in our Tamarin models for AKMA, which was also not done before.

Finally, our AKMA^p protocol for privacy-enhanced AKMA is new, with nothing similar proposed before, and so our model for AKMA^p(see Section 6) is totally new.

6. AKMA^p: Practical & Private AKMA

We now give a practical solution to the privacy attacks we exhibited in the previous section.

6.1. Privacy Failures and Patches Put Simply

Vehicles to Our Privacy Attacks in AKMA. The main reasons to our privacy attacks are:

(a) the attack against *WP_AF* (Figure 5) has its onset via an adversarial injection of the global AKID of an UE to an AF1, when this AKID was aimed for an AF2; this global nature of the AKID is also at fault for unlinkability of UE-AKIDs to AFs failing;

(b) the final parts of the attacks against *WP_AF* (Figure 5) and *PP_UE-AKID* (Figure 4) are stemming from the fact that the success or the reason of failure in the 9th message of AKMA is sent, on the Ua^* channel, to any UE (adversarial or honest) without any checks w.r.t. who this success/failure details are intended for; i.e., the attacker manipulating an UE2 learns of the AKIDs or the AFs via information meant for UE1 but sent to this UE2, on the Ua^* channel;

(c) the main vehicle by which attacker can learn AKIDs and thus have one way to start mounting most attacks, e.g., *PP_UE-AKID* (Figure 4), unlinkability of UE-AKIDs to AFs, etc., is in the case where the Ua^* channels/protocols are insecure;

(d) the attack against unlinkability of UE-SUPIs to AFs is to the fact that an attacker may query if a given SUPI/GPSI is connected to a given AF or another.

AKMA^p: Patching Our Privacy Attacks. Below and in Figure 6, we give a small adaptation of AKMA called *AKMA^p*. As the Figure 6 shows, AKMA^p inflicts minute changes to AKMA:
– changes one KDF to yield unique AKIDs;
– adds one hash, when a SUPI connects for the first time and for the first time only, to an AF;
– makes error-handling clearer.

We describe AKMA^p below. It patches all above privacy failings found in AKMA, as follows:

(i) to counteract problem (a) above, we ask that for each UE, during each of its Registrations, an AKID unique per every AFID is (re)generated.
For this, equation (1) in AKMA (see Section 2.2.2) is modified to the equation below, as follows:

$$A\text{-}TID = KDF(const, K_{AUSF}, \text{“}ATID\text{”}, \textbf{AF\_ID}, \text{SUPI})$$

See the blue box on the left hand side of Figure 6. All the other equations (i.e., derivations of identifiers and keys in AKMA) stay the same.

(ii) to counteract problem (b) above, the success or failures in AKMA's AF-key derivation which are sent from AF to the UE on the Ua* channel, should be sent not in plain text but encrypted with a key pertaining just to the UE for whom these messages are intended.

To this end, the *core* will issue UE-specific messages after step 7 of the AKMA protocol. This is because it is only the *core* (and not the AF), who –at the stage before step 9 of the AKMA protocol– has UE-specific keys. Moreover, in our trust model (namely, (T2) and (T4)), it is only the core who is trusted and therefore can construct such a message in ways unattainable to the attacker.

Concretely, if the K_{AF} generation succeeds or fails, the *core* encrypts this result with the K_{AMF} key[17]

16. The authors in [17], [11] soundly reduced the verification of weak unlinkability by Arapinis et al. [9] to the verification of two reachability properties called *well authentication (WA)* and *no desynchronisation (ND)*, and one property (which is not a trace-property) called *frame opacity (FO)*. Simply put, ND denotes that an honest interaction between A and B cannot/should not fail. And, WA encodes that whenever a conditional is positively evaluated, the agents involved are having so far an honest interaction.

17. This is the lowest-level key in the mobile-network key hierarchy which is derived out of K_{AUSF}, and re-generated at each UE re-Registration.

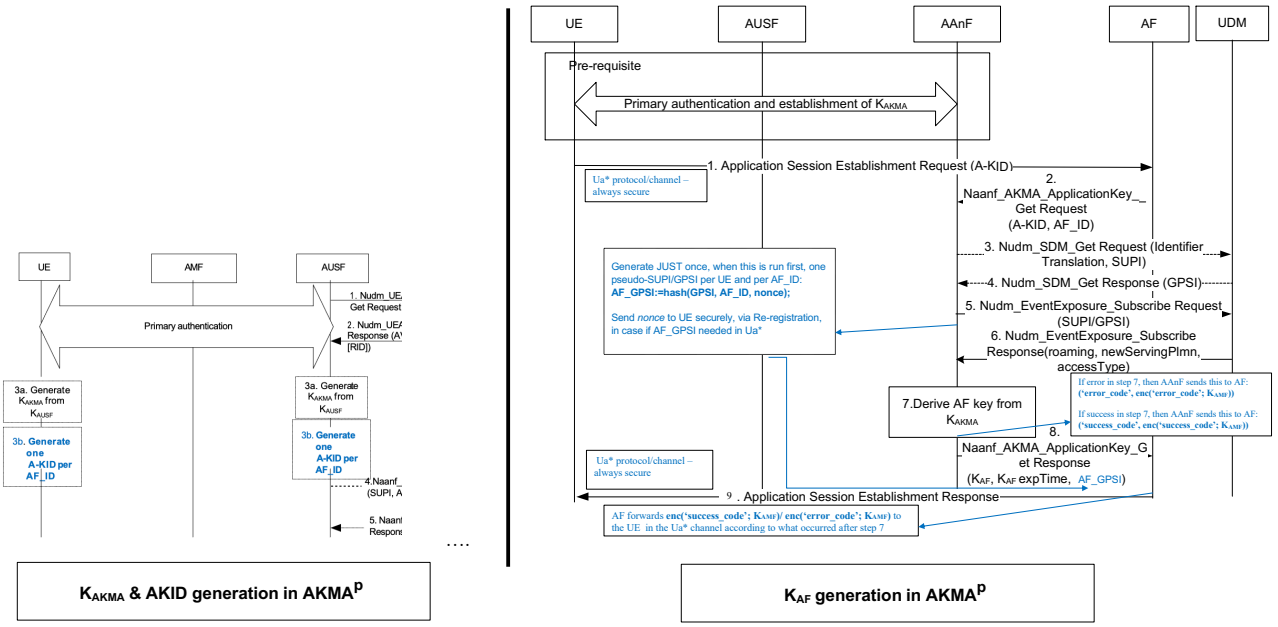

Figure 6: AKMA^p: Our Modifications to AKMA (shown in blue on Figures 6.1-1 and 6.2-1 in [6])

it shares[18] with the UE for which this *core* tried to calculate the K_{AF} key. This encryption is forwarded by the AF to the UE, in step 9, on the Ua^*-channel they share.

For a diagrammatic version of this modification, the reader can also see the blue-text boxes linked to step 7 and step 9 in Figure 6.

(iii) to counteract problem (c) above, all Ua^* channels/protocols should be secure;

(iv) to counteract problem (d) above, the core will no longer send the one SUPI/GPSI to the AFs that a UE connects to. Instead, the core will generate one "pseudo SUPI/GPSI" per AF, for each UE that ever connects to this AF. We call this AF_GPSI. We show its generation on the blue-text box linked to step 5 in Figure 6.
This AF_GPSI will be generated just once, when the UE that it is linked to connect to a given AF for the first time.

AKMA^p & Our Privacy-attacks' Counteractions. These counteractions are as follows:

• Secure Ua^* channels prevent AKID leaks, stopping the onset of attacks.
• Encrypted handling of successes/errors w.r.t. K_{AF} derivations stop malicious UEs from learning other UEs' data.
• One AKID per AFID stops the tracking of AFIDs and/or the linking of AFIDs to AKIDs.
• One pseudo-SUPI replacing each SUPI connecting to an AFID stops the linking of SUPIs to AFIDs.

18. An encryption (by the *core* or the AF) of success/failure with K_{AF} itself is not suited; it may be (e.g., in the case of failure in steps 6-8) that the right K_{AF} has not been computed, and then there will no way for the UE to decrypt the reason of failure.

Judicious error-handling is common practice when it comes to privacy preservation. One AKID per AF is also meaningful: if Twitter/X and Facebook/Meta both offered AKMA to UEs, it would not be desirable that these UEs would share the same AKID to connect over-the-air to both services.

6.2. Verifying AKMA^p's Privacy in Tamarin

We showed formally that our AKMA^p enjoys privacy guarantees w.r.t. our notions. We recount now.

Since the AKMA^p has only secure Ua^* channels, then in our Tamarin modelling we will only have $\mathcal{M}^{\text{Sec-}Ua^*}$ models and no $\mathcal{M}^{\text{Sec\_Insec-}Ua^*}$.

Since we wished to prove (not disprove) privacy for AKMA^p, we only used $\mathcal{M}^{\text{diff}}$ models, aiming that all diff-equivalences would hold (which they did).

That said, that are other clear modelling changes from the AKMA-based models, as now there is one AKID per AFID. For instance– in the $\mathcal{M}^{\text{diff}}$ models– where we restricted for AKMA to 3 AKIDs (for tractability reasons), now we will need to consider at least 4 AKIDs to be able to formulate the right `diffs` (e.g., for checking unlinkability of UE-AKIDs to AFs) . We applied such changes to our Tamarin models for AKMA, and easily produced Tamarin models for AKMA^p.

Note that, especially in AKMA^p's settings of secure communications only, it is the case that some privacy holding trivially implies other privacy-notions holding.

Concretely, if strong secrecy of UE's AKID holds ("diff-equiv" on all UE-related data holds for any two unknown AKIDs) and unlinkability of AFs to UE-AKIDs holds ("diff-equiv" on all UE-AF-related data holds even when one knows one AFID) and unlinkability of UE-AKIDs to AFs holds ("diff-equiv" on all UE-AF-related data holds even when one knows one AKID), then also

Table 3: Our Systematic Privacy Verification of AKMA^P in Tamarin (files found at [24])

Property	About	Model Class	Verification Method	Status	Ref. Model(s)	Time
strong secrecy of UE's AKID (Def. 1)	AKID	$\mathcal{M}^{\text{diff,Sec-Ua*}}$	diff-equiv on two unknown AKIDs	holds	diff/SS_AKMAP	cca. 1 min
unlinkability of UE-AKIDs to AFs (Def. 4)	AKID and AF_ID	$\mathcal{M}^{\text{Sec-Ua*}}$	no known AFID, one known AKID1, diff-equiv over AKID1-AFID	holds	indist/Unlink1_AKMAP	cca. 1 min
unlinkability of AFs to UE-AKIDs (Def. 4)	AKID and AF_ID	$\mathcal{M}^{\text{Sec-Ua*}}$	no known AKID, one known AFID1, diff-equiv over AKID-AFID1	holds	indist/Unlink2_AKMAP	1 min
post-compromise privacy of UE's AKID (Def. 2)	AKID	$\mathcal{M}^{\text{diff,Sec-Ua*}}$	diff-equiv "between a known AKID and an unknown one"	holds	from the first 3 rows	1 min
weak privacy of the AF (Def. 3)	AF_ID	$\mathcal{M}^{\text{diff,Sec-Ua*}}$	diff-equiv "between a known AFID and an unknown one"	holds	from the first 3 rows	1 min
unlinkability of UE-SUPIs and AFs (Def. 5)	SUPI/GPSI AF_ID	any/not about Ua*	diff-equiv "between 2 AKIDs contacting an AF given one known AF_GPSI"	holds	diff/Unlink_AKMAP_GPSI	23 min

post-compromise privacy of UE's AKID must hold ("diff-equiv" on all UE-related data holds even when one knows one AKID) and weak privacy of the AF("diff-equiv" on all UE-related data holds even when one knows one AFID) must hold. This is the case in AKMA^P.

Also, clearly weak privacy of UE's SUPI holds since there are no more SUPIs sent in AKMA^P.

Similarly, because of the nonce used in the generation of our AF_GPSIs, then unlinkability of UE-SUPIs to AFs also holds in AKMA^P. That is, an attacker, who is in control of a group of AFs, will only learn that a specific UE has contacted a given AF (after all the UE's AF_GPSI is a long-term identifier) but will not be able to associate this with the same UE connecting to any other AFs (since the UE's AF_GPSIs are unique for each UE/AF pair).

For good measure, in AKMA^P, we also verified all the properties we had verified for AKMA (e.g., all the via WA, ND lemmas), and they all hold for AKMA^P. For further details, please see our files.

Our verification results for AKMA^P are also summarised in Table 3.

Responsible Disclosure. We discussed with 3GPP privacy enhancements of AKMA, ours and other possibilities alongside. One aspect that 3GPP does not agree with in AKMA^P is to drop the use of SUPIs/GPSIs, in the 8th step of AKMA, as these have been added in later revision for the practical purposes of easy billing; however, we discussed the wider impact of using SUPIs/GPSIs there, and will continue to lobby for their removal. 3GPP would prefer to keep the AKIDs global, but understands the threat and we are actively looking into new ways of calculating the AKID as per AKMA^P. 3GPP would prefer to keep the $Ua*$ both secure and insecure to serve older applications based on, e.g., HTTP, however it understands the risk and a recommendation/note will likely be added in future revisions about using only/preferably $Ua*$. Finally, 3GPP agrees that the error/success codes have to be obfuscated in the last step of AKMA, as suggested by AKMA^P. Further, there is not much leeway for privacy-enhanced designs in AKMA different to these in AKMA^P, as far as 3GPP is concerned.

7. Related Work

7.1. Comparison with Privacy Analyses of AKMA

In the introduction, in paragraph "Modified Versions of AKMA", we discussed how recent privacy-enhanced versions [7] of AKMA suffer from certain shortfalls in our perspective centred on backwards compatibility. We detail here, compare and contrast further with our work, for various perspectives.

Aims. In this work, our aim is to give *formal definitions* of privacy in AKMA, from the perspective of *all identifiers* concerned, as well as their interplay (i.e., UE-AF unlinkability). Meanwhile, [7] does not aim to systematically define what privacy for AKMA means, instead it aims to hide the AKID and the SUPI from various parties; but, for instance, it is even unclear if such confidentiality leads to unlinkability. In fact, we show that unlinkability is linked to other aspects too, such as the error management in the last message of AKMA.

7.1.1 Designs. As we discussed in the introduction, our AKMA^P is constructed with backwards compatibility in mind, whilst the private AKMA presented in [7] is at odds with AKMA and 5G's infrastructure and design constraints. Without repeating the discussions in the introduction, let us give some other examples of this. Firstly, [7] changes the functionality of the AKID, which is a core identifier in AKMA. Instead of the AKID being directly consumed by the AF (which is key in AKMA, as Section 4.4.2 in its specification [6] explains), [7] makes the AKID only decryptable by the core. Worse, this also changes the flow of the AKMA protocol, as its step 1 needs new Core-to-AF exchanges. Secondly, [7] makes changes to the derivation of the K_{AF} key in such a way that the entire 3GPP-specified 5G key-hierarchy would have to change, which is virtually impossible. In our work, we avoid such essential changes to the AKMA protocol or the infrastructure, as we explained in Section 5.

7.1.2 Assumptions & Adversary Model. In our privacy analysis of AKMA, we follow the infrastructure assumptions in the 3GPP specification [6] and allow the $Ua*$ channels, between the UE and the AF, be both secure and insecure, whilst [7] assumes that they are only insecure. This means that our attacker model is stronger, and –in turn– that it shows that if a UE has a secure channel to an AF, this does not prevent it from being tracked and linked to that AF.

7.1.3 Formal Models & Verification. The formal verification by [7] and our here are orthogonal. First, the analysis in [7] was done in using ProVerif and ours is done in Tamarin. Second, as we say above, the models are under different assumptions on the attacker, infrastructure, but also w.r.t. different properties: e.g., we do UE-AF unlinkability, and tracking of the UE in AKMA's setting of co-existence of secure and insecure $Ua*$ channels, whereas [7] do strong confidentiality of AKID w.r.t. the AF. Third,

Table 4: Formal Analyses of AKMA and Variants

Work	Domain	Assumptions	Patches' Designs	Tool Used	Reusability here
[25]	security/key-agreement	insecure $Ua*$ channels	not relevant	Tamarin	yes, partial (see above)
[8]	security/key-agreement	insecure $Ua*$ channels	not relevant	Tamarin	no
[7]	strong confidentiality of AKID & SUPIs	insecure $Ua*$ channels	not backwards compatible	ProVerif	no
us	privacy of AKID, unlinkability of AKID-AFID	secure &insecure $Ua*$ channels	Tamarin	backwards compatible	not applicable

as we explained in Section 5, in our Tamarin models, we formulate some of our unlinkability properties (which do not exist in [7]), in well-known encodings of well-authentication (WA) and no-desynchronisation (ND) by [17], [11].

7.2. Comparison with Other Formal Analyses of AKMA

The verification of security properties of AKMA, not privacy-related, has also been carried out, using Dolev-Yao symbolic protocol analysers: ProVerif – by [8], and Tamarin – by [25]. Our Tamarin models are in fact inspired by those in [25], which –as we said– focused on key-agreement. As Section 5.3 showed, we modified them substantially to support privacy verification and fit our threat model and settings: i.e., (i) developed on the derivation of the AKID; (ii) added secure channels for $Ua*$; (iii) added error management on the $Ua*$ channel; (iv) added new lemmas for privacy and unlinkability in the way of well-authentication (WA) and no-desynchronisation (ND) by [17], [11] as well as with `diff`s. Just (ii) and (iii) imply much more complex models, which in turn imply much more complex proofs and the need for new and ingenious Tamarin oracle.

Take-away Message. In the Table 4, we summarise the comparisons above in Section 7.2. and 7.3, between us and other formal analyses of AKMA and patches of it.

7.3. Formal Privacy Definitions in General

W.r.t. definitions of privacy in general, Pfitzmann and Hansen offer a consolidated report of privacy-related terminology in [20]. Their report combines a number of concepts of anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management. Tsukada et al. investigate and organise some of these concepts in [23], particularly relating the concept of unlinkability to that of minimal anonymity. Goriac [16] expands on the work by Pfitzmann and Hansen by adding definitions for *involvement* and *unobservability*, and investigates privacy in terms of *behavioural equivalence*. A recent work is [13], which looks at the notion of "trackability" from various perspectives, all rooted in some practical modification. Their notion of existential trackability is closest to what is commonly known as unlinkability, and –in fact– our own notions of unlinkability are inspired by their work.

8. Conclusions

We generically defined numerous facets of privacy in the recent 5G procedure of delegated authentication called AKMA. We found privacy breaches in AKMA. Our patches minimally changie the 3GPP specifications for AKMA, and attain our privacy guarantees.

We are in ongoing talks with 3GPP about this.

Whilst our findings are supported by formal methods (i.e., state-of-art privacy analysis, in the Tamarin prover), there is generic value in the privacy definitions we put forward, and their failing on AKMA and their patches can be easily understood by non-formal-methods specialists.

References

[1] 3GPP. System architecture for the 5G System (5GS). Technical Specification (TS) 23.501, 3rd Generation Partnership Project (3GPP), 10 2020. Version 16.0.0.

[2] 3GPP. Procedures for the 5G System. Technical Specification (TS) 23.502, 3rd Generation Partnership Project (3GPP), 10 2021. Version 16.7.0.

[3] 3GPP. 5G Security Assurance Specification (SCAS); Access and Mobility management Function (AMF). Technical Specification (TS) 33.512, 3GPP, 07 2022. Version 16.3.0.

[4] 3GPP. 5G Security Assurance Specification (SCAS) for the Session Management Function (SMF) network product class. Technical Specification (TS) 33.515, 3GPP, 07 2022. Version 16.2.0.

[5] 3GPP. Digital cellular telecommunications system (Phase 2+), ... Technical Specification (TS) 33.220, 3GPP, 01 2023. Version 17.4.0.

[6] 3GPP. Authentication and Key Management for Applications (AKMA) based on 3GPP credentials in the 5G System. Technical Specification (TS) 33.535, 3GPP, 2024. Version 18.3.0.

[7] Gizem Akman, Philip Ginzboorg, Mohamed Taoufiq Damir, and Valtteri Niemi. Privacy-enhanced AKMA for multi-access edge computing mobility. *Computers*, 12(1), January 2023.

[8] Gizem Akman, Philip Ginzboorg, and Valtteri Niemi. AKMA for secure multi-access edge computing mobility in 5g. In Osvaldo Gervasi, Beniamino Murgante, Sanjay Misra, Ana Maria A. C. Rocha, and Chiara Garau, editors, *Computational Science and Its Applications - ICCSA 2022 Workshops - Malaga, Spain, July 4-7, 2022, Proceedings, Part IV*, volume 13380 of *Lecture Notes in Computer Science*, pages 432–449. Springer, 2022.

[9] Myrto Arapinis, Tom Chothia, Eike Ritter, and Mark Ryan. Analysing unlinkability and anonymity using the applied pi calculus. In *2010 23rd IEEE Computer Security Foundations Symposium*, pages 107–121, Edinburgh, UK, 2010. IEEE, IEEE Computer Society.

[10] A. Armando, D. Basin, Y. Boichut, Y. Chevalier, L. Compagna, J. Cuellar, P. Hankes Drielsma, P. C. Heám, O. Kouchnarenko, J. Mantovani, S. Mödersheim, D. von Oheimb, M. Rusinowitch, J. Santiago, M. Turuani, L. Viganò, and L. Vigneron. The AVISPA tool for the automated validation of internet security protocols and applications. In Kousha Etessami and Sriram K. Rajamani, editors, *Computer Aided Verification*, pages 281–285, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

[11] David Baelde, Stéphanie Delaune, and Solene Moreau. A method for proving unlinkability of stateful protocols. In *2020 IEEE 33rd Computer Security Foundations Symposium (CSF)*, pages 169–183, Los Alamitos, CA, USA, June 2020. IEEE Computer Society.

[12] B. Blanchet. An Efficient Cryptographic Protocol Verifier Based on Prolog Rules. In *IEEE CSFW*, 2001.

[13] K. Budykho, I. Boureanu, S. Wesemeyer, F. Rajaona, D. Romero, Lewis, Y. Rahulan, and S. Schneider. Fine-Grained Trackability in Protocol Executions. In *Network and Distributed System Security Symposium (NDSS) 2023*, 2023.

[14] David W. Chadwick. *Federated Identity Management*, pages 96–120. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[15] D. Dolev and A. Yao. On the Security of Public-Key Protocols. *IEEE Trans. Inf. Theory 29*, 29(2), 1983.

[16] Iulian Goriac. An epistemic logic based framework for reasoning about information hiding. In *2011 Sixth International Conference on Availability, Reliability and Security*, pages 286–293, Vienna, Austria, 2011. IEEE.

[17] Lucca Hirschi, David Baelde, and Stéphanie Delaune. A method for unbounded verification of privacy-type properties. *Journal of Computer Security*, 27(3):277–342, 2019.

[18] G. Lowe. A Hierarchy of Authentication Specifications. In S. Foley and J. Millen, editors, *In Proceedings of the 10th IEEE workshop on Computer Security Foundations (CSFW'97)*, pages 31–43, Massachusetts, US, 1997. IEEE Computer Society.

[19] S. Meier, B. Schmidt, C. Cremers, and D. Basin. The TAMARIN Prover for the Symbolic Analysis of Security Protocols. In *CAV*, pages 696–701, 2013.

[20] Andreas Pfitzmann and Marit Hansen. Anonymity, unlinkability, undetectability, unobservability, pseudonymity, and identity management - A consolidated proposal for terminology. *Version v0*, 31:15, 2008.

[21] Sonia Santiago, Santiago Escobar, Catherine Meadows, and José Meseguer. A formal definition of protocol indistinguishability and its verification using Maude-NPA. In *Int. Workshop on Security and Trust Management*, pages 162–177. Springer, 2014.

[22] Victor Shoup. Sequences of games: a tool for taming complexity in security proofs. Cryptology ePrint Archive, Report 2004/332, November 2004. Available at http://eprint.iacr.org/2004/332.

[23] Yasuyuki Tsukada, Ken Mano, Hideki Sakurada, and Yoshinobu Kawabe. Anonymity, privacy, onymity, and identity: A modal logic approach. In *2009 International Conference on Computational Science and Engineering*, volume 3, pages 42–51, Bellaterra, Catalonia, Spain, January 2009.

[24] Stephan Wesemeyer and Ioana Boureanu. Tamarin Files for AKMA and AKMAP. https://github.com/UoS-SCCS/AKMA-Models-Tamarin, 2024.

[25] 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.

Appendix A. Tamarin Code

```
lemma No_Desynchronisation_UE_AF [
   use_induction]:
"All AKID idAF idHN KAF SUPI count1 #
   t01 #t02 #t03 #t04
        .
        (
                AF_WA_ND(AKID, idAF, '
                   ok', '1'+'1') @ #
                   t04 //the AF was
                   contacted twice by
                   the AKID
                &
                HN_Response(idHN, idAF,
                    AKID, KAF,'ok') @
                   #t03
                &
                AF_send_KeyRequest(idAF
                   ,idHN, AKID) @ #t02
                &
                UE_WA_ND(SUPI, AKID,
                   idAF, 'secure',
                   count1) @ #t01  //
                   the channel between
                    UE and AF is
                   secure
                & #t01 < #t02
                & #t02 < #t03
                & #t03 < #t04
                & // no key reveal
                not (
                Ex X m #r.
                Reveal(X, m) @ #r
                &
                Honest(X) @ #t04 // the
                    AF has all the
                   parties that should
                    be honest and not
                   leak.
                )
        )
        ==>
        (
                count1='1'+'1' // the
                   UE must have sent
                   its AKID at least
                   twice to the AF as
                   well - which is
                   clearly impossible
                   - so this should
                   fail
        )
"
```

Figure 7: Tamarin lemma — No Desynchronisation between the UEs and the AFs in AKMA

```
lemma Well_Authentication:
"All AKID idAF #t04
        .
        (
                //AF_WA_ND(AKID, idAF,
                   'ok', '1') @ #t04
                AF_WA_ND(AKID, idAF, '
                   ok') @ #t04
        )
        ==>
        (
                Ex idHN KAF SUPI UaType
                    #t01 #t02 #t03
                .
                        HN_Response(
                           idHN, idAF,
                            AKID, KAF,
                           'ok') @ #
                           t03
                        &
                        AF_send_KeyRequest
                           (idAF,idHN,
                            AKID) @ #
                           t02
                        &
                        UE_WA_ND(SUPI,
                           AKID, idAF,
                            UaType,
                           '1') @ #t01
                        & #t01 < #t02
                        & #t02 < #t03
                        & #t03 < #t04
        )
        | //key reveal
        (
                Ex X m #r.
                Reveal(X, m) @ #r
                &
                Honest(X) @ #t04 // the
                    AF has all the
                   parties that should
                    be honest and not
                   leak.
        )
"
```

Figure 8: Tamarin lemma — Well-Authentication between the *core* and the UEs in AKMA

```
lemma No_Desynchronisation_UE_Core:
"All AKID idAF idHN SUPI count1 #t01 #
   t02 #t03
        .
        (
                Core_WA_ND(AKID, idAF,
                   'ok', '0'+'1') @ #
                   t03 //the core was
                   contacted for this
                   AKID by the AF
                   twice
                &
                AF_send_KeyRequest(idAF
                   , idHN, AKID) @ #
                   t02
                &
                UE_WA_ND(SUPI, AKID,
                   idAF, 'insecure',
                   count1) @ #t01 //
                   the channel between
                    UE and AF is
                   insecure and the UE
                    only sent count1
                   requests
                & #t01 < #t02
                & #t02 < #t03
                & // no key reveal
                not (
                Ex X m #r.
                Reveal(X, m) @ #r
                &
                Honest(X) @ #t03 // the
                    core has all the
                   parties that should
                    be honest here
                )
        )
        ==>
        (
                count1='0'+'1' //This
                   is clearly
                   impossible as the
                   UE only ever sends
                   1 request
        )
"
```

Figure 9: Tamarin lemma — No Desynchronisation between the UEs and the *core* in AKMA