Anonymity Guarantees of the UMTS/LTE Authentication and Connection Protocol

Ming-Feng Lee, Nigel P. Smart, Bogdan Warinschi, and Gaven J. Watson

University of Bristol

Abstract. The UMTS/LTE protocol for mobile phone networks has been designed to offer a limited form of anonymity for mobile phone users. In this paper we quantify precisely what this limited form of anonymity actually provides via a formal security model. The model considers an execution where the home and roaming network providers are considered as one entity. We consider two forms of anonymity, one where the mobile stations under attack are statically selected before the execution, and a second where the adversary selects these stations adaptively. We prove that the UMTS/LTE protocol meets both of these security definitions. Our analysis requires new assumptions on the underlying keyed functions for UMTS, namely that a set of pseudorandom functions are "agile". This assumption, whilst probably true, has not previously been brought to the fore.

1 Introduction

The Global System for Mobile Communications (GSM) developed by the European Telecommunications Standards Institute (ETSI) was the first cellular communication system designed to provide user authentication and data confidentiality. The evolution from GSM to the "3G" Universal Mobile Telecommunications System (UMTS), developed by the Third Generation Partnership Project (3GPP), gave the opportunity to fix some of the issues identied in the GSM security protocol. In the last few years a further update called UMTS Long Term Evolution (LTE) [5] has also been introduced by 3GPP. LTE is also referred as EUTRA (Evolved UMTS Terrestrial Radio Access) or E-UTRAN (Evolved UMTS Terrestrial Radio Access Network), but is commonly called "4G".

The 2G, 3G and 4G systems divide the participants of the protocol into the following entities; the mobile phone (called the Mobile Station in the standards), the home network of the user and the roaming, or serving, network. The serving network is often represented as the base station to which the phone is currently talking. This distinction between different network operators is to enable a phone user to "roam", and thus use their phone in different countries without needing to continually route traffic back to the home network. The basic UMTS/LTE authentication and key agreement protocol (known as AKA) retains the framework of the GSM AKA, but provides enhanced security properties. In particular the AKA protocol aims to provide entity authentication, data confidentiality and data integrity.

An additional security goal was to provide a limited form of anonymity for the user. Each user is identified by a permanent identity, called an IMSI (International Mobile Subscriber Identity). The protocol aims to minimize the use of the IMSI and instead replace the IMSI with a temporary identity, called a TMSI (Temporary Mobile Subscriber Identity). The security goal related to anonymity is to try to stop the IMSI and TMSIs used by a given mobile phone from being linked. If they could be linked then a user could be tracked through the network.

This paper aims to clarify what security properties the 3G/4G protocols provide in terms of user anonymity. In doing so we provide a formal security model, and prove that the protocol suite satisfies this model. Along the way we provide a precise description of the security properties required of the underlying keyed functions used in the UMTS/LTE protocol, which may be of independent interest.

Prior Security Analysis: A lot of prior analysis of the security of GSM/UMTS/LTE has gone into the properties of the underlying cryptographic functions, [8, 18, 20]. This work is orthogonal to the issues we are interested in. Our focus is on the protocols in which these functions are used, and to derive the required security properties which the functions need to provide so as to guarantee the protocol security properties.

Both Mitchell [22] and Pagliusi [23] have written surveys highlighting a number of folklore attacks against the GSM protocols. One such attack is the false base station attack and redirection attack. If an adversary owns a device which has the functionality of a base station, the adversary can impersonate a genuine base station and then map the victim mobile phone on the false base station. As a consequence, the adversary can redirect the outgoing traffic of the victim phone from one network to another. Zhang and Fang [24] pointed out that UMTS AKA is also vulnerable to a redirection attack, a variant of a false base station attack. Furthermore, they described an active attack by a corrupted network in which the adversary can mount a false base station attack to impersonate another uncorrupted network.

During the change over from 2G to 3G networks there were a number of possible attacks on the protocol. For example Meyer and Wetzel [21] present a roll-back attack, exploiting the situation when a UMTS subscriber roams to a GSM network which again exposes the subscriber to a false base station attack. Meyer and Wetzel extend the attack, further exploiting the lack of mutual authentication and integrity protection in GSM, to enable an active adversary to impersonate a legitimate GSM base station and hence forge a cipher mode command message. This allows the adversary to cheat a victim mobile phone into using either no encryption or a weak encryption algorithm, such as A5/2, in GSM.

We now turn to prior analysis of methods to circumvent the anonymity guarantees. An obvious trivial way in which anonymity can be revoked is by monitoring a network, whilst at the same time flooding a single phone with text messages. This will reveal the TMSI of the phone being flooded. This attack is assumed to be outside our model, we do not allow the adversary to send messages to a phone identified by its "phone number".

Arapinis et al. [7] described two vulnerabilities related to anonymity. In the first attack, called an IMSI paging attack, the adversary attacks the paging procedure used to locate the phone. If the temporary identity TMSI of the phone is not known by the serving network, the permanent identity IMSI is used to identify the phone. By injecting a paging request multiple times and observing the multiple replies, an active adversary can correlate the paged IMSI and related TMSI of a victim mobile phone in the area covered by the adversary's device (false base station). Arapinis et al. also provide an analysis, via formal methods, of a modified version of the UMTS protocol and show this meets an notion of anonymity.

In the second attack in [7], called a AKA protocol linkability attack, an active adversary which has previously intercepted an authentication request message can replay the message and check the presence of a specific phone in a particular area. Because the victims mobile phone will return a synchronization failure message after receipt of the replayed authentication request message, the adversary can trace the movements of the victim mobile phone.

Finally, the IMSI catcher attack [19] makes use of the fact that the IMSI of a mobile phone is sent in cleartext when the phone is registering for the first time in the serving network. This kind of attack can lead a mobile phone to reveal its IMSI by triggering the identification procedure from a false base station to the victim mobile phone. Such an attack was well known by the mobile industry and was previously described by Mitchell [22].

Contributions: As already remarked, anonymity in the UMTS/LTE protocol suite succumbs to a number of attacks, with most attacks relying on the use of a corrupted base station. However, whether it is secure against adversaries which do not corrupt any of the network participants is still worth investigating. To our knowledge no security analysis (in the sense of a security proof in the computational model) has been conducted for the anonymity requirement against adversaries who may intercept, transmit and replay messages between phones and the network, but who are not able to impersonate either the roaming or home networks. This attack model captures more realistically what a real attacker can actually do. In addition, most of the previous studies only concentrate on the UMTS/LTE AKA; they fail to consider the security of the whole authentication and connection establishment protocol. The security of data transmission and TMSI allocation (which allocates temporary identities to mobile phones for anonymity) followed by connection establishment are never considered.

In this paper, we focus on anonymity property of the UMTS/LTE at the "protocol level" against such adversaries. To formally analyze the protocol, we first give a modified two-party protocol which captures the security properties that the UMTS/LTE authentication and connection protocol provides. Since we focus on the security on the radio access link, we assume the links between the serving network and home environment are adequately secure. We therefore consider the serving network and home environment as a single party, which we call "the" network. Since a home network can always trace a user (as bills need to be paid) we can restrict to networks which are honest. This assumption eliminates any attack which requires an adversary to successfully impersonate a base station or roaming network. We feel this strong assumption is justified as an adversary that controls any part of the network could trivially break the confidentiality of a phone conversation. Without this a much stronger security property would be needed compared to the mild form of anonymity envisaged by the designers of the protocol. We also make no distinction between the mobile phone and the SIM in the phone. This is because we are interested in anonymity of the user (who is holding the phone) as they interact with the network.

Intuitively, anonymity means that a user can identify herself, communicate or use some service without leaking her identity. Up to now, anonymity has been formally defined for various cryptographic *schemes* in the literature, for example the definition of anonymity for group signatures [10, 12], ring signature [13], ad hoc anonymous identification [17], and direct anonymous communication (DAA) [14, 15]. We extend these definitions to a complex cryptographic *protocol.*

The anonymity notion we provide protects not only a user's identity but also the linking of protocol transactions. The two party protocol we consider includes the AKA and connection establishment phases, along with the phases related to TMSI allocation and data transmission after the authentication and connection establishment. We then propose a security model which captures the anonymity property provided by our two party variant of UMTS/LTE.

Typically, one adopts an indistinguishability based formalization to define anonymity. In such a model, the adversary selects two identities $(\mathsf{id}_{i_0}, \mathsf{id}_{i_1})$ to be challenged, then the adversary queries a challenge oracle with a hidden bit $b \in \{0, 1\}$ just once and is returned a signature or public transcript with respect to id_{i_b}. Generally, the target signature or transcript is produced by using the key of the user with id_{i_b}. The goal of the adversary in such an anonymity model is to try to determine the hidden bit b. To be deemed secure it is required that the adversary has negligible advantage over one-half in distinguishing the two identities from the given signature or transcript.

We also adopt the indistinguishability based formalization for the UMTS/LTE protocol but with two slight modifications. In the UMTS/LTE protocol, the mobile phone will be allocated a new TMSI after a TMSI *Allocation* procedure and then uses the new TMSI to identify itself when interacting with the network. For privacy, an adversary should not be able to link two transcripts from the same user, where one transcript is generated before TMSI *Allocation* and the other after. To model this kind of interaction, instead of a challenge oracle, our model has a challenge phase in which the adversary is given two freshly allocated

TMSIs (TMSI_{i_b} and $\mathsf{TMSI}_{i_{1-b}}$) in random order at the beginning of this phase and can then perform queries to some oracles multiple times with TMSI_{i_b} or $\mathsf{TMSI}_{i_{1-b}}$.

In addition our security model bears a close relationship to those used for key agreement, e.g. the BR-style models [9, 11, 16]. We can think of the TMSI as analogous to a secret key and the adversary is trying to determine to which session a secret key belongs. In particular our model has an analogue of the Reveal queries used in key agreement security to enable the adversary to determine TMSIs of sessions on which he is not being challenged.

We further refine the anonymity definition with two subcases. One subcase is the static case, in which the adversary is given two fixed phone identities and then tries to distinguish them by observing message transmissions. The other is for the dynamic case in which the adversary can dynamically choose two identities of phones on which to be challenged. Our first result shows that if the underlying primitives are secure, then the protocol indeed meets our anonymity requirement for the static case. Our second result shows that if the protocol is anonymous for the static case, then it is also anonymous for the dynamic case.

To end this introduction we review what we meant above by the underlying primitives being secure. The UMTS/LTE protocol makes use of a variety of keyed cryptographic functions, commonly referred to as $\{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}, \mathsf{f8}, \mathsf{f9}\}$. These functions are used to generate keys, authenticate messages and provide confidentiality. Informally it would appear that one needs to model the functions as Pseudo Random Functions (PRFs). However, the function subset $\{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$, whilst distinct, all take the same key as input. Thus our requirement is that this set is "PRF Agile", where we use agile in the sense of Acar et al. [6].

2 The UMTS/LTE Protocol Stack

Overview of Protocol: The UMTS/LTE protocol stack contains two main security protocols aimed at authentication and connection establishment. The overall goal is to establish a secure channel between the phone (a.k.a. the mobile station (MS)) and "the network". The network is a combination of parties consisting of a visitor location register/serving GPRS Support Node (VLR/SGSN) and a serving radio network controller (SRNC), where the SRNC is the base station controller of the serving network; however, for our purposes we will, as described in the introduction, consider the whole network as a single entity called "the network".

The UMTS/LTE authentication and connection establishment protocol's contribution is threefold:

1. Authenticate parties.
2. Establish common integrity and cipher keys, IK and CK respectively.
3. Establish temporary identities TMSI.

Initially the phone and the network do not share common integrity and cipher keys. Additionally no TMSI has been assigned and as a result the phone needs to identify itself by means of its permanent identity IMSI. The authentication and key agreement protocol AKA is run to established a shared integrity key IK and cipher key CK between each phone and the network. After a connection is established, the phone and the network can perform secure *Data Transmission* or TMSI *Allocation* to allocate a *new* temporary identity TMSI (the TMSI is encrypted by means of CK) to the phone from the network. Note that the allocation of a TMSI means that the phone can identify itself by this temporary identity so as to achieve anonymity.

We now describe the protocol, as summarized in Figure 1, in more detail. Assume a phone wants to establish a secure connection with the network, the protocol would proceed as follows. First a message

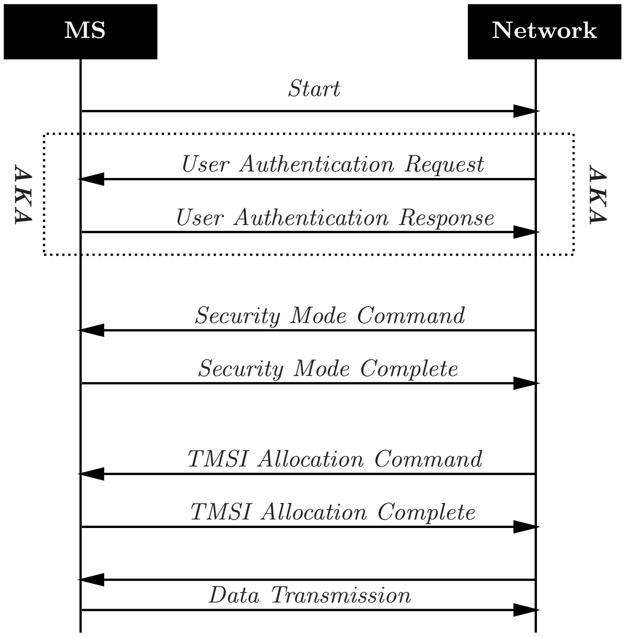

Fig. 1. Overview of the UMTS/LTS Protocol

consisting of various parameters and a START value is sent to the network. The parameters will include the precise definitions of the integrity and encryption algorithms supported by the phone. The START value acts like a counter. When a new authentication and key agreement (AKA) execution occurs, the START value is re-initialized to zero. The network (actually the SRNC) then stores the START values and the list of supported algorithms.

The parties now run the AKA protocol to authenticate each other and agree upon the integrity IK and cipher keys CK. The network chooses the highest preference integrity and encryption algorithms from the list of allowed algorithms which match the algorithms supported by the phone. The network then initiates integrity and ciphering. We provide precise details of the integrity and encryption algorithms supported and a description of the AKA later in this section.

Next the network sends to the phone: a random number FRESH, a *Security Mode Command* message m_S and the corresponding message authentication code MAC-I. The m_S message includes the security capability of the mobile equipment, the GSM ciphering capability, the selected encryption algorithm and the selected integrity algorithm. The message authentication code MAC-I is generated using the integrity key IK and the selected integrity algorithm.

On receiving this command the phone verifies the validity of the received message by checking MAC-I. If the verification passes, the phone generates a *Security Mode Complete* message and a new message authentication code MAC-I for this message. The phone sends the *Security Mode Complete* message with the MAC to the network. If verification is not successful, the phone ends the procedure.

On receipt of the *Security Mode Complete* message, the network verifies the validity of the received message authentication code MAC-I by using integrity key IK and the indicated integrity algorithm.

The value START sent by the phone is used to generate the counters COUNTER-I and COUNTER-C which are used in the integrity and ciphering algorithms. When the counters COUNTER-I and COUNTER-C are generated, the value START is also updated accordingly. For more details about the generation and updating of START, COUNTER-I and COUNTER-C, please refer to [5].

The connection is now established and both parties have been authenticated. All messages are now sent encrypted and authenticated under the keys CK and IK using the agreed algorithms. The protocol proceeds by allocating new TMSIs via the *TMSI Allocation Command* and the *TMSI Allocation Command* messages.

Ciphering and Integrity Methods: The ciphering and integrity methods are denoted by two functions, f8 and f9 respectively. The use of the block cipher Kasumi (under a particular mode of operation) for f8 and f9 is specified in ETSI TS 35.201 [1] and ETSI TS 35.202 [2], whilst the use of the stream cipher SNOW 3G is specified in ETSI TS 35.215 [3] and ETSI TS 35.216 [4]. For further details see [5].

To encrypt a message, the phone or the network computes a keystream $\mathsf{KEYSTREAM} = \mathsf{f8}_{\mathsf{CK}}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, \mathsf{DIRECTION}, \mathsf{LENGTH})$, where CK is the cipher key, COUNTER-C is a time-depended counter, BEARER is the radio bearer identifier (this is a 5 bit value with no direct effect on our analysis), DIRECTION is a transmission direction bit, LENGTH is a 16 bit field that denotes the length of the keystream block. Note that the LENGTH field only determines the output length of f8, it is not a contributor to the randomness produced; i.e. two calls to f8 with the same arguments but a different value of LENGTH will produce two streams, one of which is the prefix of the other. After the keystream is computed, the ciphertext is calculated as $\mathsf{CIPHERTEXT} = \mathsf{KEYSTREAM} \oplus \mathsf{PLAINTEXT}$. To decrypt a ciphertext, the phone or the network first computes a keystream and then derives the plaintext $\mathsf{PLAINTEXT} = \mathsf{KEYSTREAM} \oplus \mathsf{CIPHERTEXT}$. Here $\oplus$ denotes the XOR operation.

To achieve integrity, a message authentication code is attached with the message to be integrity protected (using the encrypt-then-MAC paradigm when a ciphertext is to be sent). The message authentication code of some message m is computed as:

$$\mathsf{MAC\text{-}I} = \mathsf{f9}_{\mathsf{IK}}(\mathsf{COUNTER\text{-}I}, m, \mathsf{DIRECTION}, \mathsf{FRESH}),$$

IK is the integrity key, COUNTER-I is an integrity sequence number, DIRECTION is a direction bit, FRESH is a random value.

In both cases the direction bit DIRECTION is set to zero for messages sent from the phone to the network, and set to one for the other direction.

The UMTS/LTE AKA Protocol: We now describe in detail the AKA protocol and the parameters used. Figure 2 provides an overview of the AKA protocol. Each phone SIM card and the authentication center of the network (specifically the user's home environment) share a long-term secret key K. Two counters, MS.SQN and NET.SQN are also maintained by the phone and the network respectively, to support network authentication. The sequence number NET.SQN is a counter maintained separately for each user and the counter MS.SQN is the highest sequence number the phone has accepted. The initial values for MS.SQN and NET.SQN are set to zero, with the two counters incrementing during each authentication. Intuitively the two sequence numbers MS.SQN and NET.SQN are used to guarantee the freshness of the AKA protocol.

The AKA protocols makes use of a set of three message authentication functions $\{\mathsf{f1}, \mathsf{f1}^*, \mathsf{f2}\}$, and four key generation functions $\{\mathsf{f3}, \mathsf{f4}, \mathsf{f5}, \mathsf{f5}^*\}$, all of which are controlled by the same key. In what follows we will not concern ourselves with $\mathsf{f1}^*$ and $\mathsf{f5}^*$, as they are simply variants of f1 and f5 (used in the case of resynchronization); hence we will assume them identical to f1 and f5 in our analysis.

The protocols consist of two subprocedures: The first is for the distribution of authentication data from the user's home environment to the serving network and the second is for authentication and key agreement.

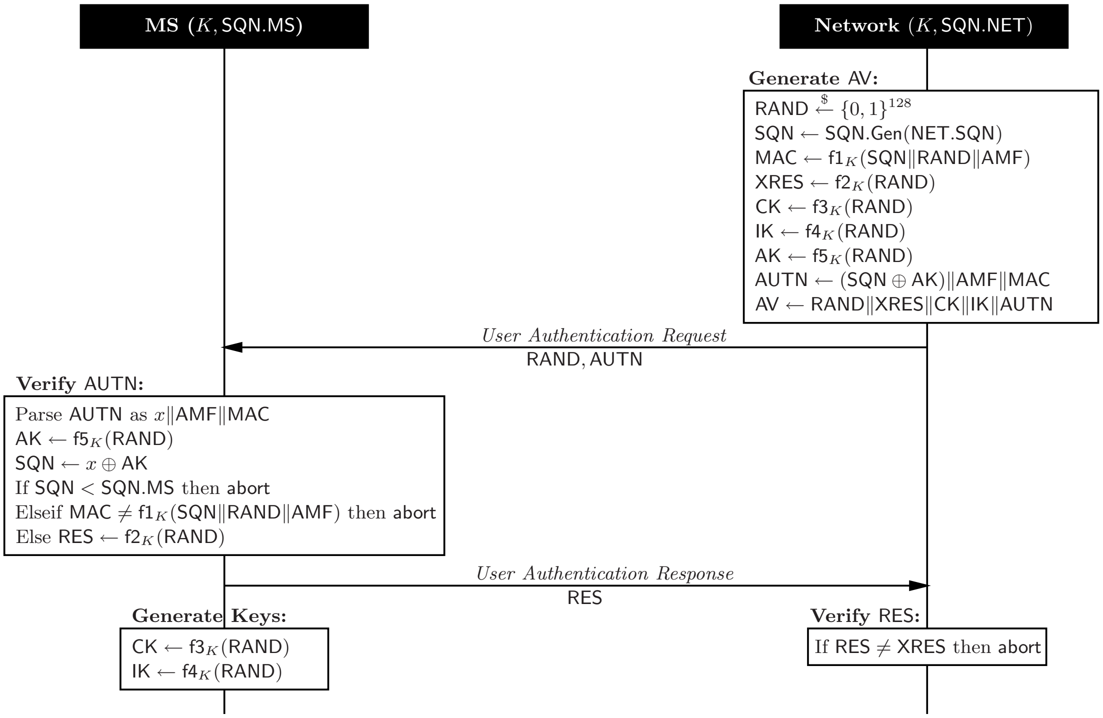
MS (K, SQN.MS)
Network (K, SQN.NET)
Generate AV:
RAND $\xleftarrow{\$} \{0,1\}^{128}$
SQN ← SQN.Gen(NET.SQN)
MAC ← $\mathsf{f1}_K$(SQN‖RAND‖AMF)
XRES ← $\mathsf{f2}_K$(RAND)
CK ← $\mathsf{f3}_K$(RAND)
IK ← $\mathsf{f4}_K$(RAND)
AK ← $\mathsf{f5}_K$(RAND)
AUTN ← (SQN ⊕ AK)‖AMF‖MAC
AV ← RAND‖XRES‖CK‖IK‖AUTN
User Authentication Request
RAND, AUTN
Verify AUTN:
Parse AUTN as x‖AMF‖MAC
AK ← $\mathsf{f5}_K$(RAND)
SQN ← x ⊕ AK
If SQN < SQN.MS then abort
Elseif MAC ≠ $\mathsf{f1}_K$(SQN‖RAND‖AMF) then abort
Else RES ← $\mathsf{f2}_K$(RAND)
User Authentication Response
RES
Generate Keys:
CK ← $\mathsf{f3}_K$(RAND)
IK ← $\mathsf{f4}_K$(RAND)
Verify RES:
If RES ≠ XRES then abort

Fig. 2. Authentication and Key Agreement (AKA) Protocol

The distribution of the data from the home to serving network is not of interest to us, since we subsume the home and serving network into one entity in our security model. However, the output of this procedure is vital to the understanding of what follows. The serving network obtains an ordered array of fresh authentication vectors $\mathsf{AV}(1 \dots n)$. The reason for the serving network obtaining an array of such vectors is to enable it to perform multiple authentications with the phone, without needing to recontact the home network. In our simplification with a single network provider we can assume that fresh individual authentication vectors are obtained for each invocation as opposed to an array of authentication vectors.

The authentication vectors AV are produced as follows: The network starts by generating an unpredictable random number RAND and a fresh sequence number SQN which is derived from NET.SQN. Typically, the sequence number SQN consists of two concatenated parts $\mathsf{SQN} = \mathsf{SEQ} \| \mathsf{IND}$, however the precise definition will not concern us (we again refer the interested reader to [5] for details). In our analysis we abstract the construction away into an algorithm SQN.Gen which takes as input NET.SQN and outputs a value SQN. The network computes a message authentication code MAC, an expected response XRES, a cipher key CK, and integrity key IK as follows.

- $\mathsf{MAC} = \mathsf{f1}_\mathsf{K}(\mathsf{SQN} \| \mathsf{RAND} \| \mathsf{AMF})$. The AMF field defines operator-specific options in the authentication process, e.g. the lifetime of integrity and cipher keys.
- $\mathsf{XRES} = \mathsf{f2}_\mathsf{K}(\mathsf{RAND})$.
- $\mathsf{CK} = \mathsf{f3}_\mathsf{K}(\mathsf{RAND})$.
- $\mathsf{IK} = \mathsf{f4}_\mathsf{K}(\mathsf{RAND})$.
- $\mathsf{AK} = \mathsf{f5}_\mathsf{K}(\mathsf{RAND})$ or $\mathsf{AK} = 0$. (the anonymity key AK is used to conceal the sequence number as the latter may expose the identity and location of the phone, if no concealment of the sequence number is needed then $\mathsf{AK} = 0$).

The network creates an authentication token $\mathsf{AUTN} = (\mathsf{SQN} \oplus \mathsf{AK}) \| \mathsf{AMF} \| \mathsf{MAC}$ and an authentication vector $\mathsf{AV} = \mathsf{RAND} \| \mathsf{XRES} \| \mathsf{CK} \| \mathsf{IK} \| \mathsf{AUTN}$. The reason for XORing SQN with AK is because the production of SQN via $\mathsf{SQN.Gen}$ produces linked values. As a result, without this masking different runs of the protocol could be linked.

The second subprocedure completes the authentication before generating the ciphering and integrity keys. The serving network first selects a fresh authentication vector AV. It then sends the random challenge RAND and the authentication token AUTN from the selected authentication vector AV to the phone. Upon receipt of RAND and AUTN, the phone computes the anonymity key $\mathsf{AK} = \mathsf{f5}_{\mathsf{K}}(\mathsf{RAND})$ and retrieves the sequence number $\mathsf{SQN} = (\mathsf{SQN} \oplus \mathsf{AK}) \oplus \mathsf{AK}$ from the authentication token AUTN. Following this the phone computes $\mathsf{XMAC} = \mathsf{f1}_{\mathsf{K}}(\mathsf{SQN} \| \mathsf{RAND} \| \mathsf{AMF})$ and compares this with MAC which was included in the received AUTN. If they are different, the user ends the procedure. The phone also verifies whether the received sequence number SQN is in the correct range, for example, $\mathsf{SQN} > \mathsf{MS.SQN}$. If the sequence number SQN is not in the correct range, the phone will abandon the procedure. If the sequence number is considered to be in the correct range, the phone computes a response $\mathsf{RES} = \mathsf{f2}_{\mathsf{K}}(\mathsf{RAND})$ and includes it in a *User Authentication Response* message returned to the network. Finally, the phone computes a cipher key $\mathsf{CK} = \mathsf{f3}_{\mathsf{K}}(\mathsf{RAND})$ and an integrity key $\mathsf{IK} = \mathsf{f4}_{\mathsf{K}}(\mathsf{RAND})$.

Upon receipt of RES, the serving network compares it with the expected response XRES given by the authentication vector AV. If RES is the same as XRES, the phone passes the authentication. The serving network then extracts the cipher key CK and integrity key IK from the selected authentication vector. If RES and XRES are different, the network abandons the authentication procedure.

3 Security model

In this section we present our security model. We first introduce the basic notation and then go onto describe the various oracles which model how the adversary can interact with the UMTS/LTE protocol. Finally, we discuss how these oracles are used to define our security experiments in the two cases of static and dynamic adversaries.

Basic Notation: If S is a set, we denote the act of sampling uniformly at random from S and assigning the result to the variable x by $x \overset{\$}{\leftarrow} S$. We let $\{0,1\}^t$ denote the set of binary strings of length t. If A is an algorithm, we write $x \leftarrow A(y_1, \ldots, y_n)$ to indicate that x is obtained by invoking A on inputs $y_1, \ldots, y_n$. The algorithms that we consider may have access to some oracles. We write $\mathcal{A}^{\mathcal{O}}$ to indicate that the algorithm $\mathcal{A}$ has access to oracle $\mathcal{O}$. We also denote concatenation of two data strings x and y as $x \| y$.

Let $\mathcal{U} = \{\mathsf{MS}_1, \ldots, \mathsf{MS}_m\}$ be the set of all phones (a.k.a. mobile stations) that register to the network. We define $\mathsf{IMSI_i}$ to be the permanent identity of MS_i and $\mathsf{TMSI_i}$ the temporary identity of MS_i. Let $\mathbf{ID}$ be a m dimensional array which is initially set to hold $\mathbf{ID}_i = \mathsf{IMSI}_i$, as the protocol proceeds this will be updated to $\mathsf{TMSI_i}$ if a phone has been allocated this temporary identity. We also let **Revealed** denote an m dimensional vector of boolean values; which are initially all set to be false. Let $\mathbf{K}$, **MS.SQN**, **NET.SQN**, **START** denote vectors of length m, where $\mathbf{K}$ is the vector of all master keys, **MS.SQN** is the vector of phone sequence numbers, **NET.SQN** the vector of sequence numbers which the network keeps for all phones, and **START** the vector of all start values. For example with index i, $\mathbf{K}_i = \mathsf{K}_i$ is the master key of the MS_i (the phone with IMSE/TMSI given by $\mathsf{IMSI}_i/\mathsf{TMSI}_i$).

We shall use the following algorithms to abstract away various generation algorithms whose details do not concern us, but whose outputs are needed to define various quantities. The precise definitions of these algorithms can be found in the UMTS/LTE standards.

- Setup: for every $\mathsf{MS}_i \in \mathcal{U}$, this algorithm generates master keys K_i, initial sequence numbers ($\mathsf{MS.SQN}_i$, $\mathsf{NET.SQN}_i$) and initial START_i values.
- SQN.Gen: takes as input NET.SQN and outputs SQN.
- FRESH.Gen: generates a fresh number FRESH.
- COUNTER-I.Gen: takes as input START and outputs the counter COUNTER-I which is used in integrity algorithm.
- COUNTER-C.Gen: takes as input START and outputs the counter COUNTER-C which is used in encryption algorithm.
- START.Update: takes as input the value START plus either COUNTER-I or COUNTER-C, and outputs an updated value for START.

Adversarial oracles: In our security analysis, there are two adversarial oracles which model the behaviour of the phone (the MS oracle) and the network (the NET oracle), and one which allows the adversary to obtain the current identity (either the IMSI or TMSI) of a specific mobile (the Reveal oracle). Both the NET and MS oracles contain program counters for each phone MS_i. The m-vector **NET.pc** is the vector of all program counters that the NET oracle maintains. The element $\mathbf{NET.pc}_i = \mathsf{NET.pc}_i$ denotes the program counter associated to mobile station i. Similarly **MS.pc** is the vector of all program counters that the MS oracle maintains.

We assume there are two globally defined identities i_0 and i_1, which informally indicate on which phones the adversary is being challenged on. How these are set will be defined later when we discuss the security experiments. At this point note that $i_0, i_1 \in \{1, \ldots, m, \perp\}$. We let $\varUpsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ be the set of functions with the same key used in the AKA protocol of UMTS/LTE and define $\mathsf{F} = \{\varUpsilon, \mathsf{f8}, \mathsf{f9}\}$.

The network oracle $\mathsf{NET}[\mathsf{X}, \mathsf{Y}](\mathsf{id}, x)$ and mobile station oracle $\mathsf{MS}[\mathsf{X}, \mathsf{Y}](\mathsf{id}, x)$ are defined in Figure 3. These oracles are parametrized by two sets X and Y, as well as two inputs id and x. As can be seen from the figure the NET and MS oracles run the functions $\mathsf{net}[\cdot](\mathsf{K}_i, \mathsf{NET.pc}_i, x)$ and $\mathsf{ms}[\cdot](\mathsf{K}_i, \mathsf{MS.pc}_i, x)$ respectively, on different parameter sets, X, Y and F. When the index of identity id is i_0, the oracles run the functions with the set X. If the index is i_1, they run the functions with the set Y. Finally, if the index is neither i_0 or i_1 with the set F, (recall that $\mathsf{F} = \{\varUpsilon, \mathsf{f8}, \mathsf{f9}\}$ as defined above). In the real world the sets X and Y would both equal F. However, we generalise the notation to allow X and Y to be different so as to provide a notational simplification for our security proof which follows.

```
Oracle $\mathsf{NET}[\mathsf{X},\mathsf{Y}](\mathsf{id}, x)$
  – find $i$ such that $\mathsf{id} = \mathbf{ID}_i$, otherwise abort
  – if $i = i_0$, $y \leftarrow \mathsf{net}[\mathsf{X}](\mathsf{K}_i, \mathsf{NET.pc}_i, x)$
  – else $i = i_1$, $y \leftarrow \mathsf{net}[\mathsf{Y}](\mathsf{K}_i, \mathsf{NET.pc}_i, x)$
  – else $y \leftarrow \mathsf{net}[\mathsf{F}](\mathsf{K}_i, \mathsf{NET.pc}_i, x)$
  – return $y$
```

```
Oracle $\mathsf{MS}[\mathsf{X},\mathsf{Y}](\mathsf{id}, x)$
  – find $i$ such that $\mathsf{id} = \mathbf{ID}_i$, otherwise abort
  – if $i = i_0$, $y \leftarrow \mathsf{ms}[\mathsf{X}](\mathsf{K}_i, \mathsf{MS.pc}_i, x)$
  – else $i = i_1$, $y \leftarrow \mathsf{ms}[\mathsf{Y}](\mathsf{K}_i, \mathsf{MS.pc}_i, x)$
  – else $y \leftarrow \mathsf{ms}[\mathsf{F}](\mathsf{K}_i, \mathsf{MS.pc}_i, x)$
  – return $y$
```

Fig. 3. NET and MS oracles defining security for modified UMTS/LTE authentication and connection protocol

The functions net and ms used by the oracles are given in Figures 7 and 8 of Appendix A. We present a textual overview here, but the reader should simply think of the oracles/functions as implementing the UMTS/LTE protocol definition but for abstract function sets $\{\{\mathsf{h1}, \mathsf{h2}, \mathsf{h3}, \mathsf{h4}, \mathsf{h5}\}, \mathsf{h8}, \mathsf{h9}\}$, which may or may not be equivalent to the functions used in the real protocol.

The oracle NET gives the adversary the ability to communicate with the network. By calling the oracle on input (id, x), this corresponds to sending the message x, from the phone with identity id, to the network. The oracle maintains **ID**, **K**, **NET.SQN**, **START**, **COUNTER-I**, **COUNTER-C** and **NET.pc**. The oracle also uses the program counter $\mathsf{NET.pc}_i$ to maintain the state of the oracle for each phone. If $\mathsf{NET.pc}_i = 1$, the NET oracle receives the security capability of the phone (supported integrity and cipher algorithms of the phone) and then starts user authentication. If $\mathsf{NET.pc}_i = 2$, the oracle receives the *User Authentication Response*. If $\mathsf{NET.pc}_i = 3$, the oracle receives the *Security Mode Complete* message. If $\mathsf{NET.pc}_i = 4$, the oracle starts TMSI *Allocation*. If $\mathsf{NET.pc}_i = 5$, the oracle receives the TMSI *Allocation Complete* message. If $\mathsf{NET.pc}_i = 6$, the oracle starts *Data Transmission*. If $\mathsf{NET.pc}_i = 7$, the oracle receives transmitted data. Note that after AKA and the negotiation of integrity and encryption algorithms has finished, the phone and the network can either perform TMSI *Allocation* or *Data Transmission*. In order to allow the adversary the choice in which to perform, we ask them to designate the next state of the oracle by appending pc to the *Security Mode Complete* message (and any further messages x for $\mathsf{NET.pc}_i \geq 3$). The inclusion of pc allows us to update the program counter $\mathsf{NET.pc}_i$ which will in turn be checked upon the next oracle call to determine the operation to perform.

The oracle MS gives the adversary the ability to communicate with the phone. The adversary can send message x to the phone with identity id by calling the oracle on input (id, x). The oracles maintains **ID**, **K**, **MS.SQN**, **START**, **COUNTER-I**, **COUNTER-C** and **MS.pc**. The oracle uses the program counter $\mathsf{MS.pc}_i$ to maintain the state of the oracle for each phone. If $\mathsf{MS.pc}_i = 1$, the MS oracle starts communication. If $\mathsf{MS.pc}_i = 2$, the oracle receives the *User Authentication Request* and outputs the *User Authentication Response*. If $\mathsf{MS.pc}_i = 3$, the oracle receives the *Security Mode Command*. If $\mathsf{MS.pc}_i = 4$, the oracle receives the TMSI *Allocation Command*. If $\mathsf{MS.pc}_i = 5$, the oracle starts *Data Transmission*. If $\mathsf{MS.pc}_i = 6$, the oracle receives transmitted data. We again ask the adversary designate the next state after AKA and the negotiation of integrity and encryption algorithms has finished. For all received messages, where $\mathsf{MS.pc}_i \geq 3$, the adversary specifies the next state of the oracle by appending an additament pc with x. This additament pc effects the program counter $\mathsf{MS.pc}_i$ and decides the next oracle state.

To be able to call the NET and MS oracles for the i-th phone the adversary needs access to the current value of $\mathbf{ID}_i$. This is given by the Reveal oracle, on input of an index $i \in \{1, \ldots, m\}$, the current value of $\mathbf{ID}_i$ is returned and $\mathbf{Revealed}_i$ is set to be true. As the protocol progresses the TMSI will be updated during the next TMSI *Allocation* command. At this point $\mathbf{Revealed}_i$ is reset to false. We stress that unlike models for secure key exchange, the Reveal oracle here has a very different function. In key-exchange models a Reveal query is permitted to model an adversary's ability to attack a participant and obtain the key established for one particular session. In contrast, our Reveal is simply used to give the adversary the information he needs to progress the conversation between NET and MS.

Security Experiments: It is clear that the authentication and connection protocol does not offer any form of strong anonymity; indeed it is not designed to. For example, when a phone communicates with the serving network for the first time, the phone needs to identify itself by its permanent identity IMSI and the downlink or uplink message is sent with a tag of IMSI. Therefore, anonymity does not hold in the first communication between the phone and the network. In addition, during the TMSI *Allocation/Reallocation* procedure, the network sends a TMSI *Allocation Command* containing the encrypted new temporary identity TMSI_n, the phone then returns a TMSI *Allocate Complete* acknowledgement. If the network does not receive the TMSI *Allocation Complete* acknowledgement from the phone, the network falls back to using the IMSI for downlink signalling and the phone should identify itself by its permanent identity IMSI again.

However, outside of these two cases and in the case of an honest network, UMTS/LTE should offer anonymity and unlinkability of communications. We first consider the case of an adversary $\mathcal{A}$ that controls the communication between the network and two fixed phones MS_{i_0} and MS_{i_1}. The adversary can eavesdrop on transcripts or send its own messages to get responses from the phones and the network. For example, by querying the oracles $\mathsf{MS}[\mathsf{F}, \mathsf{F}](\mathsf{id}, x)$ and $\mathsf{NET}[\mathsf{F}, \mathsf{F}](\mathsf{id}, x)$ with id corresponds to $\mathbf{ID}_{i_0}$ or $\mathbf{ID}_{i_1}$, the adversary gets backs public transcripts between the network and MS_{i_0} or MS_{i_1}.

Let $\varUpsilon$ and F be as before, we formally define an experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$ that depends on protocol $\varPi$ and adversary $\mathcal{A}$. This experiment is used to model the case of static security, where the adversary is told which two phones he will end up attacking. The details are given in Figure 4. The experiment starts by running Setup which generates the various required parameters. The experiment proceeds in two phases: In the first phase the adversary calls the oracles $\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x)$ and $\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x)$ just as it would in the real world. At the end of this phase, the adversary outputs some state information st with the restriction that both phones MS_{i_0} and MS_{i_1} must be in an unrevealed state, i.e. $\mathbf{Revealed}_{i_0} = \mathbf{Revealed}_{i_1} = \mathsf{false}$.

The second phase is the challenge phase. At the beginning of this phase, the adversary is given two freshly allocated TMSIs for the two phones $(\mathsf{MS}_{i_0}, \mathsf{MS}_{i_1})$ but in a random order, i.e. $\mathcal{A}$ does not know which TMSI belongs to which phone. The adversary is permitted to query $\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x)$ and $\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x)$ oracles with (id,x) where $\mathsf{id} = \mathsf{TMSI}_{i_b}$ or $\mathsf{id} = \mathsf{TMSI}_{i_{1-b}}$. During the challenge phase the adversary is not allowed to query the Reveal oracle on the indexes i_0 or i_1, as this would allow him to trivially win the game. At the end of the challenge phase, the adversary outputs a bit $\hat{b}$. The adversary is said to win the experiment if his output is correct, i.e. $\hat{b} = b$.

```
$\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$
    $(\mathbf{K}, \mathbf{MS.SQN}, \mathbf{NET.SQN}, \mathbf{START}) \leftarrow \mathsf{Setup}$
    $\mathsf{st} \leftarrow \mathcal{A}^{\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{Reveal}(i)}$
    $\hat{b} \leftarrow \mathcal{A}^{\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{Reveal}(i)}(\mathsf{st}, \mathsf{TMSI}_{i_b}, \mathsf{TMSI}_{i_{1-b}})$
    output $\hat{b}$
```

```
$\mathsf{Exp}^{d\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}]$
    $i_0, i_1 \leftarrow \perp$
    $(\mathbf{K}, \mathbf{MS.SQN}, \mathbf{NET.SQN}, \mathbf{START}) \leftarrow \mathsf{Setup}$
    $(\mathsf{st}, i_0, i_1) \leftarrow \mathcal{A}^{\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{Reveal}(i)}$
    $\hat{b} \leftarrow \mathcal{A}^{\mathsf{MS}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{NET}[\mathsf{F},\mathsf{F}](\mathsf{id},x),\mathsf{Reveal}(i)}(\mathsf{st}, \mathsf{TMSI}_{i_b}, \mathsf{TMSI}_{i_{1-b}})$
    output $\hat{b}$
```

Fig. 4. Experiments defining anonymity for static and dynamic case of UMTS/LTE protocol.

Definition 1 (Anonymity for static case). *Let* $\varUpsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ *be a set of keyed function with the same secret key,* $\mathsf{f8}$ *and* $\mathsf{f9}$ *be keyed functions, and* $\mathsf{F} = \{\varUpsilon, \mathsf{f8}, \mathsf{f9}\}$*. We define the advantage of an adversary in breaking the anonymity in the static case to be*

$$\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] = \left|\Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}1}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] = 1] - \Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}0}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] = 1]\right|.$$

Security in the dynamic case follows a similar model, but now the adversary determines which phones it wants to be challenged on, returning this information to the challenger at the end of the first phase. The two phones have the same restriction on being revealed as in the static case. We can define an experiment, $\mathsf{Exp}^{d\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}]$, that depends on protocol $\varPi$ and adversary $\mathcal{A}$ as in Figure 4.

Definition 2 (Anonymity for dynamic case). *Let* $\varUpsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ *be a set of keyed function with the same secret key,* $\mathsf{f8}$ *and* $\mathsf{f9}$ *be keyed functions, and* $\mathsf{F} = \{\varUpsilon, \mathsf{f8}, \mathsf{f9}\}$*. We define the advantage of an adversary in breaking the anonymity in the dynamic case to be*

$$\mathsf{Adv}^{d\text{-}\mathsf{anon}}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}] = \left|\Pr[\mathsf{Exp}^{d\text{-}\mathsf{anon}\text{-}1}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}] = 1] - \Pr[\mathsf{Exp}^{d\text{-}\mathsf{anon}\text{-}0}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}] = 1]\right|.$$

Informally, we say that a protocol $\varPi$ is anonymous with respect to static (respectively dynamic) adversaries if $\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$ (respectively $\mathsf{Adv}^{d\text{-}\mathsf{anon}}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}]$) is "small" for all adversaries $\mathcal{A}$.

4 Security Analysis

Our anonymity theorems are conditional in that they depend on the underlying functions $\{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}, \mathsf{f8}, \mathsf{f9}\}$ having certain properties. As remarked in the introduction the precise property is complicated by the fact that the functions $\{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ are called using the same key. In both the definition of agility and defining the security requirement for $\mathsf{f8}$ and $\mathsf{f9}$ we will need the notion of a PRF, which we recall next.

Definition 3 (Pseudo-random functions). *Let ℓ_1 and ℓ_2 be positive integers. Let $\mathcal{F} := \{F_s\}$ be a family of keyed functions under key s, where each function F_s maps $\{0,1\}^{l_1}$ to $\{0,1\}^{l_2}$. Let $\varGamma_{\ell_1,\ell_2}$ denote the set of all functions from $\{0,1\}^{l_1}$ to $\{0,1\}^{l_2}$. Consider an adversary $\mathcal{A}$ that has oracle access to a function in $\varGamma_{\ell_1,\ell_2}$, and suppose that $\mathcal{A}$ always outputs a bit. We define the PRF-advantage of $\mathcal{A}$ to be*

$$\mathsf{Adv}^{\mathsf{prf}}_{F,\mathcal{A}} = |\Pr[F_s \xleftarrow{\$} \mathcal{F} : \mathcal{A}^{F_s} = 1] - \Pr[f \xleftarrow{\$} \varGamma_{\ell_1,\ell_2} : \mathcal{A}^{f} = 1]|.$$

Informally we say that $\mathcal{F}$ is a secure prf family if $\mathsf{Adv}^{\mathsf{prf}}_{\mathcal{F},\mathcal{A}}$ is “small” for all adversaries $\mathcal{A}$.

The notion of agility [6] considers a set of schemes, all meeting some base notion of security, and requires that security is maintained when multiple schemes use the same key. Agility is thus not a property of an individual scheme but of a set of schemes relative to some (standard) security notion. In our context we take a set $\varUpsilon$ of PRFs and talk of their agility with respect to the PRF notion:

Definition 4 (PRF Agility). *Let $\mathcal{F} := \{F\}$ be a family of keyed functions. Let $\varGamma$ denote the set of all functions. Let $\varUpsilon = \{\mathsf{f1}, \mathsf{f2}, ..., \mathsf{f}n\}$ be a subset of $\mathcal{F}$ (where all $\mathsf{f}i$ are keyed by the same key), $\varPsi = \{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, ..., \mathsf{r}n\}$ be a subset of $\varGamma$, $|\varUpsilon| = |\varPsi|$ and the domain and range of $\mathsf{f}j$ is equal to that of $\mathsf{r}j$ $(1 \leq j \leq n)$. Consider an adversary $\mathcal{A}$ that has oracle access to a set of functions in $\varGamma$, and suppose that $\mathcal{A}$ always outputs a bit. Define the advantage of $\mathcal{A}$ to be*

$$\mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{A}} = |\Pr[\varUpsilon \xleftarrow{\$} \mathcal{F} : \mathcal{A}^{\varUpsilon} = 1] - \Pr[\varPsi \xleftarrow{\$} \varGamma : \mathcal{A}^{\varPsi} = 1]|.$$

Informally, we can say that a set of functions $\varUpsilon$ is PRF agile if $\mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{A}}$ is “small” for all adversaries $\mathcal{A}$.

Anonymity of the protocol for the static and the dynamic case is formalized by the following two theorems.

Theorem 1. *If there is an adversary $\mathcal{A}$ against the static anonymity of the UMTS/LTE authentication and connection protocol, then there are adversaries $\mathcal{B}$, $\mathcal{C}$, $\mathcal{D}$, $\mathcal{E}$, $\mathcal{F}$ and $\mathcal{G}$ such that*

$$\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] \leq 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{B}} + 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{C}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{D}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{E}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{F}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{G}}.$$

Theorem 2. *If the UMTS/LTE authentication and connection protocol is run with a maximum of m phones with an adversary $\mathcal{A}$, then there is an adversary $\mathcal{B}$ which satisfies*

$$\mathsf{Adv}^{d\text{-}\mathsf{anon}}_{\varPi,\mathcal{A}}[\mathsf{F},\mathsf{F}] \leq \frac{1}{2} m(m-1) \cdot \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{B},i_0,i_1}[\mathsf{F},\mathsf{F}].$$

4.1 Proof Of Theorem 1

Proof. We denote by $\varUpsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ the set of keyed functions (with the same secret key) used in the UMTS/LTE protocol, and $\varPsi = \{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}\}$ a set of random functions. The range of $\mathsf{f}j$ is equal to that of $\mathsf{r}j$, where $j \in \{1,2,3,4,5\}$. Similarly let $f8, f9$ denote the cipher and integrity algorithm used in the UMTS/LTE protocol and let $\mathsf{r8}, \mathsf{r9}$ be random functions. Define the sets $\mathsf{F} = \{\Upsilon, \mathsf{f8}, \mathsf{f9}\}$, $\mathsf{R} = \{\Psi, \mathsf{f8}, \mathsf{f9}\}$, $\mathsf{R}' = \{\Psi, \mathsf{r8}, \mathsf{f9}\}$ and $\mathsf{R}'' = \{\Psi, \mathsf{r8}, \mathsf{r9}\}$.

With these definitions $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$ is precisely the anonymity experiment of the UMTS/LTE protocol. The proof will proceed as a series of game hops which we gradually replace the set of functions F with random functions. First we switch the two usages of the set F to that of R, by a series of game hops, before switching to the set R' and finally R''.

Switching from F to R: First we alter $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$ into a modified experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}]$ such that the adversary $\mathcal{A}$ cannot obtain information about the AK, CK and IK of the mobile station MS_{i_0}. The difference between the two experiments is as follows: In the experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$, the function sets which the adversary $\mathcal{A}$ accesses are (F,F) for MS_{i_0} and MS_{i_1}, whereas in experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}]$ the functions sets the adversary $\mathcal{A}$ accesses are (R,F) for MS_{i_0} and MS_{i_1} respectively. Recall $\mathsf{F} = \{\Upsilon, \mathsf{f8}, \mathsf{f9}\}$ and $\mathsf{R} = \{\Psi, \mathsf{f8}, \mathsf{f9}\}$. Intuitively, the PRF-agility property of Υ should guarantee that this modification has only a negligible effect on the behavior of the adversary $\mathcal{A}$. More precisely we make the following claim.

Claim 1: We now claim that

$$|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}]| = 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\Upsilon,\mathcal{B}}.$$

Proof of Claim 1: We construct an adversary $\mathcal{B}$ against the PRF-agility of the set Υ which satisfies the above equality. Assume $\Upsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ is a set of keyed functions with the same secret key and $\Psi = \{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}\}$ is a set of random functions. The range of $\mathsf{f}j$ is equal to that of $\mathsf{r}j$, where $j \in \{1,2,3,4,5\}$. The adversary $\mathcal{B}$ against the PRF-agility property has its own oracle Fn. On input (j,x), oracle Fn returns $\mathsf{f}j_K(x)$ or $\mathsf{r}j(x)$. The adversary $\mathcal{B}$ proceeds as in Figure 5. To simulate the experiment for $\mathcal{A}$, the adversary $\mathcal{B}$ generates $\mathsf{MS.SQN}_{i_0}$, $\mathsf{NET.SQN}_{i_0}$, START_{i_0} for MS_{i_0} and lets the key of its oracle K be the master key K_{i_0} of MS_{i_0}. The adversary $\mathcal{B}$ also generates K_{i_1}, $\mathsf{MS.SQN}_{i_1}$, $\mathsf{NET.SQN}_{i_1}$, START_{i_1} for MS_{i_1} and maintains the above parameters. Algorithm $\mathcal{B}$ then runs adversary $\mathcal{A}$.

```
Adversary $\mathcal{B}$
    $(\mathsf{K}_{i_1}, \textbf{MS.SQN}, \textbf{NET.SQN}, \textbf{START}) \leftarrow \mathsf{Setup}$
    $\mathsf{st} \leftarrow \mathcal{A}^{\mathsf{MS[X,F](id},x\mathsf{)},\mathsf{NET[X,F](id},x\mathsf{)}}$,   where $\mathsf{X} = \{\mathsf{Fn}, \mathsf{f8}, \mathsf{f9}\}$
    $b \xleftarrow{\$} \{0,1\}$
    $\hat{b} \leftarrow \mathcal{A}^{\mathsf{MS[X,F](id},x\mathsf{)},\mathsf{NET[X,R](id},x\mathsf{)},\mathsf{Reveal}(i)}(\mathsf{st}, \mathsf{TMSI}_{i_b}, \mathsf{TMSI}_{i_{1-b}})$
    output 1 if $\hat{b} = b$, else output 0
```

Fig. 5. Adversary $\mathcal{B}$.

In the normal phase, when $\mathcal{A}$'s queries corresponds to MS_{i_1}, the set of functions $\mathcal{A}$ accesses is given by $\mathsf{F} = \{\Upsilon, \mathsf{f8}, \mathsf{f9}\}$. Specifically, if $\mathcal{A}$'s query corresponds to MS_{i_1}, $\mathcal{B}$ follows the processes of Figure 7 and Figure 8 and generates $(\mathsf{MAC}, \mathsf{RES}, \mathsf{CK}, \mathsf{IK}, \mathsf{AK})$ for MS_{i_1} by means of the set F. The algorithm $\mathcal{B}$ then computes $(\mathsf{KEYSTREAM},\ \mathsf{MAC\text{-}I})$ by means of $(\mathsf{f8}, \mathsf{f9})$ under $(\mathsf{CK}, \mathsf{IK})$. When $\mathcal{A}$'s query corresponds to MS_{i_0}, the set of functions $\mathcal{A}$ accesses is $\mathsf{F} = \{\Upsilon, \mathsf{f8}, \mathsf{f9}\}$ or $\mathsf{R} = \{\Psi, \mathsf{f8}, \mathsf{f9}\}$ depending on $\mathcal{B}$'s oracle Fn. Specifically, $\mathcal{B}$ follows the processes of Figure 7 and Figure 8 but when $(\mathsf{MAC},\ \mathsf{RES},\ \mathsf{CK},\ \mathsf{IK},\mathsf{AK})$ are needed, $\mathcal{B}$ queries its oracle Fn to get back responses to answer $\mathcal{A}$. So $\mathcal{B}$ either generates $(\mathsf{MAC}, \mathsf{RES}, \mathsf{CK}, \mathsf{IK}, \mathsf{AK})$ for MS_{i_0} by means of $\Upsilon = \{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}\}$ under K or $\Psi = \{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}\}$. Following this $\mathcal{B}$ will use $(\mathsf{f8}, \mathsf{f9})$, under $(\mathsf{CK}, \mathsf{IK})$, to generate $(\mathsf{KEYSTREAM},\ \mathsf{MAC\text{-}I})$. At the end of the initial phase, $\mathcal{A}$ outputs some state information st.

In the challenge phase, $\mathcal{B}$ picks $b \xleftarrow{\$} \{0,1\}$ and returns two TMSIs, associated to MS_{i_0} and MS_{i_1}, to $\mathcal{A}$, in an order dependent on b. This phase then proceed as before and $\mathcal{B}$ models any oracle queries appropriately. If $\mathsf{Fn} = \varUpsilon$, the view of $\mathcal{A}$ is exactly as in $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]$. If $\mathsf{Fn} = \varPsi$ then the view of $\mathcal{A}$ is exactly as in $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}]$. Finally, $\mathcal{A}$ outputs a decision bit $\hat{b}$. Algorithm $\mathcal{B}$ outputs 1 if $\hat{b} = b$ and 0 otherwise.

To prove the claim we first note that:

$$\Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] = b] = \frac{1}{2}(1 + \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]).$$

Now using this result and Definition 4 we have,

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{B}} &= |\Pr[\varUpsilon \xleftarrow{\$} \mathcal{F} : \mathcal{B}^{\varUpsilon} = 1] - \Pr[\varPsi \xleftarrow{\$} \varGamma : \mathcal{B}^{\varPsi} = 1]| \\
&= |\Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] = b] - \Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}] = b]| \\
&= |\frac{1}{2}(1 + \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}]) - \frac{1}{2}(1 + \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}])| \\
&= \frac{1}{2}(|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{F},\mathsf{F}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}]|).
\end{aligned}$$

Thus proving Claim 1.

We next switch the experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{R},\mathsf{F}]$ into a modified experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{R},\mathsf{R}]$ such that the adversary $\mathcal{A}$ cannot obtain information about AK, CK and IK of the mobile station MS_{i_1}. The difference is that in $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{R},\mathsf{R}]$, instead of the outputs of $\varUpsilon$, we replay with the outputs of $\varPsi$ when $\mathcal{A}$'s query corresponds to MS_{i_1}. In the experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{R},\mathsf{R}](p)$, the functions $\mathcal{A}$ accesses are (R,R) whereas in the experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\varPi,\mathcal{A}}[\mathsf{R},\mathsf{F}]$ the functions $\mathcal{A}$ accesses are (R,F). Recall $\mathsf{R} = \{\varUpsilon, \mathsf{f8}, \mathsf{f9}\}$ and $\mathsf{F} = \{\varPsi, \mathsf{f8}, \mathsf{f9}\}$, the PRF-agility property of $\varUpsilon$ should guarantee that this modification has only a negligible effect on the behavior of the adversary $\mathcal{A}$.

Claim 2: We claim that

$$|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{F}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}]| = 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\varUpsilon,\mathcal{C}}.$$

Proof of Claim 2: This follows in a similar way to that of Claim 1. The algorithm $\mathcal{C}$ against the agility of $\varUpsilon$ is similar to $\mathcal{B}$. The differences are as follows. The algorithm $\mathcal{C}$ generates the master key for MS_{i_0} and lets the key K of its oracle be the master key of MS_{i_1}. $\mathcal{C}$ then answers $\mathcal{A}$'s query by means of R when $\mathcal{A}$'s query corresponds to MS_{i_0} and answers by means of F or R (depending on the oracle Fn) when $\mathcal{A}$'s query corresponds to MS_{i_1}. We omit the details.

Switching to R' and R'': We now wish to take care of $\mathsf{f8}$ and $\mathsf{f9}$ and switch the experiment $\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}]$ to a modified experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}\mathsf{b}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}]$ as follows. The difference in the modified experiment is that when $\mathcal{A}$'s query corresponds to MS_{i_0}, the ciphering keystream $\mathsf{KEYSTREAM}$ is computed from a random function $\mathsf{r8}$ rather than $\mathsf{f8}$ under the key CK_{i_0}. The functions $\mathcal{A}$ accesses are R' and R for MS_{i_0} and MS_{i_1} respectively. Recall that $\mathsf{R} = \{\varPsi, \mathsf{f8}, \mathsf{f9}\}$ and $\mathsf{R}' = \{\varPsi, \mathsf{r8}, \mathsf{f9}\}$. If $\mathsf{f8}$ is a pseudorandom function, then this modification has only a negligible effect on the behavior of the adversary $\mathcal{A}$.

Claim 3: We claim that

$$|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\varPi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}]| = 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{D}}.$$

Proof of Claim 3: Assume there exists a prf-adversary $\mathcal{D}$ with access to its own oracle Fn. On input x, the oracle Fn returns either $\mathsf{f8}_K(x)$ or $\mathsf{r8}(x)$, where $\mathsf{r8}$ is a random function with the same range as $\mathsf{f8}$. The adversary $\mathcal{D}$ proceeds as in Figure 6.

To simulate the experiment for $\mathcal{A}$, $\mathcal{D}$ first generates $\mathsf{MS.SQN}_{i_0}$, $\mathsf{MS.SQN}_{i_1}$, $\mathsf{NET.SQN}_{i_0}$, $\mathsf{NET.SQN}_{i_1}$, START_{i_0}, START_{i_1} for MS_{i_0} and MS_{i_1}. Then $\mathcal{D}$ runs $\mathcal{A}$. During the experiment, $\mathcal{D}$ proceeds as in Figure 7 and Figure 8. $\mathcal{D}$ begins by deriving $(\mathsf{MAC}, \mathsf{RES}, \mathsf{CK}, \mathsf{IK}, \mathsf{AK})$ for MS_{i_0} and MS_{i_1} from the (random) functions $\Psi = \{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}\}$. For MS_{i_0}, $\mathcal{D}$ will not derive the cipher key CK_{i_0}, instead this is chosen by $\mathcal{D}$'s challenger in the prf security experiment. Effectively we set $\mathsf{CK}_{i_0} = K$. Note that since Ψ is a set of random functions by definition, this means both the cipher keys CK_{i_0} and CK_{i_1} are indistinguishable from random. As a result this will not affect $\mathcal{D}$'s simulation of the environment for $\mathcal{A}$.

If the cipher keystream is needed for MS_{i_1}, $\mathcal{D}$ computes the keystream by means of $\mathsf{f8}$ under the cipher key CK_{i_1} of MS_{i_1}. If the cipher key stream is needed for MS_{i_0}, $\mathcal{D}$ queries its own oracle Fn and receives either the output of $\mathsf{f8}$ under $K = \mathsf{CK}_{i_0}$ or $\mathsf{r8}$. At the end of normal phase, $\mathcal{A}$ outputs some state information st.

```
Adversary $\mathcal{D}^b_{i_0,i_1}$
    $(\mathbf{MS.SQN}, \mathbf{NET.SQN}, \mathbf{START}) \leftarrow \mathsf{Setup}$
    $\mathsf{st} \leftarrow \mathcal{A}^{\mathsf{MS[X,R](id},x\mathsf{)},\mathsf{NET[X,R](id},x\mathsf{)}}$,where $\mathsf{X} = \{\Psi, \mathsf{Fn}, \mathsf{f9}\}$
    $b \xleftarrow{\$} \{0,1\}$
    $\hat{b} \leftarrow \mathcal{A}^{\mathsf{MS[X,R](id},x\mathsf{)},\mathsf{NET[X,R](id},x\mathsf{)}}(\mathsf{st}, \mathsf{TMSI}_{i_b}, \mathsf{TMSI}_{i_{1-b}})$
    output 1 if $\hat{b} = b$, else output 0
```

Fig. 6. Adversary $\mathcal{D}$.

In the challenge phase, $\mathcal{D}$ picks $b \xleftarrow{\$} \{0,1\}$ and returns to $\mathcal{A}$ two TMSIs associated to MS_{i_0} and MS_{i_1}, in an order dependent on b. If $\mathsf{Fn} = \mathsf{f8}$, the view of $\mathcal{A}$ is exactly as in $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}]$. If $\mathsf{Fn} = \mathsf{r8}$, the view of $\mathcal{A}$ is just in $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}]$. Finally, $\mathcal{A}$ outputs the decision bit $\hat{b}$, $\mathcal{D}$ outputs 1 if $\hat{b} = b$, else outputs 0.

We now prove the claim in a similar way to Claim 1, this time using Definition 3.

$$\begin{aligned}
\mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{D}} &= |\Pr[\mathsf{f8} \xleftarrow{\$} \mathcal{F} : \mathcal{D}^{\mathsf{f8}} = 1] - \Pr[\mathsf{r8} \xleftarrow{\$} \Gamma_{\ell_1,\ell_2} : \mathcal{D}^{\mathsf{r8}} = 1]| \\
&= |\Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}] = b] - \Pr[\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}] = b]| \\
&= |\frac{1}{2}(1 + \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}]) - \frac{1}{2}(1 + \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}])| \\
&= \frac{1}{2}(|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R},\mathsf{R}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}]|).
\end{aligned}$$

Thus proving Claim 3.

Next we switch $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}]$ to the modified experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'',\mathsf{R}]$. Recall that $\mathsf{R}' = \{\Psi, \mathsf{r8}, \mathsf{f9}\}$ and $\mathsf{R}'' = \{\Psi, \mathsf{r8}, \mathsf{r9}\}$. In the modified experiment if $\mathcal{A}$'s query corresponds to MS_{i_0}, then the message authentication tag will be computed from the random function $\mathsf{r9}$ rather than $\mathsf{f9}$ under the integrity key IK_{i_0}. If $\mathsf{f9}$ is a pseudo random function, then this modification has only a negligible effect on the behavior of the adversary $\mathcal{A}$.

Claim 4: We claim that

$$|\mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}',\mathsf{R}] - \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'',\mathsf{R}]| = 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{E}}.$$

Proof of Claim 4: We construct an algorithm $\mathcal{E}$ against the prf security of $\mathsf{f9}$ in a similar way to $\mathcal{D}$. During the experiment, $\mathcal{E}$ proceeds as in Figure 7 and Figure 8 initialising values and answering oracle queries appropriately. For MS_{i_0}, $\mathcal{E}$ will not derive the integrity key IK_{i_0}, instead this is chosen by $\mathcal{E}$'s challenger in the prf security experiment. If the integrity keystream is needed for MS_{i_1}, $\mathcal{E}$ computes the keystream using $\mathsf{f9}$ under the integrity key IK_{i_1} of MS_{i_1}. If the integrity keystream is needed for MS_{i_0}, $\mathcal{E}$ queries its own oracle Fn and receives either the output of $\mathsf{f9}$ (under $K = \mathsf{IK}_{i_0}$) or $\mathsf{r9}$. As the rest of the proof proceeds in a similar way to Claim 3 we omit the details.

We now switch $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}]$ to the modified experiment $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}']$. In this new experiment if $\mathcal{A}$'s query corresponds to MS_{i_1}, then the ciphering keystream is computed from a random function $\mathsf{r8}$ rather than $\mathsf{f8}$ under the cipher key CK_{i_1}. Again, if $\mathsf{f8}$ is a pseudo random function, then this modification has only a negligible effect on the behavior of the adversary $\mathcal{A}$.

Claim 5: We claim that

$$|\mathsf{Adv}^{\textit{s-anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}] - \mathsf{Adv}^{\textit{s-anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}']| = 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{F}}.$$

Proof of Claim 5: This proceeds in a similar way to the proof of Claim 3. We construct an algorithm $\mathcal{F}$ against the prf security of $\mathsf{f8}$ in the same fashion as $\mathcal{D}$. In $\mathcal{F}$ we answer oracle queries using R'' when $\mathcal{A}$'s query corresponds to MS_{i_0} and by means of R or R' (depending on the oracle Fn) when $\mathcal{A}$'s query corresponds to MS_{i_1}. We again omit the full details as these proceed as in previous proofs.

Finally we switch $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}']$ to the modified experiment $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'']$. Now if $\mathcal{A}$'s query corresponds to MS_{i_1}, then the message authentication tag is computed from a random function $\mathsf{r9}$ rather than using $\mathsf{f9}$ under the integrity key IK_{i_1}. If $\mathsf{f9}$ is a pseudo random function, then this modification has only a negligible effect on the behavior of adversary $\mathcal{A}$.

Claim 6: We claim that

$$|\mathsf{Adv}^{\textit{s-anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'] - \mathsf{Adv}^{\textit{s-anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'']| = 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{G}}.$$

Proof of Claim 6: Again the proof follows that of Claim 3. We construct an algorithm $\mathcal{G}$ against the prf security of $\mathsf{f9}$ in a similar way to $\mathcal{D}$. In $\mathcal{G}$ we answer oracle queries using R'' when $\mathcal{A}$'s query corresponds to MS_{i_0} and by means of R' or R'' (depending on the oracle Fn) when $\mathcal{A}$'s query corresponds to MS_{i_1}. We again omit the full details.

Bounding the advantage under random functions: *Claim 7:* We claim that $\mathsf{Adv}^{\textit{s-anon}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}''] = 0$.

Proof of Claim 7: The adversary $\mathcal{A}$ breaks the anonymity of $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'']$ only when one of the following cases occur:

– *Case 1:* $\mathcal{A}$ obtains information about the sequence number or the ciphering and integrity keys of the phone from the response of the oracle.
– *Case 2:* $\mathcal{A}$ queries the same message in both phases of the experiment and the oracle returns the same response. For example, $\mathcal{A}$ queries the message x with id_{i_0} in the normal phase and gets the response y. Following receipt of the challenge TMSIs, $\mathcal{A}$ queries message x with TMSI_{i_b}. If $b = 0$ and the oracle returns response y, $\mathcal{A}$ may be able link both queries to the same phone MS_{i_0}.

Now we discuss the above two cases in turn. In the experiment $\mathsf{Exp}^{\textit{s-anon-b}}_{\Pi,\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'']$, no matter what $\mathcal{A}$ queries for MS_{i_0} or MS_{i_1}, the values $(\mathsf{MAC}, \mathsf{RES}, \mathsf{CK}, \mathsf{IK}, \mathsf{AK}, \mathsf{MAC\text{-}I}, \mathsf{KEYSTREAM})$ are all computed from the random functions $\{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}, \mathsf{r8}, \mathsf{r9}\}$ rather than from $\{\mathsf{f1}, \mathsf{f2}, \mathsf{f3}, \mathsf{f4}, \mathsf{f5}, \mathsf{f8}, \mathsf{f9}\}$ under K_{i_b} (in the original anonymity experiment). The key K_{i_0} of MS_{i_0} and the key K_{i_1} of MS_{i_1} are no longer used to generate anything. Moreover, the phone sequence number SQN is always masked by the random anonymity key $\mathsf{AK} = \mathsf{r5}(\mathsf{RAND})$, and will be increased with each query. Thus the adversary $\mathcal{A}$ cannot obtain information about the sequence number from the public authentication token AUTN. Therefore, the adversary $\mathcal{A}$ gains no information about the bit b from the response of the query, i.e. Case 1 cannot happen.

Furthermore, notice that the random numbers, fresh numbers and counters used in the functions $\{\mathsf{r1}, \mathsf{r2}, \mathsf{r3}, \mathsf{r4}, \mathsf{r5}, \mathsf{r8}, \mathsf{r9}\}$ are different for each query. Therefore, even if $\mathcal{A}$ queries the same message at different times, the oracle always outputs a different response. As a result the adversary $\mathcal{A}$ cannot distinguish the two phones by sending the same message twice, i.e. Case 2 does not occur.

The only strategy remaining for $\mathcal{A}$ is to output a random guess $\hat{b}$. Therefore, $\Pr[\mathsf{Exp}^{\textit{s}\text{-}\mathsf{anon}\text{-}\mathsf{0}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}''] = 1] = 1/2$ and $\Pr[\mathsf{Exp}^{\textit{s}\text{-}\mathsf{anon}\text{-}\mathsf{1}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}''] = 1] = 1/2$, proving that Claim 7 holds.

We are now in position to prove the theorem using the above claims. In the following the number at the end of each line corresponds to the associated claim.

$$\begin{aligned}
&\mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{F}, \mathsf{F}] \\
&= \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{F}, \mathsf{F}] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}, \mathsf{F}] && (1)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}, \mathsf{F}] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}, \mathsf{R}] && (2)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}, \mathsf{R}] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}', \mathsf{R}] && (3)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}', \mathsf{R}] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}] && (4)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'] && (5)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}'] - \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}''] && (6)\\
&\quad + \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A},i_0,i_1}[\mathsf{R}'', \mathsf{R}''] && (7)\\
&\leq 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\Upsilon,\mathcal{B}} + 2 \cdot \mathsf{Adv}^{\mathsf{PR}}_{\Upsilon,\mathcal{C}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{D}} \\
&\quad + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{E}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f8},\mathcal{F}} + 2 \cdot \mathsf{Adv}^{\mathsf{prf}}_{\mathsf{f9},\mathcal{G}}. \qquad \square
\end{aligned}$$

4.2 Proof Of Theorem 2

Proof. Let $\mathcal{A}$ be an algorithm attacking the anonymity of the UMTS/LTE authentication and connection protocol in the dynamic experiment. We denote by $\mathsf{Exp}^{\textit{d}\text{-}\mathsf{anon}\text{-}b}_{\mathit{\Pi},\mathcal{A}}[\mathsf{F}, \mathsf{F}]$ the experiment that $\mathcal{A}$ engages in, $\mathsf{NET}^{\mathcal{A}}$ and $\mathsf{MS}^{\mathcal{A}}$ the oracles that $\mathcal{A}$ may query, and $\mathsf{Adv}^{\textit{d}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A}}[\mathsf{F}, \mathsf{F}]$ the advantage of $\mathcal{A}$.

We shall construct a new adversary $\mathcal{B}$ attacking the anonymity for the static case by running algorithm $\mathcal{A}$ and simulating the required oracles needed by $\mathcal{A}$ in its experiment. Let us denote by $\mathsf{Exp}^{\textit{s}\text{-}\mathsf{anon}\text{-}b}_{\mathit{\Pi},\mathcal{B},i_0,i_1}[\mathsf{F}, \mathsf{F}]$ the experiment $\mathcal{B}$ performs for the static case, $\mathsf{NET}^{\mathcal{B}}$ and $\mathsf{MS}^{\mathcal{B}}$ the oracles that $\mathcal{B}$ may query, and $\mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{B},i_0,i_1}[\mathsf{F}, \mathsf{F}]$ the advantage of $\mathcal{B}$. We will show that

$$\mathsf{Adv}^{\textit{d}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{A}}[\mathsf{F}, \mathsf{F}] \leq m(m-1) \cdot \mathsf{Adv}^{\textit{s}\text{-}\mathsf{anon}}_{\mathit{\Pi},\mathcal{B},i_0,i_1}[\mathsf{F}, \mathsf{F}],$$

where m is the number of phones registering to the network.

To simulate the environment for $\mathcal{A}$, the adversary $\mathcal{B}$ first generates master keys, sequence numbers and start values for all phones except the two fixed phones MS_{i_0} and MS_{i_1} that $\mathcal{B}$ needs to distinguish. Next $\mathcal{B}$ runs $\mathcal{A}$ and answers $\mathcal{A}$'s queries to $\mathsf{NET}^{\mathcal{A}}$ and $\mathsf{MS}^{\mathcal{A}}$ as follows.

In the normal phase of the simulated experiment, if $\mathcal{A}$'s query corresponds to MS_i where $i \neq i_0$ or i_1, $\mathcal{B}$ proceeds as in Figure 7 and Figure 8 to answer $\mathcal{A}$'s query. Since $\mathcal{B}$ generated the master key of MS_i, $\mathcal{B}$ can answer the query perfectly. If $\mathcal{A}$'s query corresponds to MS_{i_0} and MS_{i_1}, $\mathcal{B}$ first queries its oracles $\mathsf{NET}^{\mathcal{B}}$ and $\mathsf{MS}^{\mathcal{B}}$ and then passes the responses to $\mathcal{A}$. At the end of normal phase, $\mathcal{A}$ outputs state information st and two identity indexes of its choice i'_0 and i'_1. Following this $\mathcal{B}$ checks whether (i_0, i_1) and (i'_0, i'_1) are equal. If $(i_0, i_1) \neq (i'_0, i'_1)$ or (i'_1, i'_0) then $\mathcal{B}$ aborts. Otherwise, $\mathcal{B}$ distinguishes i_0 from i_1 by continuing to simulate the experiment for $\mathcal{A}$. In the challenge phase $\mathcal{B}$ sends $\mathcal{A}$ the two TMSIs (TMSI_{i_b} and $\mathsf{TMSI}_{i_{1-b}}$) that $\mathcal{B}$ received in the challenge phase of its own experiment $\mathsf{Exp}^{s\text{-}\mathsf{anon}\text{-}b}_{\Pi,\mathcal{B},i_0,i_1}[\mathsf{F},\mathsf{F}]$. When $\mathcal{A}$ queries with either of these two TMSIs, $\mathcal{B}$ shall query $\mathsf{NET}^{\mathcal{B}}$ and $\mathsf{MS}^{\mathcal{B}}$ and pass the response to $\mathcal{A}$. At the end of the challenge phase, $\mathcal{A}$ halts with output $\hat{b}$, if $(i_0, i_1) = (i'_0, i'_1)$ then $\mathcal{B}$ then takes $\hat{b}$ as its output and if $(i_0, i_1) = (i'_1, i'_0)$ then $\mathcal{B}$ then takes $1 - \hat{b}$ as its output

In the case of $(i_0, i_1) = (i'_0, i'_1)$ or (i'_1, i'_0), the bit $\hat{b}$ (that $\mathcal{A}$ outputs in $\mathcal{B}$'s simulation and $\mathcal{B}$ takes to generate his output) is with respect to the two phones MS_{i_0} and MS_{i_1} (that $\mathcal{B}$ needs to distinguish). Adversary $\mathcal{B}$ shall win (i.e. $\mathcal{B}$ breaks anonymity for static case) if $\mathcal{B}$ does not abort and $\mathcal{A}$ wins (i.e. $\mathcal{A}$ breaks anonymity for dynamic case). Therefore, we have

$$\Pr[\mathcal{B} \text{ wins}] = \Pr[\mathcal{B} \text{ does not abort } \wedge \ \mathcal{A} \text{ wins}].$$

Note the event that $\mathcal{B}$ does not abort and the event that $\mathcal{A}$ wins do not affect each other, the two events are independent, so

$$\Pr[\mathcal{B} \text{ does not abort } \wedge \ \mathcal{A} \text{ wins}] = \Pr[\mathcal{B} \text{ does not abort}] \cdot \Pr[\mathcal{A} \text{ wins}].$$

The fixed indexes (i_0, i_1) to be distinguished for $\mathcal{B}$ need to match the selected indexes (i'_0, i'_1) of $\mathcal{A}$ (i.e. $i_0 = i'_0$ and $i_1 = i'_1$, or $i_0 = i'_1$ and $i_1 = i'_0$). Since there are m phones registering to the network, this happens with probability at least $\frac{2}{m} \cdot \frac{1}{m-1}$, i.e.

$$\Pr[\mathcal{B} \text{ does not abort}] \geq \frac{2}{m(m-1)}.$$

We therefore derive

$$\begin{aligned}\Pr[\mathcal{B} \text{ wins}] &= \Pr[\mathcal{B} \text{ does not abort} \wedge \mathcal{A} \text{ wins}] \\ &= \Pr[\mathcal{B} \text{ does not abort}] \cdot \Pr[\mathcal{A} \text{ wins}] \\ &\geq \frac{2}{m(m-1)} \cdot \Pr[\mathcal{A} \text{ wins}].\end{aligned}$$

Finally we bound the advantage of $\mathcal{A}$ by

$$\mathsf{Adv}^{d\text{-}\mathsf{anon}}_{\Pi,\mathcal{A}}[\mathsf{F},\mathsf{F}] \leq \frac{1}{2} m(m-1) \cdot \mathsf{Adv}^{s\text{-}\mathsf{anon}}_{\Pi,\mathcal{B},i_0,i_1}[\mathsf{F},\mathsf{F}].$$

□

5 Acknowledgements

This work has been supported in part by ERC Advanced Grant ERC-2010-AdG-267188-CRIPTO, the second author was also supported in part by a Royal Society Wolfson Merit Award. We thank Steve Babbage for comments on an earlier version of this manuscript.

References

1. 3GPP. Specification of the 3GPP Confidentiality and Integrity Algorithms. Document 1: f8 and f9 Specifications. ETSI TS 135 201 V11.0.0 (2012-11), 2012.
2. 3GPP. Specification of the 3GPP Confidentiality and Integrity Algorithms. Document 2: Kasumi Algorithm Specification. ETSI TS 135 202 V11.0.0 (2012-11), 2012.
3. 3GPP. Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 and UIA2. Document 1: UEA2 and UIA2 Specification. ETSI TS 135 215 V11.0.0 (2012-11), 2012.
4. 3GPP. Specification of the 3GPP Confidentiality and Integrity Algorithms UEA2 and UIA2; Document 2: SNOW 3G Specification. ETSI TS 135 216 V11.0.0 (2012-11), 2012.
5. 3GPP. Universal Mobile Telecommunications System (UMTS); 3G security; Security architecture. ETSI TS 133 102 V11.5.1 (2013-07), 2013.
6. Tolga Acar, Mira Belenkiy, Mihir Bellare, and David Cash. Cryptographic agility and its relation to circular encryption. In Henri Gilbert, editor, *EUROCRYPT*, volume 6110 of *Lecture Notes in Computer Science*, pages 403–422. Springer, 2010.
7. Myrto Arapinis, Loretta Ilaria Mancini, Eike Ritter, Mark Ryan, Nico Golde, Kevin Redon, and Ravishankar Borgaonkar. New privacy issues in mobile telephony: fix and verification. In Ting Yu, George Danezis, and Virgil D. Gligor, editors, *ACM Conference on Computer and Communications Security*, pages 205–216. ACM, 2012.
8. Elad Barkan, Eli Biham, and Nathan Keller. Instant ciphertext-only cryptanalysis of GSM encrypted communication. *J. Cryptology*, 21(3):392–429, 2008.
9. Mihir Bellare, Ran Canetti, and Hugo Krawczyk. A modular approach to the design and analysis of authentication and key exchange protocols (extended abstract). In Jeffrey Scott Vitter, editor, *STOC*, pages 419–428. ACM, 1998.
10. Mihir Bellare, Daniele Micciancio, and Bogdan Warinschi. Foundations of group signatures: Formal definitions, simplified requirements, and a construction based on general assumptions. In Eli Biham, editor, *EUROCRYPT*, volume 2656 of *Lecture Notes in Computer Science*, pages 614–629. Springer, 2003.
11. Mihir Bellare and Phillip Rogaway. Entity authentication and key distribution. In Douglas R. Stinson, editor, *CRYPTO*, volume 773 of *Lecture Notes in Computer Science*, pages 232–249. Springer, 1993.
12. Mihir Bellare, Haixia Shi, and Chong Zhang. Foundations of group signatures: The case of dynamic groups. In Alfred Menezes, editor, *CT-RSA*, volume 3376 of *Lecture Notes in Computer Science*, pages 136–153. Springer, 2005.
13. Adam Bender, Jonathan Katz, and Ruggero Morselli. Ring signatures: Stronger definitions, and constructions without random oracles. *J. Cryptology*, 22(1):114–138, 2009.
14. Ernest F. Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Vijayalakshmi Atluri, Birgit Pfitzmann, and Patrick Drew McDaniel, editors, *ACM Conference on Computer and Communications Security*, pages 132–145. ACM, 2004.
15. Ernie Brickell, Liqun Chen, and Jiangtao Li. Simplified security notions of direct anonymous attestation and a concrete scheme from pairings. *Int. J. Inf. Secur.*, 8(5):315–330, September 2009.
16. Ran Canetti and Hugo Krawczyk. Analysis of key-exchange protocols and their use for building secure channels. In Birgit Pfitzmann, editor, *EUROCRYPT*, volume 2045 of *Lecture Notes in Computer Science*, pages 453–474. Springer, 2001.
17. Yevgeniy Dodis, Aggelos Kiayias, Antonio Nicolosi, and Victor Shoup. Anonymous identification in ad hoc groups. In Christian Cachin and Jan Camenisch, editors, *EUROCRYPT*, volume 3027 of *Lecture Notes in Computer Science*, pages 609–626. Springer, 2004.
18. Orr Dunkelman, Nathan Keller, and Adi Shamir. A practical-time related-key attack on the KASUMI cryptosystem used in gsm and 3g telephony. In Tal Rabin, editor, *CRYPTO*, volume 6223 of *Lecture Notes in Computer Science*, pages 393–410. Springer, 2010.
19. Dirk Fox. Der IMSI-catcher. *Datenschutz und Datensicherheit*, 26(4), 2002.
20. Aleksandar Kircanski and Amr M. Youssef. On the sliding property of SNOW3G and SNOW 2.0. *IET Information Security*, 5(4):199–206, 2011.
21. Ulrike Meyer and Susanne Wetzel. A man-in-the-middle attack on UMTS. In Markus Jakobsson and Adrian Perrig, editors, *Workshop on Wireless Security*, pages 90–97. ACM, 2004.
22. Chris J. Mitchell. The security of the GSM air interface protocol. Technical Report RHUL-MA-2001-3, Royal Holloway University of London, 2001.
23. Paulo S. Pagliusi. A contemporary foreword on GSM security. In *Proceedings of the International Conference on Infrastructure Security*, InfraSec '02, pages 129–144, London, UK, UK, 2002. Springer-Verlag.

24. Muxiang Zhang and Yuguang Fang. Security analysis and enhancements of 3GPP authentication and key agreement protocol. *IEEE Transactions on Wireless Communications*, 4(2):734–742, 2005.

A Figures

```
$\mathsf{net}[\{\{\mathsf{h1}, \mathsf{h2}, \mathsf{h3}, \mathsf{h4}, \mathsf{h5}\}, \mathsf{h8}, \mathsf{h9}\}](\mathsf{K}_i, \mathsf{NET.pc}_i, x)$

– if $\mathsf{NET.pc}_i = 1$ and $x =(\mathsf{START}_i$, security capability) //receive $\mathsf{START}_i$, supported integrity and cipher algorithms of the phone
    • store security capability of the phone
    • store $\mathsf{START}_i$ in **START** according to index $i$
    • $\mathsf{RAND} \overset{\$}{\leftarrow} \{0,1\}^{128}$
    • $\mathsf{SQN} \leftarrow \mathsf{SQN.Gen}(\mathsf{NET.SQN}_i)$
    • $\mathsf{NET.SQN}_i \leftarrow \mathsf{NET.SQN}_i + 1$
    • $\mathsf{MAC} \leftarrow \mathsf{h1}_{\mathsf{K}_i}(\mathsf{SQN}||\mathsf{RAND}||\mathsf{AMF})$
    • $\mathsf{XRES} \leftarrow \mathsf{h2}_{\mathsf{K}_i}(\mathsf{RAND})$
    • $\mathsf{CK}_i \leftarrow \mathsf{h3}_{\mathsf{K}_i}(\mathsf{RAND})$
    • $\mathsf{IK}_i \leftarrow \mathsf{h4}_{\mathsf{K}_i}(\mathsf{RAND})$
    • $\mathsf{AK}_i \leftarrow \mathsf{h5}_{\mathsf{K}_i}(\mathsf{RAND})$
    • $\mathsf{AUTN} \leftarrow \mathsf{SQN} \bigoplus \mathsf{AK}_i||\mathsf{AMF}||\mathsf{MAC}$
    • $\mathsf{NET.pc}_i \leftarrow 2$
    • return $\mathsf{RAND}||\mathsf{AUTN}$ //*User Authentication Request*
– if $\mathsf{NET.pc}_i = 2$ and $x = \mathsf{RES}$ //receive *User Authentication Response*
    • if $\mathsf{RES} \neq \mathsf{XRES}$ then abort
    • select integrity and encryption algorithms supported by the phone
    • $\mathsf{FRESH} \leftarrow \mathsf{FRESH.Gen}(k)$
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • $m_S$ be the *Security Mode Command* message
    • $\mathsf{MAC\text{-}I}_S \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, m_S, 1, \mathsf{FRESH})$
    • $\mathsf{NET.pc}_i \leftarrow 3$
    • return $(m_S, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_S)$ //*Security Mode Command*
– if $\mathsf{NET.pc}_i = 3$ and $x = (m_i, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_i)||\mathsf{pc}$ //receive *Security Mode Complete*
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • if $\mathsf{MAC\text{-}I}_i \neq \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, m_i, 0, \mathsf{FRESH})$ then abort
    • if $\mathsf{pc} = 4$, 6 or 7, $\mathsf{NET.pc}_i \leftarrow \mathsf{pc}$ else abort
    • return "OK"
– if $\mathsf{NET.pc}_i = 4$ and $x = \mathsf{allocate}||\mathsf{pc}$ //start TMSI *Allocation*
    • $\mathsf{TMSI}_{i_n} \leftarrow \mathsf{TMSI.Gen}(p)$
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{FRESH} \leftarrow \mathsf{FRESH.Gen}(k)$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 1, |m|)$
    • $c_{\mathsf{TMSI}} \leftarrow \mathsf{KEYSTREAM} \oplus \mathsf{TMSI}_{i_n}$
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • $\mathsf{MAC\text{-}I}_S \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, c_{\mathsf{TMSI}}, 1, \mathsf{FRESH})$
    • $\mathsf{NET.pc}_i \leftarrow 5$
    • return $(c_{\mathsf{TMSI}}, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_S)$ //TMSI *Allocation Command*
– if $\mathsf{NET.pc}_i = 5$ and $x = (\mathsf{ack}, \mathsf{MAC\text{-}I}_i)||\mathsf{pc}$ //receive TMSI *allocation complete*
    • if $\mathsf{MAC\text{-}I}_i \neq \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, \mathsf{ack}, 0, \mathsf{FRESH})$ then abort
    • if $\mathsf{pc} = 6$ or 7 $\mathsf{NET.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return "OK"
– if $\mathsf{NET.pc}_i = 6$ and $x = m||\mathsf{pc}$ //start *Data Transmission*
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 1, |m|)$
    • $c_S \leftarrow \mathsf{KEYSTREAM} \oplus m$
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • $\mathsf{FRESH} \leftarrow \mathsf{FRESH.Gen}(k)$
    • $\mathsf{MAC\text{-}I}_S \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, c_S, 1, \mathsf{FRESH})$
    • if $\mathsf{pc} = 4$ or 7, $\mathsf{NET.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return $(c_S, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_S)$
– if $\mathsf{NET.pc}_i = 7$ and $x = (c_i, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_i)||\mathsf{pc})$ //receive transmitted message
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • if $\mathsf{MAC\text{-}I} \neq \mathsf{h9}_{\mathsf{IK}}(\mathsf{COUNTER\text{-}I}, c_i, 0, \mathsf{FRESH})$ then abort
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 0, |m|)$
    • $m_i \leftarrow \mathsf{KEYSTREAM} \oplus c_i$
    • if $\mathsf{pc} = 4$ or 6, $\mathsf{NET.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return "OK"
– else abort
```

Fig. 7. net function for NET oracle

```
$\mathsf{ms}[\{\{\mathsf{h1},\mathsf{h2},\mathsf{h3},\mathsf{h4},\mathsf{h5}\},\mathsf{h8},\mathsf{h9}\}](\mathsf{K}_i,\mathsf{MS.pc}_i,x)$

– if $\mathsf{MS.pc}_i = 1$ and $x = \mathsf{init}$ //start communication
    • $\mathsf{MS.pc}_i \leftarrow 2$
    • return $\mathsf{START}_i$, security capability (supported integrity and cipher algorithms of the phone)
– if $\mathsf{MS.pc}_i = 2$ and $x = \mathsf{RAND}||\mathsf{AUTN}$ //receive *User Authentication Request*
    • parse $x$ as $x_1||x_2||x_3||x_4$ where $x_1 = \mathsf{RAND}$,
      $x_2 = \mathsf{SQN} \bigoplus \mathsf{AK}$, $x_3 = \mathsf{AMF}$, $x_4 = \mathsf{MAC}$
    • $\mathsf{AK}_i \leftarrow \mathsf{h5}_{\mathsf{K}_i}(x_1)$
    • $\mathsf{SQN} \leftarrow x_2 \bigoplus \mathsf{AK}_i$
        ∗ if $\mathsf{SQN} > \mathsf{MS.SQN}_i$ then $\mathsf{MS.SQN}_i \leftarrow \mathsf{SQN}$
          else abort
    • if $x_4 = \mathsf{h1}_{\mathsf{K}_i}(\mathsf{SQN}||x_1||x_3)$
      then $\mathsf{RES} \leftarrow \mathsf{h2}_{\mathsf{K}_i}(x_1)$ else abort
    • $\mathsf{CK}_i \leftarrow \mathsf{h3}_{\mathsf{K}_i}(x_1)$
    • $\mathsf{IK}_i \leftarrow \mathsf{h4}_{\mathsf{K}_i}(x_1)$
    • $\mathsf{MS.pc}_i \leftarrow 3$
    • return $\mathsf{RES}$ //*User Authentication Response*
– if $\mathsf{MS.pc}_i = 3$ and $x = (m_S, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_S)||\mathsf{pc}$ //receive *Security Mode Command*
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • if $\mathsf{MAC\text{-}I}_S \neq \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, m_S, 1, \mathsf{FRESH})$ then abort
    • control security capability
    • let $m_i$ be *Security Mode Complete* message
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • $\mathsf{MAC\text{-}I}_i \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, m_i, 0, \mathsf{FRESH})$
    • if $\mathsf{pc} = 4$, 5 or 6, $\mathsf{MS.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return $(m_i, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_i)$ //*Security Mode Complete*
– if $\mathsf{MS.pc}_i = 4$ and $x = (c_{\mathsf{TMSI}}, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_S)||\mathsf{pc}$ //receive TMSI allocation command
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • if $\mathsf{MAC\text{-}I} \neq \mathsf{h9}_{\mathsf{IK}}(\mathsf{COUNTER\text{-}I}, c_{\mathsf{TMSI}}, 1, \mathsf{FRESH})$ then abort
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 1, |m|)$
    • $\mathsf{TMSI}_i \leftarrow \mathsf{KEYSTREAM} \oplus c_{\mathsf{TMSI}}$
    • $\mathbf{ID}_i \leftarrow \mathsf{TMSI}_{i_n}$
    • $\mathbf{Revealed}_i \leftarrow \mathsf{false}$
    • let $\mathsf{ack}$ be TMSI *Allocation Complete* acknowledgment
    • $\mathsf{MAC\text{-}I}_i \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, \mathsf{ack}, 0, \mathsf{FRESH})$
    • if $\mathsf{pc} = 5$ or 6, $\mathsf{MS.pc}_i \leftarrow \mathsf{pc}$
    • return $(\mathsf{ack}, \mathsf{MAC\text{-}I}_i)$ //TMSI *Allocation Complete*
– if $\mathsf{MS.pc}_i = 5$ and $x = m||\mathsf{pc}$ //start *Data Transmission*
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 0, |m|)$
    • $c_i \leftarrow \mathsf{KEYSTREAM} \oplus m$
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • $\mathsf{FRESH} \leftarrow \mathsf{FRESH.Gen}(k)$
    • $\mathsf{MAC\text{-}I}_i \leftarrow \mathsf{h9}_{\mathsf{IK}_i}(\mathsf{COUNTER\text{-}I}, c_i, 0, \mathsf{FRESH})$
    • if $\mathsf{pc} = 4$ or 6, $\mathsf{MS.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return $(c_i, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_i)$
– if $\mathsf{MS.pc}_i = 6$ and $x = (c_S, \mathsf{FRESH}, \mathsf{MAC\text{-}I}_i)||\mathsf{pc}$ //receive transmitted message
    • $\mathsf{COUNTER\text{-}I} \leftarrow \mathsf{COUNTER\text{-}I.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}I})$
    • if $\mathsf{MAC\text{-}I}_S \neq \mathsf{h9}_{\mathsf{IK}}(\mathsf{COUNTER\text{-}I}, c_S, 1, \mathsf{FRESH})$ then abort
    • $\mathsf{COUNTER\text{-}C} \leftarrow \mathsf{COUNTER\text{-}C.Gen}(\mathsf{START}_i)$
    • $\mathsf{START}_i \leftarrow \mathsf{START.Update}(\mathsf{START}_i, \mathsf{COUNTER\text{-}C})$
    • $\mathsf{KEYSTREAM} \leftarrow \mathsf{h8}_{\mathsf{CK}_i}(\mathsf{COUNTER\text{-}C}, \mathsf{BEARER}, 1, |m|)$
    • $m_S \leftarrow \mathsf{KEYSTREAM} \oplus c_S$
    • if $\mathsf{pc} = 4$ or 5, $\mathsf{MS.pc}_i \leftarrow \mathsf{pc}$, else abort
    • return "OK"
– else abort
```

Fig. 8. ms function for MS oracle