An Enhanced Symmetric-key Based 5G-AKA Protocol

Jorge Munilla[a,*], Mike Burmester[b], Raquel Barco[a]

[a] *E.T.S.I.Telecomuicación. Universidad de Málaga, 29071, Málaga, Spain.*
[b] *Dept. of Computer Science, Florida State University, Tallahassee, FL 32306-4530, USA.*

Abstract

5G technology is called to support the next generation of wireless communications and realize the "Internet of Everything" through its mMTC (massive Machine-Type-Communications) service. The recently standardized 5G-AKA protocol is intended to deal with security and privacy issues detected in earlier generations. Nevertheless, several 5G-AKA shortcomings have been reported, including a possibly excessive computational complexity for many IoT devices. To address these, a promising lightweight 2-pass authentication and key agreement (AKA) protocol for 5G mobile communications has recently been proposed by Braeken. Compared to the 5G-AKA protocol, this does not require the use of public key encryption. This paper analyzes the security claims of Braeken's protocol and shows that it does not provide full unlinkability, but only *session* unlinkability, and is (still) subject to Linkability of AKA Failure Messages (LFM) attacks. We propose solutions to such problems and prove that symmetric-key based protocols cannot offer higher privacy protection levels without compromising availability. We then describe an enhanced version of this protocol with modifications that address these vulnerabilities and support forward secrecy, which is a desirable feature for low-cost IoT devices.

Keywords: 5G AKA protocol, symmetric-key, privacy, unlinkability, IoT, forward secrecy.

*Corresponding author
Email addresses: munilla@ic.uma.es (Jorge Munilla), burmester@cs.fsu.edu (Mike Burmester), rbm@ic.uma.es (Raquel Barco)

1. Introduction

Recent advances in wireless and mobile technologies have led to massive growth in mobile services. By the end of 2019, more than 5.2 billion people were subscribed to mobile services, accounting for 67% of the global population. Although currently the 4th Generation (4G) mobile technology is the dominant technology, the evolving 5th Generation (5G) technology is gaining pace and is expected to account for over 20% of global connections by 2025 [1]. The Internet of Things (IoT) networks will be an integral part of the 5G evolution through the new mMTC (massive Machine Type Communications) service, which improves the existing NB-IoT (Narrow Band-IoT) and LTE-M (Long Term Evolution Cat-M1) services introduced in 2015. With the number of global IoT connections expected to more than double, to almost 25 billion, between 2019 and 2025, mMTC is intended to support connection densities of up to one million devices/km^2 and ultra-low cost devices, with ultra-low cost operation and maintenance (battery life of 10-15 years) [2].

Securing mobile technologies is a major challenge. Privacy issues detected in earlier mobile network generations have increased the distrust in this technology and security has been revealed to be a crucial issue that may derail or, at least, delay large-scale deployment. The 3GPP consortium (3rd Generation Partnership Project [3]), which designed the 3G and 4G standards and is now involved in the development of 5G, has already defined a security architecture for 5G systems [4]. Security and privacy are mainly guaranteed by the Authentication and Key Agreement (AKA) protocols: 5G-AKA and EAP-AKA, which mutually authenticate subscribers and operator networks. These protocols have been revised and standardized to improve protection against prior privacy attacks and, in particular, the well-known "IMSI (International Mobile Subscriber Identity) catcher" attacks [5, 6], which compromised the privacy of subscribers by exploiting the fact that their identities were not protected during transmission. Thus, the main novelty of these AKA protocols is the inclusion of randomized public key encryption to protect the subscribers' identity. Unfor-

tunately, despite these changes, it has been shown that these protocols remain subject to different privacy attacks [7, 8]. As a consequence, variants of the 5G-AKA protocols have been proposed in the literature by Koutsos [7] and Braeken et al. [9].

These variants still rely on a public key encryption, which could be an issue for ultra-low cost IoT devices, so that, recently, Braeken published an efficient and promising lightweight 2-pass AKA protocol for 5G networks [10]. This is symmetric-key based and uses the exclusive OR (XOR) operation for obfuscation and a cryptographic hash function for authentication. It is claimed to be resistant to all known attacks in the literature and offers anonymity, unlinkability, mutual authentication and confidentiality.

This paper analyzes the security of the Braeken AKA protocol and enhances/refines some of its security claims. In particular, we show that it is still vulnerable to location confidentiality attacks that exploit the Linkability of Failure Messages (LFM) and that it only provides *session* unlinkability [11], and not *full* unlinkability. We then propose solutions for these problems and show that one may have to compromise availability to achieve full unlinkability. We prove that this is not the consequence of a faulty design but the result of using symmetric-key protection. There is a fundamental trade-off between privacy (unlinkability) and availability [12] that can only be circumvented by using asymmetric protection.

Finally, we present an enhanced version of Braeken's protocol that overcomes the described weaknesses and supports forward-secrecy. This security feature, which is not provided by the current version of the protocol, guarantees that session keys are protected if, in the future, long-term keys get compromised. This is particularly important for low-cost IoT devices that may be deployed in unsupervised sites and whose low cost does not guarantee top-level anti-tamper protection.

Thus, the main contributions of this paper are: (a) the analysis of the Braeken lightweight 2-pass AKA protocol for 5G networks, pointing out its limitations and weaknesses, (b) the required modifications to address such weak-

nesses, and (c) an enhanced version of this protocol that includes these modifications and provides forward secrecy. With these goals, the outline of the paper is as follows: Section 2 reviews the 5G-AKA protocol and the main 5G modules involved in the authentication procedure. Section 3 describes in detail the Braeken 2-pass symmetric-key AKA protocol. Section 4 analyzes this protocol focusing on privacy issues, and Section 5 proposes an enhanced version of the protocol that is resistant to forward-secrecy attacks and discusses the modifications introduced. Finally, Section 6 concludes the paper.

2. Security Architecture and Procedures for 5G

The security architecture and procedures for 5G wireless systems are defined in the technical specification 3GPP TS 33.501 [4]. This section summarizes the main aspects extracted from the specifications and related documents.

2.1. Architecture

Figure 1 illustrates the 5G communication architecture, with three main parties: the User Equipment (UE), Serving Network (SN) and Home Network (HN), indicating the most relevant functions and modules (from a security point of view) that can be found in each of them.

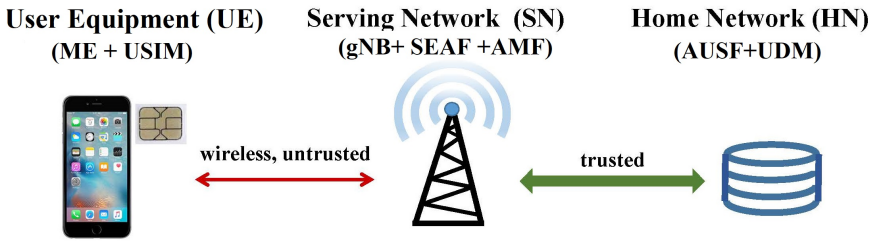

Figure 1: 5G communication architecture.

The UE contains the Mobile Equipment (ME) of the subscriber, typically a smartphone or an IoT device equipped with a Universal Subscriber Identity Module (USIM) that has cryptographic capabilities and stores the subscriber's

credentials. Plastic SIM cards have been used traditionally but their cost, which includes a SIM card reader and the difficulty associated with their substitution, make them unsuitable for IoT devices. In such cases, embedded SIMs (eSIM), implemented on chips where the credentials are provisioned remotely, are preferred. HN belongs to the subscriber's operator, manages subscriber information at the UDM (Unified Data Management) and handles authentication requests at the AUSF (Authentication Server Function). Finally, the SN provides the UE with physical access (wireless) to the network. SN contains the base stations, called gNB (Next Generation NodeB), carries out confidentiality and integrity functions of the data exchanged with UE, using keys derived from an anchor key K_{SEAF} supplied by HN and stored in the Security Anchor Function (SEAF), and also provides connection management services such as registration, reachability and mobility, implemented in the Access and Mobility Management Function (AMF) (similar to the Mobility Management Entity, MME, in 4G). The SN may or may not belong to the same operator as HN (as in roaming). From a threat model point of view, ME uses an untrusted wireless channel to communicate with SN, while SN shares a trusted (according to Clause 5.9.3 [4]), possibly wired, channel with HN.

2.2. The 5G AKA protocol (sub-clause 6.1.3.2)

A primary authentication is compulsory for all devices regardless of the service or network access these require (agnostic access network). The purpose of this primary authentication and key agreement is to enable mutual authentication between the UE and the network and provide keying material (K_{SEAF}) that can be used between the UE and the SN. A secondary authentication, intended for services provided over 5G, such as authenticated access to corporate data, is also possible in 5G but is optional.

For the primary authentication, 3GPP proposes two authentication and key agreement protocols: EAP-AKA and 5G-AKA. Here, we describe 5G-AKA since it is the most recent and used as a guideline by Braeken. EAP-AKA is quite similar, changing some exchanged messages and the key derivation slightly. For

more details, we refer the reader to [13] and Subclause 6.1.3.1 of [4].

The 5G-AKA protocol employs the exclusive-or operation "$\oplus$" for one-time pad encryption, a public key encryption function $enc_{pk_{HN}}(\cdot)$, with public key pk_{HN} and secret key stored in HN, a key derivation function (KDF), based on SHA256 [14, 15], and seven one-way keyed authentication and key generation functions: f^1, f^2, f^3, f^4, f^5, and f^{1*}, f^{5*}. The standard does not specify an implementation for these keyed functions but only security requirements. In particular, they should be cryptographically secure and mutually independent. That is, without knowing the key, their outputs should be practically *indistinguishable* from independent random functions, and it should be infeasible to determine any part of the key, or the operator variant configuration field, by manipulating their inputs and examining their straightforward or combined outputs. An example set of authentication and key generation functions, called the MILENAGE algorithms [16], has been developed by the 3GPP partners.

The 5G-AKA protocol involves the three entities: UE, SN and HN, with different elements within them, as explained, carrying out separate tasks. However, to simplify the description of our protocol, since SN and HN are assumed to be trusted and securely connected, we model them as a single entity, with SN being part of HN — see Figure 2. The USIM (at UE) stores privately: a Subscription Permanent Identifier ($SUPI$), which uniquely identifies UE, a long-term secret key k, which is different for each subscriber, the public key pk_{HN} of HN, and a sequence number SQN, used to synchronize UE and HN. These values are shared with HN, which stores them and the secret key sk_{HN} of the public key cryptosystem in a secured database. Since SQN is incremented separately on UE and HN, we distinguish the stored values by SQN_{UE} and SQN_{HN}. Randomized public key encryption (with ne a random bitstring) is used to conceal the subscriber identity: $SUCI = enc_{pk_{HN}}(SUPI, ne)$, corresponding, in practice, to the "Scheme-Output" field within the SUCI packet, which also includes information needed for routing and setting the protection scheme. Thus, the $SUPI$ is never sent in the clear to prevent the above-mentioned "IMSI catcher attack". Additionally, a Global Unique Temporary Identifier, $GUTI$, can also

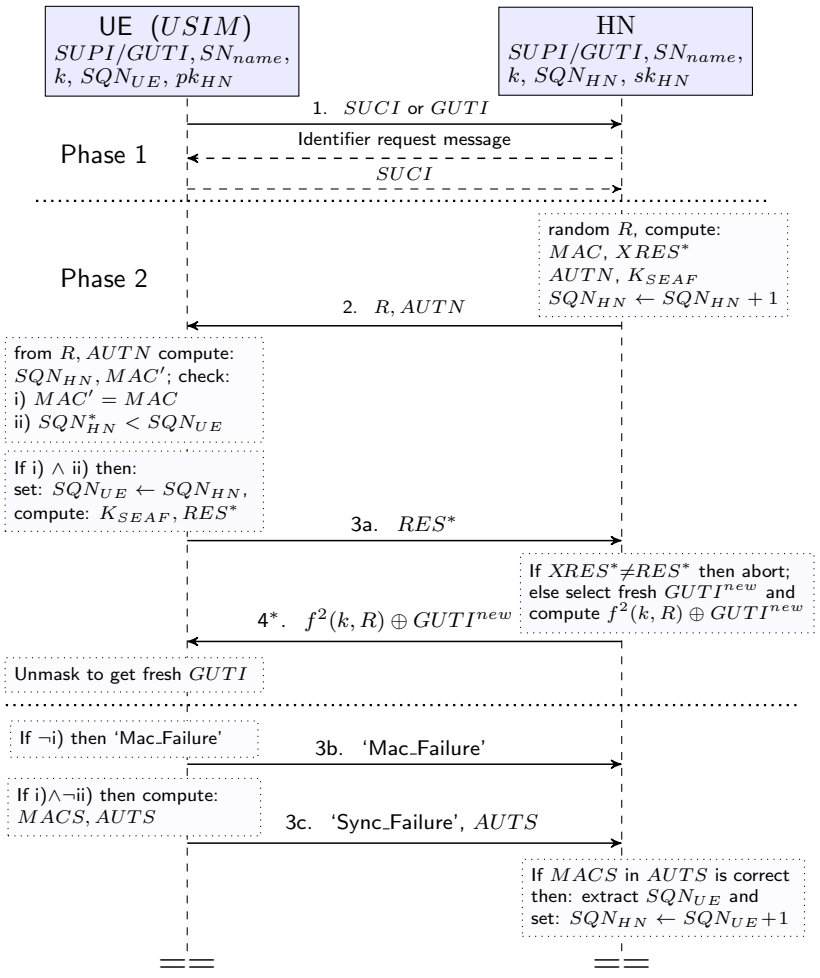

Figure 2: The 5G AKA protocol (using the descriptions in Table 1).

be used to identify the UE. This is a temporary identity assigned and sent to UE by HN after a successful authentication run, which replaces the encrypted $SUPI$, and avoids a public key encryption and a random number generation.

The flows of 5G-AKA are sketched in Figure 2, using the acronyms in Table 1, and briefly explained next ("⇒" stands for "send" and "←" for assignment):

1. *Identifier Request/Registration Request.* UE ⇒ HN : $GUTI$ or $SUCI$.

Upon receiving a $GUTI$, HN extracts the $SUPI$ from (the stored values in)

5G Acronyms	Description
$SUPI$, $GUTI$	Subscription Permanent Identifier , Globally Unique Temporary Identifier
$SUCI$	Subscription Concealed Identifier. $SUCI = enc_{pk_{HN}}(SUPI, ne)$
AMF, SN_{name}	Authentication Management Field , Serving Network Name
R, SQN_{UE}/SQN_{HN}	random challenge of HN , Sequence Number of UE/HN
$MAC, MACS$	$f^1(k, \langle SQN_{HN} \| R \| AMF \rangle), f^{*1}(k, \langle SQN_{UE} \| R \| AMF \rangle)$
CK, IK, AK/AK^*	cipher-, integrity-, authentication-key: $f^3(k,R)$, $f^4(k,R)$, $f^5(k,R)/f^{*5}(k,R)$
$AUTN, AUTS$	$\langle SQN_{HN} \oplus AK \ \| AMF \| MAC \rangle, \langle SQN_{UE} \oplus AK^* \ \| AMF \| MACS \rangle$
RES, $xRES$	actual and expected response, $f^2(k,R)$
$RES^*/XRES^*$	actual/expected derivated response, $KDF(\langle CK \| IK \rangle, \langle SN_{name} \| R \| XRES \rangle)$
K_{AUSF}	authentication server key: $KDF(\langle CK \| IK \rangle, \langle SNname \| \ SQN \oplus AK \rangle)$
K_{SEAF}	security anchor key $KDF(K_{AUSF}, SNname)$

Table 1: 5G acronyms and their descriptions.

its database. If a $SUPI$ was received, then HN decrypts $SUCI$ using its private key sk_{HN} to get $SUPI$.

2. *Challenge.* HN $\Rightarrow$ UE : $\langle R, AUTN \rangle$, where R is a fresh nonce, $AUTN = \langle SQN_{HN} \oplus AK \ \| AMF \| MAC \rangle$, with $AK = f^5(k,R)$, $MAC = f^1(k, \langle SQN_{HN} \| R \| AMF \rangle)$. HN updates the counter $SQN_{HN} \leftarrow SQN_{HN} + 1$.

   Upon receiving this message, UE computes AK, extracts SQN_{HN} from the challenge, and checks: (i.) MAC for correctness and, (ii.) SQN_{HN} for freshness: $SQN_{UE} < SQN_{HN} < SQN_{UE} + \Delta$. Δ is used to prevent desynchronization attacks by forcing the counter to wrap around.

3. *Response.* Three cases can be distinguished depending of the checks.

   a.) (i.$\wedge$ii.): UE $\Rightarrow$ HN: $RES^* = f^2(k,R)$, and UE sets $SQN_{UE} \leftarrow SQN_{HN}$.

   b.) ($\neg$i.): UE $\Rightarrow$ HN: Mac_Failure.

   c.) (i$\wedge\neg$ii.): UE $\Rightarrow$ HN: $\langle$Sync_Failure, $AUTS\rangle$, where $AUTS$ is computed as $AUTS = \langle SQN_{UE} \oplus AK^* \ \| AMF \| MACS \rangle$, with $AK^* = f^{5*}(k,R)$ and $MACS = f^{1*}(k, \langle SQN_{UE} \| R \| AMF \rangle)$. HN computes AK^*, extracts SQN_{UE}, computes $MACS$, and if correct, sets $SQN_{HN} \leftarrow SQN_{UE} + 1$.

In the first case, HN is authenticated (the MAC is correct and the challenge is fresh) and the response RES^* is sent. HN compares the received response RES^* with the expected response $XRES^*$, and if correct, the authentication protocol has ended successfully. In the second case, the authentication is aborted because the message authentication codes (MAC) cannot be verified. In the last case, a desynchronization is detected and SQN_{UE} is sent for re-syncrhonization. The authentication protocol must be repeated with these re-synchronized values.

At the end of a successful run, UE and HN share the anchor key K_{SEAF}, from which session keys for the communication between the subscriber and the HN are derived. Authentication is implicit [17], meaning that it is only effective and confirmed when the parties succeed in exchanging messages using the derived keys. The standard does not specify any additional key confirmation round, although there exists a procedure that UE and SN (AMF) has to carry out to finally establish the security context ("NAS Security Mode Command Procedure"). If the security context is successfully established, then HN can provide SN with a fresh $GUTI$ (using masked values to prevent IMSI-catching attacks).

2.3. A Brief security analysis of the 5G-AKA protocol

Although they are sometimes underspecified, as pointed out in [17], we can list the following integrity and privacy goals for 5G-AKA according to [4]:

SG1. Mutual authentication (implicit) between UE and SN, and UE and HN.

SG2. SN is authorized by HN.

SG3. Confidentiality for K_{SEAF} even if the attacker learns session keys established in other sessions (previous or consequent).

SG4. Anonymity. $SUPI$ and SQN shall remain secret in the presence of a passive attacker in order to guarantee activity privacy.

SG5. Unlinkability (user location confidentiality and user untraceability) against passive adversaries. An attacker cannot deduce the presence of a subscriber in a certain area or whether different services are delivered to the same user by eavesdropping on the radio access link.

It is not difficult to see that 5G-AKA is subject to the LFM attacks [18, 19]. This attack exploits the fact that in the event of an erroneous authentication challenge, the reason for the failure is exposed to the attacker; i.e., either a MAC_Failure or a Sync_Failure, so that the attacker can link two different sessions and identify a target user. In this attack, the adversary, after eavesdropping on a session of a target UE, acts as a fake base station (active attack) and replies the second message (authentication challenge: $R, AUTN$) to an UE: if the response of UE is Sync_Failure then the target is the same user, while if the response is MAC_Failure, then the target is some other user. This simple attack compromises subscription location, allowing, as an extension, user-traceability, although it does not contradict the SG5 goal, as this protection is only required against passive adversaries.

3. The Braeken symmetric-key based 5G AKA protocol

Braeken's protocol [10] implements a symmetric-key AKA between UE and HN (see Figure 3). It is preceded by a registration phase, where HN securely shares the parameters id, K, n with UE, with id the identity of UE, K a long-term secret key and n the sequence number (corresponding to $SUPI$, k and SQN in 5G-AKA). HN also has a master key k_m, and a temporary random key k_n. These keys are used to compute:

$$a_n = id \oplus h(k_m, k_n), \quad b_n = a_n \oplus k_m \oplus k_n, \quad c = h(k_m, id),$$

where h is a cryptographic hash function. The parameter c can be seen [10] as a certificate for the subscriber's identity and replaces pk_{HN}, while a_n, b_n, which are updated in each phase of the authentication procedure, represent the temporary identity of the subscriber (replacing the $GUTI$).

When the registration phase is completed, the parties are ready to initiate the authentication procedure. This has two operation modes: synchronized and de-synchronized. In the synchronized mode, the sequence number n is used and accepted if it is within a certain margin (as in 5G-AKA). In the de-synchronized mode, additional values are exchanged so that HN can learn the value of the sequence number that UE is using. HN will accept this value provided that it is higher than the locally stored pre-used value for that UE.

In the synchronized mode, UE sends to HN the message $[a_n, b_n, h_n]$, with $h_n = h(K, id, c, a_n, b_n, n)$, and then increments $n \leftarrow n + 1$. Otherwise (de-synchronized), UE generates a random number r_n, computes and includes $y_n = a_n \oplus id \oplus r_n$ and $z_n = n \oplus h(K, r_n, y_n)$ in the message and computes a modified $h_n = h(K, id, c, a_n, b_n, n, z_n)$.

When HN receives the message, it computes the temporary value $k_n \leftarrow a_n \oplus b_n \oplus k_m$ and finds $id \leftarrow a_n \oplus h(k_m, k_n)$ and $c \leftarrow h(k_m, id)$. Next, HN looks up in its database the secret key K and the sequence number n_{id} of the subscriber id. In the synchronized mode, HN computes $h_{n^*} \leftarrow h(K, id, c, a_n, b_n, n^*)$ with $n^* \in \{n_{id}, \ldots, n_{id} + \Delta\}$, Δ a predefined threshold, and checks if any of these matches the received value: $h_{n^*} = h_n$. In the de-synchronized mode HN retrieves $r_n \leftarrow a_n \oplus id \oplus y_n$ and $n^* \leftarrow z_n \oplus h(K, r_n, y_n)$ and computes $h_{n^*} \leftarrow h(K, id, c, a_n, b_n, n^*, z_n)$. It then checks if this value matches the received value $h_{n^*} = h_n$ and that $n^* \geq n_{id}$. If the checks are correct, the parameters are updated: $n_{id} \leftarrow n^* + 1$, two random values are drawn: k_{n+1}, f_{n+1}, with the former used to compute $a_{n+1} \leftarrow id \oplus h(k_m, k_{n+1})$ and $b_{n+1} \leftarrow a_{n+1} \oplus k_m \oplus k_{n+1}$. These values are then masked to get: $\eta \leftarrow h(f_{n+1}, c) \oplus a_{n+1}$ and $\mu \leftarrow h(c, f_{n+1}) \oplus b_{n+1}$, and the anchor key is computed as $K_{SEAF} \leftarrow h(K, f_{n+1}, \eta, \mu, n+1)$. The values $[\alpha, \beta, \eta, \mu]$ are sent, where $\alpha = c \oplus f_{n+1}$ and $\beta = h(K_{SEAF}, a_{n+1}, b_{n+1}, id, c)$. This message is received by UE, which extracts $f_{n+1} \leftarrow \alpha \oplus c$, and then computes the updated values: $a_{n+1} \leftarrow h(f_{n+1}, c) \oplus \eta$ and $b_{n+1} \leftarrow h(c, f_{n+1}) \oplus \mu$. UE also computes the anchor key $K_{SEAF} \leftarrow h(K, f_{n+1}, \eta, \mu, n+1)$ and $\beta' \leftarrow h(K_{SEAF}, a_{n+1}, b_{n+1}, id, c)$, and checks if it matches with the received value: $\beta' = \beta$. If correct, UE replaces

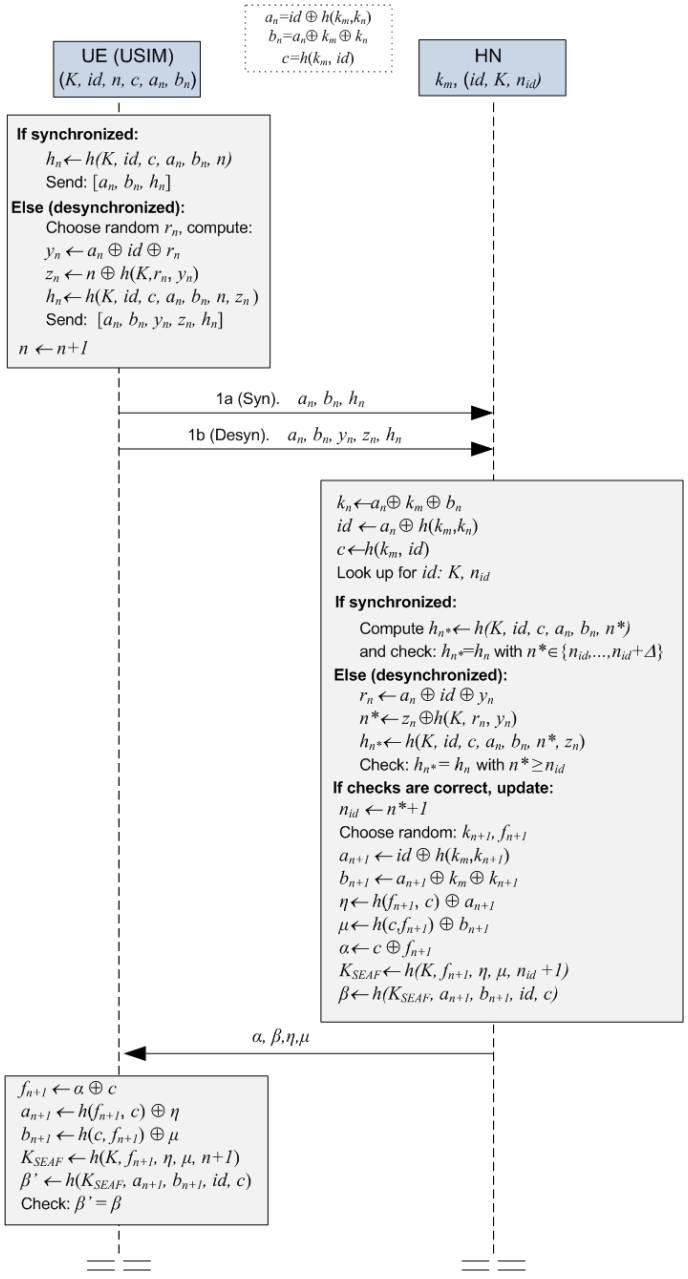

Figure 3: The Braeken AKA protocol

$[a_n, b_n]$ with $[a_{n+1}, b_{n+1}]$.

Thus, the UE updates $[a_n, b_n]$ only after a successful authentication, while n is updated after the first message is sent (even if the authentication is not

successful). For HN, n_{id} is the sole updated parameter, and only when the checks are correct.

Finally, we note that the Braeken protocol does not specify how UE enters the synchronized or de-synchronized mode. We will address this matter later.

4. Analysis of the Braeken symmetric-key based 5G AKA protocol

Braeken's protocol is claimed to offer anonymity, unlinkability, confidentiality and mutual authentication, and in particular, resistance to all known 5G AKA attacks. The security has been formally verified using RUBIN logic [20]. Compared to other AKA proposals, it is highly efficient, requiring just two flows, and relying exclusively on symmetric-key based operations, making this protocol especially suitable for IoT devices.

In this section, we shall cryptanalyze the Braeken protocol, pointing out several security aspects that could result in vulnerabilities for certain practical implementations, and discussing some of its security features; in particular, those regarding unlinkability and forward-secrecy. We shall show that Braeken's protocol only provides "session unlinkability" and is subject to LFM and forward-secrecy attacks. Forward-secrecy was not a goal of the original design still, it is a highly desirable security feature for IoT devices, which cannot afford top-level anti-tamper protection and may be deployed at unsupervised sites. An enhanced version of this protocol that implements solutions for the identified vulnerabilities and includes forward-secrecy protection is described in the next section.

4.1. Implicit authentication and replay attacks

The authentication of UE is implicit, and therefore, the protocol is subject to replay attacks. In fact, if an adversary intercepts the message (synchronized or de-synchronised) sent by UE, or gets it by using a fake base station, and re-transmits it later to HN before UE and HN have completed a new authentication protocol, then HN will accept it as valid and generate a K_{SEAF}. This

vulnerability, shared with the 5G-AKA protocol but with the implicit authentication of HN (see Section 2), does not go further because the adversary is not able to generate the session key, but in certain situations, the attack may cause functionality problems since the session remains open until a period of inactivity is detected [17]. Since this vulnerability cannot be exploited and can only be prevented by including additional communication flows, we do not propose to address it in the enhanced version.

4.2. Master key and offline brute force attacks

The Braeken protocol uses a master key (k_m) that is shared implicitly with all users. This is a potential vulnerability that may expose the entire system to an *offline* brute force attack: an adversary could register an authorized device (or compromise a genuine one), getting access to id and $c = h(k_m, id)$, and then perform an offline brute force attack to obtain the key k_m. This attack's effectiveness is inversely proportional to the length of the key so that it can be prevented by assuring that k_m is sufficiently long.

4.3. Vulnerability to LFM attacks

Because of the different formats between UE's messages in the synchronized and de-synchronized modes, Braeken's protocol shares with the 5G-AKA protocol (see Section 2.3) the LFM vulnerability. The way to address this is to make the synchronized and de-synchronized messages have the same format so that a passive adversary cannot distinguish them. For this solution, two new values A,B are included in the message exchanged in the synchronized mode: $[a_n, b_n, A, B, h_n]$, with $A = T[h(K, a_n))]$ and B a random number, that have the same bit length as y_n and z_n, respectively, where T is the truncation or padding of random bits to reach the desired length. HN must then check the value A to determine if the received message corresponds to the synchronized or de-synchronized mode.

This solution, however, to be effective, also requires: (a) to address unlinkability, which is discussed in the next section, and (b) to define the logic that

leads UE to select a particular operation mode appropriately. In fact, if this selection was based, as in 5G-AKA, on an unmasked response message from HN that specified the mode, then LFM attacks would still be possible. Thus, we alternatively propose that UE uses a counter to determine the operation mode; i.e. the de-synchronized mode is activated if there are more than Δ incomplete sessions since the last complete execution. This also avoids the inclusion of new (Failure) messages.

4.4. Privacy issues

The Braeken protocol offers 3GPP unlikability protection (Section 2.2, SG5) for subscriber location privacy under passive attacks. In this section, we discuss the limitations of this protection against active attacks.

In both the synchronized and de-synchronized modes of operation, the values a_n, b_n are repeated if the authentication is not successful. As a result, an (active) adversary that gets these values can trace UE by checking that the same values a_n, b_n are used, violating location privacy. This linking (traceability) of messages is possible until the protocol is completed and the values a_n, b_n are updated. As a result, the Braeken protocol does not provide *full unlinkability* but only a weaker version of unlinkability defined in [21, 11] as *session unlinkability*. With session unlinkability, two interrogations of a UE cannot be linked if, either the first one completed successfully, or an intermediate interrogation completed successfully.

To get full-unlinkability, these values must be randomized in each session. A proposed solution is then to replace the values a_n and b_n with randomized versions a_n^* and b_n^* that are masked as follows:

$a_n^* = a_n \oplus h(K, rn_1),\ \ b_n^* = b_n \oplus h(K, rn_2),$

with rn_1, rn_2 nonces generated by UE. Then, for every pair (id^*, K^*) in its database, HN has to compute $k_n^* = a_n^* \oplus h(K^*, rn_1) \oplus b_n^* \oplus h(K^*, rn_2) \oplus k_m$ and check if $id^* = a_n^* \oplus h(K^*, rn_1) \oplus h(k_m, k_n^*)$.

This solution achieves unlinkability but has a computational cost that could compromise availability if the number of pairs (id, K) is large. This problem

cannot be avoided when using symmetric-key protection mechanisms. If UE uses randomized symmetric-key encryption $(E_K(id, rn_1), rn_1)$ for identification, then HN has to check the received value for each pair (id, K) in its database. Alternatively, if HK shared a symmetric master key K^m with all UEs, so that these could use K^m to cipher its identities, then we would face the problem already described in Section 4.2 (without requiring brute-force computation). As shown in [12], with symmetric-key based architectures there is a trade-off between privacy and availability, and any attempt to improve privacy unavoidable impacts on the protection against DoS (Denial of Service) attacks.

For full unlinkability, UE must be able to send to HN a randomized encryption of its identifier *id*, that does not require any (fresh) input from HN (the adversary can block or forge such messages), and HN must be able to decrypt it without knowing in advance which UE sent it. If $F_{key}(id, rn)$ is this encryption, where F a pseudo-random function, then HN, and *o*nly HN, must be able to recover the identifier *id*, *for every* UE. The capability of "inverting" F corresponds to a trapdoor, that only HN should possess. The key that UE uses for the encryption cannot be used for decryption and therefore can be a public key. Consequently, F must be an asymmetric encryption function.

We conclude that session unlinkability can be considered as a constraining factor of keyed hash-based architectures (and symmetric-key based proposals in general) if an exhaustive search in the database must be avoided, which is advisable when the number of users is large. Otherwise, if full unlinkability is required, even at a higher computational cost, the proposed solution can be implemented.

Finally, we note that the 5G-AKA protocol also leaks the HN of the UE (in the clear within the SUCI packet), as this information is required for SN to route the packets correctly. This means that even using public key cryptography, avoiding any kind of linking is sometimes very hard to achieve in practice.

4.5. Forward Secrecy

Both Braeken's protocol and the 5G-AKA protocol ensure the confidentiality of K_{SEAF} even when the session keys of previous or subsequent sessions gets compromised (SG3, cf. Section 2.2). However, they do not address *forward secrecy*, which provides confidentiality assurances for the session keys when the long-term key is compromised. The importance of forward secrecy is recognized in the literature (e.g., [9]), and in particular with protocols intended for IoT devices.

As previously, we shall first describe a forward secrecy attack on the Braeken protocol and then propose a solution that is resistant to such attack. The attack of the adversary involves the following four steps:

1. Eavesdrop on the exchanged messages of a session i of Braeken's AKA protocol between the target UE and HN, and store:
   Synchronised mode: $[a_i, b_i, h_i, \alpha_i, \beta_i, \eta_i, \mu_i]$,
   De-synchronized mode: $[a_i, b_i, y_i, z_i, h_i, \alpha_i, \beta_i, \eta_i, \mu_i]$.

2. Get access to the private data on the USIM card of UE: (id, K, c, n).

3. Compute the sequence number n_i for session i:
   Synchronised mode: compute $h^* \leftarrow h(K, id, c, a_i, b_i, j)$ for $j = n-1 : -1 : 1$ until $h^* = h_i$; then $n_i \leftarrow j$.
   De-synchronized mode: $n_i \leftarrow z_n \oplus h(K, y_n \oplus a_n \oplus id, y_n)$

4. Compute the i-th session key: $K^i_{SEAF} \leftarrow h(K, \alpha_i \oplus c, \eta_i, \mu_i, n_i + 1)$.

To guarantee forward secrecy, the long-term key must be updated with each protocol execution. Consequently, in the proposed solution, the static key K is replaced with a dynamic key K_{FS} that is updated with each iteration of the protocol using the one-way hash function h: $K_{FS} \leftarrow h(K_{FS})$. The security of this solution relies on the non-invertibility of h. Thus, an adversary that gets access to K_{FS} will not be able to recover the session key of a previous communication, as the key K^i_{FS} used for its computation is different from the

currently stored key in the USIM ($K_{FS}^{j} = h^{(j-i)}(K_{FS}^{i})$ for $j > i$), and it is not possible to obtain previous values from current values, given that h is a one-way function. On the other hand, since UE authentication is implicit and subject to replay attacks, to prevent synchronization problems, HN has to keep the previous version K_{FS}^{*} until it checks that this has been correctly updated by UE. Only then, this value is updated. This also implies an extra check with this previous value when the last updated value is not valid. The complete details on the implementation of this solution are given in the next section, where the enhanced protocol is described.

5. An enhanced version of the symmetric-key AKA protocol that supports forward secrecy

5.1. Description of the modifications

Figure 4 describes the enhanced version of the protocol. The modifications are highlighted in the figure and described in the following paragraphs. The descriptions are ordered according to the convenience, in our opinion, of their implementation.

Mod 1: Forward-secrecy support by replacing the static key K with a dynamic key K_{FS}. To prevent synchronization problems, the values stored by the parties are different; HN stores the pre-updated value, K_{FS}^{*}, whereas UE stores the updated value, $K_{FS} = h(K_{FS}^{*})$.

UE side: The key K_{FS} is used to compute h_n for the first flow. If the second flow is received and the last verification is correct ($\beta' = \beta$), then K_{FS} is updated.

HN side: There are two main differences. First, the check that the received message is correct is carried out using the updated/current value ($h(K_{FS}^{*})$), and the stored/previous values (K_{FS}^{*}) of K_{FS}. Second, HN updates K_{FS} if, and only if, the checks are correct for the updated value, since this implies that UE has already updated it.

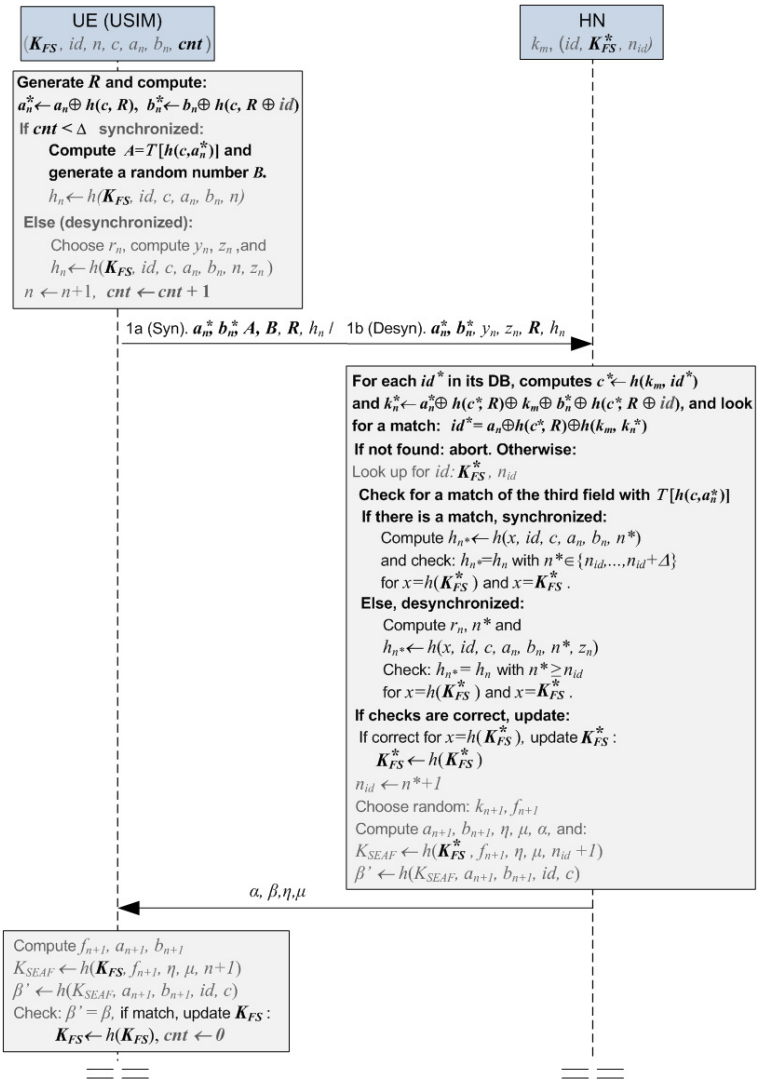

Figure 4: The enhanced Braeken AKA protocol with forward-secrecy.

Note that the solution implemented here protects previous session keys K_{SEAF} but the fact that certain parameters, such as c and id, do not change in each session makes subtler forward-secrecy attacks against privacy are still possible. More specifically, an adversary who manages to compromise a device and get access to the information stored in it, can

determine if the device was involved in a previous eavesdropped communication. Preventing this requires a more complex protocol redesign that faces the, already discussed, inherent trade-off between privacy and availability.

Mod 2: Randomization of a_n, b_n in the first flow to get full-unlikability. In contrast to the approach used in the previous section, to implement the enhanced protocol, we propose using c, instead of K, for masking. This reduces the computations when combined with forward-secrecy protection since using K_{FS} would require checking two possible masked values (the pre and the updated value). Additionally, to reduce the communication overhead, only one random number is generated and transmitted.

UE side: A new random number R is generated and used to mask the original values: $a_n^* = a_n \oplus h(c, R)$ and $b_n^* = b_n \oplus h(c, R \oplus id)$. Then, these new values together with R replace a_n and b_n in the first flow.

HN side: For every id^* in its database, HN computes $c* \leftarrow h(k_m, id^*)$ and checks for a match:

$$id^* = a_n^* \oplus h(c^*, R) \oplus h(k_m, a_n^* \oplus h(c^*, R) \oplus k_m \oplus b_n^* \oplus h(c^*, R \oplus id)).$$

If a match is not found, the message is discarded. Otherwise, the original values a_n and b_n are retrieved, and the protocol can continue as in the original protocol.

Mod 3: Equalization of message formats and inclusion of a counter to prevent LFM attacks. As in the previous case, to reduce the computation and communication cost, the value c is used, instead of K, to check the communication mode (synchronized or de-synchronized). A counter *cnt* is also included at the UE side to determine the operation mode.

UE side: Two new values, A and B, are generated with the bitlength of y_n and z_n, respectively: $A = T[h(c, a_n^*)]$ and B a random bitstring. The counter *cnt* is increased with every session and is only

reset to zero when the protocol is completed successfully (β is checked correctly).

HN side: Determine if the received message corresponds to a synchronized or de-synchronized type by checking if the third field matches $T[h(c, a_n^*)]$. If it does, HN assumes the synchronized mode. Otherwise, HN assumes the de-synchronized mode and the third field equals y_n.

5.2. Security Analysis

Threat Model. We use the standard model for semantic security in which all parties, including adversaries, are represented by probabilistic polynomial-time (PPT) Turing machines (algorithms) [22]. Adversaries control the network and can eavesdrop on the communication (passive adversaries), as well as intercept, inject, manipulate or drop messages (active adversaries). Security is based on indistinguishability. In particular, an adversary cannot distinguish a pseudo-random bitstring from a uniformly random bitstring of the same length s with non-negligible probability, where negligible means less than $1/p(s)$, p any polynomial.

The Braeken protocol uses a keyed hash function $h(K, x) : \{0,1\}^s \times \{0,1\}^* \rightarrow \{0,1\}^n$, $n = p(s)$, as a *pseudorandom* function (PRF). The input to h is a uniformly distributed key $K \in \{0,1\}^s$, and a bitstring x of arbitrary length. We call s the *security parameter*, and a computation (algorithm) efficient, or feasible, if it runs in polynomial time in s. We shall assume that: (a) the value $h(K, x)$ can be efficiently computed given K, x, and (b) the function $h(K, \cdot)$ cannot be efficiently distinguished from a uniformly random function $R : \{0,1\}^* \rightarrow \{0,1\}^n$, given access to pairs $(x_i, h(K, x_i))$, where x_i can be adaptively selected by the distinguisher.

We shall also use h as a *cryptographic* hash function $h(x) : \{0,1\}^* \rightarrow \{0,1\}^n$. Such functions: (a) can be computed efficiently, (b) are *one-way*: that is given $y \in \{0,1\}^n$ it is infeasible to find $x \in \{0,1\}^*$ such that $y = h(x)$, and (c) are *collision-resistant*: that is it is infeasible to find inputs $x, x' \in \{0,1\}^*$ with $x \neq x'$ such that $h(x) = h(x')$.

We shall first prove the unlinkability of the enhanced Braeken AKA Protocol, based on the pseudorandomness of the exchanged messages.

Theorem 1. *An adversary $\mathcal{A}$ cannot distinguish the messages exchanged during the enhanced Braeken AKA protocol from uniformly random messages with probability better than negligible (in terms of the length of the key K).*

Proof. We first prove the result for the first flow (Lemma 1) and then for the second flow (Lemma 2). □

Lemma 1. *An adversary $\mathcal{A}$ cannot distinguish the messages exchanged in the first flow from uniformly random messages with probability better than negligible.*

Proof. In the first flow the messages $\langle a_n^*, b_n^*, A, B, y_n, z_n, R, h_n \rangle$ are exchanged. We must show that these are pseudorandom and (mutually) independent. R, B are uniformly random numbers. For the remaining messages, pseudorandomness and independence are guaranteed for each session provided that the keyed hash functions:

$h(c, R)$	for	a_n^*
$h(c, R \oplus id)$	for	b_n^*
$T[h(c, a_n^*)]$	for	A
$h(K, r_n, y_n)$	for	z_n
$h(K, id, c, a_n, b_n, n)$	for	h_n(synchronized)
$h(K, id, c, a_n, b_n, n, z_n)$	for	h_n(desynchronized)

have fresh inputs and are pseudorandom. We get freshness because the same input is not repeated within the same session and at least one of the terms: R, $R \oplus id$, a_n^*, r_n, n and n, respectively, is refreshed. We get pseudorandomness because in each input there is at least one value: c, c, c, K, K and K, respectively, that serves as a key (not known to $\mathcal{A}$). For a_n^*, b_n^* we note that pseudorandomness and freshness is preserved when XORing with a fixed number. The pseudorandomness and freshness of y_n comes from the uniformly random number r_n. It follows that the messages exchanged in the first flow are pseudorandom and independent. □

Lemma 2. *An adversary $\mathcal{A}$ cannot distinguish the messages exchanged in the second flow from uniformly random messages with probability better than negligible.*

Proof. This is similar. In the second flow $\langle \alpha, \beta, \eta, \mu \rangle$ are exchanged. The pseudorandomness and freshness of $\alpha = c \oplus f_{n+1}$ comes from the uniformly random number f_{n+1}. The pseudorandomness and freshness of

$$\beta = h(K_{SEAF}, a_{n+1}, b_{n+1}, id, c),\ \eta = h(f_{n+1}, c) \text{ and } \mu = h(c, f_{n+1}) \oplus b_{n+1},$$

is a consequence of using the (secret) key values K_{SEAF}, c and c, respectively, and the fresh random numbers a_{n+1}, f_{n+1} and f_{n+1} respectively, as inputs to the keyed hash function. □

Corollary 1. *An adversary $\mathcal{A}$ cannot link the messages of the same UE for two different sessions of the enhanced Braeken AKA protocol, as it is infeasible for $\mathcal{A}$ to distinguish such messages from uniformly random messages with probability better than negligible.*

We next prove the security of the protocol against forward-secrecy attacks.

Theorem 2. *An adversary $\mathcal{A}$ with access to the information stored on the tag and the messages from previously exchanged flows, cannot compute the previous session keys with probability better than negligible.*

Proof. By contradiction. Assume $\mathcal{A}$ can compute the session key of a previous session i: K^i_{SEAF}, using the messages exchanged in that session and information stored on the tag. This means that $\mathcal{A}$ can either predict the output K^i_{SEAF} of a pseudorandom function without knowing the key K^i_{FS}, or compute the key K^i_{FS} given the information stored on the tag with probability better than negligible. The former contradicts the pseudorandomness of the function $K^i_{SEAF} = h(K^i_{FS}, \cdot)$, while the latter contradicts the one-wayness of the hash function h, since if the dynamic key stored on the tag is K^j_{SF}, $j > i$, then we have: $K^j_{SF} = h^{j-i}(K^i_{SF})$. □

5.3. Protocol Discussion

Three main modifications have been described in this section. However, it may not be advisable to implement all of them. As security experts know, security features are rarely realized "for free" and typically incur a cost in terms of hardware, communication and/or computational load. Consequently, the final design criteria and security features must depend on the specific application, the efficiency requirements, and the required security level. Considering that each case must be analyzed separately, we briefly review the proposed modifications and provide brief guidelines for their implementation.

1. *Original Protocol.* The protocol proposed by Braeken, taking into account the limitations of symmetric-key cryptography, obtains a good balance between privacy and efficiency. Designers must nevertheless be aware of its security limitations, which have been discussed in this paper. Additionally, for practical implementations, the logic to select the operation mode must be addressed. If no extra security features are going to be included, this could be implemented by using Failures Messages sent by HN (as in the 5G-AKA), or using counters as discussed in the enhanced version of the protocol. The second option is necessary if protection against LFM attack is required.

2. *Original protocol + Forward Secrecy.* This is the optimal balance between security and efficiency for lightweight applications. IoT devices are particularly vulnerable to forward secrecy attacks, and consequently, this security feature is remarkably advisable. It does not have any extra communication cost and the computational cost is negligible.

3. *Original protocol + Forward Secrecy + Full Unlinkability.* As proved in this paper, the current version of the Braeken protocol does not provide full unlinkability, but just session unlinkability. The modifications needed for full unlinkability and protection against LFM attacks should be implemented jointly, since each one of these features impacts on the other.

However, the cost for full unlinkability is high and could compromise availability, making the protocol vulnerable to DoS attacks (trade-off between privacy and availability). This vulnerability is proportional to the size of the database in HN (number of registered users), so that we only recommend this modification if full unlinkability is an essential feature for the specific implementation and the database is not large.

6. Conclusions

The solution proposed by 3GPP for 5G wireless networks regarding subscriber privacy relies on the use of a public key (of HN) that allows UE to encrypt the subscriber's identity. A detailed analysis of the recently proposed Braeken protocol has shown that this protocol cannot provide the same privacy protection level as the 5G-AKA protocol because of its use of symmetric-key protection. The Braeken protocol is also shown to be subject to LFM attacks.

We have shown that it is possible to achieve full unlinkability and overcome LFM attacks with symmetric-key protection but at the cost of a trade-off between privacy and availability, which could make the protocol vulnerable to DoS attacks. However, having slightly lower subscriber privacy protection is compensated by a much more efficient protocol, particularly considering IoT devices. 5G wireless technology aims to support future IoT communication with its mMTC service. IoT devices store secret information in their SIMs, which are supposed to be tamper-resistant. Nevertheless, such information can be compromised, particularly with IoT applications that use low cost cards with weak protection. In such cases, protection against forward secrecy is advisable.

We have proposed an enhanced version of the Braeken protocol with modifications that address forward secrecy, by updating the shared key after each successful authentication, as well as modifications that address the previous mentioned security requirements: unlinkability, protection against LFM attacks and a logic for the selection mode. These modifications are flexible and may be applied selectively. We have also discussed the advantages and disadvantages of

the proposed modifications from a system design perspective that considers the essential features of a specific implementation, including the cost for the service provided, the efficiency and the security.

Funding

This work was supported in part by FEDER funds (Junta de Andalucía-University of Málaga) under Project UMA18- FEDERJA-172, and in part by NSF under Grant DUE 1241525.

References

[1] GSMA the mobile economy, https://www.gsma.com/mobileeconomy/wp-content/uploads/2020/03/GSMA_MobileEconomy2020_Global.pdf, accessed: 2020-09-30.

[2] 3GPP, Study on Scenarios and Requirements for Next Generation Access Technologies, (3GPP), TR 38.913. Available Online (last access October 2020): https://www.3gpp.org/ftp//Specs/archive/38_series/38.913/.

[3] 3rd Generation Partnership Project (3GPP), https://www.3gpp.org/, accessed: 2020-09-30.

[4] 3GPP, Security architecture and procedures for 5G system, (3GPP), TS 33.501. Available Online (last access September 2020): https://www.3gpp.org/ftp/Specs/archive/33_series/33.501/.

[5] A. Shaik, R. Borgaonkar, J.-P. Seifert, N. Asokan, V. Niemi, Practical attacks against privacy and availability in 4g/lte, in: 23nd Annual Network and Distributed System Security Symposium, NDSS 2016, San Diego, California, USA, February 21-24, 2016, 2016.
URL http://www.internetsociety.org/events/ndss-symposium-2016

[6] D. Fox, Der imsi-catcher, Datenschutz und Datensicherheit 26 (4) (2002) 212–215.

[7] A. Koutsos, The 5g-aka authentication protocol privacy, CoRR abs/1811.06922 (2018). arXiv:1811.06922.
URL http://arxiv.org/abs/1811.06922

[8] D. Basin, S. Radomirovic, J. Dreier, R. Sasse, L. Hirschi, V. Stettler, A formal analysis of 5g authentication, 2018, pp. 1383–1396, cited By 43. doi:10.1145/3243734.3243846.
URL https://www.scopus.com/inward/record.uri?eid=2-s2.0-85056873194&doi=10.1145%2f3243734.3243846&partnerID=40&md5=f09fb3d827820f654dd0736aa34caa93

[9] A. Braeken, M. Liyanage, P. Kumar, J. Murphy, Novel 5g authentication protocol to improve the resistance against active attacks and malicious serving networks, IEEE Access 7 (2019) 64040–64052.

[10] A. Braeken, Symmetric key based 5g aka authentication protocol satisfying anonymity and unlinkability, Computer Networks 181 (2020) 107424.

[11] M. Burmester, J. Munilla, Lightweight RFID authentication With Forward and Backward Security, ACM Trans. Inf. Syst. Secur. 14 (1) (2011) 11.

[12] M. Burmester, J. Munilla, Pre vs post state update: Trading privacy for availability in RFID, IEEE Wirel. Commun. Lett. 3 (3) (2014) 317–320. doi:10.1109/WCL.2014.032814.140043.
URL https://doi.org/10.1109/WCL.2014.032814.140043

[13] V. Arkko, J.and Lehtovirta, P. Eronen, RFC 5448:Improved Extensible Authentication Protocol Method for 3rd Generation Authentication and Key Agreement (EAP-AKA), 2009.

[14] 3GPP, Generic Authentication Architecture (GAA); Generic Bootstrapping Architecture (GBA), (3GPP), TS 33.220. Available Online (last ac-

cess September 2020): https://www.3gpp.org/ftp/Specs/archive/33_series/33.220/.

[15] IETF, RFC 2104: "HMAC: Keyed-Hashing for Message Authentication".

[16] 3GPP, 3G Security; Specification of the MILENAGE algorithm set: an example algorithm set for the 3GPP authentication and key generation functions f1, f1*, f2, f3, f4, f5 and f5*; Document 5: Summary and results of design and evaluation, (3GPP), TR 35.909. Available Online (last access September 2020): https://www.3gpp.org/ftp/Specs/archive/35_series/35.909/.

[17] D. Basin, J. Dreier, L. Hirschi, S. Radomirovic, R. Sasse, V. Stettler, A formal analysis of 5g authentication, in: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security, 2018, pp. 1383–1396.

[18] H. Khan, K. M. Martin, A survey of subscription privacy on the 5g radio interface - the past, present and future, IACR Cryptol. ePrint Arch. 2020 (2020) 101.
URL https://eprint.iacr.org/2020/101

[19] M. Arapinis, L. Mancini, E. Ritter, M. Ryan, N. Golde, K. Redon, R. Borgaonkar, New privacy issues in mobile telephony: fix and verification, in: Proceedings of the 2012 ACM conference on Computer and communications security, 2012, pp. 205–216.

[20] A. D. Rubin, P. Honeyman, Nonmonotonic cryptographic protocols, in: Proceedings The Computer Security Foundations Workshop VII, 1994, pp. 100–116.

[21] M. Burmester, J. Munilla, A Flyweight RFID Authentication Protocol, in: Workshop on RFID Security – RFIDSec'09, Leuven, Belgium, 2009.

[22] S. Goldwasser, S. Micali, Probabilistic encryption & how to play mental poker keeping secret all partial information, in: Proceedings of the Four-

teenth Annual ACM Symposium on Theory of Computing, STOC '82, Association for Computing Machinery, New York, NY, USA, 1982, p. 365–377. doi:10.1145/800070.802212.
URL https://doi.org/10.1145/800070.802212