Privacy Attacks to the 4G and 5G Cellular Paging Protocols Using Side Channel Information

Syed Rafiul Hussain*, Mitziu Echeverria†, Omar Chowdhury†, Ninghui Li* and Elisa Bertino*
* Purdue University, † The University of Iowa
Email: *{hussain1, ninghui, bertino}@purdue.edu, †{mitziu-echeverria, omar-chowdhury}@uiowa.edu

***Abstract*—The cellular paging (broadcast) protocol strives to balance between a cellular device's energy consumption and quality-of-service by allowing the device to *only* periodically poll for pending services in its idle, low-power state. For a given cellular device and serving network, the exact time periods when the device polls for services (called the *paging occasion*) are fixed by design in the 4G/5G cellular protocol. In this paper, we show that the fixed nature of paging occasions can be exploited by an adversary in the vicinity of a victim to associate the victim's soft-identity (e.g., phone number, Twitter handle) with its paging occasion, with only a modest cost, through an attack dubbed `ToRPEDO`. Consequently, `ToRPEDO` can enable an adversary to verify a victim's coarse-grained location information, inject fabricated paging messages, and mount denial-of-service attacks. We also demonstrate that, in 4G and 5G, it is plausible for an adversary to retrieve a victim device's persistent identity (i.e., IMSI) with a brute-force `IMSI-Cracking` attack while using `ToRPEDO` as an attack sub-step. Our further investigation on 4G paging protocol deployments also identified an *implementation oversight* of several network providers which enables the adversary to launch an attack, named `PIERCER`, for associating a victim's phone number with its IMSI; subsequently allowing targeted user location tracking. All of our attacks have been validated and evaluated in the wild using commodity hardware and software. We finally discuss potential countermeasures against the presented attacks.**

I. Introduction

When a cellular device is not actively communicating with a base station, it enters an idle, low-energy mode to conserve battery power. When there is a phone call or an SMS message for the device, it needs to be notified. This is achieved by the paging protocol, which strives to achieve the right balance between the device's energy consumption and timely delivery of services such as phone calls. When there is one or more pending services for a device, the network's Mobile Management Entity (MME) asks base station(s) to broadcast a paging message, which includes the Temporary Mobile Subscriber Identity (TMSI) of the device. TMSI is randomly assigned by the MME to the device, and it is recommended that the TMSI for a device changes frequently.

Kune et al. [1] showed that a user's presence in a geographical area can be identified by a sniffing attack that exploits the fact that in practice the TMSI is changed infrequently. An attacker places multiple phone calls to the victim device in a short period of time and sniffs the paging messages. If the most frequent **TMSI** among the paging messages appears frequently enough, then the attacker concludes that the victim device is present. Shaik et al. [2] found that paging messages can be triggered with SMS as well as notifications from instant messengers; consequently, the same attack in [1] can be mounted by these means. These attacks exploit the deployment weakness that TMSI is infrequently changed. Hong et al. [3] showed that some deployments choose the new TMSI predictably even when it is changed. Furthermore, such attacks can be made stealthy in the sense that the attacker can make phone calls and send SMS messages that trigger paging messages without alerting the user of the victim device.

The natural defense against those attacks is to change TMSI frequently and use random, unpredictable values for new TMSI. This renders existing attacks ineffective. However, in this paper, we show that *even if different and unrelated TMSIs are used* in every two subsequent paging messages, it is possible to carry out a similar attack to verify whether a victim user is present in a geographical cell.

We propose the `ToRPEDO` (TRacking via Paging mEssage DistributiOn) attack, which is able to verify whether a victim device is present in a geographical cell with less than 10 calls, even under the assumption that the TMSI changes after each call. Furthermore, in the process, the attacker learns exactly when a device wakes up to check for paging messages and 7 bits of information of the device's International Mobile Subscriber Identity (IMSI). This knowledge enables two other new attacks that lead to full recovery of the device's IMSI.

When the TMSI is changed each time, one may no longer link a call made by the attacker and the resulting paging message. The key insight under our novel attack is that the paging protocol requires synchronization between the base station and the device. The LTE paging protocol uses a paging cycle of T frames, each of which is 10ms long. The default value of T is 128. Each device has a Paging Frame Index (PFI), which is determined by its IMSI, and the device wakes up only once during a paging cycle, at the frame indexed by its PFI. The base station broadcasts the paging message for the device at these frames. When multiple calls for a device are made, their corresponding paging messages will occur in frames indexed by the same PFI. When the base rate of paging messages is low, that is, paging messages only appear in a small fraction of all frames, the attacker can identify which PFI is "too busy", and thus the victim device's PFI.

Leveraging this insight, we first designed two simple attacks. A *filtering-based* attack incrementally rules out candi-

Network and Distributed Systems Security (NDSS) Symposium 2019
24-27 February 2019, San Diego, CA, USA
ISBN 1-891562-55-X
https://dx.doi.org/10.14722/ndss.2019.23442
www.ndss-symposium.org

date PFI values in which no paging messages were received. This attack, however, is ineffective when expected paging messages are absent or seriously delayed, which can occur due to network congestion as well as sniffer errors. A *counting-based* attack accounts for possibly absent or delayed paging messages by incrementally ruling out PFI values that have seen too few paging messages. Although the accuracy increased in comparison to the *filtering-based* approach, the number of phone calls required to reliably calculate the victim's PFI is still relatively high (i.e., ~ 15), and the accuracy decreases when the absence rate for paging messages increases. This led us to our final ToRPEDO attack which leverages all available information, including the exact delay between the time when the call is made and the time when the paging message is observed, and the exact number of paging records in each frame. ToRPEDO calculates the likelihood of seeing the observations of paging messages when the victim device's PFI takes any one value in $0, 1, \cdots, \mathrm{T}-1$ as well as when the victim's device is not present. If the ratio between the top two candidates' likelihood is above a predefined threshold, we can conclude that the user is present in the current cell and the user's PFI is the candidate with the highest likelihood. In our experiments, this approach yields the highest accuracy (100%) while requiring only 8 phone calls on average.

ToRPEDO impacts. ToRPEDO is not only applicable to 4G but also to the current version of **5G**. Once the attacker knows the victim's paging occasion from ToRPEDO, the attacker can hijack the victim's paging channel [4]. This would consequently enable the attacker to mount a denial-of-service attack by injecting fabricated, empty paging messages, thus blocking the victim from receiving any pending services (e.g., SMS). The attacker can also inject fabricated emergency messages (e.g., Amber alert) using paging channel hijacking [4]. *With* ToRPEDO*, the attacker can also detect the victim's presence in any cellular area provided that the attacker has a sniffer in that area*. Additionally, for a targeted attack, if the attacker is aware of the victim's often-visited locations, then the attacker can set up sniffers on those locations to create the victim's cell-level mobility profile. ToRPEDO can also enable the attacker to detect the connection status (i.e., idle/connected) of the victim's device leading to privacy issues. Finally, ToRPEDO can also be used to mount other attacks, for instance, the PIERCER and IMSI-Cracking attacks discussed below.

PIERCER attack for 4G. Our investigation of paging protocol deployments revealed that in some exceptional cases, contrary to conventional wisdom and 3GPP recommendations, some service providers use IMSIs instead of TMSIs in paging messages to identify devices with pending services. A simple manual testing revealed that it is possible to give the service provider the impression that the exceptional case is occurring which forces it to reveal the victim's IMSI. We exploited this weakness to design the PIERCER (Persistent Information ExposuRe by the CorE netwoRk) attack which enables an attacker with the knowledge of the victim's phone number, a sniffer, and a fake base station in the victim's cell to associate the victim device's IMSI with its phone number while using ToRPEDO as an attack sub-step. The dangers of PIERCER are well known. Precisely, PIERCER can enhance prior attacks, which require knowledge of victim's IMSI, to a level where just knowing the victim's phone number is sufficient [2], [4]–[7].

IMSI-Cracking attack for 4G/5G. We also observed that ToRPEDO enables an attacker with the knowledge of the victim's phone number to retrieve the victim's IMSI by launching a brute-force attack. For US subscribers, IMSIs can be represented as 49-bit binary numbers. IMSI's leading 18-bits (i.e., the mobile country code and the mobile network code) can be obtained from the phone number using paid, Internet-based home location register lookup services [8]. Identifying the victim's paging occasion with ToRPEDO additionally leaks the trailing 7 IMSI bits for US subscribers leaving 24 bits for the attacker to guess. Using a brute-force attack and two oracles (one for 4G and another for 5G) we designed, the attacker can guess the victim's IMSI in less than 13 hours.

Attack validation. We have verified ToRPEDO against 3 Canadian service providers and all the US service providers. PIERCER, on the other hand, has been verified against one major US service provider and 3 major service providers of a South Asian country. We have also noticed the similar pattern of broadcasting IMSIs in paging messages by three German, four Austrian, one Icelandic, two Chinese, and one Russian service providers and speculate that PIERCER may be feasible for those service providers.

Countermeasures. Since ToRPEDO is the precursor of PIERCER and IMSI-Cracking attacks, we primarily focus on designing and evaluating possible defenses against ToRPEDO and outline additional approaches for the other two attacks. For defending against ToRPEDO, we design and evaluate a countermeasure that adds noise in the form of fake paging messages for perturbing the underlying paging message distribution. Our evaluation suggests that this can make mounting the ToRPEDO attack prohibitively expensive while incurring only moderate energy overhead for the devices.

Contributions. This paper makes the following contributions:

- We present the ToRPEDO attack that exploits a 4G/5G paging protocol weakness to enable an attacker that knows a victim's phone number to verify the victim's presence in a particular cellular area and in the process identifies the victim's paging occasion. It not only elevates prior attacks but also facilitates other newer attacks.
- We present the PIERCER attack that exploits a 4G paging protocol deployment vulnerability to allow an attacker to associate a victim's phone number with its IMSI. Apart from its immediate implication on victim's location tracking, PIERCER can also lift prior attacks that require knowledge of the victim's IMSI to only require knowledge of the victim's phone number.
- We also show that ToRPEDO can enable an attacker to mount a brute-force IMSI-Cracking attack leaking a victim's IMSI for both 4G and 5G.
- All of our attacks for 4G have been validated against real networks. We also design and evaluate a countermeasure for ToRPEDO.

II. Background

We now briefly describe the 4G LTE network architecture, and then review relevant concepts of 4G LTE paging protocol.

A. LTE Network Architecture Overview

In an LTE network, cellular devices are called User Equipments (**UE**). The core network is called the Evolved

Packet Core (**EPC**). Geographic locations are partitioned into hexagonal cell areas, each of which is serviced by a designated base station (**eNodeB**), which enables connectivity of UEs in that cell to the EPC.

Fig. 1 illustrates the architecture of a cellular network. The Mobility Management Entity (**MME**) manages the connectivity and mobility of UEs in a particular tracking area (a set of cell areas). Other EPC components include Serving Gateway (SGW), Home Subscriber Server (HSS), the PCRF (Policy and Charging Rules Function) server, and Packet Data Network Gateway (PGW).

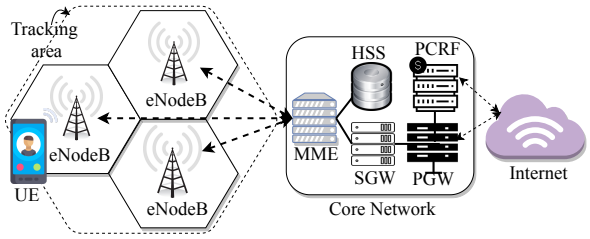


Fig. 1: The LTE Network Architecture.

4G LTE provides packet switch (PS) services (for Internet access) and VoLTE services (for Voice over LTE). It also supports circuit switch (CS) voice services using the Circuit-Switched Fallback (CSFB) technique which moves UEs from the 4G to 3G network to access 3G voice services, and then returns them to the 4G network.

A UE, equipped with a SIM-card, has a unique International Mobile Subscriber Identity (IMSI). IMSI is stored on a SIM card and generally does not change. A UE also has a Temporary Mobile Subscriber Identity (TMSI), which is the identity that is most commonly sent between the UE and the network. TMSI is randomly assigned by the MME to a UE, when the UE first connects to a base station in the tracking area. The TMSI is local to a tracking area, and so it has to be updated each time the UE moves to a new geographical area. The MME can also update a UE's TMSI if it desires to do so.

B. Network Time/Frame Synchronization in LTE

LTE supports full duplex radio communication between a UE and a base station through frequency division duplex (FDD) mode in which the transmitter and receiver operate on different carrier frequencies. In LTE-FDD, communications are carried out through *radio frames* (also called *system frames* or type-1 LTE frames) each of which spans 10 milliseconds. In this paper, we simply call them *frames*. They are indexed with a 10-bit circular counter (resetting to 0 after counting up to 1023), and thus have System Frame Numbers (SFN) in the range of $[0, 1023]$. Thus for every 10.24 seconds, SFN will repeat. Each frame is further partitioned into 10 *sub-frames* each of which spans 1 millisecond (Fig. 2).

Connection bootstrapping starts with a UE capturing the `master_info_block` (MIB) message, which is periodically (more precisely, every 40 milliseconds) broadcast by base stations. The MIB includes the current SFN and other connection-related parameters used by the UE uses to synchronize itself and connect to the base station.

C. Paging Protocol Overview

When a UE is not actively communicating with a base station, it enters an idle, low-energy mode for saving battery

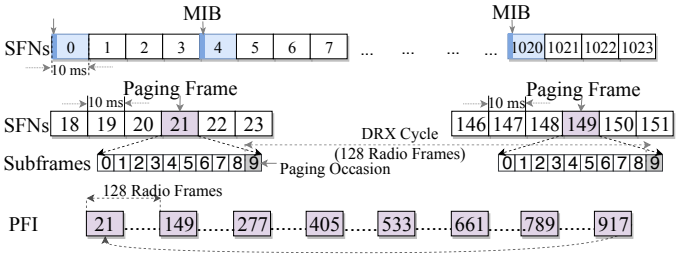


Fig. 2: Network time/frame synchronization, and paging frame and paging occasion calculation.

power. When the UE is in the idle mode, the base station uses the paging protocol to notify the UE about emergencies (e.g., Tsunami warning) or pending network services (e.g., incoming calls). This is called Discontinuous Reception (DRX).

Service paging. For notifying an idle UE about a pending service, if *smart paging* is used, the MME first asks the UE's last connected base station to broadcast a paging message for the UE. If there is no response from the UE, then the MME asks all base stations in its tracking area to broadcast the paging message. For non-smart paging, the first step is skipped. If the UE still does not respond, it is assumed that the UE either left the tracking area or is not communicating with the network.

Paging messages. A paging message contains a maximum of 16 *paging records*. Each paging record notifies one UE that there are incoming services for it. Such a record contains the MME identifier, the domain (PS or CS), and the target UE's paging identity, which can be either IMSI or TMSI.

Paging occasion. A UE in idle state wakes up periodically to check whether there is a paging message. If there is a paging message, the UE iterates over the paging records in the message while searching for its paging identity (IMSI or TMSI). It re-establishes connection with the base station if it finds its identity. The paging protocol ensures that when a base station sends a UE's paging record at a given time, the UE also wakes up at that time to check, i.e., a base station and a UE must agree on when to send/receive paging records for the UE.

D. Paging Synchronization

The paging occasion for a UE (i.e., when it wakes up to check for paging messages) is given by three numbers: the paging cycle $\mathrm{T} \in \{32, 64, 128, 256\}$; the Paging Frame Index PFI, which is an integer between 0 and $\mathrm{T} - 1$; and a sub-frame index s, $0 \leq s \leq 9$. The UE wakes up at sub-frame s in any frame whose SFN is congruent to PFI modulo T. For example, when $s = 9$, $\mathrm{T} = 128$ and $\mathrm{PFI} = 21$, the UE will wake up at sub-frame 9 in frames with SFN $21 + i * 128$ for $0 \leq i \leq 7$. For every cycle of T frames (of total length 10T ms), the UE needs to wake up for only 1 ms. We now explain how these numbers are computed.

Paging Cycle (T). The base station broadcasts a proposed value for T. The UE can choose to use that value or propose another value, in which case the minimum of these two values is chosen.

Paging Frame Index PFI. Computing the PFI requires the UE's UE_ID, defined as

$$\text{UE\_ID} = \text{IMSI} \bmod 1024.$$

In addition, it requires another public parameter (**nB**) set by the base station, and chosen from the set

$\{4\mathrm{T}, 2\mathrm{T}, \mathrm{T}, \frac{\mathrm{T}}{2}, \frac{\mathrm{T}}{4}, \frac{\mathrm{T}}{8}, \frac{\mathrm{T}}{16}, \frac{\mathrm{T}}{32}\}$. The PFI is defined using the following formula:

$$\mathrm{PFI} = \frac{\mathrm{T}}{N} \times (\mathrm{UE\_ID} \bmod N) \text{ where } N = \min(\mathrm{T}, \mathrm{nB}).$$

Equivalently,

$$\mathrm{PFI} = \begin{cases} \mathrm{UE\_ID} \bmod \mathrm{T} & \text{when } \mathrm{T} \leq nB \\ \frac{\mathrm{T}}{\mathrm{nB}} \times (\mathrm{UE\_ID} \bmod \mathrm{nB}) & \text{when } \mathrm{T} > \mathrm{nB} \end{cases}$$

Sub-frame Index. The sub-frame index s can be calculated using the lookup Table I where

$$N_s = \max\left(1, \frac{\mathrm{nB}}{\mathrm{T}}\right); \quad i_s = \left\lfloor \frac{\mathrm{UE\_ID}}{N} \right\rfloor \bmod N_s.$$

Note that when $\mathrm{nB} \leq \mathrm{T}$ and $N_s = 1$, all UEs will use sub-frame 9. When $\mathrm{nB} = 2\mathrm{T}$, a UE uses either sub-frame 4 or sub-frame 9, depending on whether its UE_ID is even or odd. When $\mathrm{nB} = 4\mathrm{T}$, the UEs are partitioned into 4 groups, each using one sub-frame.

	$i_s = 0$	$i_s = 1$	$i_s = 2$	$i_s = 3$
$N_s = 1$	9	**N/A**	**N/A**	**N/A**
$N_s = 2$	4	9	**N/A**	**N/A**
$N_s = 4$	0	4	5	9

TABLE I: Sub-frame index s

Example. An example is shown in Fig. 2 illustrating the calculation of the PFI, the System Frame Numbers during which the UE should wake up, and the sub-frame index, where nB=T=128 and UE_ID=21.

Paging occasion in 5G. In 5G, the calculating of the paging occasion is very similar to that for LTE. The paging occasion for a UE is given by the same three numbers: the paging cycle T; the Paging Frame Index PFI; and the sub-frame index s, which are computed exactly as in LTE. The only difference is that 5G introduced another public-parameter broadcast called PF_offset (clause 7 of RRC Idle mode specification [9]). The UE wakes up at the sub-frame s in any system frame whose SFN+PF_offset is congruent to PFI modulo T.

E. Abstraction of Paging

Abstractly, UEs are partitioned into a number of paging groups that time-share the channel through which paging messages are sent. Paging messages for UEs from two different groups will be sent at different times, and can be identified as such. For ease of exposition, we consider the case where T=nB when describing our attacks. Under this case, each UE wakes up once every T frames. Three of the four wireless carriers we have observed use T=nB=128, while the other uses T=128, nB=8. Our attacks can be generalized to the case where $\mathrm{T} \neq \mathrm{nB}$, since the same time-sharing principle applies.

III. ToRPEDO Attack

A. Problem setting

For our purpose, UEs are partitioned into T groups, as given by their Paging Frame Indexes (PFI). A UE's PFI depends on its IMSI. Time is divided into cycles of length 10T ms. Such a cycle consists of T frames, each of length 10ms. We number the frames within one cycle from 0 to $\mathrm{T} - 1$.

When a MME receives a service for a UE, it asks base station(s) to broadcast a paging record at the next frame that has the same number as the UE's PFI. We assume that the paging record uses the TMSI (and not the IMSI) to identify the UE. Furthermore, the phone's TMSI is updated to a new one (randomly chosen by the MME) each time the phone has responded to a call. That is, the carriers have already deployed defenses suggested by earlier work. We show that attacks are nonetheless possible.

B. Adversary Model

We assume that the adversary knows the soft identity of the target UE u_t, such as the phone number, e-mail address, or social network handler of u_t. The adversary also knows the geographical area that u_t is likely to be in (called the target area), and sets up a sniffer in that area to listen on the paging broadcast channel. That is, the adversary can make a good guess about the geographical location of u_t, although the guess does not need to be correct. Through the attack, the adversary is able to find out whether the guess is correct or not. Also note that the adversary can carry out this attack simultaneously in multiple geographical areas against a target u_t, provided that the adversary is willing to spend the resources for doing so.

The goal of the adversary is to (i) *confirm whether or not u_t is indeed in the target area*, and (ii) *when u_t is in the area, identify the UE's* PFI *(which we denote by* PFI_t*), which also yields information about the phone's IMSI.*

The adversary can make a call to u_t to trigger a paging message for u_t, and listens to the paging broadcast channel. While we use the term "*make a call*" to describe the adversary's action, the action could take the form of VoLTE or CSFB calls, SMS's, tweets, and so on.

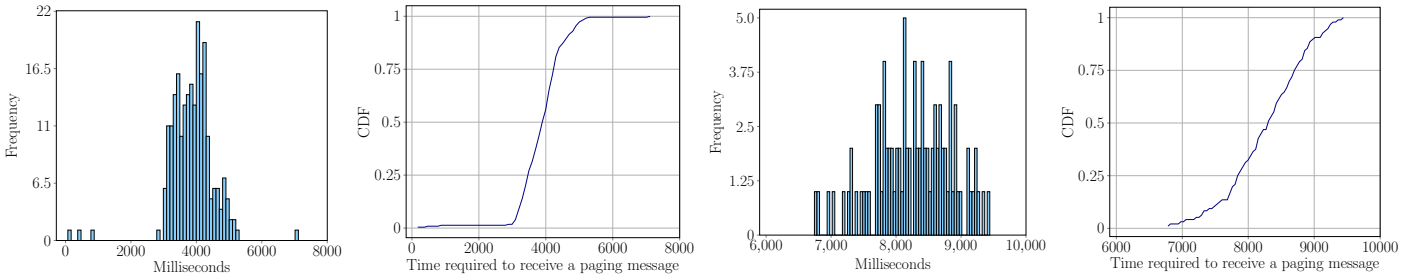
The adversary can repeat this process of making a call and listening for paging messages multiple times. We note that these calls do not need to be made continuously. The adversary can wait between calls. The only restriction is that if the total duration of the attack is too long, then the UE may move out of the target area during that time.

To assess the effectiveness of such location tracking attacks, we consider the following criteria: (1) identification accuracy: when u_t is present in the area, the rate with which the attack outputs is correct PFI, (2) presence accuracy: when u_t is not present in the area, the rate with which the attack correctly concludes that u_t is not present, and (3) the number of calls required for the attacking algorithm to reach a decision.

C. High-level Intuition of the Attack

The intuition behind the attack is as follows. Suppose, for the moment, that there is no other paging message in the system. When the adversary makes a call to u_t, there are two cases. Case one: If u_t is in the area, a paging message will be sent, and the adversary will also see the paging message at a particular PFI. Case two: u_t is not in the area, then the adversary will not see any paging message. By making a single call to u_t and checking whether there is a corresponding paging message, the adversary can infer whether u_t is in the target area, and, if u_t is, what is its PFI.

The challenge of the attack is the unrealistic assumption that there is no other paging message in the system. When there are other paging messages, it is difficult for an adversary to associate a call she made with a particular paging record. Recall that we assume that the TMSI (instead of the IMSI) is used in a paging record, and the TMSI changes each time



(a) Histogram of paging delay for SMS.

(b) Cumulative Distribution Function of paging delay for SMS.

(c) Histogram of paging delay for VoLTE phone calls.

(d) Cumulative Distribution Func. of paging delay for VoLTE calls.

Fig. 3: Distribution of paging delay i.e., the time between the event of initiating a phone call or SMS and the event of reaching a paging message to the receiver for that phone call/SMS.

a UE connects to the base station. The only information to associate a call with its corresponding paging record is timing. The paging record should be sent soon after the adversary makes a call. We call the interval between the time when the adversary makes a call and the time when the paging record is sent the *paging delay*. Unfortunately for the adversary, the paging delay is affected by many factors, and is randomized from the adversary's perspective. Furthermore, there are paging records for other UEs in the area, as well as other services for u_t that are not from the adversary. What the adversary needs to do is to test whether paging records due to adversary-initiated calls are present for each PFI in the noise of paging records generated by the background processes.

One way to establish an association between calls and the resulting paging records is obtaining a probability distribution of paging delays, and then determining a *delivery window*. Let us denote the time at which the adversary makes a call t_0; then the delivery window is given by two times t_b, t_e. Any paging record received after t_b and before t_e is considered to be in the delivery window. The choice of t_b, t_e needs to be carefully made. A window that is too narrow will miss associated paging records. On the other hand, a window that is too wide increases the probability that paging records resulting from the background sources as associated with the call.

The distributions of paging delays differ based on the type of services, e.g., SMS, or tweets. They are also dependent on the cell area and the load. The adversary can get a distribution of paging delays for each type of service, either by using historical knowledge, or by estimating the distributions for a particular incoming service before actually carrying out attacks. In Section III-F, we discuss this aspect in details.

Fig. 3 shows the empirical distributions of paging delays we observed in our experiments. Fig. 3a and Fig. 3b show the histogram and cumulative distribution for paging delay of SMS. Fig. 3c and Fig. 3d show the same for VoLTE calls. We made 500 calls for each. From the figure, one can observe that paging delays for SMS messages are between approximately 2.8 and 5.3 seconds, whereas the paging delays for VoLTE phone calls are between approximately 6.8 and 9.4 seconds.

D. Two Simple Attacks

Because of the background traffic of paging messages, the adversary is unable to use a single call to carry out the attack, and needs to make multiple calls. When the adversary makes multiple calls over time, she expects to see a paging message after each call, and all the paging messages are delivered in frames with the same number. Here the adversary relies on the observation that the base rate of paging messages in each frame is typically low. We first present two simple attacks.

1) Filtering: This attack assumes *perfect delivery of paging messages*, that is, it is assumed that each time the adversary makes a call, a paging message will be reliably received during the delivery window. Starting with the set of all possible PFI values, the adversary repeats the following steps: (1) Make a call. (2) Listen for paging messages during the delivery window. (3) Remove from the set all PFI values that do not have a paging message during the window. (4) If only one PFI value remains in the set, then it concludes that this is u_t's PFI. If the set is empty, it concludes that u_t is not in the target area.

For the Filtering attack to work reliably, paging messages need to be delivered/captured almost perfectly, which is not always possible due to different forms of noise in the sniffer-captured data. One reason is that the sniffer is not totally reliable and may miss paging messages because of signal interference. Another reason is that the call may be dropped or delayed due to network congestion, causing the paging message to either be missing, or fall outside the delivery window. Yet another reason is that the device u_t may be in the connected mode, and a paging message is not needed.

2) Counting: When paging messages are delivered imperfectly, we rely on the fact that a call will result in a paging message in the delivery window with high probability. The algorithm uses a parameter ϕ, which models the probability that a paging message is received by the adversary within the paging window. We set $\phi = 0.85$ for our experiments by assuming 15% as the upper bound on paging miss rate. The adversary maintains a counter (initialized to 0) for each PFI value, and iterates through the following steps: (1) Make a call. (2) Listen for paging messages during the delivery window. (3) For each PFI value, if there is a paging message for that value during the delivery window, increment the corresponding counter by 1. (4) For each PFI value that is still in the set, let v be the counter value, and n be the number of rounds, remove the PFI value if

$$\mathsf{Pr}(v : n, \phi) = \binom{n}{v} \phi^v (1-\phi)^{n-v} < \theta = 0.1$$

(5) If only one PFI value remains in the set, then it concludes that this is u_t's PFI. If the set is empty, it concludes that u_t is not in the target area.

E. The ToRPEDO Attack with Likelihood Analysis

The Counting attack does not use the information about how many paging records arrive in each frame (only whether

there is at least a paging record), nor does it use the information about the timing of the message's arrival. From Fig. 3, we can see that even though the delivery window for SMS is roughly from 2.8 to 5.4 seconds, a paging message is much more likely to arrive at, e.g., 4 seconds, than at 5.3 seconds. We now present the ToRPEDO attack, which utilizes all information to conduct a likelihood analysis and decide u_t's PFI.

Let the time at which the adversary makes a call be $t = 0$. ToRPEDO takes as input $F(\cdot)$, the cumulative distribution of paging delay. That is, if the paging message corresponding to the call is received at all, then with probability $F(t)$, it will be received by time t. Let t_m be the smallest t such that $F(t) = 1$. Then the adversary listens for c cycles after making a call, where c is computed as follows:

$$c = \left\lceil \frac{t_m}{10\mathrm{T\ ms}} \right\rceil,$$

For each $i \in [0..\mathrm{T}-1]$, during the whole observation period, there are c frames with SFN congruent to i modulo T. Let $v_{i,j}$ (where $0 \le i < \mathrm{T}$ and $1 \le j \le c$) denote the number of paging records received at the j-th frame that has SFN congruent to i modulo T. The time for $v_{i,j}$ can be computed as:

$$\mathsf{time}(i,j) = \begin{cases} 0 & \text{when } j = 0 \\ ((j-1)\mathrm{T}+i-b)\cdot 10\mathrm{ms} & \text{when } j \ge 1, b \le i \\ (j\cdot \mathrm{T}+i-b)\cdot 10\mathrm{ms} & \text{when } j \ge 1, b > i \end{cases} \tag{1}$$

where $b = S_0 \bmod \mathrm{T}$, where S_0 is the SFN when the call occurs, denotes the frame index within the length-T cycle at the time the call is made.

For each i, we compute the likelihood of observing the sequence $V = v_{i,1}, v_{i,2}, \cdots, v_{i,c}$ when $\mathsf{PFI}_t \neq i$ and when $\mathsf{PFI}_t = i$. When $\mathsf{PFI}_t \neq i$, the sequence V is due to the background paging records. We use the Poisson distribution to model the probability that we observe a certain number of paging records in a given frame. The Poisson distribution is appropriate because whether there exists a paging record for a user depends on whether other users call her at this instant and such events may originate from a large number of independent users. It expresses the probability of a given number of events occurring in a fixed interval of time if these events occur with a known constant rate and independently of the time since the last event. It is parameterized by a base rate λ_b, which the adversary can estimate empirically.

We use ℓ'_i to denote the likelihood of observing $v_{i,1}, v_{i,2}, \cdots, v_{i,c}$ when $\mathsf{PFI}_t \neq i$. It can be computed as:

$$\ell'_i = \prod_{j=1}^{c} \mathsf{Pr}\big[P(\lambda_b) = v_{i,j}\big] \tag{2}$$

Here $P(\lambda_b)$ is a Random Variable following the Poisson distribution with parameter λ_b.

We use ℓ_i to denote the likelihood of observing the sequence V when $\mathsf{PFI}_t = i$. To compute ℓ_i, we need to consider two cases.

- First, it may be that even though $\mathsf{PFI}_t = i$, the paging message is not observed. This happens with probability $1 - \phi$, where ϕ is the same parameter as used in the Counting attack. It is an estimation of the probability that a paging message is received within the delivery window.
- Second, the paging record may be delivered during any of the c cycles, and we have to sum up the likelihood of each case. The likelihood of the paging record being delivered during the j-th cycle is a product of 3 probabilities: (1) the probability that the delay for the paging message is such that the message arrives at the j-th cycle; (2) the probability that we observe $v_{i,j}$ records given that the paging record arrives in this cycle (i.e., the background contributes $v_{i,j} - 1$ paging records); (3) observations of $v_{i,k}$ where $k \neq j$ are from the background. Thus ℓ_i can be computed as follows:

$$\begin{aligned} \ell_i = (1-\phi) \prod_{j=1}^{c} \mathsf{Pr}\big[P(\lambda_b) = v_{i,j}\big] \\ + \phi \sum_{j=1}^{c} \left(F(\mathsf{time}(i,j)) - F(\mathsf{time}(i,j-1))\right) \\ \times \mathsf{Pr}\big[P(\lambda_b) = v_{i,j}-1\big] \prod_{k=1,k\neq j}^{c} \mathsf{Pr}\big[P(\lambda_b) = v_{i,k}\big] \end{aligned} \tag{3}$$

where:

ϕ	is the same parameter as used in the counting attack
$P(\lambda_b)$	is a Random Variable following the Poisson distribution with parameter λ_b
F	is the CDF of paging delay given in Fig. 3
time	is defined in Eq.(1)

The global likelihood of a PFI to be the victim's PFI. After making each call, the adversary computes the global likelihood, $\mathcal{L}_i$ for each $i = \mathrm{PFI}$, $0 \le i < \mathrm{T}$, to belong to u_t. The adversary also computes the global likelihood, $\mathcal{L}_{-1}$ for the case that u_t is not present. The global likelihood $\mathcal{L}_i$ and $\mathcal{L}_{-1}$ after making n calls are computed as follows:

$$\begin{aligned} \mathcal{L}_i &= \prod_{n \text{ trials}} \ell_i \prod_{m=0, m\neq i}^{T-1} \ell'_m \\ \mathcal{L}_{-1} &= \prod_{n \text{ trials}} \prod_{m=0}^{T-1} l'_m \end{aligned} \tag{4}$$

Identifying the victim's PFI. After each new trial, if the maximum global likelihood ($\mathcal{L}_i$) for any i becomes significantly larger (i.e., by an order of a set threshold value τ) than the second largest global likelihood value, the adversary identifies i as the PFI of the u_t. If -1 is identified this way, it concludes that u_t is not in the area.

$$\frac{\mathcal{L}_i}{\max\ \mathcal{L}_j, \text{where j} \neq \text{i}} \ge 10^{\tau} \tag{5}$$

F. Discussions

Estimating empirical distributions of paging delays. We assume that the adversary $\mathcal{A}$ owns a UE, $u_1^{\mathcal{A}}$, that is, it is a subscriber of the target network located at cell area c. Furthermore, the adversary knows $u_1^{\mathcal{A}}$'s IMSI and TMSI, which is feasible in many cases through cellular debugging tools such as MobileInsight [10]. $\mathcal{A}$ thus can compute the PFI value for $u_1^{\mathcal{A}}$. Using another adversary-controlled UE, $u_2^{\mathcal{A}}$, the adversary sends $\mathcal{N}$ phone calls, $\mathcal{N}$ SMS, or $\mathcal{N}$ tweets to $u_1^{\mathcal{A}}$ and observes the time (in milliseconds) required to receive paging messages at $u_1^{\mathcal{A}}$ for the corresponding phone calls, SMS and tweets. Using these observations, $\mathcal{A}$ can establish an empirical paging delay distribution for each service. Note that it is necessary to wait for the device to move to the idle mode before making the next call, otherwise paging messages

Carrier	Total hours of observation	Total number of paging with IMSI	Percentage of paging with IMSI
US-1	44 hours	171	0.274%
CH-1	10 hours	8	3.404%
CH-2	7 hours	1	0.028%
US-MVNO-1	15 hours	78	0.324%
US-MVNO-2	18 hours	146	0.336 %
RU-1	4 hours	2	3.175%

TABLE II: Number of `paging_imsi` messages observed by a single UE for different network operators.

will not be triggered. For our experiment, we choose the conservative value of ~ 40 seconds between two consecutive calls (or, SMS/tweets) to ensure this is the case.

Clandestine location tracking. The adversary can carry out ToRPEDO, clandestinely, i.e., without alerting the human user using u_t. For example, a phone call can be made silent [2], [3] in 4G LTE by making a call and then terminating it in a few seconds. In this case, the base station will broadcast a `paging` message, but the phone will not ring because the call is hanged up before the call establishment procedure succeeds. Similarly, SMS and other messages can be made silent.

Smart and non-smart paging. Wireless providers generally use non-smart paging (all base stations in a tracking area broadcast paging messages) for VoLTE phone calls, and smart paging (only one base station broadcasts paging messages) for SMS services. The adversary can thus choose which one to use based on whether adversary is certain of u_t's exact location.

IV. The PIERCER Attack for 4G

This section describes the PIERCER attack that enables an attacker to associate a victim's phone number with its IMSI by exploiting a deployment oversight of service providers.

A. Attack Surface

Our investigation towards PIERCER began due to an observation we made while inspecting the network traces of different service providers' paging protocol deployment. The traces we analyzed were contributed by devices across the world in the MobileInsight platform [10]. In those traces, we observed that a non-negligible amount of paging messages originating from 1 major US network operator, 2 Chinese network operators, 1 Russian network operator, and 2 US mobile virtual network operators [11] that contain IMSI as the identifier (i.e., `paging_imsi`). Table II shows a summary.

Some of these observed `paging_imsi` messages, however, were not intended for the UEs that actually contributed the traces. They rather were intended for other UEs sharing the same paging occasion as the trace-contributing UEs. Due to the lack of contextual information about UEs for which the observed `paging_imsi` messages were intended, we were unable to conclude, only from the traces, the condition(s) under which the operators sent `paging_imsi`. To increase the confidence of our observations, we collected network traces containing paging messages originating from all major US network operators and validated that one operator (the same one from the MobileInsight traces) sent `paging_imsi`. Additionally, we observed the same behavior from major network operators at different countries in Europe (e.g., 3 providers from Germany, 4 provides from Austria, 1 provider from Iceland) and Asia (e.g., 3 providers from a South Asian country).

B. Curious Case of Paging Containing IMSIs

To establish the condition(s) under which `paging_imsi` was sent, we consulted the 3GPP standard and searched the Internet for relevant documentation. A manual testing of the deployments, however, revealed that the actual condition is somewhat nuanced compared to the ones found in the standard and in the Internet documentation.

3GPP Standard. According to the 3GPP standard, it is permissible for a network provider to send out a `paging_imsi` message in the following two cases:

- **A.** When a lower-layer failure occurs for a UE during an interleaved TMSI reallocation and paging procedures, and the core network does not receive any response to an implementation-dependent number of `paging` messages containing either the old TMSI or new TMSI.
- **B.** When the device's TMSI is unavailable due to network failure.

A recent work by Hong et al. [3], however, observed that the network operators tend to disallow the overlap of the TMSI reallocation procedure and paging protocol by ignoring the TMSI reallocation. The authors used this insight to block the network from changing a device's TMSI which can then be used to track the user. As a result, case **A** cannot be the right condition for our observation. For case **B**, the standard does not clearly describe what constitutes a network failure preventing us to draw any conclusions.

Practitioner's observation. A quick web search led us to an article [12] discussing the following three cases in which an operator may send `paging_imsi` in 2G networks. The considered cases, however, focus on the GSM network instead of the newer 4G LTE network.

1) When the VLR (Visitor Location Registry)—similar functionality as in MME—uses a volatile storage to store the different IMSI-TMSI mappings, a VLR restart would induce an inaccessible IMSI-TMSI mappings. When VLR restarts, any paging will be with IMSI.
2) If the VLR has a limited amount of RAM, a user's IMSI-TMSI mapping may be evicted when a new mapping needs to be inserted. Any paging for users whose mapping are not in the RAM will be with IMSI.
3) The IMSI-TMSI mapping expires for users with long inactivity period. Any paging for users whose IMSI-TMSI mappings have expired will result in `paging_imsi`.

The current cellular infrastructure arguably does not have the same limitations of using an unreliable, small volatile storage to store the IMSI-TMSI mapping for devices. This argument is further strengthened by our observation that network operators always try with 1-2 `paging_tmsi` messages before trying with a `paging_imsi`. This means that network operators usually do not lose the IMSI-TMSI mappings neither because of limited volatile storage nor due to network nodes' abrupt restart. Nonetheless, the findings by Shaik et al. [2] who observed that operators did not change TMSIs for some devices for up to 7 days also invalidates the practitioner's remaining observation.

Manual testing. Since neither the standard nor the web article [12] provided a convincing condition to why the network may send `paging_imsi`, we resort to a manual testing approach. We

first collected traces in a cellular device $\mathrm{UE}_{\mathrm{test}}$ equipped with the offending network's SIM card while placing phone calls from another device periodically, at a regular interval. We, however, did not see any `paging_imsi` intended for $\mathrm{UE}_{\mathrm{test}}$. This led us to the conclusion that `paging_imsi` is sent only in exceptional cases.

One exceptional case we considered is to block $\mathrm{UE}_{\mathrm{test}}$ from receiving the paging message from the network. For this, we rely on a prior attack called paging channel hijacking [4]. We also established a sniffer to pick up any paging messages containing IMSI.

We observed that if we call $\mathrm{UE}_{\mathrm{test}}$ but block the corresponding paging message in the VoLTE (PS domain) from reaching the $\mathrm{UE}_{\mathrm{test}}$, the offending network retries to send the paging messages in the PS domain twice. After two unsuccessful retries in the PS domain, it then sends a `paging_imsi` in the non-VoLTE (CS domain). We validated this by matching the IMSI with $\mathrm{UE}_{\mathrm{test}}$'s IMSI. We repeated this multiple times and observed the same phenomenon.

C. Attack Description

The threat model and `PIERCER` attack steps are given below.

Threat model. For `PIERCER`, we assume an attacker who knows the victim's phone number, and can set up a paging message sniffer and a fake base station (with higher signal strength) in any cell including the victim's.

Description. An attacker initiates `PIERCER` by identifying the victim's paging occasion and current cell-level location with `ToRPEDO`. The attacker then installs a paging message sniffer and a fake base station in the victim's cell. After which the attacker hijacks the victim's paging channel and then places a **single silent** phone call. Vulnerable operators will send `paging_imsi` after two failed attempts with `paging_tmsi` (due to hijacked paging channel). The attacker's sniffer can capture the IMSI when `paging_imsi` is sent; completing the attack. The attacker may repeat the last step to gain higher confidence.

D. Discussion

Impact. `PIERCER` can also enhance the attacker capability to effectively mount some prior attacks that require the knowledge of the victim's IMSI [2], [4], [6], [7].

Defense. The defense for `PIERCER` is to ensure that the network operator never sends the `paging_imsi` message.

Observation. We attempted `PIERCER` with SMS or Twitter messages to no avail. We speculate that unlike phone calls other services do not have real-time requirements.

V. The IMSI Cracking Attack for 4G and 5G

We now present the bruteforce IMSI cracking attack, and also describe the oracles we have exploited for 4G and 5G paging protocols to decide whether a guessed IMSI belongs to the victim device. It is natural to question the rationale of designing a brute-force IMSI cracking attack for 4G where other legitimate means (e.g., `identity_request`) are available to retrieve the victim's IMSI in the clear. We wanted to demonstrate that the attack is feasible even when the IMSI is never released in the clear, as in 5G, where the IMSI (or, SUPI) is encrypted with the operators' public key. In fact, it is more efficient for 5G as one does not need to wait for the appropriate PFI.

Threat model: For the cracking attack, we assume the adversary to have the same capabilities as described in the `PIERCER` attack in Section IV-C.

A. 5G-SUPI/IMSI Representation and Information Leakage

Representation. The persistent SIM card-specific identity in 5G is called the Subscriber Permanent Identifier (SUPI). SUPI [13] can be either of the IMSI form or of the network access identifier (NAI) form. For our discussion, we focus on IMSI which is not only used in 5G but also used in 4G to uniquely identify the subscriber for authentication.

IMSIs are represented as a 15-digit (14-digit for Europe) binary-coded decimal (BCD). The first 3 digits (resp., 2 digits for Europe) of an IMSI represent the mobile country code (MCC) whereas the next 3 digits (same in Europe) represent the mobile network code (MNC) identifying the specific network operator. The rest of the 9 BCD digits of IMSI, called mobile subscription identification number (MSIN), are unique to the subscriber.

Leakage. (1) Given a user's phone number, it is possible to look up the MCC and MNC (i.e., the first $5/6$ BCD-digits) corresponding to that device using paid, Internet-based home location register lookup services [8]. This leaves 9 BCD-digits of the IMSI for the adversary to guess.

(2) Recall that, the last 10 bits of the IMSI are used for calculating the paging occasion of a device. In that calculation, however, the IMSI is considered to be a 14-$/15$-digit decimal number instead of a BCD number. Without loss of generality, if network operator base stations have $\mathrm{T}{=}\mathrm{nB}{=}128$, then calculating the victim's paging occasion will leak the last 7 bits of the victim's IMSI.

We will describe how these two forms of leakage can be combined by the adversary to decrease the search space of the brute-force search. For example, suppose that the IMSI the attacker intends to crack belongs to a US subscriber. The current maximum value of MCC for US subscribers is 316 whereas the maximum value of MNC is 990. If we consider the rest of the 9 digits of the IMSI to be 9, then the corresponding decimal number's (i.e., $316990[9]^9$) binary representation yields a 49-bit binary number whose leading 18 bits and the trailing 7 bits are known to the attacker, leaving only 24 bits for him to guess which would take the attacker 2^{24} (i.e., ~ 16.77 million) guesses in the worst case.

B. The IMSI-Cracking Attack Against 4G

This section describes the `IMSI-Cracking` attack against 4G including the oracle that enables the attacker to check the correctness of his IMSI guesses.

Oracle for 4G. The main insight we use for designing the oracle for 4G is that the legitimate responses against a `paging_imsi` and a `paging_tmsi` are different. When a device receives `paging_tmsi`, it responds with a RRC layer connection request message which includes the device's TMSI. On the contrary, when a device receives a `paging_imsi` message,

it invalidates its TMSI and the established security context (if any), and sends a RRC layer connection request followed by a NAS layer attach request. In this case, the RRC layer connection request message contains a random identity instead of its TMSI, which has been invalidated.

Recall that a paging message can contain up to 16 paging records each of which identifies a device for which there is a pending service. When a device wakes up to find a paging message, it goes through the paging records—in the order of their appearances—stopping at the first record whose identity field value matches the device's identity (i.e., IMSI/TMSI). We leverage this observation in the following insight. Suppose that the attacker knows the victim's TMSI T_{victim} but not his IMSI. The attacker makes a guess I_{guess} of the victim's IMSI and wants to check whether I_{guess} is the victim's IMSI. For this, the attacker can inject a fabricated paging message for the victim containing the following two paging records:

Paging record 1 containing I_{guess} in the identity field;
Paging record 2 containing T_{victim} in the identity field.

After receiving the above paging message, if the victim responds with a RRC layer connection request containing an identifier whose value is not equal to T_{victim}, then I_{guess} is the victim's IMSI as it is responding to paging record 1. If the victim, on the other hand, responds with a RRC layer connection request containing T_{victim} as the identifier, it means that the attacker's guess is wrong because the victim is responding to the paging record 2.

The complete attack. The attacker starts off by using the `ToRPEDO` attack to identify the victim's coarse-grained location, paging occasion, and the current TMSI. Note that, the victim's paging occasion can be shared by multiple non-targeted users inducing an implicit K-anonymity set where (K-1) is the number of non-targeted users sharing the victim's paging occasion. The attacker then hijacks the victim's (and, also the other K-1 users') paging channel as described by prior work [4]. The attacker creates a fabricated paging message containing 16 paging records where the first 15 records contain different IMSI guesses from the adversary whereas the last paging record contains the victim's TMSI as the identifier.

For each fabricated paging message, if the attacker receives an RRC layer connection request with the victim's TMSI, it means that none of the 15 IMSI guesses belong to the victim. On the other hand, if the attacker does not receive a RRC layer connection request with the victim's TMSI, it suggests that one of the 15 IMSI guesses belongs to the victim, although the attacker does not know which one it is. Also, as the victim received `paging_imsi`, it would invalidate the current TMSI. To narrow down which of the 15 guessed IMSIs belongs to the victim, the attacker stops the paging channel hijacking attack and lets the victim connect to the legitimate base station.

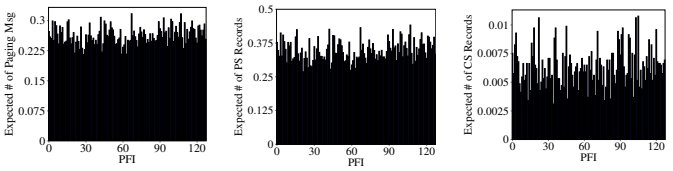
Then the attacker again uses `ToRPEDO` to identify the victim's current TMSI T^c_{victim}. Suppose the victim's IMSI belongs to the following set: $\mathcal{G} = \{I^i_{\text{guess}} | 1 \leq \text{i} \leq 15\}$ identified from the previous guess. Again, the attacker hijacks the paging channel of the victim including the other devices sharing the same paging occasion. Then the attacker sends a maximum of 15 paging messages each of which contains two paging records. For the first paging message, the first record contains $I^1_{\text{guess}} \in \mathcal{G}$ as the identifier whereas the second record contains T^c_{victim}. Similarly, for the second paging message, the first record contains $I^2_{\text{guess}} \in \mathcal{G}$ as the identifier whereas the second record contains T^c_{victim}, and so on. For the jth paging message where $1 \leq j \leq 15$, if the attacker receives an RRC layer connection request with a random identifier, then it suggests that the guess I^j_{guess} belongs to the victim.

C. The IMSI-Cracking Attack Against 5G

This section presents the oracle needed for carrying out the `IMSI-Cracking` attack against 5G.

Oracle for 5G. Since the unencrypted IMSI does not appear in 5G [14], we leverage three insights drawn from the attach (i.e., the registration) procedure to design the oracle for 5G. (i) The core network's response to a `registration_request` message (resp., `attach_request` message in 4G) is different depending on whether the message contains a valid IMSI. If the core network receives a `registration_request` message with a non-existent/invalid IMSI, the network issues a `registration_reject` (cause #9: UE identity cannot be derived by the network) message (clause 5.5.1.2.5 of the 5G NAS standard [14]) to the device, whereas the network sends an `auth_request` message (*challenge*) in response to a `registration_request` message with an existent/valid IMSI. (ii) There is a one-to-one relationship between the cryptographic master key (K) and the IMSI of each device in the network which means that a device with IMSI_i cannot solve an authentication challenge c_j derived from K_j of the device with IMSI_j, where $i \neq j$. (iii) A device's response to a valid `auth_request` message is different from the response to an invalid `auth_request` message. For an `auth_request` message, the device responds with an `auth_response` message if it can solve the challenge; otherwise, it responds with an `auth_failure` with an indication to the message authentication code (MAC) or the sequence number verification failure.

If a device's initial `registration_request` message (containing MCC, MNC, and the encrypted MSIN of the user) is not integrity protected, the network initiates an authentication procedure with the device (clause 6.4.6 of 5G Security Architecture [15]). The encrypted MSIN, also called concealed subscription identifier (SUCI), is a function of the home network's public key CN_pk and the MSIN of the user, i.e., $\text{SUCI} = \text{f}(\text{CN}_\text{pk}, \text{MSIN})$. We leverage this observation in the following insight. Suppose the attacker knows the core network's public key provisioned in the attacker-controlled SIM card/UE, and makes a guess I_{guess} of the victim's IMSI and wants to check whether I_{guess} is the victim's IMSI. For this, the attacker calculates $\text{SUCI}_{\text{guess}}$ corresponding to the I_{guess} and sends a fabricated `registration_request` message to the core network. Let $\mathcal{I}$ be set of all valid/active IMSIs for a test network. After receiving the `registration_request` message, if the network responds with a `registration_reject` message, it means that $I_{\text{guess}} \notin \mathcal{I}$. If the network, on the contrary, responds with an `auth_request` message, it means that $I_{\text{guess}} \in \mathcal{I}$ and the attacker considers I_{guess} as a potential candidate of the victim's IMSI. To validate such a guess, the attacker forwards the `auth_request` message to the victim's device. If the victim responds with an `auth_failure` message indicating a MAC verification failure, the attacker infers that the `auth_request` message generated by the network is not for the victim's IMSI and thus that I_{guess} is not the victim's



(a) Paging message base rate ($\lambda^{b}_{\text{paging}}$). **(b)** PS record base rate (λ^{b}_{ps}). **(c)** CS record base rate (λ^{b}_{cs}).

Fig. 4: Average number of paging message, PS record, and CS record arrivals in any PFI within one paging cycle during peak-time of a day.

IMSI. If the device, however, responds with an `auth_response` message, the attacker infers that the I_{guess} is the victim's IMSI.

VI. Testbed Setup

We now describe our testbed that we used for validating ToRPEDO, PIERCER, and IMSI-Cracking attacks in 4G.

Paging sniffer. For capturing broadcast messages we set up a sniffer using a Universal Software-defined Radio Peripheral device, i.e., USRP B210 [16] (costs $1300) connected to an Intel Core i7 machine running Ubuntu 16.04 as the hardware component and srsLTE [17], an open source LTE protocol stack implementation. We modified the srsLTE's `pdsch_ue` application to enable the sniffer to periodically (∼10 minutes) switch its decoding mode between the `master_info_block` and `paging` channels. Thus the sniffer periodically synchronizes the network time/frame similar to commercial-off-the-shelf (COTS) UEs and reliably computes the SFN value for a received paging message. We also use the sniffer to capture and decode the `sys_info_block_1` and `sys_info_block_2` messages and learn the parameters relevant for computing the victim UE's paging occasion (e.g., T and nB).

Malicious eNodeB. We use another USRP B210 [16] connected to an Intel Core i7 machine running Ubuntu 16.04 and modified srsENB [17] to set up a malicious eNodeB. There are other more economical options for setting up a rogue eNodeB using LimeSDR [18] (costs around $299) which has also been shown to be effective [19].

VII. ToRPEDO Evaluation

In this section, we validate and evaluate the *filtering*, *counting*, and *likelihood* variants of ToRPEDO.

Effectiveness metrics. For assessing the effectiveness of all variants of ToRPEDO, we use the following metrics: (1) **accuracy** (defined below), and (2) **number of trials** (i.e., calls/SMSs) required to correctly identify a victim UE's paging occasion. We also evaluate the same for the case when the victim is not present in a cell area.

$$\textbf{accuracy} = \frac{\text{total \# of attacks - \# of mis-identifications}}{\text{total \# of attacks}} * 100\%$$

A. Evaluation Setting

We evaluated the ToRPEDO variants in both peak (12:00 PM noon) and off-peak (12:00 AM midnight) times as the number of users as well as the paging message distribution tends to vary with time of the day [1], [3]. We also carried out the attacks in two different geographical locations, although we present results from one location due to space constraints. In the similar vein, we include results for only one major US network provider (i.e., **US-I**) as we have mostly observed the similar trends for the rest of the network providers.

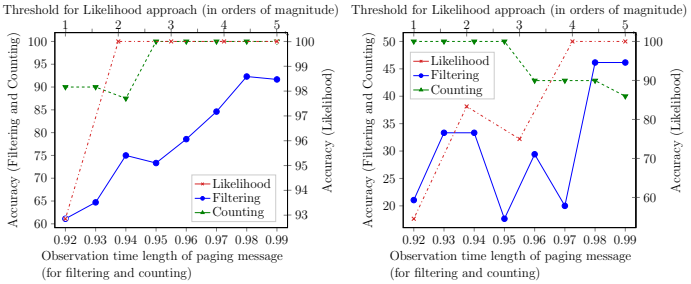
We have considered both VoLTE and CSFB phone calls while validating ToRPEDO variants. Similarly, we also considered paging in both PS and CS domains. We particularly considered paging with CS domain as our analysis of network traces from 34 different service providers [10] revealed that 14 of them use paging with CS domain. Finally, we demonstrate ToRPEDO variants' effectiveness in identifying the victim's presence and also absence in a cell.

B. Baseline for Likelihood Variant of ToRPEDO

Carrying out the likelihood variant of ToRPEDO requires the attacker to establish a baseline paging message (resp., records) distribution. For the baseline, the attacker first uses a sniffer to capture paging messages received at different PFIs for 15 minutes. The adversary then computes the following Poisson distribution parameters: (i) $\lambda^{b}_{\text{paging}}$ = average number of paging message arrivals for any PFI within one paging cycle (T); (ii) λ^{b}_{PS} = average number of PS record arrivals for any PFI within one paging cycle; and (iii) λ^{b}_{CS} = average number of CS record arrivals for any PFI within one paging cycle. Fig. 4a, 4b, and 4c show the average number of paging message $\lambda^{b}_{\text{paging}} = 0.26$, PS record $\lambda^{b}_{\text{PS}} = 0.34$ and CS record $\lambda^{b}_{\text{CS}} = 0.0065$ arrivals, respectively, in any PFI within one paging cycle during the peak time of a day. This distribution is related to the number of users in an area as we observed ∼20982 and ∼8062 unique TMSIs per 30 minutes during the peak and off-peak time, respectively. We require $\frac{10}{k}$ minutes (k is the number of devices) to get this empirical distribution which may vary based on time, location, network and load.

C. Identifying Victim's Presence with ToRPEDO

We now describe validation results of ToRPEDO variants with different trial types (e.g., VoLTE phone call, SMS).



(a) No missing paging. **(b)** 15% missing paging

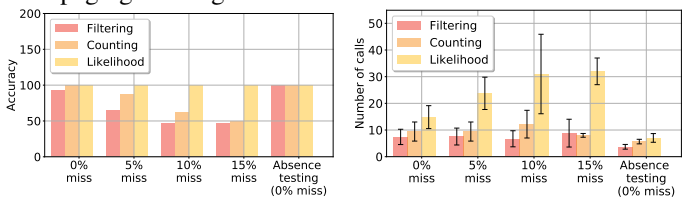
Fig. 5: Accuracy and number of trials against thresholds for ToRPEDO using VoLTE calls during the peak hour (around noon) of a day.

1) VoLTE Call: (*Parameter selection*). Carrying out ToRPEDO variants requires fixing few parameters. For the filtering and counting variants the important parameter is the *delivery window* (or, *observation interval*). Given a time period, an observation interval of 0.95 means that 95% of the observed paging messages arrived within the considered time period. The observation interval can be computed from the histogram of the paging delay similar to ones in Fig. 3a. In the similar vein, the necessary parameter for the likelihood variant of ToRPEDO is the threshold value τ. τ is used to compare the likelihood of the two PFIs with highest likelihood values.

Fig. 5a shows the accuracy of three approaches for different observation intervals and thresholds. Note that the scale of Y-axis for the likelihood based approach is different from that of the filtering and counting based approaches. As we increase the observation interval, the accuracy of the Filtering and Counting attacks improve whereas the accuracy of the likelihood approach improves with increasing value of threshold (τ).

One can then choose the parameter values for each ToRPEDO variant that results in the highest accuracy in case of 15% paging missing rate by consulting Fig. 5b. Precisely, we set the observation intervals for *Filtering* and *Counting* attacks to 0.98 and 0.95, respectively, whereas we set the threshold for *likelihood* to $\tau = 4$. We omit the parameter selection process for the latter trial types as they are similar to the VoLTE case.

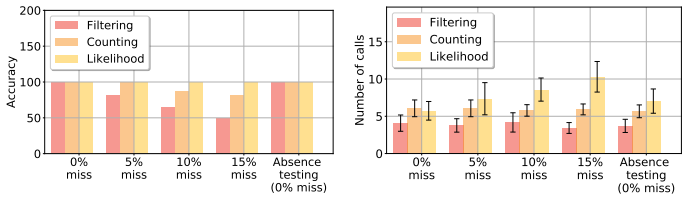
We observed on average 3.5% and maximum 8% paging missing rate during our evaluation. As ToRPEDO becomes more difficult when this rate increases, to determine whether the attack works under higher rates, we considered 5%, 10%, and 15% paging missing rates in the evaluation.



(a) Accuracy (at peak hour) **(b)** Number of trials (at peak hour)

Fig. 6: Accuracy and number of trials against the selected observation intervals (98% for filtering and 95% for counting) and the threshold value ($\tau = 4$ in order of magnitude for likelihood) for ToRPEDO using **VoLTE** calls during the **peak** time of a day.

Accuracy for VoLTE calls. Fig. 6a and Fig. 7a show the accuracy of the *filtering*, *counting*, and *likelihood* based approaches in the peak and off-peak hours of a day for different paging missing rates. The accuracy drops for filtering and counting with increasing paging missing rates. For instance, the accuracy for counting drops from 100% to 48% (Fig. 6a) as the paging missing rate increases from 0% to 15% whereas the accuracy for likelihood approach remains 100% throughout.



(a) Accuracy (at off-peak hour) **(b)** Number of trials (at off-peak)

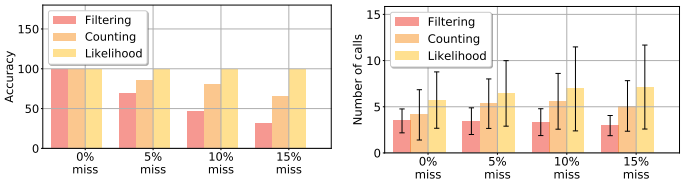
Fig. 7: Accuracy and number of trials against the selected observation intervals (98% for both filtering and counting) and the threshold value, τ=3 (in order of magnitude) for likelihood based ToRPEDO using **VoLTE** calls during the **off-peak** time of a day.

Number of VoLTE calls required. Fig. 6b and Fig. 7b show the number of silent VoLTE phone calls that an adversary requires in the peak and off-peak hours of a day for successful identification of victim's PFI using *filtering*, *counting*, and *likelihood* based approaches for different paging missing rates. The number of silent phone calls for the likelihood based approach increases with increasing paging missing rates whereas the number of silent calls for filtering and counting based approaches remains consistently low for all paging missing rates. This is because both the filtering and counting approach remove the victim's PFI from the set of candidate PFIs (as discussed in Section III) if the paging is missed for a silent call or a couple of silent calls. In contrast, the likelihood based approach takes the error rate into account and thus requires additional silent phone calls to reach the threshold (e.g., $\tau = 4$) in the case where paging messages are missed. Note that, since the base paging rate during off-peak hours is significantly lower than that of the peak hours, the adversary requires less phone calls at off-peak hours as shown in Fig. 7b. For instance, the adversary requires only $\sim$5 silent phone calls with the likelihood approach (for no missing paging rate) at off-peak hours which increases to $\sim$13 calls at peak hours. This indicates that the number of phone calls required for ToRPEDO is proportional to the base rates $\lambda^b_{\text{paging}}$, λ^b_{PS}, and λ^b_{CS}.

2) CSFB Phone Call: For the network operators which choose to generate mobile terminated CSFB call for the callee based on VoLTE capability of the callee device and the network, the adversary takes into account both the PS and CS records in the paging messages after a silent phone call. In contrast, for other types of network which always trigger paging in CS domain for CSFB calls, the adversary considers only the CS records for inferring victim's PFI.

Accuracy for CSFB calls. Fig. 8a and Fig. 9a show the accuracy of the *filtering*, *counting*, and *likelihood* based approaches during peak and off-peak time for different paging missing rates. In general, the accuracy trend for CSFB calls is similar to that for the VoLTE calls as shown in Fig. 6a and Fig. 7a.

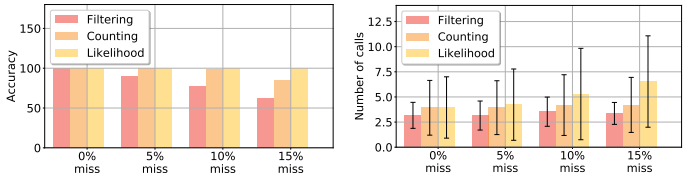
Number of CSFB calls required. Fig. 8b and Fig. 8b show the number of silent CSFB phone calls that an adversary requires at peak and off-peak time for successfully identifying victim's PFI using the *filtering*, *counting*, and *likelihood* based approaches for different paging missing rates. Since the base rate of CS domain records (λ^b_{CS}) is as strikingly low as 0.0065 for any PFI within one paging cycle (Fig. 4c), the number of silent CSFB phone calls for ToRPEDO with all three approaches is significantly lower than that of the VoLTE phone calls (Fig. 6b and Fig. 7b). The results mean that it is easier to identify the victim's PFI and its presence if the victim UE or the serving eNodeB are not VoLTE capable. One implication could be that for locations where eNodeBs are not VoLTE capable, it may be easier to identify a particular user's presence with only a small number of phone calls.



(a) Accuracy (at off-peak hour) **(b)** Number of calls (at off-peak)

Fig. 8: Accuracy and number of trials against the selected observation intervals (98% for both filtering and counting) and threshold value ($\tau = 2$ in order of magnitude for likelihood) for ToRPEDO using **CSFB** calls during the **peak** time of a day.

3) SMS and Tweets: We successfully validated ToRPEDO using SMS and tweets. A tweet mentioning the victim's Twitter handle triggers paging if the victim sets the Twitter app with "Push Notification" on. The accuracy for SMS and Tweets



(a) Accuracy (at off-peak hour) **(b)** Number of trials (at off-peak)

Fig. 9: Accuracy and number of trials against the selected observation intervals (98% for both filtering and counting) and threshold value ($\tau = 2$ in order of magnitude for likelihood) for ToRPEDO using **CSFB** calls during the **off-peak** time of a day.

follow the similar trend of phone calls. The number of required trials, however, was as low as 6-9 at peak time and 4-6 at off-peak time because of a smaller paging delay (3-5 seconds as shown in Fig. 3a). While tweets are less stealthy, Tweeter handle is public, and the adversary can carry out the attack without the victim's phone number.

4) Time Requirement.: The phone calls/SMS/tweets placed by the adversary to perform ToRPEDO attack do not have any temporal dependency, i.e., the trials can be made at arbitrary times. It is, however, necessary to wait (about ∼30-35 seconds) for the device to move to the idle mode before making the next trial. Therefore, for ToRPEDO to be successful, it requires 2.4–4.3 minutes on average when successive phone calls are placed with an interval of ∼30-35 seconds.

D. Sensing Victim's Absence with ToRPEDO

We also perform the ToRPEDO attack when the victim device and the adversary's sniffer are not in the same tracking area and evaluate the accuracy and the number of silent calls required for identifying victim's absence during the peak and off-peak hours of a day. The accuracy of the filtering, counting, and likelihood based approaches follow a trend similar to identifying victim's PFI (i.e., presence) as shown in Fig. 6a and Fig. 7a. Fig. 6b and Fig. 7b show that the adversary with likelihood based approach requires slightly less calls (8 calls for threshold $\tau = 4$) during peak hours for identifying victim's absence whereas it requires $\sim$ 13 calls for identifying victim's PFI (i.e., presence) as shown in Fig. 6.

VIII. Validating PIERCER and IMSI-Cracking Attacks

In this section, we describe our validation process of PIERCER and IMSI-Cracking attacks for 4G.

A. PIERCER Evaluation

The goal of our PIERCER validation is to determine how many calls the adversary needs to reliably retrieve the victim UE's IMSI given the victim's phone number and PFI (assumed to be obtained with ToRPEDO attack). We describe the validation process for **US-I** only, although similar behavior was observed for three providers of Germany, four providers of Austria, one provider of Iceland, and three other network operators of a South Asian country. Note that the list of network operators identified to be using paging_imsi is not exhaustive as our evaluation does not include all the network operators in every country of the world.

Paging channel hijacking. After identifying the victim's PFI, we carried out the paging channel hijacking attack by faithfully following the steps described by Hussain et al. [4].

The attack. Once the victim's paging channel was hijacked, we made a single phone call to the victim's phone number which caused the legitimate eNodeB/MME to first send a paging_tmsi in the PS domain for the victim UE. Once a paging_tmsi is issued, we observed that the US-I network sets a timer for the response. Since the victim is unaware of the actual paging_tmsi message, the timer at MME expires because of UE's unresponsiveness. We then observed that the MME initiates another paging_tmsi in the PS domain and continues to do so until it reaches the maximum_paging_attempt which was set to 2 by US-I. Upon expiration of retries, the MME of US-I first sends paging_tmsi in the CS domain and then, when its timer expires, it broadcasts the paging_imsi in the CS domain. If the adversary now makes a second phone call, the MME initiates another paging_tmsi in the CS domain without trying with the PS domain first, and then, upon expiration of the timer, the MME broadcasts paging_imsi again which validates our attack.

Total call trials. Excluding the phone calls needed for the prerequisite ToRPEDO attack, we validated that indeed a single phone call was sufficient to expose the victim's IMSI.

B. IMSI-Cracking Evaluation

The goal of our IMSI-Cracking attack evaluation is to determine the time and the number of paging messages that an adversary needs to make to identify the victim's IMSI given that the adversary knows the victim's phone number (in our case, MCC=310 and MNC=260 retrieved from [8]), PFI (in our case, 21), and TMSI (using techniques from [1], [2]).

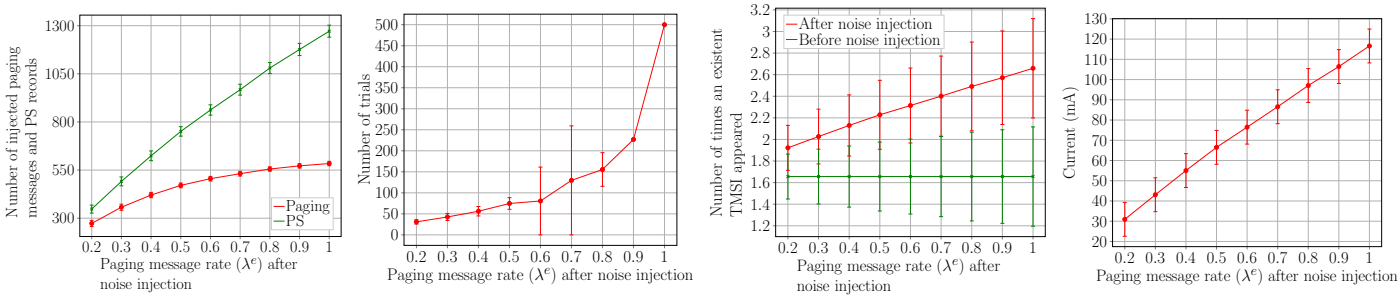
We compute the $I^{\max}_{\text{guess}}$ and $I^{\min}_{\text{guess}}$, i.e, the maximum and minimum possible IMSI values (in binary) that have the value 310260 in 18-bits MSB and the value 21 in 7-bits LSB. We found $I^{\max}_{\text{guess}} = 310260999999381$ and $I^{\min}_{\text{guess}} = 310260000000021$ for the given PFI, MCC, and MNC. We started with $I^{\max}_{\text{guess}}$ in descending order and fabricated paging messages with 14 I_{guess} each. Note that, though the standard specifies that up to 16 paging records can be accommodated into a single paging message, we observed that our test UE device accepts paging messages containing at most 15 paging records which indicates a deviation from the standard and an interoperability issue.

Number of paging messages. To identify the victim's IMSI (=310260628687893), the attacker needed to try a total of 2900876 IMSIs and thus sent $\lceil \frac{2900876}{14} \rceil + 14$ = 207220 paging messages through the malicious eNodeB.

Time requirement. We set the values of paging cycle T and nB to 128 for hijacking paging channel and, therefore, sent one paging message every 1.28 seconds. The total time required to crack the victim's IMSI was ∼74 hours. We performed this experiment for 7 days in the off-peak hours. Note that the attack can be expedited by at most 32 times by setting the value of $\text{nB} = \frac{\text{T}}{32}$ while querying with fake paging messages by hijacking victim's paging channel.

IX. Countermeasure

Since ToRPEDO is the precursor of both PIERCER and IMSI-Cracking attacks, in this section, we mainly focus on possible defenses against ToRPEDO. We also evaluate the effectiveness and overhead of a countermeasure we suggest.



(a) # paging and PS records injected to increase the base paging rate $\lambda^b_{\text{paging}}$=0.13 and λ^{ps}_b=0.15.

(b) Avg. # of trials for ToRPEDO against the increased paging rate generated through noise injection.

(c) Total # of paging (combining noisy and actual) containing an existent TMSI in 38.4 seconds.

(d) Extra energy/power consumed by a device for different levels of noises.

Fig. 10: Effectiveness and overhead of our proposed noise-based defense mechanism.

A. *Defense against* ToRPEDO

It is only natural to consider countermeasures that primarily focus on either thwarting the root cause (i.e., fixed paging occasion) of ToRPEDO or defending against ToRPEDO through the detection of its (behavioral) signature. Such a view induces the following two categories of defenses, referred to as *Protocol-level* and *Signature-based* countermeasures. As we demonstrate below, these categories of countermeasures, however, are ineffective due to deployment constraints. This inspired us to design a countermeasure which prevents the adversary from retrieving accurate side channel information through the addition of noise. We call this the *Noise-based* countermeasure and demonstrate that it can effectively thwart ToRPEDO without incurring substantial overhead.

1) Protocol-level defenses: The main philosophy of protocol-level defenses is thwarting the root cause of ToRPEDO, that is, to ensure that a UE's paging occasion does not remain fixed in a particular cell area. Having the paging occasion rely on the TMSI instead of the IMSI can be a plausible solution. At a first glance, it seems that this solution would work, but a recent study [2] has shown that network operators do not change TMSI frequently and even when they do, the TMSI remains predictable [3]. As most network operators reallocate the TMSI only after the CSFB, a TMSI would remain fixed for a device until a CSFB is executed. This may give the adversary ample time to launch ToRPEDO using VoLTE calls, SMS, or data services. Another naive solution would be to ensure that a device's TMSI is reallocated after it receives a `paging` message. Such a solution, however, is infeasible in practice due to its high energy demand for a device and high protocol overhead.

Another effective solution could potentially attempt to induce unpredictability of the following: (1) the identifier included in the `paging` message; (2) the paging occasion of a UE in a cell. To introduce randomness, however, it is crucial to ensure that both the UE and the eNodeB (and also the MME) share a common source of randomness from which to generate subsequent pseudorandom numbers to be used as both the identifier and paging occasion. The cellular paging protocol is designed so to enable a UE to receive paging messages without going through the connection bootstrapping [14], [20]. Without a successful bootstrapping, it is not clear how the UE and the eNodeB (resp., the MME) could establish the necessary shared, random seeds. Even when this deployment constraint is ignored, deploying this defense would require major overhaul in both the UE and network operator sides.

2) Signature-based defense: Another dimension of defense that can be adopted by the network operators would be to use machine learning algorithms and deep packet inspection techniques to develop a signature of the ToRPEDO attack (e.g., a lot of silent calls or SMS on the victim's phone number within a particular time interval) and preventing ToRPEDO by applying countermeasures (e.g., rate-limiting) whenever such a signature is detected [21]–[23]. An adversary, however, may evade the detection of such signatures by increasing delays between subsequent phone calls. Along with the resource overhead on the network operator side, another critical challenge of deploying such a defense is to balance the detection rate without compromising the quality of service for benign users.

B. *Our Proposed Noise-based Countermeasure*

We now describe our noise-based countermeasure.

The high-level idea. The basic idea of our proposed countermeasure is to increase the current paging rate ($\lambda^c_{\text{paging}}$) of all paging occasions to a certain level (λ^e) so that the adversary would need a high number of silent calls to sufficiently differentiate the paging rate of victim's paging occasion from others. To increase the paging rate, we propose that an eNodeB injects new paging messages at the paging occasions for which the paging rate is relatively lower than the expected rate (λ^e). The eNodeB will also add new paging records to both the actual paging messages (note that, at most 16 paging records can be sent in a single paging message) and the noisy paging messages to increase the current base rate of paging records.

Noisy paging messages. An intuitive and seemingly practical choice for creating noisy paging messages would be to use fake/non-existing TMSIs so that the current devices in the network do not respond to the noisy paging messages containing the fake TMSIs. However, the adversary may distinguish the fake paging records by identifying the messages/TMSIs for which there is no response from the devices. Although identifying fake paging messages is exceedingly difficult, we do not rule out this possibility and thereby propose an eNodeB to add existing TMSIs for which the actual paging messages were recently (e.g., previous 10 minutes) requested by the network. Such noisy paging messages with legitimate TMSIs may cause a device to respond to the additional paging messages for which there is no actual pending incoming services.

CS (circuit-switch) domain records. As shown in ToRPEDO attack validation, the base rate of paging containing CS records is substantially low. Consequently, the adversary only needs 2-4 silent phone calls for a successful ToRPEDO attack. To

protect against this, especially in case of an incoming CSFB service, we suggest that the network sends a paging message in the PS domain first, and following it up with an encrypted `cs_service_notification` message to the UE through a dedicated logical channel.

C. Evaluation of Noise-based Countermeasure

In this section, we evaluate our countermeasure based on its effectiveness and overhead.

Network bandwidth overhead. For evaluating the operator's bandwidth requirement, we varied λ^e in the range of [0.13, 1.0] when the base paging rates were $\lambda^b_{\text{paging}}$=0.13 (resp., λ^{ps}_b=0.15) and measured the number of fake paging messages (resp., PS records) injected by the network operator. The results are shown in Fig. 10a. As expected, increasing λ^e results in an increase in the number of injected fake paging messages (resp., records). For $\lambda^e = 1.0$, the maximum number of injected paging messages (resp., PS records) is ~ 600 (resp., ~ 1200).

Countermeasure effectiveness. For measuring the effectiveness of our countermeasure, we calculated the number of calls the adversary would need for a successful `ToRPEDO` attack for different values of λ_e varied between [0.13, 1.0]. The results are shown in Fig. 10b. Increasing the injected noise (i.e., increase of λ^e) slows the attacker down by requiring more calls for a successful `ToRPEDO` attack. When λ^e reaches 1.0, we observed that `ToRPEDO` did not succeed with 500 calls.

UE overhead. For measuring the UE-side overhead when our countermeasure is deployed, we first calculated the number of spurious paging messages a UE would have to respond due to fake paging message injection. Fig. 10c shows the results about the number of paging messages (both actual and noisy) containing an existing TMSI in a 30 paging-cycles (=38.4 seconds) time interval when the eNodeB uses actual TMSIs to generate noisy paging messages. It also shows that a device would receive ~ 1 additional spurious paging message.

Power cost. We used an existing power model [24] for our test phone to calculate the energy a UE would require to respond to additional spurious paging message(s). Fig. 10d shows results of the additional energy requirement in terms of electric current flow (in milliampere or mA) with varying λ^e. As expected, increasing λ^e results in an increase of the UE's energy requirement; the maximum value being ~ 117 mA.

X. Discussion

Limitations. For `ToRPEDO` to be successful, an attacker needs to have a sniffer in the same cellular area as the victim. If the number of possible locations that the victim can be in is large, the expense of installing sniffers (i.e., $200 each) could be an impediment to carrying out a successful attack. In a similar vein, for a successful `PIERCER`, the attacker needs to have a paging message sniffer and also a fake base station which would cost around $400. The `IMSI-Cracking` attack for 4G will be feasible only in cases where the attacker can carry out his attack without the victim noticing that his device is not receiving any notifications, for instance, when the victim is sleeping at night. `ToRPEDO` and `IMSI-Cracking` attacks on 5G were not validated due to the lack of deployed networks.

Responsible disclosure. We reported our findings to GSMA through the coordinated vulnerability disclosure (CVD) program and our findings were acknowledged by GSMA [25].

XI. Related Work

This section presents existing efforts that expose users' persistent information and thus facilitate location tracking.

IMSI catching attacks. The IMSI catching attacks still prevail for 4G LTE networks [2] as they did for 2G and 3G networks [26]–[28]. In contrast to traditional IMSI catchers where the adversary forces the UE to expose the IMSI/IMEI, the `PIERCER` demonstrated in this paper forces the network to expose the user's IMSI/IMEI and thus uniquely maps a phone number to its IMSI which was supposed to be only possible by law enforcement agencies with operators' cooperation. To the best of our knowledge, we are the first to develop and demonstrate a complete attack in both 4G and 5G.

Location tracking through TMSI and C-RNTI. The 3GPP standard [29] suggests to change TMSI frequently to hide users' IMSI/IMEIs. However, prior work [1], [2] havs shown that an adversary can still exploit the operational network's misconfiguration of frequent TMSI change, and thus track a user by uniquely mapping a phone number to his TMSI which often does not get replaced even for three days [2]. The most recent work [3] along this direction exposes the operational networks' vulnerability of not properly randomizing the TMSIs while reallocating them to the subscribers. As a result, some of the bytes are fixed [30] between the old and new TMSIs through which the adversary can still infer the new TMSI and track the subscriber. In contrast, our `ToRPEDO` exploits the protocol standard's vulnerability of choosing fixed paging frames for a subscriber which makes all the network operators vulnerable to this attack. Rupprecht et al. [31] and Jover et al. [32] demonstrated that an adversary can identify victim UE's short-lived, lower-layer identifier (C-RNTI) when given the UE's TMSI and thus track the victim's UE. `ToRPEDO`, on the contrary, recovers the UE's PFI (and TMSI as a side effect) which enables tracking victim's UE irrespective of short-lived C-RNTIs or TMSIs.

Location tracking through traceability attacks. Arapinis et al. [33] presented a traceability attack by exploiting an implementation bug in the 3G network which violates the 3GPP standard recommendation of adopting new temporary identity for the UE with a tracking area change. In another work, Arapinis et al. [34] demonstrated another traceability attack in which the adversary replays the `auth_request` for the victim UE to all the UEs in an area and detects the presence of the victim UE from the causes (MAC failure or SQN synchronization failure). In contrast, the traceability attack [4] with the *security mode command* procedure exploits the missing nonces in `security_mode_command`, a different implementation bug of the commercial networks.

XII. Conclusion and Future Work

Our paper sheds light on an inherent design weakness of the 4G/5G cellular paging protocol which can be exploited by an attacker to not only obtain the victim's paging occasion but also to identify the victim's presence in a particular cell area just from the victim's soft-identity (e.g., phone number, Twitter handle) with a novel attack called `ToRPEDO`. We also

demonstrate that ToRPEDO can enable an attacker to exploit a deployment oversight of several network operators to retrieve a victim's IMSI from his phone number using the PIERCER attack. To further provide evidence of ToRPEDO's potency, we show that it empowers an attacker to launch a brute-force IMSI-Cracking attack through the use of two novels oracles we designed for 4G and 5G, respectively. Each of these attacks can also be leveraged to enhance prior attacks. All of our attacks have been validated in realistic setting for 4G using cheap software-defined radio and open-source protocol stack. As part of our investigation on the cellular paging protocol, we also design and evaluate a novel countermeasure for ToRPEDO that raises the bar for the attacker without incurring substantial overhead or violating common-sense deployment constraints.

In future, we will explore other forms of side channel information, exposed by cellular network protocols, that can be exploited to launch novel security and privacy attacks. We will also evaluate different facets of the solution space possibly in collaboration with the stakeholders.

ACKNOWLEDGEMENT

We thank the anonymous reviewers for their valuable suggestions. This work was supported by NSF grant CNS-1657124, United States ARO grant W911NF-16-1-0127, NSA's Science of Security Lablet program through North Carolina State University, NSF grant CNS-1719369 and Intel as part of the NSF/Intel ICN-WEN program.

REFERENCES

[1] D. F. Kune, J. Koelndorfer, and Y. Kim, "Location Leaks on the GSM Air Interface," in *NDSS*, 2012.

[2] A. Shaik, J. Seifert, R. Borgaonkar, N. Asokan, and V. Niemi, "Practical Attacks Against Privacy and Availability in 4G/LTE Mobile Communication Systems," in *23nd Annual Network and Distributed System Security Symposium, NDSS, CA, USA, February 21-24*, 2016.

[3] B. Hong, S. Bae, and Y. Kim, "GUTI Reallocation Demystified: Cellular Location Tracking with Changing Temporary Identifier," in *25th Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 18-21*, 2018.

[4] S. R. Hussain, O. Chowdhury, S. Mehnaz, and E. Bertino, "LTEInspector: A Systematic Approach for Adversarial Testing of 4G LTE," in *25th Annual Network and Distributed System Security Symposium, NDSS, San Diego, CA, USA, February 18-21*, 2018.

[5] A. Dabrowski, N. Pianta, T. Klepp, M. Mulazzani, and E. Weippl, "Imsi-catch me if you can: Imsi-catcher-catchers," in *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2014, pp. 246–255.

[6] U. Meyer and S. Wetzel, "A Man-in-the-Middle Attack on UMTS," in *Proceedings of the 3rd ACM Workshop on Wireless Security*, ser. WiSe '04. ACM, 2004, pp. 90–97.

[7] I. Androulidakis, "Intercepting Mobile Phone Calls and Short Messages Using a GSM Tester," in *18th Conference on Computer Networks, Poland*, A. Kwiecień, P. Gaj, and P. Stera, Eds., 2011, pp. 281–288.

[8] *HLRLOOKUPS*, https://www.hlr-lookups.com/.

[9] *3GPP; Technical Specification Group Radio Access Network; V15.0.0 NR; (2018-06) User Equipment (UE) procedures in Idle mode and RRC Inactive state (Rel. 15)*, http://www.3gpp.org/DynaReport/38304.htm.

[10] Y. Li, C. Peng, Z. Yuan, J. Li, H. Deng, and T. Wang, "MobileInsight: Extracting and Analyzing Cellular Network Information on Smartphones," in *Proceedings of the 22nd Annual International Conference on Mobile Computing and Networking*, ser. MobiCom, New York, NY, USA, 2016, pp. 202–215.

[11] F. Zarinni, A. Chakraborty, V. Sekar, S. R. Das, and P. Gill, "A First Look at Performance in Mobile Virtual Network Operators," in *Proceedings of Internet Measurement Conference (IMC)*, 2014, pp. 165–172.

[12] H. Welte, *The reason why you see paging by IMSI in real-world GSM networks*, http://laforge.gnumonks.org/blog/20100628-the_reason_for_paging_by_imsi, 2010.

[13] *5G; System Architecture for the 5G System (3GPP TS 23.501 version 15.2.0 Release 15).*, https://www.etsi.org/deliver/etsi_ts/123500_123599/123501/15.02.00_60/ts_123501v150200p.pdf.

[14] *5G; Non-Access-Stratum (NAS) protocol for 5G System (5GS); Stage 3 (3GPP TS 24.501 version 15.0.0 Release 15).*, https://www.etsi.org/deliver/etsi_ts/124500_124599/124501/15.00.00_60/ts_124501v150000p.pdf.

[15] *5G; Security architecture and procedures for 5G System (3GPP TS 33.501 version 15.1.0 Release 15).*, https://www.etsi.org/deliver/etsi_ts/133500_133599/133501/15.01.00_60/ts_133501v150100p.pdf.

[16] *USRP B210*, https://www.ettus.com/product/details/UB210-KIT.

[17] *srsLTE*, https://github.com/srsLTE.

[18] *LimeSDR*, https://www.crowdsupply.com/lime-micro/limesdr.

[19] A. Shaik, R. Borgaonkar, S. Park, and J.-P. Seifert, "On the Impact of Rogue Base Stations in 4G/LTE Self Organizing Networks," in *Proceedings of the 11th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, NY, USA, 2018, pp. 75–86.

[20] *3GPP. Non-Access-Stratum (NAS) protocol for Evolved Packet System (EPS); Stage 3 Specification 3GPP TS 24.301 ver. 12.8.0 Rel. 12.*, [Online]. Available: http://www.3gpp.org/dynareport/24301.htm.

[21] H. Li, X. Xu, C. Liu, T. Ren, K. Wu, Z. W. Cao, Xuezhi, Z. Y. Yu, and D. Song, "A Machine Learning Approach To Prevent Malicious Calls Over Telephony Networks," in *Proceedings of the 2018 IEEE Symposium of Security and Privacy*, 2018.

[22] B. Reaves, L. Blue, H. Abdullah, L. Vargas, P. Traynor, and T. Shrimpton, "AuthentiCall: Efficient Identity and Content Authentication for Phone Calls," in *26th USENIX Security Symposium*, Vancouver, BC, 2017, pp. 575–592.

[23] B. Reaves, L. Blue, D. Tian, P. Traynor, and K. R. Butler, "Detecting SMS Spam in the Age of Legitimate Bulk Messaging," in *Proceedings of the 9th ACM Conference on Security & Privacy in Wireless and Mobile Networks (WiSec)*, New York, NY, USA, 2016, pp. 165–170.

[24] X. Chen, J. Meng, Y. C. Hu, M. Gupta, R. Hasholzner, V. N. Ekambaram, A. Singh, and S. Srikanteswara, "A Fine-grained Event-based Modem Power Model for Enabling In-depth Modem Energy Drain Analysis," *Proc. ACM Meas. Anal. Comput. Syst.*, 2017.

[25] *Mobile Security Research Hall of Fame*, https://www.gsma.com/aboutus/workinggroups/working-groups/fraud-security-group/mobile-security-research-hall-fame.

[26] R. Borgaonkar and S. Udar, "Understanding IMSI Privacy," in *BlackHat*, 2014.

[27] D. Abodunrin, Y. Miche, and S. Holtmanns, "Some dangers from 2G networks legacy support and a possible mitigation," in *Communications and Network Security (CNS)*, Sept 2015, pp. 585–593.

[28] K. Nohl, "Mobile self-defense." [Online]. Available: https://events.ccc.de/congress/2014/Fahrplan/system/attachments/2493/original/Mobile_Self_Defense-Karsten_Nohl-31C3-v1.pdf

[29] *3GPP Standard*, www.3gpp.org.

[30] N. Golde, K. Redon, and J.-P. Seifert, "Let Me Answer That for You: Exploiting Broadcast Information in Cellular Networks," in *Proceedings of the 22Nd USENIX Conference on Security*, 2013, pp. 33–48.

[31] D. Rupprecht, K. Kohls, T. Holz, and C. Pöpper, "Breaking LTE on Layer Two," in *IEEE Symposium on Security & Privacy*, May 2019.

[32] R. P. Jover, "LTE security, protocol exploits and location tracking experimentation with low-cost software radio," *CoRR*, vol. abs/1607.05171, 2016. [Online]. Available: http://arxiv.org/abs/1607.05171

[33] M. Arapinis, L. I. Mancini, E. Ritter, and M. Ryan, "Privacy through Pseudonymity in Mobile Telephony Systems," in *NDSS*, 2014.

[34] M. Arapinis, L. Mancini, E. Ritter, M. Ryan, N. Golde, K. Redon, and R. Borgaonkar, "New Privacy Issues in Mobile Telephony: Fix and Verification," in *Proceedings of the 2012 ACM Conference on Computer and Communications Security*, ser. CCS '12. ACM, 2012, pp. 205–216.