Ankush Singla*, Syed Rafiul Hussain, Omar Chowdhury, Elisa Bertino, and Ninghui Li

Protecting the 4G and 5G Cellular Paging Protocols against Security and Privacy Attacks

Abstract: This paper focuses on protecting the cellular paging protocol — which balances between the quality-of-service and battery consumption of a device — against security and privacy attacks. Attacks against this protocol can have severe repercussions, for instance, allowing attacker to infer a victim's location, leak a victim's IMSI, and inject fabricated emergency alerts. To secure the protocol, we first identify the underlying design weaknesses enabling such attacks and then propose efficient and backward-compatible approaches to address these weaknesses. We also demonstrate the deployment feasibility of our enhanced paging protocol by implementing it on an open-source cellular protocol library and commodity hardware. Our evaluation demonstrates that the enhanced protocol can thwart attacks without incurring substantial overhead.

Keywords: Cellular Networks, Paging Procedure, Broadcast Authentication

DOI 10.2478/popets-2020-0008
Received 2019-05-31; revised 2019-09-15; accepted 2019-09-16.

1 Introduction

Receiving and transmitting radio packets as part of the cellular communication protocol are arguably two of the most demanding functions of a cellular device with respect to energy consumption. To save device battery, the cellular protocols allow a device to transition to a low-power, idle state when the network detects a predefined period of cellular inactivity from the device. It is, however, crucial to ensure that when the device is in such an idle state, it does not overlook any pending network services (e.g., phone calls). This is where the cellular paging protocol comes into play.

By adhering to the cellular paging protocol, a device periodically wakes up from its idle state to poll for any paging messages triggered by the core network for notifying any pending services. On receiving a paging message (containing the device's Temporary Mobile Subscriber Identity or TMSI), the device gets ready for the service by re-establishing a secure connection to the core network. The exact time periods when the cellular device wakes up and polls for paging messages (also known as the device's *paging occasion*) is **fixed by design**—a deterministic function of the device's **persistent** identity (International Mobile Subscriber Identity or IMSI) and some public parameters (broadcast by the serving network)—in the **4G** cellular protocol. Apart from service notifications, paging messages are also used to disseminate emergency messages, such as earthquake and tsunami warnings. Because of such critical use cases, the paging protocol is an attractive attack target for motivated adversaries (e.g., nation-states, terrorist organizations) for nefarious purposes (e.g., surveillance, creating artificial panic). *It is, thus, paramount to analyze the security and privacy threats of paging protocol, and develop robust defense mechanisms for mitigating the detected vulnerabilities.*

Prior work [13, 15, 16, 18, 23] has identified a number of exploitable weaknesses in the 4G and 5G paging protocols. We classify these vulnerabilities into three categories: ❶ *Lack of confidentiality and anonymity:* A device's TMSI (originally designed to obfuscate the persistent identity) is sent in the paging message in plaintext. Furthermore, the core network often chooses not to update TMSI frequently, because of the expensive cryptographic operations and protocol interactions required for changing the TMSI [13, 18, 23]. The adversary exploiting such weak anonymity policies can map a user's phone number to its TMSI and track the user's location in a particular area. ❷ *Fixed paging occasion:* The specific time frame at which the device wakes up from the idle state is fixed. This creates a side-channel, which enables the adversary to map a user's phone number to its paging occasion, track the user, and recover the

***Corresponding Author: Ankush Singla:** Purdue University, E-mail: asingla@purdue.edu
Syed Rafiul Hussain: Purdue University, E-mail: hussain1@purdue.edu
Omar Chowdhury: The University of Iowa, E-mail: omar-chowdhury@uiowa.edu
Elisa Bertino: Purdue University, E-mail: bertino@purdue.edu
Ninghui Li: Purdue University, E-mail: ninghui@cs.purdue.edu

user's IMSI [16]. ❸ *Lack of authentication:* The paging protocol does not have authentication or integrity protection. This allows an adversary to hijack the paging channel [15] and push fabricated paging messages including emergency alerts to all devices in a target area [15].

In this paper, we aim to design and evaluate defense mechanisms that would mitigate the above mentioned vulnerabilities without incurring prohibitive performance and communication overhead, or requiring significant protocol or infrastructural changes.

Existing proposals for mitigating the paging protocol vulnerabilities [11, 12, 16, 19, 24] require either heavy signaling overhead or major overhauls of the cellular systems, rendering legacy devices incompatible. Also, there are currently no proposed techniques that can efficiently provide authentication for paging messages.

Our approach: Our defense mechanisms address each of the paging protocols' vulnerabilities discussed above. To protect devices from unauthorized tracking due to infrequent TMSI updates, we propose a new ephemeral identifier called P-TMSI (*Pseudo-TMSI*). In contrast to TMSI, which is explicitly assigned and sent to a device securely and reliably (with acknowledgement) by the core network, the P-TMSI is sequentially selected by the core network and the device from a list of P-TMSIs generated from a pre-shared, secret seed agreed upon during the mutual authentication and thus does not require any additional interactions.

To defend against the side-channel attacks exploiting the weakness of fixed paging occasion, we propose a variable paging occasion such that any two successive paging messages for the same UE are not sent/received at the same paging occasion. To realize this, we propose to compute the paging occasion based on the frequently updated P-TMSI value of the device instead of its IMSI.

We also propose two policies governing the update frequency of the paging occasion and P-TMSI— (1) to refresh after each paging occasion, or (2) only after a paging message is sent to the device. With the first approach the network and the device select a new P-TMSI and compute the new paging occasion after every paging cycle. Since this approach guarantees that no two consecutive paging occasions are the same, it offers a stronger security guarantee against a resourceful adversary who already knows the current P-TMSI of the user and tries to inject fake paging messages at the victim's paging occasion [15]. In this approach, however, the paging occasion is renewed even in the absence of any paging message sent to the cellular device and thus induces a high computation and storage overhead. On the contrary, with the second approach the network and the device seamlessly select the next P-TMSI only when a paging message is sent to the cellular device. This policy trades off strong anonymity guarantees for significant overhead reduction.

Finally, to protect the devices against paging channel hijacking and unauthorized injection of fake paging messages, we propose a symmetric-key based broadcast authentication scheme [21]. We sign the paging messages in a particular paging cycle using the corresponding signing key from a pre-generated one-way key-chain and attach the signature to the paging message. The network reveals the verification key in the next paging cycle, which the cellular device uses to verify the paging message sent in the previous cycle. Our approach, thus, enables an efficient authentication/verification scheme, both in terms of computation and communication overhead at the expense of negligible latency.

Our contributions: The paper has the following technical contributions:

1. We design a defense mechanism which mitigates the design vulnerabilities of the cellular paging protocol. Particularly, we address each of the paging protocol vulnerabilities in the following way:
   (a) We design an approach to prevent tracking of a UE in a particular area enabled by observing the TMSIs included in the paging messages. Our approach introduces an ephemeral identifier called the P-TMSI.
   (b) We design a secure and lightweight mechanism to randomize the paging occasion, so that it does not leak information about the IMSI. Our mechanism, thus, provides privacy protection against the ToRPEDO attack [16].
   (c) We evaluate the cost of adding broadcast authentication to paging messages to prevent malicious tampering or injection of fake alerts.
2. We implemented our enhanced protocol using open-source libraries and commodity hardware, and carried out an extensive evaluation. The evaluation shows that our protocol is efficient in terms of execution, memory, and performance overhead.

2 Background

In this section, we briefly introduce the architecture of 4G and 5G cellular networks, the attach and the paging procedures, and how the paging occasion is computed.

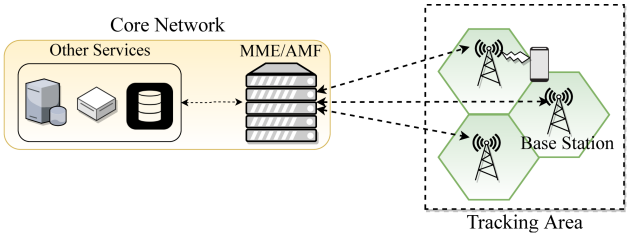

Fig. 1. Cellular Network Architecture.

2.1 Cellular Architecture

Cellular architecture. In cellular networks, a geographical area is partitioned into hexagonal cells managed by one or more base-stations. These base-stations provide connectivity between the core network and the nearby cellular devices (as shown in Fig. 1). A *Mobility Management Entity* referred to as MME (resp., Access and Mobility Management Function or AMF in 5G) in the core network manages the connectivity and mobility of the devices in its *tracking area* (TA) consisting of one or more cells.

Device identity. A cellular device (also referred to as User Equipment or UE) equipped with a SIM (*Subscriber Identification Module*) card is assigned an International Mobile Subscriber Identity (IMSI) which uniquely identifies the UE. The IMSI typically does not change once assigned, and its leakage can enable tracking and impersonation of unsuspecting victims. To prevent unwanted exposure of a user's IMSI, the MME assigns a randomly generated Temporary Mobile Subscriber Identity (TMSI) to the UE for further communication with the core network. The TMSI has to be updated every time the UE moves to a new tracking area or after a certain time interval. The Third Generation Partnership Project (3GPP) [2], the standards body for cellular network protocols, recommends changing the TMSI frequently to prevent user tracking.

Time synchronization. In 4G and 5G cellular networks, the communication between a UE and a base-station is carried out using *radio frames* (also called *system frames*) each of which spans 10 milliseconds. These radio frames are indexed using a circular counter from 0 to 1023, which is also called its system frame number (SFN). Each SFN is further partitioned into 10 sub-frames of 1 millisecond each. A base-station periodically broadcasts a `master_information_block` message which includes the current SFN of the network and other parameters used by UE to synchronize itself with the base-station.

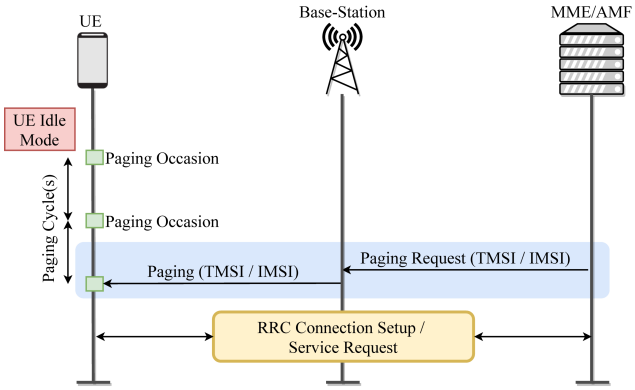

Fig. 2. Paging Procedure.

2.2 Attach Procedure

Whenever a UE is switched on with a valid SIM card, it first scans the network and selects the base-station that satisfies its selection criteria. To establish a connection with the core network, the UE then sends an `attach_request` message to the MME, containing its IMSI/TMSI and the supported cipher suites. The UE and the core network authenticate each other using a challenge-response protocol (using a pre-installed symmetric master key in the SIM card) and then negotiate the cipher suite to be used for encryption and message authentication based on their individual capabilities. Finally, the MME completes the attach procedure by sending an encrypted and integrity protected `attach_accept` message containing the UE's TMSI.

2.3 Paging Procedure

The paging procedure (see Fig. 2) allows a UE to enter a low power-consumption mode only when there are no uplink (from UE to network) or downlink (from network to UE) messages for a pre-defined amount of time.

Paging cycle. When in *idle* mode, the UE periodically wakes up (for ∼1 ms) to check if there is any notification for pending service(s) (e.g., phone call, SMS, or incoming data), at a predetermined time-frame once every *paging cycle*. A paging cycle can have any time duration from 320 ms (32 radio frames) to 2.56 seconds (256 radio frames) [3] depending on network parameters.

Paging occasion. The radio frame at which the UE wakes up in every paging cycle to check for a paging message is known as the *paging frame (PF)*. It is computed as follows using the paging cycle value $\mathsf{T} \in \{32, 64, 128, 256\}$, another public parameter $\mathsf{nB} \in \{4\mathsf{T}, 2\mathsf{T}, \mathsf{T}, \frac{\mathsf{T}}{2}, \frac{\mathsf{T}}{4}, \frac{\mathsf{T}}{8}, \frac{\mathsf{T}}{16}, \frac{\mathsf{T}}{32}\}$, and the UE's identifier UE_ID, where $\mathsf{UE\_ID} = \mathsf{IMSI} \bmod 1024$.

$$\mathsf{PF} = (\tfrac{\mathsf{T}}{\mathsf{N}}) \times (\mathsf{UE\_ID} \bmod \mathsf{N}), \text{ where } \mathsf{N} = \mathsf{Min}(\mathsf{T}, \mathsf{nB})$$

T and nB are system parameters shared in the system_info_block messages broadcast by the base-station. The value of PF varies between 0 and 255 due to the above mentioned constraints. The specific sub-frame of the paging frame at which the UE wakes up is also computed using the above mentioned parameters and a simple look-up table. The paging frame and the sub-frame together form a UE's *paging occasion*.

Emergency notifications. Paging is also used to disseminate emergency information, such as tsunami warnings or amber alerts, and to notify any changes in system configuration to all the UEs in an area.

Response to a paging message. A single paging message can contain up to 16 *paging records* addressed to multiple UEs. Each record contains the UE identifier (TMSI/IMSI). If a UE notices its own identifier, it switches to an *active* state and initiates a connection with the core network to get the service completed.

3 Exploitation of Existing Security and Privacy Policies

In this section, we first discuss the existing security and privacy policies prescribed by the 3GPP standards for cellular paging protocols. Existing attacks on paging protocols, however, have shown these policies to be inadequate. Based on the analysis of these attacks, we then distill design and deployment weaknesses of the cellular paging protocols that enable these attacks (see Table 1).

3.1 Security and Privacy Policies of Paging Protocols

Unfortunately, the paging protocol neither provides confidentiality nor authentication guarantees. This lack of guarantees and the fact that paging messages are broadcast imply that an adversary can not only sniff a paging message over-the-air, but also inject fabricated paging messages. One can attribute the lack of confidentiality guarantees to the paging protocol's original goal of balancing between the device's battery consumption and quality-of-service. More precisely, providing confidentiality guarantees through the use of encryption would require the device to perform up to 16 expensive decryption operations—one for each paging record—per paging occasion (i.e., ∼1.280 seconds in practice) to check whether there is a pending service for the device, thus defeating the purpose of conserving battery power.

The 3GPP, however, aims at providing some anonymity guarantees for the paging protocol, that is, an attacker should not be able correlate two paging messages intended for the same user. This guarantee is achieved by using the transient TMSI as device identifier in the paging records instead of the persistent IMSI. Note that, one can aim at maintaining this guarantee as long as TMSIs are changed frequently and the protocol does not leak any side-channel information.

3.2 Exploitable Design & Deployment Weaknesses

In this section, we discuss paging protocols' design and deployment weaknesses that have been exploited by prior attacks. Any robust defense mechanisms must mitigate these weaknesses.

3.2.1 Design Weaknesses

This section discusses protocols' design vulnerabilities.

Side-channel information due to fixed paging occasion. A closer inspection of the 4G paging protocol reveals a critical and fundamental weakness. For a particular device in a specific cell, the time intervals when the device wakes up from the low-power state to check for paging messages (i.e., the paging occasions) are fixed [16]. This is because the paging occasion is computed from the device's persistent IMSI. This essentially exposes side-channel information which is shown to be exploitable by the ToRPEDO (TRacking via Paging mEssage DistributiOn) attack [16].

To track the location of the victim, the adversary first sniffs the paging messages broadcast by a legitimate base-station serving a particular target area and learns the distribution of paging message arrival (shown in Fig. 4) at every paging occasion. The adversary then continues making silent phone calls until a paging occasion is found to receive a significantly higher number of paging messages than the others. By comparing the paging message distribution under attack (see Fig. 4) with the benign paging message distribution (see Fig. 3) one can identify the victim's presence in the target area and obtain the victim's paging occasion, which is 21 as is evident from Fig. 4. The reason is that the probability of receiving paging messages in the victim's paging occasion under attack is substantially higher (a spike at paging occasion 21 in Fig. 4) than the rest of the paging occasions both under attack and benign conditions. The exposure of paging occasion, i.e., the $\mathsf{UE\_ID} = \mathsf{IMSI} \bmod 1024$ reveals the trailing 7-10 bits of the victim's IMSI (when base-stations set their public parameters T and nB both to 128). This can further enable the adversary to learn the victim's IMSI through the IMSI-cracking attacks [16].

Gen.	Design Weakness		Deployment Weakness	
	Vulnerability	Attacks	Vulnerability	Attacks
4G	TMSI update frequency is under-specified [1]	TMSI exposure and location tracking [18, 23]	TMSI is updated infrequently or predictably	TMSI exposure and location tracking [13, 18]
	Paging occasion is based on IMSI [2]	ToRPEDO [16] and IMSI-Cracking [16]	Paging contains IMSI as device identifier	IMSI-Catching with PIERCER [16]
	Lack of authentication [2]	Paging-channel hijacking [15], panic attack by broadcasting fake emergency alerts [15], stealthy deregistration attack [15]	—	—
5G	TMSI update requires additional interactions	Potential location tracking [18, 23]	—	—
	Paging occasion is based on TMSI, but requires additional protocol interactions to change	TMSI exposure and location tracking [13, 18]	—	—
	Lack of authentication or integrity protection [1]	Paging-channel hijacking [15], panic attack by broadcasting fake emergency alerts [15], stealthy deregistration attack [15]	—	—

Table 1. Security and privacy policies for the 4G/5G paging protocols and the corresponding attacks exploiting them.

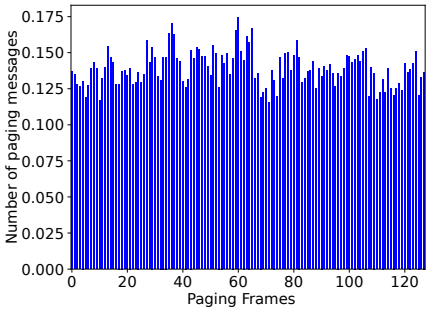

Fig. 3. Average number of paging message arrivals at different paging occasions within one paging cycle, where T=nB=128 and the adversary made **no phone calls**.

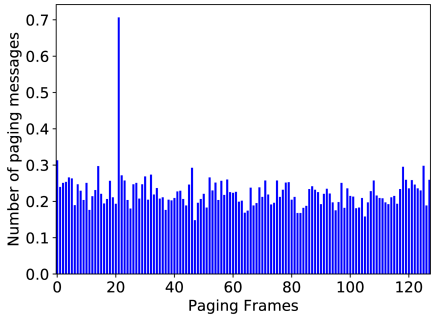

Fig. 4. Average number of paging message arrivals in different paging occasions within one paging cycle, where T=nB=128 when the adversary makes **multiple phone calls**.

To prevent such attacks, 5G specification has recently proposed using variable paging occasions computed using the TMSI, i.e., UE_ID = TMSI mod 1024. The specification, however, requires the device and the core network to perform the `configuration update procedure` every time TMSI is updated.

Lack of authentication. Another critical design weakness of the existing paging protocols is the lack of cryptographic protections for paging message authentication. This makes it possible for an attacker to inject fake paging messages including malicious emergency warnings (e.g., tsunami, amber alerts) by simply installing a base-station with a higher signal strength near the victim UE(s), and start flooding all the paging frames with the fake alerts [15]. This can be used to create a widespread panic in a certain area. The attacker can also carry out a DoS attack by transmitting empty fake paging messages [15] to prevent UEs in certain area from getting legitimate paging messages. This consequently prevents UEs from receiving any notifications (e.g., call, SMS) from the core network. The fact that UEs cannot differentiate between a fake and a legitimate paging message makes this attack very powerful.

3.2.2 Deployment Weaknesses

This section presents deployment slip ups by carriers that have been exploited by prior attacks.

Privacy loss due to the use of IMSI as the device identifier. Certain network carriers have been shown to be using IMSI in the paging records as the device identifier instead of the 3GPP prescribed TMSI. Hussain et al. proposed the PIERCER attack (Persistent Information ExposuRe by the CorE netwoRk) [16] which ex-

ploits this deployment weakness along with the design weakness of using a fixed paging occasion and lack of confidentiality for the victim's IMSI. Access to the victim's IMSI can enable the attacker to launch further targeted attacks [15].

Side-channel information due to infrequent updates of TMSI. Although the 3GPP standard for 4G networks [1] suggests changing the TMSI frequently to prevent mapping (e.g., phone number to TMSI) attacks, it does not clearly outline the rate at which a UE's TMSI should be changed. Due to under-specification, the practical deployments of the paging protocol have shown to either not update the TMSI frequently [18, 23] to avoid having to carry out the additional protocol interactions, or choose the next TMSI predictably, even when changed [13]. Operational networks' reluctance to change TMSI frequently allows an attacker to identify and track a user's presence in a target area [18]. For this, the attacker makes multiple silent phone calls to the victim's device for which the network triggers paging messages. The adversary equipped with a low-cost software-defined radio (SDR) can then sniff the paging messages in the target area. If the adversary observes a unique TMSI appearing in the paging messages, she infers that the victim is present in the current base-station's coverage area, and thus track the victim's coarse-grained location. Tracking a victim UE over a large geographic area is also possible for a resourceful adversary by deploying multiple such SDRs at different locations.

The 5G specification [2], on the other hand, clearly outlines that every service completion (triggered by either the device or the network) calls for a change of TMSI through the `configuration update procedure`. Since the configuration update procedure requires additional interactions between the device and the core network, the upcoming 5G deployments may similarly try to get away without introducing such additional interactions and run into similar issues as 4G operational networks – thus becoming susceptible to location tracking attacks.

4 Overview of Proposed Defenses

In this section, we first present our adversary model and then discuss the challenges in designing a secure paging protocol. Finally, we provide a high-level overview of our proposed defense mechanisms.

4.1 Adversary Model

For designing our defense mechanisms, we consider the following adversarial capabilities. This adversary model is consistent with prior work on this area.

1. We assume that the adversary knows the phone number (or, other soft identity) of the target UE and can trigger multiple paging messages from the MME by placing calls or sending SMSes.
2. The adversary is able to eavesdrop on the paging broadcast channel as well as to create and inject fake paging messages at the paging frames of his choice using a malicious base-station.
3. The attacker-controlled base-stations can broadcast with a higher signal strength than the legitimate one forcing the UE to receive the fabricated paging messages instead of legitimate ones.

We aim to design defense mechanisms that can provide the authentication and guarantees about the lack of side-channel information for paging protocols in the presence of an adversary with the above capabilities. We do not aim to provide confidentiality guarantees as it can incur prohibitive overhead for the device. Finally, we consider denial-of-service attacks, such as, jamming at the physical layer to be outside the scope of this paper.

4.2 Challenges

To incorporate security and privacy mechanisms in the existing paging protocol without breaking backward compatibility, while being incentive compatible at the same time, one has to address the following challenges: ① *Paging packet format*: Modifying the packet format of the paging protocol to include additional cryptographic information is an approach unlikely to be deployed in practice because of its associated deployment cost and backward incompatibility. ② *Protocol overhaul:* Any defense requiring substantial changes in the current protocol is also unlikely to be accepted following the above argument. ③ *Overhead:* As a cellular device processes roughly one paging message per second, any sophisticated cryptographic scheme might incur prohibitive overhead with respect to packet processing time and energy consumption.

4.3 Overview

This section provides a high-level overview of our proposed defenses.

4.3.1 Preventing Side-channel Information due to Infrequent TMSI Updates

Ephemeral identifier P-TMSI. To prevent cellular devices from being illegitimately tracked due to the infrequent update of TMSIs, we propose a new ephemeral UE identifier, P-TMSI (or pseudo-TMSI). The P-TMSI of a device will be seamlessly refreshed by both the time-synchronized UE and core network simultaneously. We propose to use P-TMSI as the device identifier in the paging records instead of TMSI. The transient nature of P-TMSI prevents an attacker from being able to correlate two paging messages sent to the same UE.

The motivation for using P-TMSI instead of TMSI stems from the fact that a TMSI update in 4G is performed by executing the `GUTI reallocation procedure` (resp., `configuration update procedure` for 5G), which requires sending an encrypted and integrity protected `GUTI_reallocation_command` message from the network to the UE and a `GUTI_reallocation_complete` message from the UE to the network as an acknowledgement. On the other hand, with our proposed mechanism, updating P-TMSI will not require any additional protocol steps to synchronize its values between the UE and MME. This design choice can encourage the network operators to adopt this mechanism as updating P-TMSI does not incur any communication overhead.

P-TMSI update mechanism. The core network will share a random secret seed with the UE through the secure and authenticated channel established after completing the `attach procedure`. Using this secret seed, the UE and the core network will both generate a list of random numbers using a secure pseudo-random number generator (PRNG) and store this list in their respective memories. The UE and the core network use this list to sequentially select the next P-TMSI values. The frequency of the P-TMSI updates can be governed by two approaches as discussed in the next section.

4.3.2 Preventing Side-channel Information due to Fixed Paging Occasion

Variable paging occasion. To prevent against the attacks exploiting the fixed paging occasion [16], we propose a mechanism to generate variable paging occasions so that two consecutive paging messages do not have the same paging occasion for a particular UE. For this, we propose to use the frequently updated P-TMSI values instead of the static IMSI to compute the paging occasion of a device. We thus modify UE_ID (used for computing the paging occasion) calculation by using P-TMSI instead of IMSI, that is, $\mathsf{UE\_ID} = \mathsf{P\text{-}TMSI}_{\text{current}} \bmod 1024$ instead of $\mathsf{UE\_ID} = \mathsf{IMSI} \bmod 1024$. Such an approach prevents any kind of leakage of IMSI, thus protecting against the ToRPEDO and PIERCER attacks [16].

Update frequency for the paging occasion. One natural policy to randomize the paging occasion would be to change the paging occasion after every paging cycle regardless of whether the UE actually received any paging message in that paging cycle. Since we compute paging occasion based on P-TMSI, this design choice naturally forces us to change the P-TMSI after each paging cycle. The main intuition for this is to enable the device to evade a powerful adversary that keeps sending empty paging messages to the UE, by knowing the UE's current P-TMSI, and thus tries to occupy victim's paging channel. Such a paging-channel hijacking attack becomes ineffective if the UE moves to a different paging occasion after every paging cycle. This approach, however, exhausts the list of P-TMSIs rapidly as it requires the UE to compute its next paging occasion at every paging cycle. This also requires a strict time synchronization between the UE and the base-station.

To address these challenges, we propose to update the P-TMSI and paging occasion only when there is an actual paging message for the UE and the UE successfully reconnects to the MME using a `service_request` message. This is computationally more efficient as it requires paging occasion updates at a much slower rate. In this approach, an adversary, however, can hijack the paging channel knowing the P-TMSI of a victim and never let any legitimate paging message reach the victim and thus force the paging occasion to remain fixed. To allow the victim to detect a paging channel hijacking attempt, we rely on the paging message authentication which we discuss below.

4.3.3 Preventing Lack of Paging Message Authentication

To prevent injections of fake paging messages and to enable the UE to detect any paging channel hijacking attempts, we propose to use a broadcast authentication/integrity protection scheme that allows each receiver to verify if the received message is intact and originated from the claimed sender [20].

Choice of broadcast authentication scheme. There are two possible design approaches one can consider: one based on asymmetric-key cryptography and another based on symmetric-key cryptography. Both these approaches have their strengths and limitations.

Digital signatures based on asymmetric cryptography can be a straightforward choice for broadcast authentication, since they are scalable and also provide public verifiability without the need for setting up individual secret keys for every recipient. Digital signature schemes, like RSA [22] and ECDSA [8], however, incur significant performance overhead as they involve expensive cryptographic operations that may slow down the signing and verification process and thus affect the timely delivery of service notifications and emergency warnings. Digital signatures also require Certification Authorities (CAs) to generate and maintain the public keys of different entities which is currently absent for 4G and 5G cellular protocols. To make matters worse, paging messages are only of a certain fixed maximum length, and it is crucial to fit the extra information for digital signature and the certificate-chain within the current protocol packet format. There is just not enough space for this extra information making such a solution infeasible.

Symmetric-key based authentication schemes, on the other hand, generally rely on Message Authentication Codes (MAC) [17]. These schemes are, however, not ideal for large-scale broadcast authentication as they require pairwise secret keys for every signer and verifier. This adds a significant management and storage overhead to maintain and distribute all these keys making it infeasible for practical deployments. Furthermore, if pairwise secret-keys were used for paging message authentication, the base-station would need to include a separate MAC for each paging record in the paging message. Since one paging message can contain a maximum of 16 paging records, this would allow the network to fit only a maximum of 8 paging records and their corresponding MACs in a paging message without breaking backward compatibility. This would result in a severe degradation in the quality of service as the base-stations might have to wait multiple paging cycles to communicate an incoming call or a service request to the UE.

To address these challenges, we leverage the TESLA broadcast authentication protocol [21] and instantiate it in the context of paging message authentication. Our instantiation, dubbed PTESLA, uses symmetric cryptographic functions (MAC) but provides asymmetric-key properties by *delayed key disclosure*, i.e., the verification key is disclosed after a fixed interval of time allowing receivers to verify the messages sent in the previous time-interval. TESLA addresses the scalability issue of symmetric-key schemes by removing the need for disseminating pairwise keys to the recipients. The signing and verification overhead incurred by TESLA is very low and the size of the extra information sent for authentication is also small when compared to the asymmetric-key based techniques, which makes it a perfect candidate for our purposes.

For PTESLA to work, the UE and the base-station establish the required bootstrapping parameters during the initial `attach procedure` after mutual authentication. Afterwards, for every paging message, the base-station calculates an authentication tag and attaches it to the paging message. Upon reception, the UE buffers the paging message along with the authentication tag. The base-station releases the corresponding verification key along with the paging messages in the next paging cycle. The UE determines whether the verification key is valid and uses it to authenticate the previous paging message.

5 Protocol Design

In this section, we provide details about our proposed solutions, including the rationale behind the design decisions.

5.1 Refreshing P-TMSI and Paging Occasion

A straightforward approach to assign new P-TMSI values would have been to generate them on-the-fly when needed, but this might result in performance issues in real deployments. We, therefore, introduce the concept of a *P-TMSI store*, $\mathcal{L}_s$, which stores a list of 32-bit random numbers serving as future P-TMSI values. These values can be accessed and used in a sequential manner at runtime. The P-TMSI store allows the UE and the network to generate/regenerate P-TMSI values in batches and store them in memory for quick access when required. In what follows, we discuss the specific details of the two approaches to periodically refresh the P-TMSI values for the UEs.

5.1.1 Refreshing P-TMSI after each Paging Cycle

Our first approach refreshes the P-TMSI and the paging occasion after each paging cycle. The three main steps of this approach are as follows (see also Fig. 5):

(1) Bootstrapping: When a UE initially wants to connect/register to the core network, it initiates the `attach procedure` by sending the `rrc_connection_request` message to the base-station. The base-station generates a 32-bit random secret seed $\mathsf{K} \longleftarrow \{0,1\}^{32}$ for the UE and shares it with the UE in the encrypted and

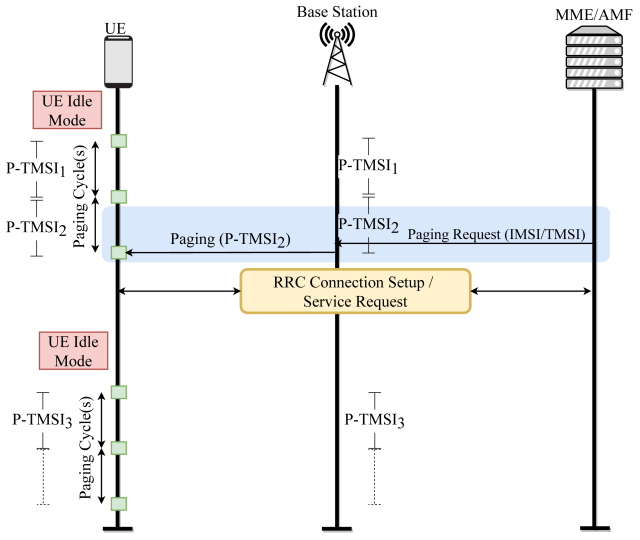

Fig. 5. Refreshing P-TMSI after each paging cycle.

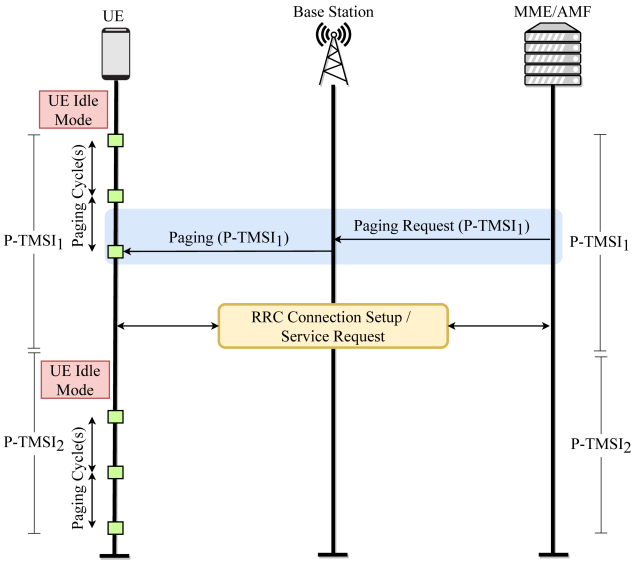

Fig. 6. Refreshing P-TMSI after every paging message.

integrity protected `rrc_connection_reconfiguration` message sent at the end of the `attach procedure`.

The base-station and the UE then use a cryptographically secure pseudo-random number generator (CSPRNG) with seed K to generate a list $\mathcal{L}_s$ of n 32-bit random numbers (value of n is configurable) and store it in memory. In our instantiation, we use a Hash based Message Authentication Code–Deterministic Random Bit Generator (HMAC_DRBG) [9] to generate this list.

$$\mathsf{HMAC\_DRBG(K)} \longrightarrow \{\mathsf{P\text{-}TMSI}_1 \cdots \mathsf{P\text{-}TMSI}_n\}$$
$$\text{where, } \mathsf{P\text{-}TMSI}_i \in \{0,1\}^{32}$$

(2) Updating P-TMSI$_{\text{current}}$ at runtime: Whenever the UE enters the idle mode, the base-station and the UE choose the first P-TMSI from the list $\mathcal{L}_s$ as the P-TMSI for the current paging cycle and set the P-TMSI index i to 1. At every paging cycle, the base-station and the UE increment the index i and pick-up the corresponding P-TMSI from $\mathcal{L}_s$.

$$\mathsf{P\text{-}TMSI}_{\text{current}} = \mathsf{P\text{-}TMSI}_i$$
$$i = i + 1, \text{ where } i \in \{1 \ldots n\}$$

(3) Regenerating the P-TMSI store: The base-station and the UE store the current state of the CSPRNG in the context CTX. This context is used to generate the next batch of random numbers once the current list is depleted.

$$\mathsf{HMAC\_DRBG(CTX)} \longrightarrow \{\mathsf{P\text{-}TMSI}_{n+1} \cdots \mathsf{P\text{-}TMSI}_{2n}\}$$

Synchronization challenge. Since the P-TMSI is being continuously refreshed after each paging cycle, it requires synchronization between the base-station and the UE to ensure they have same value for current P-TMSI. Otherwise, there can be a situation where the UE has not updated its current P-TMSI value and is expecting a paging message addressed to the stale P-TMSI, but the base-station sends the paging message containing the new P-TMSI as the UE identifier. This situation can arise because of UE crashes, execution errors, or some other device malfunction. To prevent desynchronization, we rely on the in-built *synchronization* procedure of the cellular protocol with which the base-station and UE periodically synchronize their time and radio frames leveraging `master_info_block` and `system_info_block` messages.

Deployment limitations. Though this approach can prevent an adversary from hijacking the victim's paging channel, the UE in idle mode needs to spend its resources to compute new paging occasion after each paging cycle. This exhausts the list of P-TMSIs $\mathcal{L}_s$ rapidly and forces the UE and the base-station to regenerate it frequently.

Further enhancements. To make this approach more efficient, one can update P-TMSIs after every N-th paging cycles instead of each paging cycle. The UE and base-station can negotiate N (e.g., 10) during the `attach procedure`. This will significantly improve the performance (by a factor of N), while keeping the window for performing any attack fairly short (e.g., N=10 paging cycles or approximately ∼10 seconds).

5.1.2 Refreshing P-TMSI after every Paging Message

Refreshing the P-TMSI after each paging message (Fig. 6) instead of every paging cycle overcomes the drawbacks of the previous approach and is computationally more efficient. It also obviates the responsibility of the

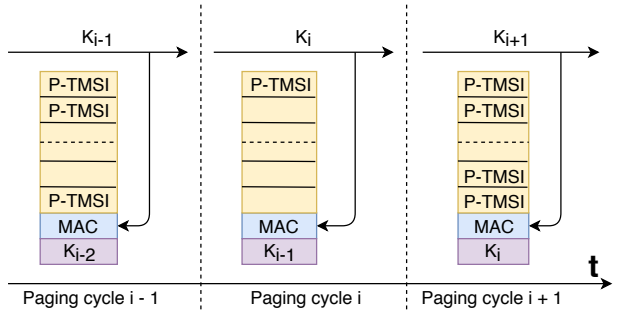

Fig. 7. At each paging cycle, the base-station selects the current signing key from the one-way key-chain and calculates the MAC on the paging message using this key. It then attaches the MAC and the previous paging cycle's key to the paging message.

base-station to maintain the list of P-TMSIs, $\mathcal{L}_s$. The current approach has the following three steps:

(1) Bootstrapping: When a UE sends the initial `attach_request` message to connect to the core network, the corresponding MME generates a 32-bit random secret seed K for that UE and shares it using the encrypted and integrity protected `attach_accept` message at the end of the initial `attach procedure`. The UE and the MME then use a CSPRNG to generate a list $\mathcal{L}_s$ of n 32-bit random numbers.

(2) Updating P-TMSI$_{\text{current}}$ at runtime: Initially, the UE and the MME choose the first P-TMSI from $\mathcal{L}_s$ as the P-TMSI$_{\text{current}}$. In the event of a paging message being sent, the MME uses the P-TMSI$_{\text{current}}$ to address the UE. Upon receiving the paging message, the UE re-connects to the base-station and to the core network, and sends a `service_request` message to the MME. The UE then updates its P-TMSI$_{\text{current}}$ with the next P-TMSI value in $\mathcal{L}_s$. Once the MME receives the `service_request` message from the UE, it also updates its P-TMSI$_{\text{current}}$ accordingly.

(3) Regenerating the P-TMSI store: The UE and the MME regenerate $\mathcal{L}_s$ after it gets depleted, using the saved context CTX.

5.2 TESLA-Based Authentication for Paging Messages (PTESLA)

For PTESLA to work, the UE and the base-station need to be time-synchronized which is inherently provided by the `master_info_block` message during the UE's initial bootstrapping phase. The base-station also communicates the disclosure delay and the initial key to the subscribers though a secure channel which is established at the end of the initial `attach procedure`. In what follows, we specify the details of our adaptation of the TESLA authentication protocol.

Base-station bootstrapping. The base-station requires a one-way key-chain for PTESLA to work. To generate this key-chain, the base-station creates a random 32-bit seed S. It then applies the SHA-256 hash function to S to generate a 256-bit hash (K'_{n}) and truncates it to the first 32-bits (K_n). This operation is performed repeatedly to generate a one-way key-chain $(K_n, K_{n-1}, \ldots, K_0)$ of length *n+1*, where each key is a 32-bit value. This key-chain is used in the reverse order of generation $(K_0, K_1, \ldots, K_n)$, to prevent an attacker from inferring the next key by using the previous key.

$$\mathsf{SHA\text{-}256(S)} \longrightarrow \mathsf{K_n}' \longrightarrow \mathsf{trunc(K_n}') \longrightarrow \mathsf{K_n}$$
$$\mathsf{SHA\text{-}256(K_n)} \longrightarrow \mathsf{K_{n\text{-}1}}' \longrightarrow \mathsf{trunc(K_{n\text{-}1}}') \longrightarrow \mathsf{K_{n\text{-}1}}$$
$$\ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots\ \ldots$$
$$\mathsf{SHA\text{-}256(K_1)} \longrightarrow \mathsf{K_0}' \longrightarrow \mathsf{trunc(K_0}') \longrightarrow \mathsf{K_0}$$
$$where \{\mathsf{K_n}, \mathsf{K_{n\text{-}1}} \cdots \mathsf{K_0}\} \in \{0,1\}^{32}$$

The base-station then divides the time into equal intervals of duration $\mathrm{T_{int}}$, which is set to be equal to the duration of the paging cycle (already defined by the cellular protocol). The base-station then assigns each interval a key from the one-way key-chain. Every paging message in a paging cycle will use the key assigned to the current time interval. We choose the disclosure delay (i.e., how many intervals must pass before a given key is disclosed) to be equal to one paging cycle.

UE bootstrapping. When the UE initiates the `attach procedure`, the base-station shares the key for the previous interval $\mathsf{K_{i\text{-}1}}$ with the encrypted and integrity protected `rrc_connection_reconfiguration` message. The UE stores it and uses it later (after reception of a paging message) to verify whether the disclosed key belong to base-station's legitimate key-chain.

Signing. In each paging cycle, the base-station selects the corresponding 32-bit key $\mathsf{K_i}$ from the one-way chain and uses it to compute the MAC for any paging message m sent in that paging cycle. For this, the base-station uses the HMAC-SHA256 algorithm with the chosen key $\mathsf{K_i}$. Finally, base-station appends the truncated first 32-bits of this MAC ($\mathsf{MAC_m} \in \{0,1\}^{32}$) to the paging message, along with the key for the previous paging cycle ($\mathsf{K_{i\text{-}1}}$). The signing process is illustrated in the Figure 7.

$$\mathsf{HMAC_{K_i}(m)} \longrightarrow \mathsf{MAC_m}'$$
$$\mathsf{trunc(MAC_m}') \longrightarrow \mathsf{MAC_m}$$

Verification. The UE wakes up at its paging occasion and checks for any paging message sent by the base-station. If it receives a paging message, it buffers the message along with the 32-bit MAC value for verification. The UE then waits for the next paging cycle to obtain the key used for signing messages in the last paging

cycle. The UE first verifies if the disclosed key is valid by checking if the previously disclosed key can be derived from it. After verifying the key's authenticity, the UE then verifies the MAC of the previously buffered paging message using the currently disclosed key.

Embedding MAC and key in the paging message and maintaining backward compatibility. In 4G and 5G cellular networks, a paging message contains two sections: 16 UE-specific paging records; 1 shared record. Each of the 16 paging records (of maximum 48 bit length) contain the identity of the UE for which there is a pending network service. The shared record essentially contains information for emergency notifications for all the UEs who share the same paging occasion. Since the current paging packet-format does not have any provision for including MAC and key, the base-station uses 2 paging records (out of 16) to accommodate the 32-bit key (in 1 paging record) and the 32-bit MAC (in 1 paging record) in the paging message. Our proposed defense mechanisms, thereby, do not break any existing functionality of the cellular devices and are thus backward-compatible with the legacy devices.

If the cellular networks deploy our solution and a cellular device unable of handling the P-TMSI or the message authentication code (MAC) wants to connect, the UE and the network falls back to the existing paging protocol. The capability of the UE to handle P-TMSI and MAC in the paging message is notified to the core network with the UE initiated `attach_request` message during the initial `attach procedure`.

Verification frequency. Our design requires empty paging messages (i.e., no paging records) to be signed too. When a UE receives a paging message on its paging occasion, there are three possible approaches to verify the authenticity of the paging message.

The first approach is to *authenticate a paging message first and then check if it contains the UE's identity (P-TMSI)*. This approach allows the UE to detect any paging channel hijacking attempt because the MAC contained in fake paging messages would fail the verification step. The UE may decide to connect to another base-station once a certain number of paging message verification checks fail. With this approach, however, the UE has to authenticate every paging message, even though the message does not contain UE's P-TMSI.

To address this, we propose the second approach by which the UE *first checks if the paging message contains the UE's P-TMSI and then authenticates the message*. This approach removes the overhead of the UE having to verify messages not addressed to itself. However, with this approach, the attacker might be able to perform a paging channel hijacking attack by flooding the paging channel with fake/empty paging messages [15] to prevent legitimate ones from being received. In this case, the UE will not find its P-TMSI and thus discard the messages.

To prevent such paging channel hijacking attack, we extend the second approach by also requiring the UE to *randomly authenticate paging messages even when it does not contain any paging record for the UE*. This hybrid approach provides protection against paging channel hijacking attacks while minimizing the computational overhead.

Impact on quality of service(QoS). A drawback of this approach for broadcast authentication is the introduction of a slight delay in the verification of the paging message. The UE has to buffer the paging message for one paging cycle (320 msec to 2.56 sec) till it receives the verification key and is able to verify the paging message. This will have a small impact on the QoS, as the notification of a call or an SMS will be delayed by 320 msec to 2.56 sec, i.e., the length of a single paging cycle.

5.3 Handling Exceptional Scenarios

What if the RRC connection request is dropped? When refreshing P-TMSI and paging occasion after receiving a paging message, a case of de-synchronization could arise among the P-TMSI values at the UE and the MME if the service request procedure is disrupted (due to dropped/lost `rrc_connection_request` message) before completion. This may create a situation where the MME updates its P-TMSI, but the UE does not. In this case, the UE will look for both its current P-TMSI as well as the next P-TMSI in the identifiers/paging records in the paging messages and will maintain two paging occasions per paging cycle until it receives a paging message for a particular P-TMSI.

What if the UE switches from one base-station to another? The UE will set up the (RRC layer) connection with the target base-station first and then perform the `tracking area update` (TAU) procedure to reconnect to the core network. In the case of refreshing P-TMSI in each paging cycle, the target base-station will generate a new random secret seed and share it with the UE to *reestablish* the P-TMSI lists.

What would be the impact of broadcast authentication on emergency warnings? Our broadcast authentication approach requires the UE to buffer the paging message for one paging cycle until it receives the verification key in the next paging cycle. Thus, in order to verify the paging messages containing emergency

warnings, the UE has to wait for anywhere between 320 ms and 2.56 seconds which is fairly insignificant and will not affect the quick dissemination of the emergency alerts.

UE without SIM card. Typically, a UE without a SIM card also listens to the paging broadcast channel [3] to receive emergency notifications or emergency phone calls, if there is any. In such cases, the UE and the network do not perform any mutual authentication and thus they compute the paging occasion based on the UE's IMEI (International Mobile Equipment Identity) instead of IMSI/TMSI. We also comply with the standard's suggestion to use IMEI for computing the paging occasion of a device without SIM card, because our proposed defenses are based on the assumption that a UE with a valid SIM card will generate the list of P-TMSIs with a secret seed which is shared only after the mutual authentication with the core network. However, since emergency warnings and emergency phone calls cannot be triggered by the adversary, we argue that paging occasion based on the IMEI for a UE without SIM card is not vulnerable to linkability (correlation) and location tracking attacks.

6 Security Analysis

In this section, we discuss how our proposed defense mechanisms provide the security and the privacy guarantees we set out to achieve in Section 4.1. As discussed earlier, our defenses do not aim to provide resiliency against DoS attacks. We assume that the initialization/bootstrapping phases for the P-TMSI refresh scheme and PTESLA are performed in accordance with the protocol definitions provided in Section 5. We also assume that the attacker is not able to tamper with or learn the information contained in the encrypted and integrity protected messages `rrc_connection_reconfiguration` and `attach_accept` belonging to the `attach procedure`.

• *The adversary is not able to confirm whether the user is present in the target area:* Our proposed schemes achieve this by introducing a new ephemeral UE identifier P-TMSI with policies to refresh the P-TMSI values, either at every paging cycle or after every paging message reception, such that no two paging messages addressed to a UE have the same UE identifier. This design protects against identity correlation attack [18, 23]. We also propose to update the paging occasion based on the UE's P-TMSI to protect against the ToRPEDO-type linkability attack [16] that exploits static paging occasions for the UE and maps the user's phone number to a unique paging occasion. Both these defenses prevent location tracking attacks against a UE.

• *The adversary cannot learn any information about the IMSI of the target UE by observing the paging messages or the paging occasions.* Since our proposed approaches remove the dependency of the paging occasion on the UE's static IMSI and instead use the variable P-TMSI to compute paging occasion and address the UE in paging records, there is no leakage of any information about the UE's IMSI value.

• *The adversary cannot inject fake paging messages or emergency alerts without being detected.* The UE, with our proposed broadcast authentication scheme (PTESLA), always verifies the authenticity of a paging message addressed to itself and also randomly verifies even when the paging message does not contain any paging record for the UE. The UE, therefore, can detect the fake paging messages and any suspicious paging channel hijacking attempts when the MAC verification fails.

• *The adversary cannot predict the value of next TMSI by observing current value.* With our proposed defenses in place, the cellular device and the core network (or the base-station) will use CSPRNG [14], a random number generator, which has been proven to be secure against crypt-analysis, to generate the list of P-TMSIs. An important property of a CSPRNG is that observing previously generated random numbers from the CSPRNG gives no usable information about what the next random numbers are going to be. This makes it impossible for the attacker to predict the next P-TMSI value by observing the P-TMSIs, making the attacker unable to correlate two paging messages addressed to the same UE. We use an HMAC_DRBG [9] for our purposes, which uses HMAC to generate random numbers. It has a formal proof of security and has been proven to be cryptographically secure [25].

• *The adversary cannot perform a Man-in-the-Middle (MitM) attack without being detected.* Since PTESLA sends the signing keys in plain-text, a MitM attacker may try to capture the key to sign a fake paging message and send that to the UE. However, if the attacker observes key K_i, and uses it to generate and sign a fake paging message in the current paging cycle $i + 1$, the authentication check will fail as the UE will use the key K_{i+1} to authenticate that message instead of K_i. It is, therefore, impossible for the attacker to sign and inject a message in a paging cycle without being detected.

7 Evaluation

The goal of this section is to evaluate the effectiveness and the overhead induced by our proposed defense mechanisms with respect to computational, memory, and communication costs.

7.1 Testbed Setup

Cellular network setup. We implement and evaluate our proposed schemes on a testbed setup for 4G cellular networks. Since the paging procedures for 4G and 5G cellular networks are fairly similar, the overhead and security guarantees for 5G network can be seamlessly extrapolated from the results for the 4G cellular network.

Hardware and software components. For our testbed, we use a USRP B210 [6] software-defined-radio (SDR) board connected to a desktop PC with an Intel Core i7-6700K at 4.00 GHz and 32 GB DDR4 RAM to function as a legitimate cellular base-station. The PC runs Ubuntu 18.04 operating system. The base-station and the core network are set up on the same machine using the open-source implementations, srsENB and srsEPC, respectively, provided by the srsLTE [5] library. We modify these open-source libraries to evaluate our defense mechanisms.

To mimic a UE, we use another USRP B210 [6] SDR board connected to a laptop. We modify the srsUE implementation provided by srsLTE open-source library to evaluate our solutions. We were, however, unable to use commercial mobile devices to test our implementation, because the modems' firmware for those devices are proprietary and closed-source. It is, therefore, important to point out that the computational costs obtained from the experimental analysis of our protocols will not be indicative of the computational costs for actual UEs. However, our results show that the resource costs of our solutions are fairly negligible and we expect those costs will also be negligible for the actual UEs.

Cryptographic libraries used. We use the mbedTLS crypto library [4] for generating cryptographically secure random numbers. We use HMAC_DRBG to generate the random numbers required for the updating the P-TMSI values.

Parameter setup. For our evaluations, we set up the length of paging cycle T to 128 radio frames (1.28 second) and nB to 128.

7.2 Evaluation Results

In this subsection, we discuss the computational, memory, and communication costs incurred by our P-TMSI and paging occasion refresh policies and the PTESLA broadcast authentication scheme.

	Refreshing after each paging cycle	Refreshing after every paging message
Number of elements in $\mathcal{L}_s$	100	10
Time taken to generate/regenerate $\mathcal{L}_s$	0.14 msec	0.017 msec
$\mathcal{L}_s$ regeneration required after	128 seconds	Depends on number of notifications
Memory requirement for MME (per UE)	0 bytes	40 bytes
Memory requirement for base-station (per UE)	400 bytes	0 bytes
Memory requirement for UE	400 bytes	40 bytes

Table 2. Comparison between two P-TMSI refresh policies based on computation and memory requirements.

7.2.1 Results for P-TMSI Refresh Policies

Computational overhead: For the P-TMSI refresh policy in which the *P-TMSI is updated after each paging cycle*, we set up the list of P-TMSIs $\mathcal{L}_s$ to contain 100 random numbers. The UE and the base-station, therefore, regenerate the $\mathcal{L}_s$ periodically after every 100-th paging cycle, i.e., 128 seconds (each paging cycle lasts 1.28 seconds). On the other hand, the policy of *updating P-TMSI after a paging message reception* requires a smaller number of P-TMSI renewals since the number of paging messages usually sent to a UE within a particular time-interval is fairly low in practice [16, 18]. Therefore, in the second case, we set up the list $\mathcal{L}_s$ to contain only 10 random numbers. To summarize, the policy of refreshing TMSI after each paging cycle requires the base-station to generate 100 random numbers for every UE in its service area every 128 seconds whereas the policy of refreshing P-TMSI after every paging message reception requires only 10 random numbers (assuming 10 paging messages arrive for the UE in that period) and thus significantly reduces the computation overhead.

Updating P-TMSI after each paging cycle requires a 32-bit random seed to be generated by the base-station and shared with the UE during the `attach procedure`. The base-station takes 0.03 msec to generate the random seed for a specific UE during the initial context setup request. This random seed is sent with the `rrc_connection_reconfiguration` message to the UE. Both the base-station and the UE initialize the HMAC_DRBG context using this seed and generate a list $\mathcal{L}_s$ consisting of one-hundred 32-bit random numbers. This list $\mathcal{L}_s$ requires 0.14 msec to be generated (or regenerated once depleted). The policy of *updating P-*

	P-TMSI update policy 1/policy 2	Hussain et al. [16]	3GPP proposal [2]
Fake paging messages injected by base-station	0/0	18740	0
Fake paging messages received by UE	0/0	31	0
GUTI reallocation procedures executed	0/0	0	10
New TMSI/P-TMSI assigned	937/10	0	10
UE Battery drain	≈ 0 mA	3654 mA	1170 mA

Table 3. Comparison of our P-TMSI refresh solution with other defenses to prevent TMSI correlation attacks. We assume a 20 minute time-period (937 paging cycles) for the comparison and assume that the UE receives 10 legitimate paging messages during this period [16].

TMSI after a paging message is computationally much more efficient. In this policy, the MME generates a 32-bit random seed and shares it with the UE using the `attach_accept` message. Both the MME and the UE initialize the HMAC_DRBG context using this seed and generate a list $\mathcal{L}_s$ consisting of ten 32-bit random numbers in 0.017 msec.

Memory overhead: When updating the P-TMSI after each paging cycle, our approach requires the device and the base-station to maintain the P-TMSI store as a list $\mathcal{L}_s$ of 100 random numbers where each number is 32 bit long. With this approach, the UE and the base-station, therefore, have to maintain 400 bytes of data for every connected UE. On the other hand, the P-TMSI update policy in which the P-TMSI is changed after a paging message arrival, uses the P-TMSI store as a list $\mathcal{L}_s$ of 10 random numbers, which implies that the UE needs to store only 40 bytes of data and the MME requires the same amount of data for every UE in its tracking area. Note that the variable size of $\mathcal{L}_s$ allows a trade-off opportunity between performance and memory overhead. For instance, a larger list $\mathcal{L}_s$ with 1000 random numbers will require less frequent regenerations, but will take up to 4 KB memory space whereas an $\mathcal{L}_s$ with 100 random numbers requires only 400 bytes of memory.

Communication overhead: The communication overhead is modestly low for both the P-TMSI refresh policies since only the 32-bit random seed K needs to be communicated among the MME, base-station, and UE.

We summarize the performance, memory and communication overhead of the two P-TMSI refresh approaches in Table 2. As demonstrated, the overheads induced by these P-TMSI refresh approaches are minimal and thus signify the efficiency of our proposed schemes.

Comparison with other TMSI correlation defenses. Table 3 presents a comparison of our proposed P-TMSI refresh policies with existing approaches [2, 16] to prevent TMSI correlation attacks. For this, we consider a 20 minute time-window which corresponds to 937 paging cycles (assuming a paging cycle lasting 1.28 sec) and assume that the UE receives 10 legitimate paging messages containing the device's identity during this period. To protect against the ToRPEDO-style correlation attack, Hussain et al. [16] propose adding fake paging messages (600 messages per 30 paging cycles) to normalize the TMSI distribution, thus resulting in each UE receiving an average 31 fake paging messages during this period. The new 3GPP 5G specifications [3] suggest running the `configuration update procedure` (resp., `GUTI reallocation procedure` for 4G) to assign a new TMSI every time a paging message is sent/received. We consider this approach to be the *baseline*. We leverage an existing power model [10] to calculate the approximate amount of energy overhead incurred by the UEs due to all these security approaches. As is evident from Table 3, our approaches for P-TMSI refresh have negligible battery consumption (≈ 0 mA) overhead because they do not require the UE to handle any extra injected paging messages or initiate the `GUTI reallocation procedure` for refreshing the TMSI which would require 3654 mA and 1170 mA for the solutions proposed by Hussain et al. [16] and 3GPP [2], respectively. Our P-TMSI refresh solutions, therefore, would thus be more cost-efficient for cellular network providers as compared to other proposals.

7.2.2 Results for Paging Occasion Randomization

There is no extra memory required or communication cost for enabling the paging occasion randomization because it relies on the refresh policies of P-TMSI values themselves. Also, the arithmetic operations involved in the paging occasion calculation are trivial and, thus, unlikely to have any impact on the computational overhead.

7.2.3 PTESLA Authentication Results

For the base-station bootstrapping step of PTESLA, the base-station takes 14 msec to generate a key-chain containing 10,000 keys (32-bit each) and requires a memory-space of 40 KB for storing it. It takes the base-station 0.0029 msec to sign a paging message. Due to the delayed key-disclosure required for PTESLA, the base-station sends the verification key for the previous paging messages along with the paging message in the current paging cycle. So, the UE has to buffer the paging message for a duration of one paging cycle (1.28

Number of keys in the key-chain	10,000
Time to generate the key-chain	14 msec
Time to sign a message	0.0029 msec
Time to verify a message	0.0031 msec
Communication overhead	64 bits
Memory requirement for MME	0 bytes
Memory requirement for base-station	40 KB
Memory requirement for UE	0 bytes
End-to-end delay	1280.006 msec

Table 4. PTESLA authentication overhead.

msec default) and then it takes 0.0031 msec for verification. The end-to-end message authentication delay turns out to be equal to 1280.006 msec, assuming the paging occasion stays the same for consecutive paging cycles. We do not consider the communication latency for this calculation as it is negligible (less than 1 ms) compared to the total end-to-end delay. PTESLA also uses 64 bits of a paging message, i.e., 2 paging records for incorporating 32-bit MAC and 32-bit key in every paging message. The results are summarized in Table 4. As demonstrated, the performance, memory, and communication overheads induced by PTESLA are modest and will not have a negative impact on the QoS in a cellular deployment.

7.2.4 Total Overhead of our Defenses

In this section, we summarize the total overhead incurred per paging cycle by our proposed solutions. The number of UEs served by a base-station and an MME (controlling multiple base-stations) varies with the area (e.g., mall/movie-theater, stadium versus rural area) and the time of the day (rush hours versus midnight). For evaluating the total overhead of our techniques, we assume there are 1000 UEs being served by a particular base-station and 10,000 UEs being served by an MME in a particular tracking area. A paging cycle with system parameters $\mathsf{T} = \mathsf{nB} = 128$ can have maximum 128 paging messages. We also assume that every UE will receive one paging message in this period. For the sake of comparing the two refresh policies, we divide the cost of generating P-TMSI list L_s by the number of elements in L_s to compute the cost for generating a single P-TMSI value (0.0014 msec). Table 5 presents the summary of total overhead for the two P-TMSI refresh policies.

Performance overhead: When P-TMSI is updated after each paging cycle, our proposed solution requires approximately 1.77 msec at the base-station (1.4 msec for generating P-TMSI for 1000 UEs + 0.37 msec for signing 128 paging messages) and 0.0045 msec at the UE side (0.0014 msec for generating one P-TMSI + 0.0031 msec for verification of one paging message) at the UE side. On the other hand, when the P-TMSI is updated only after an actual paging message is received by the UE, our proposed defense mechanism induces an overhead of 14 msec at the MME (to generate P-TMSI for 10,000 UEs) and 0.0014 msec at the UE (to generate 1 P-TMSI) for every paging message transmission. The signing overhead at the base-station and the verification overhead at the UE (0.37 msec and 0.0031 msec respectively) remains the same as the baseline solution proposed by 3GPP [2].

	Policy 1 + PTESLA	**Policy 2 + PTESLA**
Performance overhead (UE)	0.0045 msec	0.0045 msec
Performance overhead (base-station)	2.88 msec	1.48 msec
Performance overhead (MME)	0 msec	14 msec
Memory overhead (UE)	400 bytes	40 bytes
Memory overhead (base-station)	440 KB	40 KB
Memory overhead (MME)	0 KB	400 KB
Communication overhead	64 bits	64 bits

Table 5. Comparison of P-TMSI and paging occasion refresh policies accompanied with the proposed authentication scheme PTESLA with respect to performance, memory and communication overhead. Policy 1 is when the P-TMSI is updated every paging cycle and policy 2 refers to the update of P-TMSI only after a paging message reception.

Memory overhead: The P-TMSI refresh policy where the P-TMSI is updated at each paging cycle would require a memory of 440 KB (400 KB for 1000 P-TMSI stores with 100 elements each + 40 KB for TESLA key-chain) at the base-station and only 400 bytes at the UE side. The reason for UE's low memory overhead is due to the fact that a UE only needs to maintain the P-TMSI store for itself and does not require to maintain the TESLA key-chain. Similarly, the P-TMSI refresh policy where the P-TMSI is updated after a paging message reception would require a memory space of 400 KB at the MME side (for 10,000 P-TMSI stores with 10 elements each), 40 KB at the base-station (for TESLA key-chain) and only 40 bytes (for P-TMSI store) at the UE.

Communication overhead: Our solutions incur a minimal communication overhead of 64 bits while bootstrapping and a total overhead of 64 bits for the TESLA MAC (32-bit) and key (32-bit) attached to every paging message.

As is clear from this discussion, our proposed solutions have minimal performance, memory, and communication overhead for practical deployments. This keeps the UE battery consumption overhead negligible and makes these solutions scalable at the MME and the

base-station, and viable even while serving large numbers of UEs.

8 Related Work

Although a few defenses have been proposed against ToRPEDO-style correlation attacks [16], they induce significant performance overhead or require changing the paging protocol substantially. Furthermore, no solutions have been proposed for efficient broadcast authentication of paging messages. In contrast, our proposed approach addresses both these security and privacy concerns of the paging protocol.

Federrath et al. [12] proposed a sequential search strategy for determining UE locations instead of storing these locations in a centralized database, to protect the UE's location privacy. This approach, however, introduces significant delays in call establishment, requires a significant overhaul of the paging protocol, and does not address the possibility of injection of malicious messages by attackers.

Kune et al. [18] proposed various solutions to the location tracking problem, including sending the paging messages to multiple location areas where the device is present most of the time. Such an approach, however, introduces significant overhead in terms of number of location areas to which the paging message is sent and may adversely affect user experience due to the introduced delay.

Ta et al. [24] proposed assigning each UE a unique tag and superimposing it onto the paging transmitted waveform. This approach, however, requires physical layer modifications, may degrade the user experience when the signal-to-noise ratio is too low, and still leaks information about the IMSI based on observing the paging occasion by the ToRPEDO attack [16].

Nicanfar et al. [19] proposed changing the temporary ID (GUTI) and secret key (to access the new base-station) for UEs by the handover mechanism, during the movement of the UE between two location areas, to hide the real user ID. They, however do not address the location tracking issue if the user stays in the same location area by sniffing the paging messages.

Dittler et al. [11] proposed ANOTEL, an overhaul of location management in cellular networks. They introduced two new entities, called pseudonym provider and location provider, to map users to their respective locations. This approach, however, introduces significant performance overhead and requires introduction of two new trusted third-parties.

Hussain et al. [16] outlined a defense against their ToRPEDO attack by adding noise in the form of fake paging messages for perturbing the underlying paging message distribution. Such an approach, however, cannot completely prevent ToRPEDO attack, it just raises the bar for the attack. On top of that, it incurs a significant performance overhead in areas where the number of UEs is small.

The latest 3GPP 5G standard [2] mentions the use of the variable TMSI to calculate the paging occasion similar to our proposal. It also mentions the invocation of the `GUTI reallocation procedure` after the UE receives a paging message to update the TMSI assigned to the UE. However, our P-TMSI update mechanisms do not require the explicit invocation of the `GUTI reallocation procedure`. Our solutions assign a new P-TMSI value to the UE with a non-interactive way preventing several round-trips of communication between the UE and the core network.

9 Conclusion and Future Work

The paper proposes a retrospective defense mechanism for protecting cellular paging protocols from security and privacy attacks. By analyzing prior attacks, we identified design weaknesses which our proposed defense mechanism thwarts. To demonstrate the enhanced paging protocol's deployability, we implement the protocol using software-defined radios and open-source protocol stack. We have also performed empirical evaluation which suggests that our paging protocol can be deployed without incurring substantial overhead while maintaining backward-compatibility and effectiveness.

In future, we will explore if our proposed defenses can be applied to Bluetooth, Bluetooth Low Energy, ZigBee [7], and similar IoT transport protocols in which broadcast messages sent by the controller/master devices do not have confidentiality, adequate anonymity, and authentication/integrity protection and are subject to similar exploitations. Finally, a formal security proof of our mechanism is also the subject of future work.

Acknowledgement

We thank our shepherd, Boris Koepf, and the anonymous reviewers for their valuable suggestions. This work is supported by NSF grants CNS-1657124 and CNS-1719369, United States ARO grant W911NF-16-1-0127, NSA's Science of Security Lablet program through North Carolina State University, Intel, and a grant by Purdue Research Foundation.

References

[1] 3GPP, Specification number TS 24.301, Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS). https://www.etsi.org/deliver/etsi_ts/124300_124399/124301/10.03.00_60/ts_124301v100300p.pdf.

[2] 3GPP, Specification number TS 24.501, Non-Access-Stratum (NAS) protocol for 5G System (5GS). https://www.etsi.org/deliver/etsi_ts/124500_124599/124501/15.00.00_60/ts_124501v150000p.pdf.

[3] 3GPP, Specification number TS 38.304, User Equipment (UE) procedures in idle mode and in RRC Inactive state. https://www.etsi.org/deliver/etsi_ts/138300_138399/138304/15.03.00_60/ts_138304v150300p.pdf.

[4] mbedTLS. https://tls.mbed.org/source-code.

[5] srsLTE. https://github.com/srsLTE/srsLTE.

[6] USRP B210. https://www.ettus.com/all-products/UB210-KIT/.

[7] Zigbee. https://zigbee.org/.

[8] American Bankers Association et al. Public key cryptography for the financial services industry: The elliptic curve digital signature algorithm (ecdsa). *ANSI X9*, pages 62–1998.

[9] Elaine Barker, John Kelsey, et al. NIST special publication 800-90A: Recommendation for random number generation using deterministic random bit generators. 2012.

[10] Xiaomeng Chen, Jiayi Meng, Y Charlie Hu, Maruti Gupta, Ralph Hasholzner, Venkatesan Nallampatti Ekambaram, Ashish Singh, and Srikathyayani Srikanteswara. A Fine-grained Event-based Modem Power Model for Enabling In-depth Modem Energy Drain Analysis. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):45, 2017.

[11] Tim Dittler, Florian Tschorsch, Stefan Dietzel, and Björn Scheuermann. ANOTEL: Cellular networks with location privacy. In *Proceedings of the 2016 IEEE 41st Conference on Local Computer Networks (LCN)*.

[12] Hannes Federrath, Anja Jerichow, Dogan Kesdogan, Andreas Pfitzmann, and Dirk Trossen. Minimizing the average cost of paging on the air interface-an approach considering privacy. In *Proceedings of the 1997 IEEE 47th Vehicular Technology Conference. Technology in Motion*, volume 2, pages 1253–1257. IEEE, 1997.

[13] Byeongdo Hong, Sangwook Bae, and Yongdae Kim. GUTI Reallocation Demystified: Cellular Location Tracking with Changing Temporary Identifier. In *Proceedings of the 25th Annual Network and Distributed System Security Symposium, NDSS 2018*.

[14] Andrew Huang. Hacking the Xbox: an introduction to reverse engineering. 2002.

[15] Syed Rafiul Hussain, Omar Chowdhury, Shagufta Mehnaz, and Elisa Bertino. LTEInspector: A systematic approach for adversarial testing of 4G LTE. In *Proceedings of the 25th Annual Network and Distributed System Security Symposium, NDSS 2018*.

[16] Syed Rafiul Hussain, Mitziu Echeverria, Omar Chowdhury, Ninghui Li, and Elisa Bertino. Privacy Attacks to the 4G and 5G Cellular Paging Protocols Using Side Channel Information.

[17] Jonathan Katz, Alfred J Menezes, Paul C Van Oorschot, and Scott A Vanstone. *Handbook of applied cryptography*. CRC press, 1996.

[18] Denis Foo Kune, John Koelndorfer, Nicholas Hopper, and Yongdae Kim. Location leaks on the GSM air interface. In *Proceedings of the 19th Annual Network and Distributed System Security Symposium, NDSS 2012*.

[19] Hasen Nicanfar, Javad Hajipour, Farshid Agharebparast, Peyman TalebiFard, and Victor CM Leung. Privacy-preserving handover mechanism in 4G. In *Proceedings of the 2013 IEEE Conference on Communications and Network Security (CNS)*.

[20] Adrian Perrig, Ran Canetti, J Doug Tygar, and Dawn Song. Efficient authentication and signing of multicast streams over lossy channels. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy, S&P 2000*.

[21] Adrian Perrig, Ran Canetti, J Doug Tygar, and Dawn Song. The TESLA broadcast authentication protocol. *Rsa Cryptobytes*, 5(2):2–13, 2002.

[22] Ronald L Rivest, Adi Shamir, and Leonard Adleman. A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2):120–126, 1978.

[23] Altaf Shaik, Ravishankar Borgaonkar, N Asokan, Valtteri Niemi, and Jean-Pierre Seifert. Practical attacks against privacy and availability in 4G/LTE mobile communication systems. *arXiv preprint arXiv:1510.07563*, 2015.

[24] Tuan Ta and John S Baras. Enhancing privacy in LTE paging system using physical layer identification. In *Data Privacy Management and Autonomous Spontaneous Security*, pages 15–28. Springer, 2012.

[25] Katherine Q Ye, Matthew Green, Naphat Sanguansin, Lennart Beringer, Adam Petcher, and Andrew W Appel. Verified correctness and security of mbedTLS HMAC-DRBG. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*.