

# You Can Run, but You Can't Hide: Exposing Network Location for Targeted DoS Attacks in Cellular Networks

Zhiyun Qian<sup>1</sup> Zhaoguang Wang<sup>1</sup> Qiang Xu<sup>1</sup> Z. Morley Mao<sup>1</sup> Ming Zhang<sup>2</sup> Yi-Min Wang<sup>2</sup>

<sup>1</sup> University of Michigan <sup>2</sup> Microsoft Research

## Abstract

*An important class of attacks against cellular network infrastructures, i.e., signaling DoS attack, paging channel overload, and channel exhaustion attack, operates by sending low rate data traffic to a large number of mobile devices at a particular location to exhaust bottleneck resources such as radio resource and radio resource controller. We term this class of attack targeted DoS attack on cellular networks, given the focus on a specific location. The success of such attacks depends on an important but unvalidated assumption that it is possible to create a relatively accurate hit list of mobile devices associated with the target network location to which attackers can direct traffic. In this study, we take an important first step to validate this assumption by performing a large-scale study on the feasibility of these attacks. In addition, we propose accurate and efficient techniques for generating IP address to network location association in real time for existing commercial UMTS networks.*

*Our technique relies on developing and measuring network signatures consisting of both static and dynamic features of key network elements such as Radio Network Controllers (RNCs) that are stable within the same location but distinct across locations. We find that a single feature in some cases is unique enough to locate a city such as NYC. With as little as 800kbps probe rate, we can identify a sufficient number of IPs to accurately target a network location, after scanning millions of IPs within 1.2 hours to effectively impose 2.5 to 3.5 times the normal load on the network.*

## 1 Introduction

Data cellular networks are perennially constrained by limited radio resources due to ever-increasing user demand. To enable efficient resource reuse, there exists built-in support for intelligently allocating radio resources among multiple users and releasing them when they are perceived no longer actively in use.

However, the mechanism for implementing the radio resource management primitives requires complex and heavy signaling procedures, rendering cellular networks vulnerable to a variety of low-rate *targeted DoS attacks* [25, 35, 32]. Due to the low bandwidth property, they can be launched from merely one or more external Internet hosts and can deny service to a large number of legitimate users in a particular area. In Figure 1, this class of attacks is illustrated at the bottom left side. Different from another class of attacks such as HLR attack [34] (on the top right of the figure) that requires access to cellular botnets, the low-rate targeted DoS attacks are considered much more realistic given the weaker requirements.

Although targeted DoS attacks are seemingly serious threats, they are still hypothetical. In particular, an important requirement, *i.e.*, locating a sufficient number of mobile devices in the same area (the so-called hit-list) is not fully investigated. Previous work [19] on exploiting SMS functionality to overload cellular voice service proposed using phone numbers under specific area codes to generate the hit-list. However, phone numbers cannot be directly translated into IP addresses needed for launching targeted DoS attacks on cellular data services and have limited location accuracy (detailed in §5.3).

To bridge this gap, we develop localization techniques to identify IP addresses under a given area via active probing and fingerprinting. It works by controlling a probe phone under the target area to measure the baseline signature of that location, then searching for other devices with sufficiently similar signature to the baseline. Our observation is that it is possible to actively measure characteristics and configuration parameters associated with network elements that are similar at the same location and distinct across locations. We empirically evaluate this approach on two large UMTS carriers in the U.S. (anonymized as Carrier 1 and Carrier 2 for privacy concerns). We find the approach promising in identifying a set of near-by mobile IPs with high accuracy. This is particularly true for big cities that often

have unique configurations or features to satisfy their load requirement. In certain cases, it is even possible to uniquely identify a big city such as NYC with only a single feature. Thus, our work demonstrates that the threat of targeted DoS attacks is real.

Besides exposing the potential abuse of network information to enable this class of targeted attack against cellular networks, our technique can also support legitimate privacy-preserving applications that rely to the knowledge of the number of nearby users to determine whether a user should send his location information while ensuring  $k$ -anonymous properties [21]. More generally, our technique opens up a new direction in understanding how critical infrastructures like cellular networks can leak information about their networks which leads to privacy implications, *e.g.*, in the form of location exposure.

In this work, we make the following contributions:

- We conduct the first large-scale empirical study on the feasibility of targeted DoS attack against commercial cellular data networks (with overview shown in Table 1), using data collected through our deployed mobile app on major smartphone platforms (with details of the app covered in §3). We show that 80% of the devices keep their device IPs for more than 4 hours, leaving ample time for attack reconnaissance.
- We develop novel techniques to map IP addresses of mobile devices to a geographic area, using a combination of network features including static configuration settings, topological properties, and dynamic features.
- Using the proposed network signatures, we empirically validate the evidence of diverse network signatures across Radio Network Controllers (RNCs). We show that in big cities, the signature is typically unique enough to allow an attacker to locate enough IPs to impose 2.5 to 3.5 times the normal load on the network.

The rest of the paper is organized as follows. §2 presents the background on UMTS network as well as the attack overview, followed by the IP-related feasibility analysis in §3. §4 describes the methodology for localization, and §5 discusses the evaluation results. Possible defense solutions are described in §6. Finally we cover the related work in §7 and conclude in §8.

## 2 Background and Overview

In this section, we first describe the UMTS background with a focus on the measurable parameters which can be configured differently across network locations. We then discuss the intuition of the technique and the detailed steps of the attack.

### 2.1 UMTS Background

**UMTS architecture.** A UMTS network consists of three subsystems: Devices, Radio Access Network and Core Network [24]. They form a hierarchical structure where the lowest layer is the device, followed by radio access network consisting of base-stations and Radio Network Controllers (RNCs). At the highest level is the Core Network which has Serving GPRS Support Node (SGSN) and Gateway GPRS Support Node (GGSN). The latter connects to the Internet. Our focus is on the radio access network—the DoS target.

**Radio Resource Control and State Machine.** In UMTS, to efficiently utilize the limited radio resources (*i.e.*, physical communication channels), the radio resource control (RRC) protocol uses a state machine (shown in Figure 2) associated with each device typically with three RRC states—**IDLE**, **CELL\_DCH** (or DCH) and **CELL\_FACH** (or FACH). Different states have different amount of radio resource assigned. **IDLE** has no radio resource allocated. To send or receive any traffic, it has to promote to **DCH** where a dedicated channel is assigned to the device. In **FACH**, devices are assigned a shared channel which only provides limited throughput (less than 20kbps). It is designed for applications requiring a very low data rate. There are several configurable static parameters associated with state transitions. For example, Downlink/Uplink (DL/UL) Radio Link Controller (RLC) buffer size is used to determine when the FACH → DCH transition occurs (we refer to it as **queue size threshold** thereafter). If the number of bytes sent or received exceeds the threshold, the state transition will be triggered. Another example is the **inactivity timer** which determines when the DCH → FACH transition occurs. They are all configured at the RNC and may differ across RNCs in different locations, making them good candidates to construct the network signatures. Note that different RNC implementations may also employ slightly different state machines due to vendor differences. For instance, there could be a **IDLE** → **FACH** transition instead of **IDLE** → **DCH**. In some cases, new states (*e.g.*, **CELL\_PCH**) are introduced to further optimize radio resource. These differences can also be part of the signatures.

It is worth mentioning that in the next generation LTE networks, similar resource control mechanism and control parameters also exist [5].

### 2.2 Attack Outline

**RNC-level Localization.** At a high level, our intuition is that network elements (*e.g.*, RNC) in different locations may be configured differently,

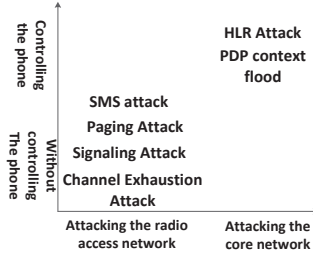


Figure 1. Cellular attacks

Category	Measurement target
MobileApp	Cellular IP reachability analysis (§3.1)
MobileApp	IP duration analysis (§3.2)
Probing	Active vs. Idle IPs (§3.3)
MobileApp	IP locality analysis (§4.1)
MobileApp	Phone-number-based localization accuracy (§5.3)

Table 1. Attack feasibility analysis

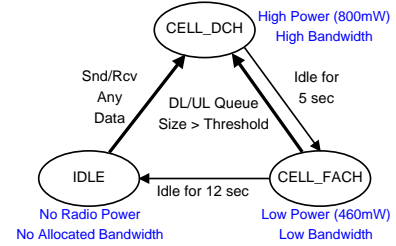


Figure 2. State machine for Carrier 1

therefore exhibit distinct network signature. Since geographically close-by devices are governed by the same RNC, they also share the same signature. We discuss a more complete list of features of our proposed network signature in §4.2.1. Since most features identified are associated with the RNC, the localization granularity is naturally at the RNC level. Typically, a big city such as NYC has multiple RNCs, while a few small cities may share the same RNC. To clearly illustrate the attack, we outline the steps.

1. The attacker has access to a *probe phone* in the target location. This phone measures the network signature of that location and uses it as the baseline to match against signatures of other probed phones. The probe phone could either be compromised or owned by the attacker. Note that we do not assume access to a cellular botnet. Otherwise, it is trivial to obtain the location information (*e.g.*, GPS) from the phone.
2. The attacker first exclude IPs that are unlikely under the target area through a coarse-grained localization step. For instance, the attacker can exploit the approximate IP locality in cellular networks, *i.e.*, each GGSN responsible for a geographic region assigns distinct IP blocks to phones (See §4.1).
3. The attacker probes the remaining IPs to measure their associated signatures (as a fine-grained localization step) with certain time and resource constraint. The signatures are matched against the baseline signature. The ones that are sufficiently similar are considered to be under the same RNC.
4. After identifying enough IP addresses for the target network location, the attacker can launch the targeted DoS attack.

### 3 IP-related Feasibility Analysis

In this section, we investigate the feasibility of our localization technique and the targeted DoS attack in general. First (§3.1), we study how many carriers allow cellular IPs to be probed directly (a requirement for scanning/fingerprinting and the DoS attack itself). Second (§3.2), we quantify how long a device keeps its cellular IP address (to ensure IPs are still valid

after fingerprinting). Third (§3.3), we measure how many devices are online versus active (to estimate the additional attack load that can be introduced).

**MobileApp.** To answer the above questions, we collect empirical data using a mobile app we deployed on major platforms (referred to as *MobileApp* thereafter) which measures various aspects of any cellular network including throughput and port blocking behavior. It also reports additional information such as Carrier, GPS, IP address, timestamp, Signal Strength, and 7-digit phone number prefix (with user’s consent). As of March, 2011, there are 100,223 users who have voluntarily downloaded and run our app all over the world. There is a small subset consisting of 1,683 users who opted to run the periodic test that report the device’s IP address every one hour and allow the signature measurement to be conducted from our server. We focus on the data set collected over a 2-week duration from Nov 7th to 21th, 2010. The complete list of analysis based on the *MobileApp* is summarized in Table 1.

#### 3.1 IP Reachability

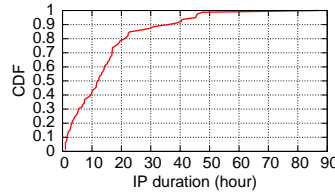
Some carriers deploy NAT or firewall to ensure traffic originated from the Internet cannot reach the mobile devices directly. They affect signature-based localization as well as the set of existing attacks assuming direct reachability to the phones. Based on the MobileApp data set, we collected reachability information of 180 UMTS carriers in different continents as summarized in Table 2. Among them, 77 of them assign public IP addresses to devices, 52 of which are directly reachable from the Internet; 103 carriers assign private IP addresses to devices (with NAT), 40 of which are reachable because they allow either device-to-device probing within the same network or one device to spoof other device’s IP address (4 out of the 40 allow IP spoofing). In summary, regardless of public or private IP carriers, the percentage of carriers that allow direct reachability is  $\frac{40+52}{180} = 51\%$ . This result motivates our localization technique.

In those cases where carriers do not allow phones to be probed in any way, existing attacks discussed

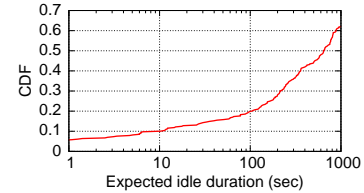
Carrier count	Private IP		Public IP	
	Reachable <sup>1</sup>	Not	Reachable	Not
180	40	63	52	25

<sup>1</sup>In-NAT device-to-device probe allowed

**Table 2. Reachability for 180 UMTS carriers.**



**Figure 3. CDF of IP duration**



**Figure 4. CDF of idle duration**

in [25, 35, 32] will also be deterred. However, denying reachability is not without any cost. For instance, it may hurt peer to peer applications as well as applications that require running servers on the phones.

### 3.2 IP Duration

It is critical to understand how long a device keeps the same IP address as it takes time to measure the signature and identify the location of an IP. Previous work has shown that when NAT boxes are deployed, the public IPs assigned by NAT can change frequently in cellular networks [15]. For instance, they show that with the same private IP address, the public IP address can change on the order of tens of seconds. However, device IPs (IPs directly assigned to devices) often change much less frequently in contrast to NAT-assigned IPs. Typically such IPs only change when the device reboots or the network interface is switched (*e.g.*, WiFi to 3G). Carriers do not have incentives to frequently reassign new device IPs as it incurs overhead.

Indeed, from the *MobileApp*'s periodic test data consisting of (deviceID, timestamp, deviceIP) tuples, we quantify the expected IP duration of each device as shown in Figure 3 and find the duration of the device IP is fairly long, with more than 80% devices keep their IP addresses longer than 4 hours.

### 3.3 Online Devices vs. Active Devices

We define *online* devices as the ones reachable but may or may not be actively using the network. *Active* devices as the ones reachable and also actively using the network. The ratio of active to online devices determines the upper bound of additional load an attacker can impose on the network.

To quantify the number of online and active devices, we conduct a TCP SYN probing experiment on 12 sampled /24 IP prefixes used in the cellular network operated by Carrier 1 in the U.S. in March, 2011 for two hours at different time-of-day (one hour at noon, one hour after midnight). We probe all IP addresses in the range every 30 seconds. Though the analysis does not cover all the IP ranges, we believe the result is still representative based on randomly sampled IP prefixes.

For validation, we confirm that TCP SYN packets can indeed elicit response from the phone instead of the firewall with spoofed responses. We verify this by launching two different “TCP traceroutes” from the server to the phone – if the probing packet is non-SYN packet, the traceroute will discontinue at the firewall hop (the firewall drops out-of-state TCP packets) while SYN packet can move forward until it reaches the phone and elicit response. To infer if a phone is actively using the network, we exploit the IP ID field in the IP header in the response packets [31]. In practice, many operating systems (such as Windows family) increment a system-wide IP ID counter for every outgoing IP packet. If two consecutive response packets have IP ID values  $X$  and  $X + 1$ , it means the phone is **idle** for the 30-second probing interval. Otherwise, it is **active**.

On average, there are 22% of the probed IPs respond (other IPs are likely unassigned at the moment we probe). Out of the responsive/online devices, the active ratio is 20% during day time and 13% at midnight, showing some time-of-day pattern. The ratio indicates that an attacker can introduce at most 5 times the normal load at day time, assuming all IPs can be correctly located. Even if the attacker finds only half of the online phones, he can still impose 3 times the normal load on average, or  $\frac{50\%}{20\%} = 2.5$  to  $\frac{50\%+20\%}{20\%} = 3.5$ , depending on how many identified IPs are overlapping with the ones that are already active. Either way, the attack load is significant enough to cause damage assuming a common over-provisioning factor of 1:2 [20].

From the probing, we can also infer the average idle duration for each IP address. This information helps us understand how likely there will be background traffic interfering with the feature measurement described in §4.2.2. Figure 4 shows that 85% of the IP addresses have an idle duration of 100 seconds, which are long enough for most measurements.

In summary, we have studied various aspects of real networks and collected evidence demonstrating that the localization scheme and the targeted DoS attack are indeed feasible.

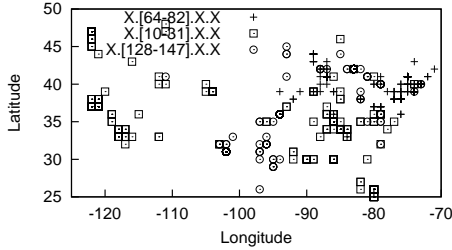


Figure 5. IP address locality

## 4 Localization Methodology

### 4.1 Coarse-grained Localization using IP Locality

IP address is known to have locality on the Internet – close-by IPs in the address space tend to share geographic proximity. However, as pointed out by previous work [36], a large geographic area may share the same set of IP pools managed at a GGSN location. We verify this using 13,784 samples of GPS-IP pairs of Carrier 1 and 17,589 pairs of Carrier 2 from the MobileApp data, during the time period from Sep 2009 to Nov, 2010.

Figure 5 shows Carrier 1’s private IP address locality. We observe the same private IP range is indeed shared by a large geographic area. For example, X.64.X.X – X.82.X.X are always allocated to devices from the East Coast, while IPs in X.128.X.X – X.147.X.X are mostly assigned to devices on the Central or Northeast. In total, we find 4 such disjoint IP ranges which likely correspond to different GGSNs [36]. Similar results are found for Carrier 2. In summary, device IPs only show coarse-grained geographic locality. Nevertheless, it allows the attacker to exclude a large range of IPs without ever probing them.

For an attacker without prior knowledge about which IP range corresponds to the target area, we devise a simple technique to obtain such knowledge — one can reset the network interface programmatically to continuously obtain new IP addresses. It takes about 30 seconds to get a new IP address every time. In our experiment, this technique allows us to obtain the IP pool of X.128.X.X – X.147.X.X within about an hour.

### 4.2 Fine-grained Localization using Network Signatures

From coarse-grained IP locality, the attacker can obtain a set of candidate IPs only knowing their rough location at the granularity of several states in the U.S. To narrow down to a more fine-grained target location,

Feature	Type	Measurement cost
Min-RTT	Dynamic	Medium
DCH tail time	Static	Medium
FACH DL queue size threshold	Static	Medium
FACH UL queue size threshold	Static	Medium
FACH queue consumption rate	Static	High
Avg promotion delay	Dynamic	Low
RRC State machine	Static	Low
Max number of RLC retransmission	Static	High

Table 3. Features of the network signature.

the next step is probe these IPs directly to obtain their network signatures.

#### 4.2.1 Network Signature Definition

Here we define network signature as a set of measurable features that are potentially distinct across different locations. We discuss why the signature exists and how we select them:

- Incremental changes:** different hardware components are purchased and upgraded at different time to improve capacity of the network or to introduce new technologies [4, 3], thus likely introducing heterogeneity in the RNC hardware. For example, min-RTTs are reduced when HSUPA are deployed on top of HSDPA compared to when HSDPA deployed alone [23]. Also, different RNC products can have slightly different state machine behavior as explained before. As 4G LTE network technology is gradually deployed, we expect even more heterogeneity across network locations.
- Connectivity between RNC and base-stations:** this may vary across locations. The network delay variation may be due to either RNC processing delay difference or base-station-to-RNC link delay difference (*e.g.*, link congestion) [25, 7], thus affecting the observed state promotion delay.
- Configurations:** Different RNCs have different static parameter settings to accommodate the load at different locations. For instance, big cities often configure resource-recycling parameters (*e.g.*, inactivity timer) more aggressively to accommodate more users.

The general criteria for selecting features is that any *measurable properties* associated with RNC can be considered. We list the potential set of features in Table 3, most of which have been introduced in §2.1. Although it may not be complete, it is a good starting point. These features can be classified as static or dynamic. Static features are configured statically on the RNC. They are thus very stable and relatively easy to measure. Dynamic features, on the other hand, are likely to vary over time thus harder to measure. Also, some features are more expensive to characterize.

For example, the maximum number of allowed RLC retransmission [26] and the FACH queue consumption rate may take many measurements to correctly identify.

In our current study, we pick the 5 features highlighted in Table 3 (and described in §2.1) that are relatively easy to measure. The methodology for measuring them is described in the next section. We did not include the RRC state machine feature because we did not find a corresponding ground truth RNC to test our measurement methodology. The local RNC on which we rely for testing did not implement the CELL\_PCH state. A dedicated attacker can however spend more effort and take advantage of the remaining features to refine the signature.

#### 4.2.2 Measurement Methodology

Our probing methodology assumes control at only one end (*i.e.*, the server side). It is different from a previous study [29] that devised a methodology to measure similar features but with complete control on both ends. In our case, we have to overcome additional challenges such as background traffic on the device which may distort the measured result. Moreover, when state promotion delay is measured from the server, there is an additional variable, *i.e.*, paging delay that needs to be factored out. Finally, without controlling the device, it is hard to control the size of the packet responded by the device (thus hard to measure the UL queue size threshold). For instance, we can use a large TCP SYN packet to probe a device, but the size of the response packet is fixed (40 bytes TCP RST). To control the uplink size, we use UDP probing where the response packet is 24 bytes larger than the original probing packet because the triggered *ICMP port unreachable* packet embed the original UDP packet and wrap it with a new ICMP and IP header. The ICMP response packet is limited to at most 576 bytes as per RFC [6]. We will discuss how to use TCP and UDP probing by adjusting the probing packet size to measure DL and UL queue size threshold respectively.

**DCH inactivity timer.** As described in §2.1, this parameter determines when to release radio resource (after  $N$  seconds of inactivity). The procedure to measure inactivity timer is as follows: (1). The attacker first sends a large probe packet (*i.e.*, 1200 bytes). If the phone is in IDLE or FACH, it will trigger a state promotion to DCH; otherwise, it will reset the inactivity timer [30]. (2). Then the attacker waits for some interval,  $X$  seconds before he sends the next large packet. (3). If the next packet incurs an RTT sufficiently large to include the state promotion, this indicates the power state has changed from DCH to FACH after  $X$  seconds of inactivity, and therefore the inactivity timer

is  $X$  seconds. (4). Else, if the next large packet experiences a small RTT, indicating that the power state is still in DCH, it then set  $X = X + \Delta$  and go back to step (2). We set the initial  $X$  to be 2.5s which is small enough to catch all the known values of inactivity timer. We use a measurement granularity of 0.2s ( $\Delta = 0.2s$ ). It is not useful to increase the granularity further since the variation of normal cellular RTT will prevent us from getting more fine-grained results.

As mentioned, one challenge is background traffic, which may prematurely reset the timer, inflating the measured inactivity timer. The counter strategy is to continue the measurement only when the initial state is in IDLE. The assumption is that if the phone is not using network now, it is likely to continue to stay idle in the near future (demonstrated in Figure 4). After all in a real attack, the timer inference can finish much more quickly given the attacker can profile the timer value in the target RNC offline using his attack phone. Specifically, instead of measuring all possible timer values, he only needs to make a binary inference – whether the tested IP has a target timer or not. Nevertheless, the above general methodology assumes no prior knowledge of the timer value; we use it to understand how diverse they can be across locations.

The other source of noise is fast dormancy [13], a mechanism implemented on some devices to release radio resource before the timer expires and save battery life on the phone, causing early transition from DCH to IDLE. However, since the power state transitions directly from DCH to IDLE instead of to FACH, the next observed RTT is much larger and thus distinguishable from the non-fast-dormancy case. More importantly, it is not always implemented or triggered every time. Thus, we can repeat the measurements several times and discard the fast dormancy cases. In practice, we find UDP probing rarely encountered fast dormancy (compared to TCP probing), likely due to more bytes being exchanged in the UDP case.

**FACH  $\rightarrow$  DCH DL/UL queue size threshold.** As described before, there exists a queue size threshold for DL and UL respectively. Using TCP probing, the size of the TCP RST response is fixed, allowing DL packet size adjusted independently without affecting the UL packet size (ensuring UL threshold is not exceeded first). The following depicts the procedures on the server side to measure DL queue size threshold: (1) send a large packet to make sure the state will be in DCH; (2) wait for  $T = \text{Timer}$  seconds to let the power state enter FACH; (3) send a packet with size  $Y$ ; (4) If a state promotion is triggered (identified by a large RTT), then record  $Y$  as the queue size threshold and exit; (5) else, try a larger packet size  $Y = Y + \xi$  and go back to (3). We set the initial  $Y = 20$  and  $\xi = 30$  in the MobileApp

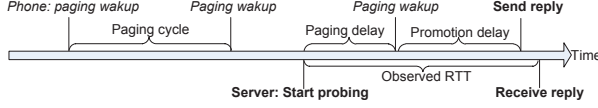


Figure 6. An example of paging delay.

experiment. In the later large-scale probing experiment, we change  $\xi = 10$  bytes in an attempt to measure more fine-grained queue size threshold. Nevertheless, the later experiment confirms that the results with  $\xi = 30$  are still valid (without missing any values in between).

Measuring UL queue size threshold is more difficult, because distinguishing between DL threshold and UL threshold being exceeded is hard — in both cases we just observe a large RTT indicating that state promotion is triggered. Fortunately, since we now already know the DL queue size threshold via TCP probing, we can send UDP packets with size smaller than the DL threshold so that they are guaranteed not to exceed the DL queue threshold. If there is still a state promotion triggered, then it has to be the UL packet that exceeds the UL queue size threshold. This is why we have to measure the DL threshold before the UL threshold. There are two constraints in the results: 1). The UL response packet is always 24 bytes larger than the DL probe packet (as described earlier), therefore, we can measure the UL threshold up to 24 bytes larger than the DL queue threshold. 2). The max size of the UL response packet is 576 bytes (as discussed earlier). As a result, the UDP probing can only measure up to 576 bytes for the UL threshold.

**IDLE → DCH Promotion delay.** Although it is easy to measure promotion delay from the phone, it is much more difficult to do so from the other end. This is because when a downlink packet is to be received in **IDLE** state, it may get delayed significantly due to a process called paging [2].

Figure 6 shows an example where the phone is in **IDLE** state, only periodically wakes up to listen for incoming packets. If a packet happens to arrive between two paging occasions, it will be delayed until the next paging occasion. In the worst case, the delay can be as large as a full paging cycle length. The paging cycle length is determined by Core Network and is stable across RNCs [2]. According to the 3GPP specification [1], the protocol should support several values: 0.64s, 1.28s, 2.56s, *etc.* In practice, we observe it to be 2.56s for Carrier 1 and 1.28s for Carrier 2. It is stable and validated through a hardware power-meter: every 2.56s a power spike is seen, and we can observe that the promotion process must wait after the spike.

To overcome the effect due to paging delays, we design a probing technique using binary search that

can distinguish the paging delay from the estimated promotion delay. It requires several rounds of probing (the exact number of rounds depends on the accuracy desired). The high-level idea is that we gradually narrow down the exact paging occasion. The more we probe, the closer to the paging occasion the probing will be. We validated the inferred promotion delays match the duration of power spikes using power-meter. Below is the detailed methodology:

We use the following notations: we denote  $C$  as the paging cycle,  $S_i$  as the start time of  $i^{th}$  probe mod by  $C$  ( $0 \leq S_i \leq C$ ),  $P_i$  as the paging delay of the  $i^{th}$  probing, and  $T_i$  to be the RTT of the  $i^{th}$  round.

Initially, without any knowledge of the paging occasion, we conduct the initial two probes spaced out with an offset of  $\frac{C}{2}$ . More specially,  $S_2 - S_1 = \frac{C}{2}$ . An example outcome is shown in Figure 7.(a). We can infer  $P_1 < P_2$  based on the fact that  $T_1 < T_2$ , and paging occasion should be somewhere between  $S_1$  and  $S_2$ . Despite potential variation in promotion delay (on the order of hundreds of milliseconds), the spacing between  $P_1$  and  $P_2$  of  $\frac{C}{2}=1.28s$  should be large enough to overcome the noise.

We select the next  $S_i$  to bound the paging occasion further. For example, in Figure 7.(b), the start time is selected to be  $S_3 = \frac{S_1+S_2}{2}$  given  $T_1 < T_2$ . Note that here the difference between  $P_3$  and  $P_2$  is even bigger than the difference of  $P_1$  and  $P_2$ , making each step of the algorithm robust to the noise of promotion delay variation.

The process is repeated until we are able to bound the  $S_i$  to be very close to the paging occasion. We pick  $i$  to be 8 since it is able to bound the distance to be less than  $\frac{2.56s}{2^8} = 10ms$ . Even though each individual promotion delay may vary significantly, the average is expected to be stable, thus collecting multiple samples of promotion delay is useful. After inferring the paging occasion, we are able to recover each previous paging delay  $P_i$ , and the corresponding promotion delay can then be computed simply as  $T_i - P_i$ .

Note that it is possible there are background traffic during the 8 rounds of probing since it spans  $8 \times 30 = 240$  seconds. However, such background traffic only matters when the measured promotion delay is very small, indicating the device is already in DCH instead of IDLE. Such cases can be easily excluded and the attacker can simply try again in the next round.

**Minimum RTT in DCH.** Although it has been shown that cellular network RTT is highly variable, we find the minimum (over 40 measurements) is surprisingly stable within the same area (See Figure 10). It is also diverse in different geographic areas. Our local experiment shows that the standard deviation of 100 min-RTTs (each is computed over 40 RTT measurements) measured in

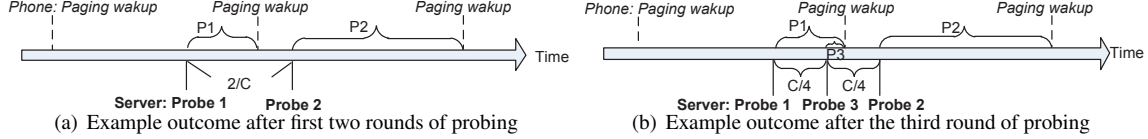


Figure 7. Illustration of our proposed paging delay elimination method to measure promotion delay.

6 hours is only 2.003. In fact, computing min-RTT over 30 measurements is also acceptable as its standard deviation is 2.07. 10 measurements will however increase the standard deviation to 3.46.

**Validation.** We thoroughly validate the proposed methodology using local experiments over two different UMTS carriers across several types of phones. The results are consistent and match with those collected assuming direct access to the phone. For instance, we verified that the UL/DL queue size threshold measurements are consistent with or without direct access to the phone.

## 5 Evaluation

In this section, we evaluate the effectiveness of the signature-based localization through empirical data collected from the MobileApp as well as a large-scale probing experiment. We focus on three main evaluations.

1. We quantify the distribution and entropy of each feature.
2. We compare the signature-based localization with the state-of-the-art phone-number-based approach.
3. We evaluate the scalability, *i.e.*, time requirement, to locate enough number IP addresses.

**Data collection.** We collect data from two main sources. One is from the periodic test of the MobileApp, which allows us to obtain ground truth RNC, timestamp, and signature to evaluate two major UMTS carriers in the U.S. covering 20 RNCs in Carrier 1 (shown in Figure 10) and 43 RNCs in Carrier 2 from Nov 8th to Nov 19, 2010. In this dataset, we collect all five features except the UL queue size threshold, which can be measured in the actual attack via UDP probing as discussed in §4.2.2. The other dataset is from large-scale probing on Carrier 1 in March 2011, the goal of which is to understand the signature distribution more comprehensively without knowing where each IP is located. The two datasets jointly allow us to evaluate the accuracy of our localization technique.

**RNC ground truth.** Even though we run our MobileApp on the phone, there is still no direct way to obtain the current RNC the phone is connected to. As an approximation, we obtain “location area code” (LAC) — an identifier that covers an area that is usually

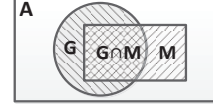


Figure 8. Illustration of coverage and accuracy

comparable to an RNC [9] through standard platform APIs.

**RNC/LAC coverage estimation.** To understand the challenge to distinguish different RNCs, we first need to estimate the number of RNCs and their coverage. To do so, we leverage the OpenCellID project [11], an open source project aimed to create a complete database of LAC number and Cell IDs worldwide. The data shows 1596 unique LAC numbers for Carrier 1 with about 3/4 of the U.S. covered. A rough estimate based on area uncovered leads to about 2000 LACs in the entire U.S. Since the number of GGSN locations is estimated to be 4 (See §4.1), a single GGSN should cover about 500 LACs/RNCs. Typically, a big city such as NYC has multiple RNCs (to accommodate the load), while a few small cities may share the same RNC (verified by LAC and location data from OpenCellID).

**Evaluation methodology.** As described in §2.2, in the actual attack, an attacker measures the baseline signature from his controlled probe phone and then compare it against signatures of other probed devices in real time. We simulate this process by considering each signature sample in the MobileApp as a baseline (where we know its ground truth RNC) and compare it with other samples for a similarity match. Static features should have exactly the same value to be considered a match. For dynamic features, *e.g.*, average promotion delay and min-RTT, we allow a **signature distance threshold parameter (SD-threshold** in short) to tolerate some variation. For example, if we set the min-RTT SD-threshold to 5ms, then all samples within  $\pm 5$ ms of the baseline sample are considered to match the baseline. For min-RTT and promotion delay SD-threshold, we select a large range of them from (5ms, 50ms) to (20ms, 400ms). We discuss later how set the SD-threshold to balance between the coverage and accuracy metrics.

**Coverage and Accuracy metrics.** As illustrated in Figure 8, the sets  $A$ ,  $G$ , and  $M$  represent different signature samples.  $A$  represents signature samples for



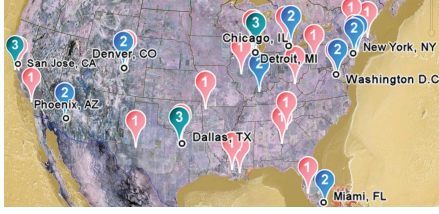


Figure 9. Carrier 1's queue threshold

all RNCs.  $G$  denotes all the signature samples under the target RNC.  $M$  corresponds to the signature samples inferred to be under the same target RNC (using the localization technique). Ideally, localization should output an  $M$  that is equivalent to  $G$ . However, in reality,  $M$  and  $G$  may only partially overlap. Based on the notations, we define two metrics to evaluate the performance of the localization:  $Coverage = \frac{|G \cap M|}{|G|}$  and  $Accuracy = \frac{|G \cap M|}{|M|}$ . Intuitively, coverage quantifies the fraction of phones that can be successfully identified under the target RNC. Higher coverage implies that the attack can impose higher load on the network. Accuracy quantifies the fraction of targeted phones that are indeed under the target RNC. Higher accuracy implies less wasted attack traffic.

### 5.1 Evaluation using the MobileApp Data

In this section, we mainly discuss the findings about the characteristics of the features and city-level properties (e.g., big cities usually have distinct values). We also briefly discuss how we can compute the localization coverage metric using just the MobileApp data.

**Static features.** We find that although inactivity timer and queue size threshold do not have many distinct values, they already show some interesting diversity patterns even from the limited samples.

For Carrier 1, we observe two main inactivity timer values: around 3s and 5s. For the queue size threshold, we also observe two main values: around 480 and 510 bytes. 480 tends to correlate with the 5s inactivity timer, while 510 tends to correlate with the 3s one. Besides these, there is one exceptionally large queue size threshold at around 1000 bytes observed at three places – Bay Area, Dallas, and Chicago Area (i.e., big cities). In Figure 9, we can clearly see the distribution of different queue size threshold values. Marker 1, 2, and 3 represent 480, 510, and 1000 bytes respectively. It is interesting that Marker 2 and 3 are always at big cities while Marker 1 is often present in small cities. In the cases where two Markers are closely located (e.g., Marker 1 and 3 in Dallas), we observe that Marker 1 is in fact at a nearby small city.

For Carrier 2, we observe similar patterns. Two

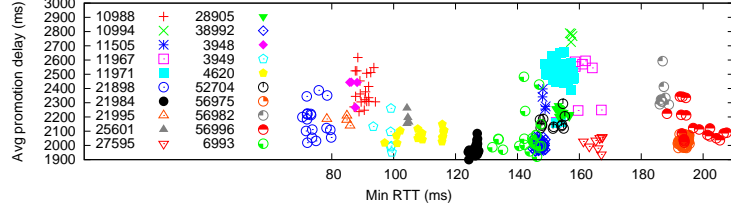


Figure 10. Carrier 1's 20 different RNCs' dynamic features

main values of inactivity timer: 3s and 8s, and two main values for queue threshold: around 110 and 300 bytes. However, there is also an exception of 1000 bytes which is only observed in New York City (multiple LACs in the city and different users consistently show this value), likely due to its higher load requirement. We also observe interesting patterns around big cities, e.g., Los Angeles is configured with 300 bytes while near-by California cities are configured with 110 bytes.

Despite the limitations of the small number of observed values, we note that the features are promising in distinguish big cities which often have bigger queue size threshold than their near-by cities. Coupled with dynamic features, the signatures will be even more unique.

**Dynamic features.** Figure 10 shows the distribution of Carrier 1's two dynamic features – promotion delay and min-RTT. Each data point is an average over 8 measurements of promotion delay (after subtracting the normal RTTs) and 40 measurements of RTT. We employ 95<sup>th</sup> percentile filtering to exclude outliers of promotion delay due to measurement noise. Data points with the same shape indicate that they are from the same LAC. We can see from Figure 10 that min-RTT is very stable in most cases, even though a user can move around in nearby locations connecting to different base-stations with varying signal strength.

For promotion delay, it is still reasonably stable but less stable than min-RTT. In Figure 10, we aggregate enough data points during off-peak hours (3AM - 6AM where promotion delays should be stable), which have relatively stable range of values and do show diversity across different LACs. In the cases where two different RNCs have similar signatures (e.g., at min-RTTs close to 150ms in the figure), we may need to combine static features to distinguish them.

**Coverage calculation.** Coverage is relatively simple to calculate using only the MobileApp data. As defined in §5,  $Coverage = \frac{|G \cap M|}{|G|}$ .  $G$  denotes all the signature samples under the target RNC. We use the samples collected in MobileApp data (where we know the ground truth RNCs) to serve as  $G$ .  $G \cap M$  is the signature samples correctly inferred to be under the target RNC. They both can be derived from the MobileApp data.

Inact. timer	queue threshold		Prob.	Inact. timer	queue threshold		Prob.	Inact. timer	queue threshold		Prob.	Inact. timer	queue threshold		Prob.
	DL	UL			DL	UL			DL	UL			DL	UL	
5s	480	484	41.1%	3s	510	>=534	4.3%	3s	480	244	1.7%	3s	1010	244	1.4%
3s	510	244	28.8%	3s	480	>=504	2.7%	3s	<=340	244	1.6%	3s	1010	>=576	1%
5s	480	>=504	6.9%	3s	480	484	2.6%	5s	<=340	244	1.5%	Others		6.4%	

Table 4. Static feature probability distribution from large-scale probing

For accuracy, however, using the MobileApp data alone is not sufficient because it is possible there are other unknown RNCs which have similar signatures, *i.e.*, collisions. This motivates our subsequent measurement on the general distribution of the signatures in an entire GGSN.

## 5.2 Evaluation via Large-Scale Probing

In this section, we focus on two results. First, we characterize the signature distribution of all five features via large-scale probing. Next, based on the distribution, we calculate the localization accuracy for the MobileApp data where we have the ground truth RNC.

**Evaluation methodology.** In our large-scale probing data, we measure all 5 features but only 4 features are measured in the MobileApp data (UL threshold was not included). When we join two datasets together, we need to make up for the missing feature in the MobileApp data. We choose a very conservative way to do this – we fill in a UL threshold such that the (DL,UL) threshold combination is the least unique possible. This way, the signature is never more unique than it should be, thus the resulting localization accuracy can only be worse.

**Signature distribution measurement for Carrier 1.** Previous results only provide a limited view on the signature distribution. To get a more comprehensive view, we sample  $\frac{1}{10}$  of an entire GGSN IP range in Carrier 1 comprising 20 /16 IP prefixes to measure all five features.

First, we use *Shannon entropy* – a metric measuring the uncertainty associated with a random variable, to measure how “random” the network signature variable is and how many RNCs can be distinguished on average in our context. Entropy is defined as  $H(X) = -\sum_x p(x) \log_2 p(x)$ , where  $X$  is the signature variable and  $x$  is a possible value for  $X$ . Here we break  $X$  into  $S$  and  $D$  representing static and dynamic features respectively. Thus  $H(X) = H(S, D) = H(S) + H(D|S)$ , by the law of entropy.

For  $H(S) = -\sum_s p(s) \log_2 p(s)$ , we need the  $s$  and  $p(s)$  which are both shown in Table 4. There are in total 11 different static feature  $s$  with at least 1% probability (appearance). Plugging in the probability  $p(s)$ , entropy  $H(S)$  is calculated to be 2.3 bits, indicating that we

can distinguish on average  $2^{2.3} = 5$  different groups of RNCs, using static features alone.

A closer examination of the table shows that, despite the inactivity timer taking on only two distinct values of 3s and 5s, the queue size threshold values are much more diverse than those from the MobileApp dataset (especially when combined with inactivity timer values). The distribution is biased towards the top two combinations where DL threshold values are 510 and 480, which match the values observed in our MobileApp dataset. This means that some RNCs will be harder to distinguish than others. Specifically, big cities with smaller inactivity timer and bigger queue size threshold are typically more distinct, considering that 41.1% of the static features contain 5s inactivity timer and 480 DL threshold (likely in small cities). The results indicate that big cities are more easily identified and targeted.

For  $H(D|S)$ , it can be computed as  $\sum p(s) \times H(D|S = s)$  by enumerating different static feature  $s$ . To compute  $H(D|S = s)$ , we slice the two-dimensional dynamic feature space into equal-sized bins using various threshold choice of min-RTT and promotion delay (as described in §5). Plugging in the probability associated with each bin, the computed  $H(D|S)$  varies from 2.5 to 5 bits. This means the combined entropy  $H(S, D)$  is between 4.8 and 7.3 bits, which can distinguish 28 – 158 different groups of RNCs on average. Remember this is without including other potential features in Table 3 which can provide additional entropy. Using the calculated entropy number and the estimated RNC number (500 per GGSN), we know on average there are  $500/158 = 3.2$  to  $500/28 = 17.9$  RNCs left indistinguishable to the attacker. Again, given the distribution is nonuniform, the actual accuracy will depend on which RNC is targeted.

**Accuracy calculation using both the MobileApp and probing data.** We already discussed how to compute coverage in §5.1. Now we discuss how to calculate  $Accuracy = \frac{|G \cap M|}{|M|}$ . First, accuracy calculation requires knowing  $M$  – the signature samples inferred to be under the target RNC. We perform the following procedure to output  $M$  – for each RNC’s signature sample in MobileApp (termed as **baseline sample**), we go through all the probed signature samples (without knowing their ground truth RNCs), and compare them against the baseline sample. The signatures that match

LAC	Signature		Location	Phone-number	
	Coverage	Accuracy		Coverage	Accuracy
11956	52%	41%	Queens, NY	N/A <sub>1</sub>	
11974	50%	83%	Newton, NJ	N/A <sub>1</sub>	
26507	50%	3%	Ann Arbor, MI	N/A <sub>1</sub>	
52704	54%	88%	Arlington, TX	15.5%	6%
52818	50%	75%	Arlington, TX		
52903	51%	77%	Fort Worth, TX	0%	0%

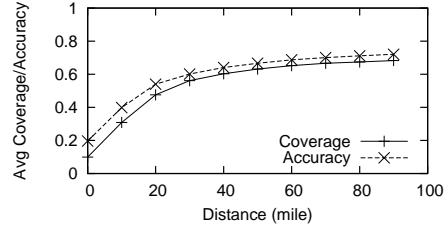
<sup>1</sup> No such phone number data from MobileApp

**Table 5. Coverage and accuracy comparison**

the baseline close enough are then included in  $M$ . Now the only unsolved part is  $|G \cap M|$  — the number of correctly inferred signature samples, which is in fact semantically equivalent to  $|G| \times coverage$  — out of all the signature samples that are under the target RNC, the covered ones are effectively the ones correctly inferred. Since we already know  $coverage$ , all we need is  $|G|$  at this point. We can estimate  $|G|$  based on the population distribution observed in our MobileApp. Specifically, if we know the fraction of users that are under a particular RNC out of all the users, then we can rewrite  $|G|$  into  $|N| \times p$  where  $|N|$  is total number of signature samples in the GGSN, and  $p$  is the fraction of population that are under the target RNC. Therefore,  $Accuracy = \frac{|G| \times coverage}{|M|} = \frac{|N| \times p \times coverage}{M}$ . Since our dataset is not complete enough to calculate  $p$  that is representative, we resort to the uniform population distribution where every  $p$  is equal only for the purpose of evaluation.

**Coverage and accuracy results.** As described in §5, the SD-threshold for deciding if two samples are close enough can be tuned according to the desired attack impact and the resource constraint. The more relaxed the SD-threshold, the more samples are considered under the target RNC and thus likely to include more correct samples (and better coverage and higher attack impact). However, this is at the cost of lower accuracy (*i.e.*, other non-related signature samples will be mistakenly inferred to be under the RNC). As an example, we tune the SD-threshold to achieve 50% coverage (which is equivalent to 2.5 to 3.5 times the normal load), the corresponding accuracy is shown in Table 5 for the 6 RNCs that are under the probed GGSN from the MobileApp data.

Here LAC 52704 and 52818 (Arlington) both belong to the “Dallas-Fort Worth-Arlington” metropolitan area that has a large population. The accuracy is high because their DL queue size threshold of 1010 is very unique. LAC 52903 (Fort Worth) also belongs to the same metropolitan area. However, its DL threshold is only 510 bytes. The reason why its accuracy is high is that its min-RTT is also very unique to distinguish it from most other LACs. LAC 26507 has a very



**Figure 11. Phone-number-based coverage and accuracy in small cities**

bad accuracy because the promotion delay is unstable and requires a large threshold value to maintain the coverage rate. Further, its static features are also the most commonly observed ones, further decreasing the accuracy. From the attacker’s perspective, he can also perform such signature distribution sampling offline regularly to estimate how distinct the target RNC’s signature is before the attack. To further improve the accuracy, we discuss an optional step to further filter IPs that are in other RNCs (in §6). Finally, the last two columns in Table 5 are the results of the phone-number-based approach described in the next section.

### 5.3 Comparison with Phone-Number-Based Localization

In the state-of-the-art phone-number-based localization [19, 35], the first seven digits of the phone number (area code) are used to map out its corresponding physical location (and carrier name) using the public NPA/NXX database [8]. The fundamental difference between signature- and phone-number-based approach is that phone number is a static property — users carry the same phone number to different places, even when they switch network providers. In contrast, network signature is dynamic and up-to-date, thus robust to user mobility. Based on the data we collected (described later), we find 75.4% of the carrier names returned from NPA/NXX database are incorrect, leading to extra effort for correction, assuming there is a way to do so.

**Data & evaluation methodology.** From our deployed MobileApp, with user’s permission we collected 15,395 entries consisting of first 7 digits of phone number, carrier name, GPS location from users all across the U.S. as ground truth. We use the data to calculate coverage and accuracy and compare them directly with the signature-based approach. Ideally we want to map each GPS location to its RNC and do an RNC-level comparison on the coverage and accuracy. However, the NPA/NXX database returns only city-level location. Thus, we approximate the RNC-level

Category	Signature-based localization/IP traffic	Phone-number-based localization/MMS
Localization overhead	× Scanning large range of IPs	✓ Crawl database, correct carrier name
Localization accuracy	– Better in large cities	– Better in small cities
Localization technique extensibility	✓ More features to improve accuracy	× Difficult to extend
Stealthiness in actual attack	✓ Hard for users to notice	× Easy for users to notice or disable
Feedback in actual attack	✓ Immediate feedback	× No feedback
Timing control in actual attack	✓ Easy to control	× Can get queued and delayed
Current defense effectiveness	– Can be hard to traverse NAT	– MMS blocked or rate limited
Detectability in actual attack	– Probing traffic can be detected	– MMS flooding can be detected

Figure 12. Comparison between signature-based and Phone-number-based attacks

comparison through city-level comparison instead. To begin, we map each GPS location to its corresponding city via a city-to-GPS database, and map the phone number prefix to its city via the NPA/NXX database. We term them *GPS-city* and *queried-city* respectively.

We treat big cities and small cities differently. For each big city *big*, which is defined to have multiple RNCs (or LACs from OpenCellID), we compute its city-level coverage as  $\frac{X}{Y1}$  where  $X$  is the number of cases where  $GPS-city = queried-city = big$ , and  $Y1$  is the number of cases where  $GPS-city = big$ . Similarly, we compute accuracy as  $\frac{X}{Y2}$  where  $X$  is the same as before, and  $Y2$  is the number of cases where  $queried-city = big$ . Note that since a phone number can only map to a city-level location, its RNC-level coverage and accuracy need to be divided further by  $n$  — the number of RNCs in that city. The coverage and accuracy are then  $\frac{X}{Y1 \times n}$  and  $\frac{X}{Y2 \times n}$ . Here we conservatively use  $n = 2$  to account for any inaccuracy when mapping from LAC to RNC. In reality,  $n$  is likely bigger and makes their results even worse. For small cities that may share a single RNC with other cities, we need a different method. Specifically, if the *GPS-city* and *queried-city* are within a certain physical distance, we consider them to be under the same RNC.

**Coverage and accuracy results.** For big cities, the average coverage and accuracy are 18.8% and 18.4% respectively for phone-number-based approach. Both values are much lower than the signature-based results in Table 5. In comparison with our signature-based localization, if we set the SD-threshold to allow the same accuracy, the coverage of the two big cities in Table 5 can increase to 60% – 80%, which is equivalent to 3 to 4 times the normal load. For small cities, Figure 11 illustrates how average coverage and accuracy over all cities change with different distance threshold. At 10 miles, which is our locally observed distance to observe two different RNCs in small-city area, the coverage and accuracy are about 30% and 40% respectively. At 30 miles, they are around 56% and 60%, indicating that the phone-number-based approach can perform better in the small city case as the signature-based approach has less

distinct signatures in small cities.

As a more direct comparison, we look at the intersection between the phone-number-based results and signature-based results from MobileApp, arriving at these following cities: Arlington and Fort Worth. As we can see in Table 5, the phone-number-based coverage and accuracy are much worse than the the signature-based approach. For Arlington,  $coverage = \frac{X}{Y1 \times 2} = \frac{11}{46 \times 2}$ ;  $accuracy = \frac{X}{Y2 \times 2} = \frac{11}{92 \times 2}$ . For Fort Worth,  $coverage = \frac{X}{Y1 \times 2} = \frac{0}{30 \times 2}$ ;  $accuracy = \frac{X}{Y2 \times 2} = \frac{0}{3 \times 2}$  ( $X, Y1, Y2$  are defined earlier). Surprisingly for city “Fort Worth”, both metrics are 0%, showing how ineffective phone-number-based approach can be. Other cities do not have enough phone number data points in MobileApp, thus are marked as N/A.

#### Other aspects.

Note that NPA/NXX database does not provide mappings from phone numbers to IPs. Therefore, the only way to attack the data channel using phone numbers is through MMS messages, as SMS uses control channel instead. To launch the attack, a large number of MMS messages must be sent regularly.

Different from sending direct IP traffic, sending MMS has a number of short-comings (shown in Table 12). We illustrate a few of them: 1). Repeated MMS messages are easy for users to notice and disable or even report to their carriers as abuse. In contrast, sending direct IP traffic is much less noticeable to users. 2). We find that sending MMS through the email interface (e.g., 123456789@mms.att.net) provides no feedback on the success of message delivery, thus potentially wasting significant effort by attacking phones that received no MMS at all (e.g., no MMS capability). 3). The timing of MMS delivery is very unpredictable, making the attack coordination difficult. In our local experiment, we sent MMS messages repeatedly every 30 seconds over 4 hours at night, which presumably should incur little delay. In contrast, we observe significant delays, causing the phone 22.7% of the time to be idle and making the attack significantly less effective. In a real attack where the MMS gateway/server is more loaded, it is expected to

observe even longer delays. In fact, since MMS still uses SMS to push the notification [10], it is likely that the control channel is overloaded first.

## 5.4 Attack Scalability Analysis

An important question is how fast the attacker can scan a sufficient number of IPs. If it takes too long, the collected IP address may change (*e.g.*, users may power off the phone). With a simple analysis, we estimate that it takes 1.2 hours to scan enough IPs with 800kbps of bandwidth. The duration can be significantly shortened using more attack machines and bandwidth.

**Input.** The prefix ranges found to be allocated for a target location under a GGSN: X.128.X.X – X.147.X.X.(§4.1) are used as input. This corresponds to 1,300,500 IPs.

**Min-RTT probing.** The attacker can first use min-RTT measurements to filter irrelevant IPs as min-RTT has the smallest probing bandwidth requirement and the highest entropy. Although in total we need 30 – 40 probes (as described in §4.2.2) to determine the min-RTT, we need only 1 probe to first check if the IP is assigned or not. Probing one IP with one probe requires bandwidth of  $\frac{40\text{byte} \times 8\text{bit}}{RTT} = 1\text{kps}$  (assuming  $RTT = 0.3\text{s}$ ). Probing 800 IPs simultaneously requires  $1\text{kps} \times 800 = 800\text{kbps}$ . Probing each IP once takes about  $\frac{1,300,500}{800} \times 0.3\text{s} = 8\text{min}$ .

Since many IPs are not allocated at run time, they can be dropped easily upon the first probe. Considering the online IP ratio of 22% (§3.3), around 286,110 IPs are left. Probing the remaining IPs with additional 29 back-to-back measurements will take  $\frac{286,110}{800} \times (0.3\text{s} \times 29) = 51\text{min}$ . In total, it takes roughly 59 minutes to scan the entire GGSN IP range.

**Static feature probing.** After min-RTT probing, on average  $\frac{1}{8}$  of IPs (around 35,763) remain since the entropy of min-RTT is 3 bits. The attacker can then measure the static features to further narrow down IPs to probe. Given that the attacker knows the exact inactivity timer and the queue size threshold in the target location from the probe phone in that area, it can directly look for these values during scanning. Depending on the timer and queue size threshold, a complete round to measure both values takes a large packet (1200 bytes) to trigger state promotion to DCH, a small packet (40 bytes) to test the inactivity timer, and a packet with size matching the queue size threshold (suppose it is 500 bytes) to see if it forces the state to go back to DCH. The time to finish such a round would be the *promotion delay*(2s) + *inactivity timer*(5s) = 7s. The throughput requirement is then  $\frac{(1200+40+500) \times 8\text{bits}}{7} = 1.94\text{kbps}$ . Scanning 412 IPs simultaneously would require  $1.94\text{kbps} \times 412 = 800\text{kbps}$  bandwidth and  $\frac{35,763}{412} \times 7\text{s} =$

10min to finish.

**Promotion delay probing.** After the static feature probing, there are on average  $\frac{1}{5}$  IPs (around 7,152) remaining for entropy of 2.3 bits. Then the promotion delay can be measured. It takes only 10.7bps uplink on the server side, since it only needs to send and receive one 40-byte packet every 30 second. It takes 4min to complete 8 rounds of probing (8 rounds is selected in §4.2.2). The attacker can easily target the rest simultaneously using  $7,152 \times 10.7\text{bps} = 76\text{kbps}$  bandwidth. The estimated time to finish probing should be 4min since all of them can be done in parallel.

In total, the time to complete the measuring of all features is  $59 + 10 + 4 = 73\text{min}$ . Considering 80% of IPs keep their IPs for more than 4 hours (Figure 3), the attacker has enough time to launch the attack thereafter for at least several hours.

## 6 Discussion and Defense

### Impact of device difference on dynamic features.

Ideally, end-devices should have little impact on the dynamic features such as RTT and promotion delays. However, we do realize that different phones supporting different technologies may co-exist. For instance, some phones may support only HSDPA while their network supports both HSUPA and HSDPA (*i.e.*, HSPA). This means that these “older” phones may experience slower network performance (*e.g.*, longer RTT and promotion delay) thus exhibiting different network signatures. In practice, however, it is likely that there will always be one or two “dominating” technologies that are most widely used in the end-devices and can be considered as the target for an attacker. For instance, devices are frequently updated given that smartphones are becoming cheaper and quickly out-dated. Regardless, such phone differences do not impact the static features such as queue length threshold and inactivity timer.

### Impact of base-station difference on dynamic features.

Besides device differences, different base-stations can also impact the measured RTTs or promotion delays, with bounded noise. Indeed, from the MobileApp data, we find that in a short duration (*e.g.*, peak hours), the measured promotion delays do not vary significantly under the same RNC considering the samples are collected across multiple base-stations already. We do find one instance where the promotion delay can vary more significantly in different base-stations. This means that we either have to relax the SD-threshold to cover enough signature samples, or need some way to account for it. As a preliminary investigation, we find that the difference could be caused by different base-stations under different load, indicated by significantly differing average RTTs (min-RTTs are

still the same) which can be used for calibration. We leave for future work to systematically address this.

**Optional final step to narrow down RNCs.** In cases where the attacker wants to ensure that the inferred IPs indeed belong to the target RNC, an optional step that the attacker can take is to actually try these IPs out. More specifically, the attacker can choose a subset of the total IPs and launch a mini-attack against the RNCs (which should increase the processing delay on the RNC) and observe the changed baseline promotion delay from his probe phone. If the promotion delays of the tested IPs match the perturbed baseline promotion delay, it is likely that they indeed belong to the same RNC. The attacker can repeat this step multiple times to perform more fine-grained filtering.

**Next generation LTE networks.** As discussed earlier, LTE networks still have the built-in radio resource control mechanism with associated configurable parameters [5]. For instance, they allow various timers such as inactivity timer, discontinuous reception (DRX) cycle timer to be configured [5]. Fundamentally, due to the scarce nature of the radio resource, the corresponding radio resource control protocol has to provide mechanisms for sharing the resource smartly and flexibly among a large number of users. It is conceivable that network signatures exist for LTE networks as well since different network locations may again use a different set of configurations to accommodate the load difference.

**Defense.** The network signatures can be considered as a type of side channel that leaks information about the network location. A common defense is to eliminate such side channel. For instance, instead of using the fixed configuration parameter for all devices at all times, carriers can assign parameters probabilistically within a range, which makes it significantly harder for an attacker to infer the actual parameter. However, this may reduce the effectiveness of the parameter for accommodating workload and still requires scrutinization. As network defenses, the carrier can disallow direct probing from the Internet or other phones. However, it also has negative impact as discussed. As host-based defenses, one simple solution is to use host firewall that drops unsolicited probing packets to prevent information leakage. This could be an effective solution in the short term. As smartphones evolve, however, they may open more ports to accept incoming connections, making host firewall protection less effective. We leave for future work to systematically study how to defend against such information leakage in critical infrastructures such as cellular networks.

## 7 Related work

The wide adoption of cellular networks increasingly draws attention to study their vulnerabilities. Various forms of attacks have been revealed in the past few years. Serror *et al.* [32] found malicious traffic from the Internet can overload the paging channel in CDMA2000 networks. Traynor *et al.* [35] presented two attacks on GPRS/EDGE networks by abusing the radio resource setup and teardown process. Lee *et al.* [25] showed well-timed attack traffic can trigger excessive radio resource control messages, which in turn overburden RNCs in UMTS networks. The success of all these attacks hinges on the ability to identify many target devices in a particular cellular network region, *e.g.*, under a RNC (which is the focus of this paper).

Our work draws on earlier cellular network measurement studies. Balakrishnan *et al.* [15] found IP address alone cannot be used to determine the geographic location of mobile phone in AT&T's 3G network. Latency measurement appears to be more informative. However, their observations are drawn from latency samples of only three locations.

Prior work leveraged phone numbers to create a hit list for the purpose of launching targeted attacks. Enck *et al.* [19] generated a hit list of phone numbers through a combination of sources such as NPA/NXX database and web scraping. However, its effectiveness is not thoroughly evaluated. Moreover, sending SMS or MMS has a number of limitations compared to sending IP traffic directly to phones.

There have been extensive studies on geolocating IP addresses in the Internet. Most of them assume a list of landmarks with well-known geographic location, extracted from manually-maintained sources like DNS names [28, 33, 12]. They then use latency constraints to multiple landmarks to geolocate an IP address [22]. Unfortunately, these techniques are ill-suited for geolocation in cellular networks because a phone is often far from its first IP hop (*e.g.*, GGSN) not only in terms of geographic distance but also in terms of network latency. Given this, instead of attempting to infer the exact location of a phone, we solve a slightly different (but easier) problem, *i.e.*, determining if two phones are under the same RNC.

There are also studies for localizing mobile devices such as Escort [18], VOR [27], Virtual Compass [16], PlaceLab [17], and Radar [14]. The key difference is that they all require participation and cooperation from the phone, which is not assumed in the attack.

## 8 Conclusion

Motivated by the recent attacks against cellular network infrastructure at a particular target region (e.g., signaling attack), we focus on a simple but important question of how to create a hit-list of reachable mobile IP addresses associated with the target location to facilitate such targeted DoS attacks. Our approach uses network signatures through active probing and is applicable for a large set of current cellular networks we studied. This exposes a new attack vector that cannot be easily eliminated. From a thorough empirical analysis, we offer suggestions on how to defend against such reconnaissance effort by attackers.

## References

- [1] 3GPP TS 25.133: "Requirements for support of radio resource management".
- [2] 3GPP TS 25.331: "RRC Protocol Specification".
- [3] AT&T: 80 percent of network now upgraded to HSPA+. <http://www.engadget.com/2010/11/17/atandt-80-percent-of-network-now-upgraded-to-hspa/>.
- [4] AT&T: wireless capacity upgrades complete in Hillsborough, Pasco. <http://www.bizjournals.com/tampabay/stories/2010/10/11/daily38.html>.
- [5] Evolved universal terrestrial radio access (e-utra); radio resource control (rrc); protocol specification. <http://www.3gpp.org/ftp/Specs/html-info/36331.htm>.
- [6] Extended icmp to support multi-part messages. <http://tools.ietf.org/html/rfc4884>.
- [7] iPhone Troubles Might Go Beyond Signaling. [http://www.lightreading.com/document.asp?doc\\_id=190764&piddl\\_msgid=226225#msg\\_226225](http://www.lightreading.com/document.asp?doc_id=190764&piddl_msgid=226225#msg_226225).
- [8] Local Calling Guide. <http://www.localcallingguide.com/index.php>.
- [9] Mobility management. [http://en.wikipedia.org/wiki/Mobility\\_management](http://en.wikipedia.org/wiki/Mobility_management).
- [10] Multimedia Messaging Service. [http://en.wikipedia.org/wiki/Multimedia\\_Messaging\\_Service](http://en.wikipedia.org/wiki/Multimedia_Messaging_Service).
- [11] OpenCellID. <http://www.opencellid.org/>.
- [12] Whois ip address/domain name lookup. <http://whois.domaintools.com>.
- [13] UE "Fast Dormancy" behavior. 3GPP discussion and decision notes R2-075251, 2007.
- [14] P. Bahl and V. N. Padmanabhan. Radar: an in-building rf-based user location and tracking system. In *Proc. of INFOCOM*, 2000.
- [15] M. Balakrishnan, I. Mohamed, and V. Ramasubramanian. Where's That Phone?: Geolocating IP Addresses on 3G Networks. In *Proc. of IMC*, 2009.
- [16] N. Banerjee, S. Agarwal, P. Bahl, R. Chandra, A. Wolman, , and M. Corner. Virtual Compass: relative positioning to sense mobile social interactions. In *IEEE Pervasive Computing*, 2010.
- [17] Y. Chen, Y. Chawathe, A. LaMarca, and J. Krumm. Accuracy characterization for metropolitan-scale wi-fi localization. In *Proc. of Mobisys*, 2005.
- [18] I. Constandache, X. Bao, M. Azizyan, and R. R. Choudhury. Did You See Bob?: Human Localization using Mobile Phones. In *Proc. of MobiCom*, 2010.
- [19] W. Enck, P. Traynor, P. McDaniel, and T. La Porta. Exploiting open functionality in SMS-capable cellular networks. In *Proc. of CCS*, 2005.
- [20] H. Galeana, R. Ferrus, and J. Olmos. Transport capacity estimations for over-provisioned utran ip-based networks. In *WCNC*, 2007.
- [21] M. Gruteser, D. Grunwalddepartment, and C. Science. Anonymous usage of location-based services through spatial and temporal cloaking. In *Proc. of Mobisys*, 2003.
- [22] B. Gueye, A. Ziviani, M. Crovella, and S. Fdida. Constraint-based Geolocation of Internet Hosts. In *Proc. of IMC*, 2004.
- [23] H. Holma and A. Toskala. HSDPA/HSUPA for UMTS: High Speed Radio Access for Mobile Communications. John Wiley and Sons, 2006.
- [24] H. Kaaranen, A. Ahtiainen, L. Laitinen, S. Naghian, and V. Niemi. UMTS Networks: Architecture, Mobility and Services (2E). John Wiley & Sons, 2005.
- [25] P. P. C. Lee, T. Bu, and T. Woo. On the Detection of Signaling DoS Attacks on 3G Wireless Networks. In *Proc. of INFOCOM*, 2007.
- [26] O. D. Mey, L. Schumacher, and X. Dubois. Optimum Number of RLC Retransmissions for Best TCP Performance in UTRAN. In *Proc. of Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, 2005.
- [27] D. Niculescu and B. Nath. VOR base stations for indoor 802.11 positioning. In *Proc. of MobiCom*, 2004.
- [28] V. Padmanabhan and L. Subramanian. An Investigation of Geographic Mapping Techniques for Internet Hosts. In *Proc. of SIGCOMM*, 2001.
- [29] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. Characterizing radio resource allocation for 3G networks. In *Proc. of IMC*, 2010.
- [30] F. Qian, Z. Wang, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. TOP: Tail Optimization Protocol for Cellular Radio Resource Allocation. In *Proc. International Conference on Network Protocols*, 2010.
- [31] Z. Qian, Z. M. Mao, Y. Xie, and F. Yu. Investigation of triangular spamming: a stealthy and efficient spamming technique. In *In Proc. of IEEE Security and Privacy*, 2010.
- [32] J. Serror, H. Zang, and J. C. Bolot. Impact of Paging Channel Overloads or Attacks on a Cellular Network. In *WiSe*, 2006.
- [33] N. Spring, R. Mahajan, and D. Wetherall. Measuring ISP Topologies with Rocketfuel. In *Proc. of SIGCOMM*, 2002.

- [34] P. Traynor, M. Lin, M. Ongtang, V. Rao, T. Jaeger, P. McDaniel, and T. La Porta. On Cellular Botnets: Measuring the Impact of Malicious Devices on a Cellular Network Core. In *Proc. of CCS*, 2009.
- [35] P. Traynor, P. McDaniel, and T. La Porta. On Attack Causality in Internet-connected Cellular Networks. In *Proc. of USENIX Security*, 2007.
- [36] Q. Xu, J. Huang, Z. Wang, F. Qian, A. Gerber, and Z. M. Mao. Cellular data network infrastructure characterization and implication on mobile content placement. In *Proc. of SIGMETRICS*, 2011.