Attacks against subscriber privacy on the LTE Authentication Protocol (EPS-AKA)

Monder Rammouz
School of Computation, Information and Technology (CIT)
Technical University of Munich
Munich, Germany
monder.rammouz@tum.de

***Abstract*—Cellular networks are, as of today, an ubiquitous aspect of daily life. Currently the latest globally deployed mobile standard, 4G/LTE (Long Term Evolution) is a significant advancement compared to previous generations, 2G/GSM and 3G/UMTS, in terms of security. It utilizes a so-called EPS-AKA protocol, mandated by the 3rd Generation Partnership Project, in order to protect the security and privacy of its subscribers. However, this currently deployed protocol presents many flaws in its specification and was subject to many attacks targeting subscribers' location and activity privacy in the past. In this summary analysis, we examine these vulnerabilities and describes how the attacks take place. We first provide an overview of the LTE architecture and explain how the EPS-AKA protocol operates. We then uncover attack vectors and explain how each one can be exploited to mount practical attacks. Finally, we present countermeasures to address each of these privacy issues.**

I. Introduction:

Given the rising significance of cellular network services (e.g. Internet, calls) as an ubiquitous aspect of daily life, with a number of mobile subscriptions of 8.6 billion in 2021, it is critical to ensure subscribers' privacy. The deployment of 4G/LTE (Long Term Evolution) networks, bringing improved functionality, security, and privacy for the mobile subscribers, was a significant advancement compared to previous generations, namely 2G/GSM and 3G/UMTS.

It utilizes what is called the Authentication and Key Agreement (AKA) protocol, established by the 3GPP organization (3rd Generation Partnership Project) in charge of standardizing technologies for mobile telecommunications. It attempts to mutually authenticate mobile subscribers having Universal Subscriber Identity Module cards (USIM) with 4G/LTE networks and establish cryptographic keys to secure ensuing communications. We delimit AKA as a challenge-response mutual authentication and key agreement protocol. With each new generation, 3GPP has improved AKA, with in total eight versions from 2G to 5G. To give a brief history, 2G/GSM was known to have a number of vulnerabilities, such as the lack of mutual authentication between mobile subscribers and the network, which allowed attackers to set up fake base stations and trick legitimate mobile devices into connecting to them. The next generation specification, 3G/UMTS, introduced mutual authentication protocols (e.g. Milenage known as the first version of AKA, XOR-AKA, H3G-AKA and TUAK) and more robust cryptographic algorithms. The 4G/LTE with the EPS-AKA specification further strengthened subscribers' security and privacy by mandating authentication and encryption in more situations. In this paper, we focus on the EPS-AKA. Indeed, although subscriber privacy is explicitly required for 4G, many attacks that exploit flaws in the EPS-AKA protocol specification have been shown previously.

In order to understand these flaws, we first and foremost present a general overview of the 4G/LTE architecture and closely examine the EPS-AKA protocol specification in Section 1. The background knowledge about the architecture of the network will be necessary for the reader to understand some practical aspects handled in the next section. In Section 2, we uncover several vulnerabilities found in the analysis of the protocol specifications. We discuss the underlying reasons for their existence, and we present the potential threats to subscribers' privacy that they represent. Furthermore, we explain how each one of these attack vectors can be exploited to mount practical attacks. We also attempt to realize the first attack using a commercially available low-cost setup. Lastly, we discuss countermeasures to address each of these privacy issues.

Contributions:

- A summarization of the main flaws affecting subscribers' privacy: Wireless practitioners frequently inquire about the reasons behind the adoption of new authentication mechanisms in the newly specified 5G-AKA protocol. This paper first aims at summarizing the main flaws in the currently globally deployed 4G EPS-AKA, to better understand newly adopted protocol design choices.
- A practical description of each attack showing its feasibility: On one hand, we briefly describe the EPS architecture and the core AKA protocol, we explain the main attack vectors and how to leverage them to break subscribers' privacy. On the other hand, we show how our attacks are feasible against widely deployed 4G devices with a low-cost setup.

II. Background

A. An overview of the EPS architecture

Defined in 3GPP Release 8 (December 2008), the Evolved Packet System or EPS designates the architecture of 4G LTE

(Long-Term Evolution) cellular networks. This architecture is divided into two main parts: the Evolved Packet Core or EPC also known as the core network and the Evolved Universal Terrestrial Radio Access Network or E-UTRAN, which serves as the access network.

EPS was designed to improve upon the previous generation networks, such as the Global System for Mobile Communications or GSM and the Universal Mobile Telecommunications System or UMTS. It uses packet-switched data communication in its core network, which allows a more reliable, efficient and flexible data transfer as multiple devices share multiple communication channels, making it fundamentally different from the older GSM and UMTS, which use circuit-switched data communication requiring a dedicated connection for each data transfer. Besides, EPS achieves a higher data transfer rate (100 Mbps for download and 50 Mbps for upload), supporting high-bandwidth applications (e.g. video streaming).

As figure 1 shows, the EPS consists of several nodes:

- User Equipment or UE, carried by the subscriber, typically a smartphone with a Universal Integrated Circuit Card (UICC) hosting at least one Universal Subscriber Identity Module (USIM) application (in short, a USIM card).
- Evolved Node B or ENodeB, a radio access equipment providing the UE with a connection to MMEs and S-GWs. It utilizes Access Stratum (AS) protocols to communicate with its UEs through signaling messages (e.g. Radio Resource Control protocol messages).
- The Mobility Management Entity or MME, a control-plane node of the EPC used for establishing sessions (Session management), maintaining continuous communications with the subscribers as they move (Mobility management) and most importantly, performing mutual authentication with the UEs (Identity management). It utilizes Non-Access Stratum (AS) protocols to communicate with its UEs.
- The Home Subscriber Service or HSS: a database of subscribers information for the purpose of authentication.
- The Serving Gateway or S-GW, a user-plane node that routes the data traffic between the UE and the EPC.
- The Packet Data Network Gateway or P-GW, an interface between the EPC and external packet data networks, such as the Internet.

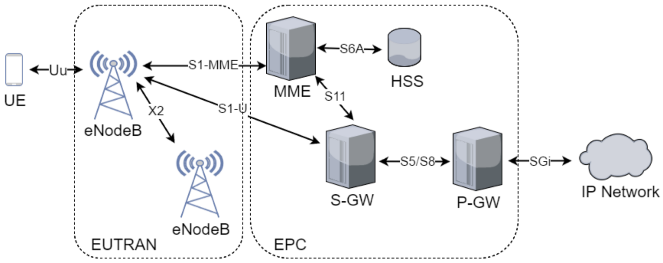


Fig. 1. Overview of the LTE architecture or EPS

When a subscriber equipped with UE (we use both terms interchangeably) requests a service such as a call or sending an SMS, he must first attach to the network. The UE first performs an AS Attach procedure. It includes a Cell Search and a Random Access Procedure to identify a suitable cell (the area covered by a eNodeB) and to synchronize with it at the physical layer [4]. The UE scans all the available LTE frequency bands and detects a Physical Broadcast Control Channel (PBCH). The PBCH unidirectionally transmits the system information of the cell, such as the Physical Cell ID (PCI) and the supported frequency bandwidth, which allows the UE to synchronize itself with the cell. The Radio Resource Control (RRC) protocol is then employed to ensure that the UE is always connected and is communicating with the best eNodeB available. The TAU Procedure is used as a part of the RRC protocol to maintain an accurate record of the UE's location in the network's databases, in order for the MME to manage the mobility of the UE. The NAS (Non-Access Stratum) Attach Procedure can start: the UE sends an initial Attach Request message to the MME via the eNodeB. And the MME then uses the UE's International Mobile Subscriber Identity (IMSI) to look up the corresponding subscribers' credentials in its HSS. A challenge-response based mutual authentication mechanism called the AKA protocol is then performed to authenticate the UE with the MME (and vice-versa) and configure security attributes (one-time secret keys) for future encryption. If the authentication is successful, so-called EPS bearer can then be established, it transports data packets between the UE and the P-GW over the S-GW. The data traffic is encrypted using keys derived by the AKA protocol. The UE can now send and receive data, make calls, send text messages, access the internet, etc. as per the operators' service plan and the quality of service associated with the EPS bearer. To release the connection, the UE sends a detach request to the MME, which finally releases the resources allocated to the UE [1].

Since the attack concepts handled in this paper revolve around the EPS-AKA protocol, we reduce the architecture into three components to better describe the protocol. First, the Home Network or HN, which hosts the HSS, the database of the subscribers' credentials and authentication attributes. Second, the UE that contains a USIM with cryptographic capabilities (e.g. symmetric encryption, MAC or Message Authentication Code for data integrity and authenticity protection) to protect signaling and user data communications. The USIM principally stores the permanent International Mobile Subscriber Identity (IMSI), a long-term shared secret key K (shared with the HN), and a 48-bit counter called Sequence Number or SQN that provides freshness as a protection measure against replay attacks. As will be discussed in the next section, both the IMSI and the SQN are crucial attributes to take into account when examining the design of the protocol. The third

do we need to know ALL of this?

component, the Serving Network or SN, consists of MME and the E-UTRAN (radio access network of eNodeBs). The SN plays the role of a relay (between the UE and the HN), to which a part of the AKA protocol run is delegated.

B. The AKA protocol

The AKA protocol is the only authentication method that is allowed for 3G, 4G, and 5G networks by the 3GPP. Although the protocol has undergone changes and improvements in each generation, its core specifications of the protocol have remained the same. The focus of the discussion will be on the core protocol, which is currently vulnerable to our attacks.

For the UE to establish a secure channel with the SN (ensuring confidentiality of user data), it must first authenticate itself to its corresponding HN (mainly for billing purpose) and authenticate its HN (so that a fake SN cannot establish such a channel and break confidentiality). The UE and the HN then derivate session keys from the long-term shared secret key K to secure subsequent communication, using a replay protection mechanism (SQN) and MAC to authenticate the exchanged messages. The AKA protocol is made up of 3 main phases: identification, challenge-response, and an optional re-synchronization, in which the SQN value is updated on the HN side, in case it becomes out of sync. The protocol flow is depicted in figure 2.

Identification: First, the SN request the UE's identity by sending an Identity Request message. The UE responds with its identity in an Identity Response message. This identity is used to request authentication material from the HN in the next phase.

Challenge-response: This phase makes use of K, the (under normal circumstances) synchronized SQNs in both the UE and the HN, respectively SQN_{UE} and SQN_{HN}, as well as a shared set of independent one-way keyed cryptographic functions f1 to f5 (and *f1 to *f5) used to compute the authentication parameters and derivate session keys (these are mostly MILENAGE confidentiality functions, widely adopted in the mobile industry since 3G and still considered secure until today) [2].

When the SN requests authentication material, the HN generates a message called AUTN. AUTN first contains the challenge, which consists of a concatenation of R with the SQN_{HN} of the subscriber. However, to protect against eavesdropping, SQN_{HN} is not transmitted in plain text. Instead, it is concealed by being XORed with an Anonymity Key $AK = f5(R, K)$ using the eXclusive-OR operator. The nonce R is used in key derivation to make sure each ciphertext is unique, making it harder for an attacker to determine K using for example a known plaintext attack. The concealed value $CONC = SQN_{HN} \oplus AK$ allows the UE to extract SQN_{HN} by computing AK. $AUTN$ then contains $CONC$ and its MAC (Message Authentication Code) in order to prove the authenticity of the challenge. Moreover, the HN calculates the expected authentication response $xRES = f2(R, K)$, the encryption key CK, and the integrity key IK. But these, in contrast to R and $AUTN$, are not sent to the UE. CK and IK are used in the newly established secure channel once the authentication is successful. In the HN side, the SQN is finally incremented.

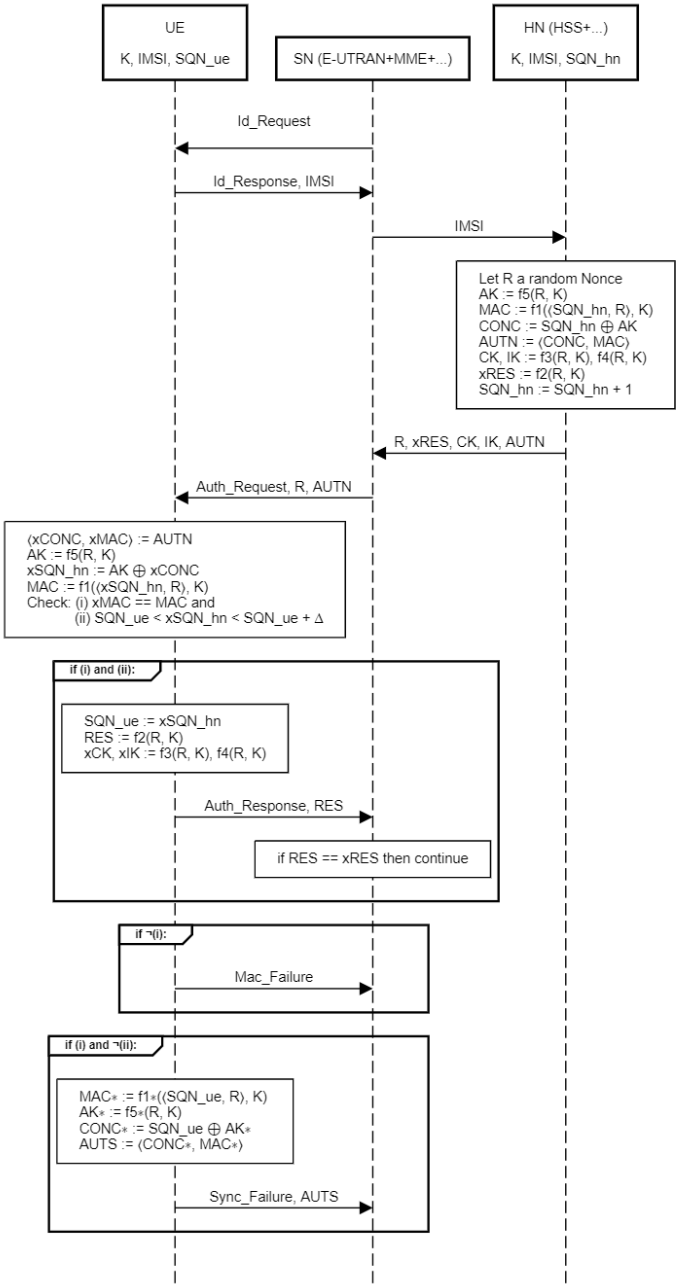


Fig. 2. Sequence diagram of the EPS-AKA protocol

The UE responds with an Authentication Response message when the authentication is successful, or an authentication failure message with the reason for failure otherwise. To determine if the authentication was successful, the UE extracts the MAC and SQN_{HN} from $AUTN$, verifies that the challenge is authentic and fresh. If the MAC is not valid, the challenge is considered not authentic and UE responds with a Mac_failure. If, otherwise, the parsed SQN_{HN} is not comprised in a defined interval from the current UE's SQN_{UE} (in other words, $SQN_{UE} < xSQN_{HN} < SQN_{UE}+$ is false), the UE replies with a Sync_failure. Replayed challenges from a potential active attacker are in this way prevented, and the SQN is considered unsynchronized. The threshold is fixed with regard to a trade-off between availability vs. security. If all checks are passed, the UE also computes CK and IK in his side and stores them for use in future messages. The UE also calculates the Authentication Response RES and sends it to the SN. The SN authenticates the UE by verifying that the received response matches $xRES$, and if it does, the AKA protocol is completed, and the subsequent communications can be secured using the secret keys IK and CK, which are, apropos, not transmitted in any of the protocol messages.

Re-synchronization: In the event of a synchronization failure (case (i) and (ii)), the UE responds with a message called Sync_failure and includes a value called $AUTS$. The purpose of the $AUTS$ is to allow the HN to resynchronize with the user equipment by using the UE's sequence number instead of its own: the new SQN_{HN} is $SQN_{UE} + 1$. Once again, $AUTS$ conceals SQN_{UE} by XORing it with a derived Anonymity Key $AK*$, and is authenticated using a message authentication code $MAC*$.

The EPS-AKA protocol, as it currently stands, poses a significant risk to the privacy of the subscribers. In the following analysis, we will examine the various vulnerabilities and threats inherent in the protocol and explore potential attacks.

III. IMSI Catchers

The first major threat associated with the EPS-AKA protocol is the potential for IMSI leakage. The current EPS-AKA protocol's design does not adequately protect the IMSI from being eavesdropped. In fact, during the Identification phase, the UE sends the Identity Response message containing its IMSI in plaintext without verifying the authenticity of the SN first [7].

A. *Vulnerability*

rough step

This is due to the design choice of a one round-trip mutual authentication protocol with a symmetric key mechanism, which eliminates the possibility of authenticating the "challenge request" part of the protocol (Identification phase) on both the UE's and SN's side. To ensure the authenticity and integrity of an authentication process carried out with only two exchanged messages (one round-trip), a synchronized counter (the SQN) was chosen as a guaranty that both parties have the same session state, which is important for preventing replay attacks. Without synchronization, an attacker could intercept and resend a modified previous message in later time, causing the authentication process to be bypassed. That said, the choice of a one-round trip protocol is due to a trade-off between privacy and network efficiency. Back when 4G with EPS-AKA was launched in 2009, UE's computational resources were too limited to generate a random Nonce instead of the SQN for synchronization [10]. Of course, adding a third message for initially authenticating the SN's Identity Request would have eliminated the need for a synchronized state and increased privacy; but this would also decrease network efficiency. That is why the SN is assumed to be a trusted entity in the network during the Identification phase by the protocol.

This lack of a mechanism for confirming the authenticity of the base station or SN prior to transmitting sensitive information, leaves the UE vulnerable to a Man-in-the-Middle attacker who can broadcast an Identity Request to all the UEs in a specific area and intercept the Identity Response messages containing their permanent identities using a so called IMSI-catcher or Stingrays.

Stingrays can be potentially used for mass surveillance of individuals in a specific location, linking a person to their identity on the network, or tracking subscribers location with fine granularity. Previous studies have indicated that the range of a rogue base station (using USRP B210) is between 50 and 100 meters [10], without any additional hardware to enhance the signal. In consequence, it undermines two main requirements in the 3GPP specification, namely subscriber's privacy and untraceability. Furthermore, this can also pose a serious risk to the integrity of the network because a malicious entity can then impersonate the subscriber and mount further attacks, as it is the case of our third attack.

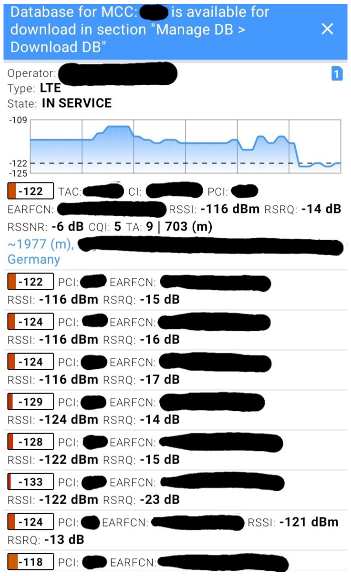
B. *Attack in practice*

We now explain how it is possible to construct a low-cost IMSI Catcher for LTE using only commercially available hardware and open-source software.

How the E-UTRAN is deployed: We first describe how mobile operators deploy the system components described in Section 1 in a specific geographic area in order to comprehend the specifics of the further explained attack. A given service area covered by the operator is divided into smaller regions called Tracking Areas (TAs), each managed by a single MME. A TA contains eNodeBs. Each eNodeB control a group of cells and operate at a certain allocated frequency range / band of LTE, identified by its EARFCN (EUTRA Absolute Radio-Frequency Channel Number) [3]. Base Stations broadcast information about the network to the UEs in their cell group in the form of messages called System Information Blocks (SIB). SIBs include a Tracking Area Code (TAC), a Mobile Country Code (e.g. MCC=262 for Germany), a Mobile Network Code (identifying the mobile operator), and cell ID, and allows UEs to conduct the aforementioned AS Attach Procedure. In the context of Mobility Management, the UE periodically triggers a TA Update with a TAU Request to the MME. This informs the MME about the UEs location in order for it to hand over the connection to another cell, in case the UE is moving out of a so-called cell coverage area, an area where the cell has the highest priority following a certain criterion. In LTE, we mainly prioritize cells that operate on higher priority frequencies. Network operators use this absolute priority frequency criterion to ensure robust coverage. They design the network such that there are multiple cells operating on different frequencies with associated priorities within a specific area. This way, in case of incidents or network congestion on the highest priority frequency, the UE will not lose connectivity, but instead switch to a cell that operates on the next priority frequency. This procedure is referred to as Inter-frequency Reselection. A list of priorities for different frequencies as well as other reselection parameters is broadcasted by the eNodeB of the cell in the SIB messages also [4]. This means that simply having a rogue eNodeB in the vicinity of the UE does not necessarily cause a reselection. An active adversary's main goal is to force Inter-frequency Reselection for the targeted UE, in order to collect its IMSI [3].

Required equipment and tools: To reproduce this attack, only Commercial Off-The-Shelf (COTS) hardware is needed. It includes a laptop with a USB3 port, a Universal Software Radio Peripheral (USRP) operating over a wide range of radio frequencies (70MHz - 6GHz) compatible with LTE frequency bands, and a phone equipped with a prepaid USIM card. The total being less than 1500 euros.
Being limited in terms of technical abilities, we avoid modifying open-source software. We configure the USRP as a rogue base station using Open Air Interface. OAI emulates a Release 10 LTE compliant E-UTRAN and EPC on a single computer, and uses the USRP as the radio interface of the simulated eNodeB. We use the phone first to identify the cells in the targeted area and EARFCNs with the highest priority. For that, we use Service Mode of the device, which provide important information about the LTE network.

Configuring the IMSI Catcher and faced difficulties: We now explain the process of configuring our LTE IMSI Catcher using OAI. Previous researches showed that one can use a second USRP, referred to as a "eNodeB Jammer" [7] which causes the UE to disconnect from its current cell and perform a reselection onto the rogue base station. Since we do not hold a second USRP, we mock the jamming by, instead, using a Faraday cage, in order to reduce the signal strength of the E-UTRAN and physically isolate the UE, but also to prevent any negative impact that jamming could have caused on other UEs that could have been present in the jammer's radius. In order to configure our one and only rogue eNodeB, we first collect the EARFCN of the several LTE cells that coexist in the area and are accessible via reselection using the Netmotinor Android app, as shown in figure 3. We also gather the TAC in Service Mode by calling *0011# [7], as shown in figure 4. We then configure our rogue base station to transmit the publicly available MCC and MNC of the operator in order to mimic the network. We also configure our rogue eNodeB to operate on the EARFCN that has the highest priority next to the available cells. By also broadcasting a TAC value different from the UE's previous cell's (for example, we set it as TACprevious + 1), we trick the UE into sending a TAU Request message. The UE attempts to connect to the cell that operates on the highest priority frequency and has the strongest signal, which is in our case is the rogue eNodeB's. The UE attempts to attach to the rogue base station, which then requests the UE's IMSI through an Identity Request message. The UE then responds by revealing its IMSI inside the Identity Response message. In order to properly configure our IMSI catcher using the previously gathered information, we referred to the article "How to Connect OAI eNB (USRP B210) with COTS UE" from OAI [8]. Unfortunately, despite conducting extensive research and studying the relevant literature, we were not fully prepared for the challenges that arose during the setup and configuration of the device. However, we insist on providing the essential information depicted earlier, to help the lector configure the USRP and further conduct the experiment.



Database for MCC: [redacted] is available for download in section "Manage DB > Download DB"

Operator: [redacted]
Type: LTE
State: IN SERVICE

-122 TAC: [redacted] CI: [redacted] PCI: [redacted]
EARFCN: [redacted] RSSI: -116 dBm RSRQ: -14 dB
RSSNR: -6 dB CQI: 5 TA: 9 | 703 (m)
~1977 (m), [redacted]
Germany

-122 PCI: [redacted] EARFCN: [redacted]
RSSI: -116 dBm RSRQ: -15 dB

-124 PCI: [redacted] EARFCN: [redacted]
RSSI: -116 dBm RSRQ: -16 dB

-124 PCI: [redacted] EARFCN: [redacted]
RSSI: -116 dBm RSRQ: -17 dB

-129 PCI: [redacted] EARFCN: [redacted]
RSSI: -124 dBm RSRQ: -14 dB

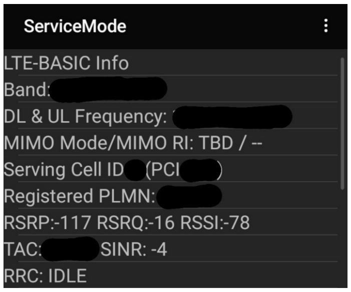
-128 PCI: [redacted] EARFCN: [redacted]
RSSI: -122 dBm RSRQ: -15 dB

-133 PCI: [redacted] EARFCN: [redacted]
RSSI: -122 dBm RSRQ: -23 dB

-124 PCI: [redacted] EARFCN: [redacted] RSSI: -121 dBm
RSRQ: -13 dB

-118 PCI: [redacted] EARFCN: [redacted]

Fig. 3. Netmotinor


ServiceMode

LTE-BASIC Info
Band: [redacted]
DL & UL Frequency: [redacted]
MIMO Mode/MIMO RI: TBD / --
Serving Cell ID [redacted] (PCI [redacted])
Registered PLMN: [redacted]
RSRP:-117 RSRQ:-16 RSSI:-78
TAC: [redacted] SINR: -4
RRC: IDLE

Fig. 4. Serving Mode

C. Countermeasures

Several studies have investigated ways to secure Identity Request messages with additional cryptographic techniques [11]. In 5G, the 3GPP has newly made changes to the identity request phase of the AKA protocol to protect the privacy of the subscriber's permanent identifier: the UE's IMSI is never transmitted in plain text, but instead is sent as a SUbscription Concealed Identifier (SUCI) that is protected by randomized, asymmetric encryption using the HN's public key K. This ensures that only the underlying HN is visible to the SNs or rogue base stations, making it more difficult for IMSI catcher attacks to succeed. However, it's important to note that even with these changes, there are still new subscriber privacy risks that can be introduced by an attack that does not rely on this identification phase.

IV. Linkability attack

The linkability of failure messages is a further vulnerability of the EPS-AKA protocol that allows to track targeted subscribers in a defined attack area covered by a rogue eNodeB. This is accomplished by replaying an old eavesdropped authentication challenge that the targeted UE has already received and then observing the response, which is either Mac_Failure or Sync_Failure. We recall that the UE first checks if the received MAC is correct. If it is not, the UE sends a Mac_Failure message to the SN. A Mac_Failure infers that the replayed message was not successfully authenticated as a challenge for the concerned UE. The targeted UE would not reply with a Mac_Failure. After this, the UE checks if the SQN inside the challenge is current. If not, it sends a Sync_Failure message along with a re-sync token AUTS. Even if the authentication fails, the attacker is then able to infer that the targeted UE is present in the attack area, also allowing him to determine the identity of the subscriber by replaying the challenge multiple times. The subscriber location privacy requirement in the 3GPP specification is broken.

An example of a practical scenario for this attack, is the use of rogue base stations in shops to gather information about customers movement, timing and habits inside the store for the purpose of a better targeted advertising. This scenario has already been reported and exploited in real-world settings, using Smartphones Wi-Fi signaling capabilities [5], analogously to the messages we explained for the linkability attack.

This vulnerability can be addressed by merging the two sources of failure into a single message [6]. Currently, 5G AKA uses encryption to conceal the type of failure in the failure message. However, this method is not completely secure and still leaves the system susceptible to linkability attacks, as an attacker can determine the type of failure by analyzing the length of the message and inferring the type of the message. Multiple studies have proposed additional methods to decrease the likelihood of linkability attacks.

V. Logical attack on SQN privacy

In this section, we present a logical vulnerability in EPS-AKA that would allow a potential malicious entity to compromise the confidentiality of the SQN. Unlike the previously attacks, which only revealed the presence of the targeted subscriber in specific areas, finding out about the SQN allows an attacker to learn the consumption patterns of mobile services for the targeted subscriber. Additionally, this attack has, in contrast to the previous attacks, the advantage that it can break a subscriber's privacy even when they are outside the attack area. It is because it based on exploiting a limited number of rogue base stations deployed in specific locations, such as busy intersections or targeted offices for example, which regularly "gather" Sync_Failure messages from the targeted subscriber and learn n significant bits of the SQN at these different times. Even when users are using mobile services outside the attack area, part of this activity may be leaked to the adversary the next time the user enters the attack area.

A. The role of the SQN

As we explained earlier, the main function of the SQN is to provide freshness to the challenges, which helps prevent replay attacks. Also, by using a SQN, the number of messages sent between the UE and the SN is reduced, as lost or delayed messages can simply be resent with the same SQN, and further processed as a retransmission. This way, the SQN helps to improve the efficiency of the network. There are different options for how the SQN can be updated [10]. These include incrementing the SQN by a fixed number, or using a time-based approach. The choice of which SQN policy to implement is left to the network operator, and it is important to note that this choice has security implications on the network. The logical attack presented in this section will only work if the SQN policy is not time-based, and we will later explain how this is the case. The SQN is incremented on both the UE and the HN upon each authentication according to an incrementing policy set by the operator. According do R. Borgaonkar in "New privacy threat on 3G, 4G, and upcoming 5G AKA protocols" [10], the value added to the SQN after each successful authentication was found to be 1 for all tested operators in the UK.

Before being sent to the UE, SQN_{HN} is masked with an anonymity key AK. Also in case of a synchronization failure, during the process of resynchronization, the UE's current SQN_{UE} is masked with the anonymity key AK*. The goal of the adversary in our scenario is first to break the privacy of the SQN. For that, we first explain the logical vulnerability in the protocol.

B. Logical vulnerability

We stressed before that the SQN is always "concealed" using XOR and never directly encrypted. This is what the attacker specifically exploits in the Concealed Sequence Number $CONC* = SQN_{UE} \oplus AK*$. By replaying a genuine challenge R, AUTN made for a targeted UE at two different times t1 and t2, the attacker can gather the values $CONC*_1$ and $CONC*_2$ sent back by the UE as he encounters synchronization failures. The attacker then calculates $CONC *_1 \oplus CONC*_2 = (SQN_{UE,1} \oplus$

$AK*_1) \oplus (SQN_{UE,2} \oplus AK*_2) = SQN_{UE,1} \oplus SQN_{UE,2}$. By carefully choosing several timestamps, the attacker can infer the entirety of the SQN bits by using the gathered values $SQN_{UE,i} \oplus SQN_{UE,j}$, as we will explain in the next subsection.

C. Attack principle and inferring the SQN

In order to execute this attack, an active attacker first needs to have knowledge of the IMSI of the targeted UE. For that, the adversary can use a Stingray to gather the victim's IMSI. We also assume that the HN increases SQN_{HN} by 1 after each successful authentication. As depicted in figure 5, the attacker fetches 2n + 2 fresh and consecutive authentication challenges $RAND_i$and $AUTN_i$ intended for the targeted UE. Then he replays a total of 2.(n + 2) of them to the UE, from which he gathers CONC* values contained in the Sync_Failure messages [10].

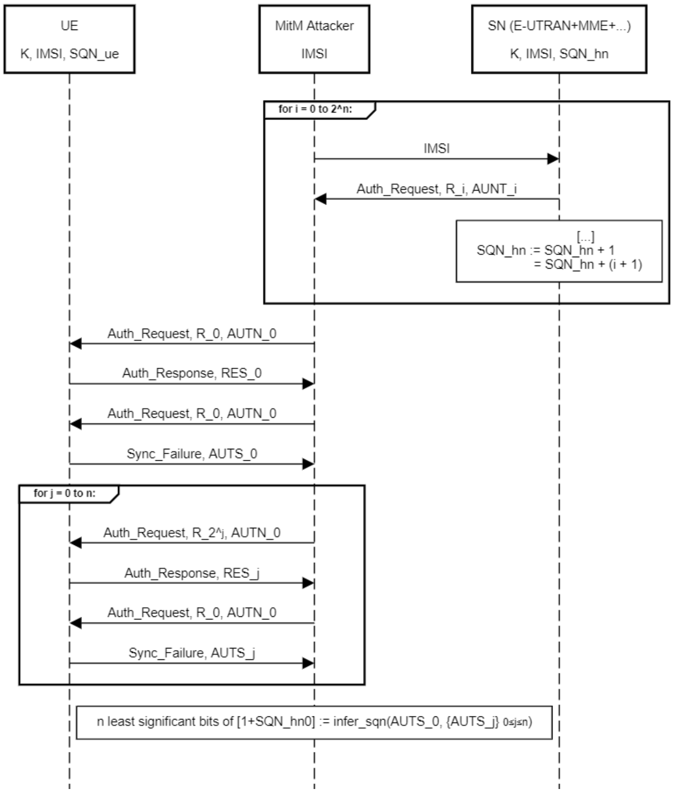


Fig. 5. Logical attack on SQN privacy

The key here is selecting appropriate injections to retrieve values $SQN_{UE} \oplus (SQN_{UE} + 2i)$ for $1 \leq i \leq n$. The attacker first replays the first challenge (R_0 and $AUTN_0$), which is interpreted as authentic and fresh from the UE's perspective. At this point SQN_{UE} is equal to $SQN_{HN,0} + 1$. The attacker then resends the same challenge, resulting in a Sync_Failure message containing $CONC*_0$ where the concealed $SQN_{UE,0}$ is equal to $(SQN_{HN,0} + 1) \oplus AK = (SQN_{HN,0} + 1) \oplus f5(R_0, K)$. Next, the attacker repeatedly replays selected challenges in the form of R_{2j} and $AUTN_{2j}$, causing the UE to update its SQN_{UE} to $SQN_{HN,0}$+2j+1. And each challenge is resent again, causing a Sync_Failure messages containing Concealed Sequence Numbers of the form $CONC*_j = (SQN_{HN,0}+2j+1) \oplus K = (SQN_{HN,0}+2j+1) \oplus f5(R_0, K)$. The SQN is then inferred offline using the algorithm that we implemented in the Python script $sqn_infer()$ [10] in figure 6.

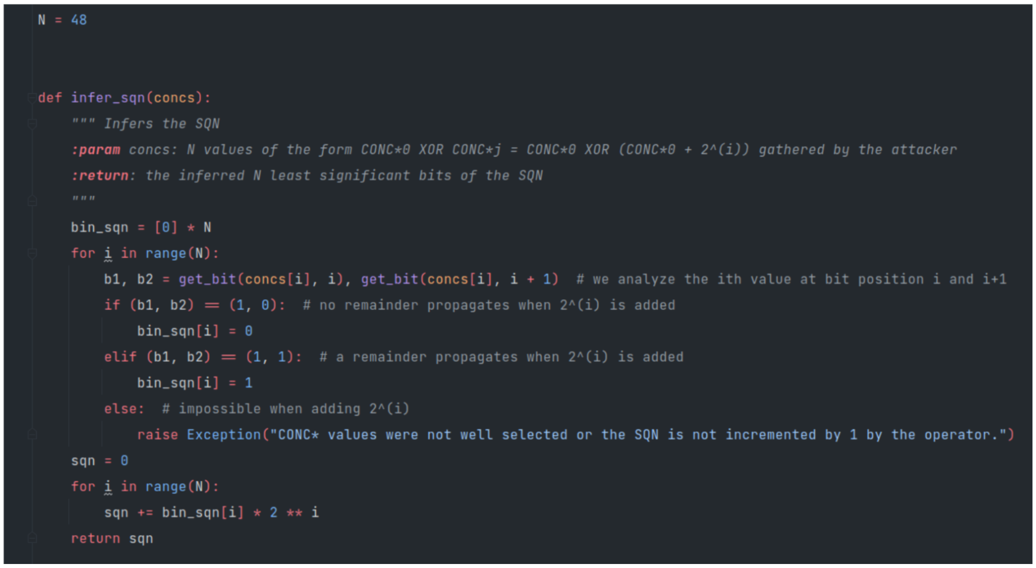
The algorithm takes n+2 fetched $CONC*_j$ values as parameters and returns the n least significant bits of $SQN_{HN,0}$+1. As per the logical flaw explained earlier, it holds for $0 \leq j \leq n$:CONC*$_0$ $\oplus$CONC*$_j$ = (($SQN_{HN,0}$+1) $\oplus AK$) $\oplus$ ((SQN$_{HN,0}$+2j+1) $\oplus AK$) = (SQN$_{HN,0}$ +1) $\oplus$(SQN$_{HN,0}$+2j+1). The algorithm uses these values to infer the n least significant bits of $SQN_{HN,0}$+1, by analyzing how remainders propagate at bit position i and i+1 as 2^i is added to each value. The algorithm can be executed offline and can be adjusted based on the increment used by the operator (the increment is 1 in the script). In the case of this attack, the UE plays the role of an oracle, as the adversary uses values contained in failure messages broadcasted by the UE to gain information about the SQN. To conclude, this logical attack breaking SQN privacy results from the improper implementation of a cryptographic mechanism in the protocol, namely here, XORing the Authentication Key AK with a monotonically incremented counter (SQN).

D. Attack in practice

Since the UE receives these challenges before authentication but after identification, the adversary only needs the UE's IMSI in order to impersonate him and obtain the challenges from the SN. This can easily be done in practice, and researchers have previously demonstrated its feasibility in a proof of concept [10]. They were able to fetch genuine authentication challenges by configuring an USRP to impersonate the target's USIM using the software srsUE from the srsRAN suite (previously known as srsLTE). Alternatively, they also explained how an Android-based phone equipped with a programmable USIM card could also be used. For that, they used their own modified version of the tool pySIM to program a SysmoUSIM card using an external SmartCard reader, in order to stores the targeted UE's IMSI. All the signaling messages were then read using the SCat tool, including the challenges originally intended for the targeted UE. To replay the challenges, the researchers used a rogue base station utilizing OpenLTE, to lure the targeted UE into attaching to it by reselection procedure.

E. Monitoring user activity

Finally, the attacker can then use the n significant bits of SQNs "gathered" at different times to determine the number of AKA sessions the subscriber has made between those times, but also his typical service consumption, even if he is not in the attack area most of the time. For that, the attacker has to exploit the fixed authentication policies of the operator to determine how frequently authentication is performed and for which activities of the UE. Researchers found that there are



```python
N = 48


def infer_sqn(concs):
    """ Infers the SQN
    :param concs: N values of the form CONC*0 XOR CONC*j = CONC*0 XOR (CONC*0 + 2^(i)) gathered by the attacker
    :return: the inferred N least significant bits of the SQN
    """
    bin_sqn = [0] * N
    for i in range(N):
        b1, b2 = get_bit(concs[i], i), get_bit(concs[i], i + 1)  # we analyze the ith value at bit position i and i+1
        if (b1, b2) == (1, 0):  # no remainder propagates when 2^(i) is added
            bin_sqn[i] = 0
        elif (b1, b2) == (1, 1):  # a remainder propagates when 2^(i) is added
            bin_sqn[i] = 1
        else:  # impossible when adding 2^(i)
            raise Exception("CONC* values were not well selected or the SQN is not incremented by 1 by the operator.")
    sqn = 0
    for i in range(N):
        sqn += bin_sqn[i] * 2 ** i
    return sqn
```

Fig. 6. SQN inference algorithm

little variations in authentication frequency among operators, and that it is possible to deduce the fixed policy for an operator by inspecting signaling messages between the SN and the UE when the attaches to the network (e.g. by calling or sending an SMS) [10]. Unfortunately, no particular work struck our attention as we were looking for an in-depth analysis of the different policies implemented by the operators.

F. Countermeasures

One potential solution suggested to mitigate this logical attack vector is to rate limit the number of authentication requests per subscriber. However, an attacker could potentially learn this rate limit by testing the network. R. Borgaonkar mentions in his research [10] that one operator has implemented a rate limit of 3 consecutive failures, but the attacker could potentially bypass this countermeasure by requesting authentication challenges from different SNs.
However, the main vulnerability that this attacks exploit is the use of XOR and the lack of randomness in the SQN, which makes the concealment of the SQN_{UE} by AK inefficient. One proposed solution is to use symmetric encryption instead of XOR. This can be easily implemented by using the existing capabilities of USIM. The symmetric key for encrypting SQN_{UE} can be derived from the key K and R in the received authentication challenge [11]. However, it may add extra processing load on the HN side due to the decryption requirement. This could be delegated to the SN by transmitting the decryption key to it. Another fix is to use asymmetric encryption for encrypting SQN_{UE} inside the concealed CONC* of the Sync_Failure messages [11]. This was not considered practical during the design of 4G, due to a trade-off between security and the cost of a public key infrastructure, but is now known to be used in 5G.

VI. Conclusion:

In this summarization work, we aimed at elucidating the main vulnerabilities and threats facing mobile LTE subscribers' privacy on the currently deployed 3GPP specification of EPS-AKA protocol. We first gave an extensive overview of the LTE architecture and explained how the EPS-AKA protocol operated, as a basis for understanding the functioning of the tree presented vulnerabilities, but also to grasp some practical aspects involved in exploiting them. We showed how three attacks could be conducted to monitor subscribers' location and activity (partly). As an example, we explained how an attacker using a low-cost and easily obtainable hardware setup and open-source software can carry out an IMSI Catcher attack, gaining access to subscribers' identity. We explained the underlying causes of the vulnerabilities and their impact on 4G security requirements, helping to provide insights for future 3GPP standardization (5G). We also cited the proposed countermeasures and fixes to these vulnerabilities according to current research. To conclude, we stress that the vulnerabilities we presented in this summarization work are a fundamental part of the 3GPP specifications of the EPS-AKA protocol and are, for the most, not caused by issues in the implementation of the LTE network nor the operating system of subscribers' devices. Therefore, any device that has a USIM card is susceptible to these attacks. In future releases of 3GPP, clever and sophisticated attackers may find new ways to exploit any obtainable information to perform further AKA protocol-related attacks. Therefore, it is crucial to secure the authentication procedures. It includes protecting the subscriber's IMSI, the failure messages, and the SQN.

REFERENCES

[1] 3GPP, “3GPP System Architecture Evolution (SAE)—Security Architecture” (Release 15), technical specification (TS) 33.401, v15.2.0 (September 2018).

[2] 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*” (Release 1999), technical specification (TS) 35.205, v17.0.0 (March 2022).

[3] Altaf Shaik, Ravishankar Borgaonkar, N. Asokan, Valtteri Niemi, and Jean-Pierre Seifert, “Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems,” Proceedings of the Internet Society Symposium on Network and Distributed System Security (NDSS) (Feburary 2016).

[4] Piqueras Jover, Roger. LTE security, protocol exploits and location tracking experimentation with low-cost software radio. (2016)

[5] S. Clifford and Q. Hardy, “Attention Shoppers: Store is Tracking Your Cell,” New York Times, vol. 14 (2013).

[6] Yue Zhang, Chengbin Huang, and Jinhua Wang ”Privacy protection methods for linkability attacks during 5G AKA”, Proc. SPIE 12506, Third International Conference on Computer Science and Communication Technology (ICCSCT 2022), 125061S (28 December 2022); https://doi.org/10.1117/12.2662492

[7] S. F. Mjølsnes and R. F. Olimid, ‘Easy 4G/LTE IMSI Catchers for Non-Programmers’, in Lecture Notes in Computer Science, Cham: Springer International Publishing, 2017, pp. 235–246.

[8] Luis Ariza, “How to Connect OAI eNB (USRP B210) with COTS UE”, Open Air Interface Wiki(February 2020)

[9] R. P. Jover, ‘LTE security, protocol exploits and location tracking experimentation with low-cost software radio’, arXiv [cs.CR], 18-Jul-2016.

[10] R. Borgaonkar, L. Hirschi, S. Park, and A. Shaik, ‘New privacy threat on 3G, 4G, and upcoming 5G AKA protocols’, Proc. Priv. Enhancing Technol., vol. 2019, no. 3, pp. 108–127, Jul. 2019.

[11] Abdeljebbar, Mourad Kouch, R. (2017). Security Improvements of EPS-AKA Protocol. International Journal of Network Security. 20.636.10.6633 / IJNS.201807_20(4).05.