Mitigating DoS Attacks on the Paging Channel by Efficient Encoding in Page Messages

Liang Cai[1], Gabriel Maganis[1], Hui Zang[2], and Hao Chen[1]

[1] Computer Science Department, University of California, Davis
{lngcai,gymaganis}@ucdavis.edu,
hchen@cs.ucdavis.edu,
[2] Sprint Advanced Technology Labs
hui.zang@sprint.com

Abstract. Paging is an important mechanism for network bandwidth efficiency and mobile terminal battery life. It has been widely adopted by mobile networks, such as cellular networks, WiMax, and Mobile IP. Due to certain mechanisms for achieving paging efficiency and the convergence of wireless voice and data networks, the paging channel is vulnerable to inexpensive DoS attacks. To mitigate these attacks, we propose to leverage the knowledge of the user population size, the slotted nature of the paging operation, and the quick paging mechanism to reduce the length of terminal identifiers. In the case of a CDMA2000 system, we can reduce each identifier from 34 bits down to 7 bits, effectively doubling the paging channel capacity. Moreover, our scheme incurs no paging latency, missed pages, or false pages. Using a simulator and data collected from a commercial cellular network, we demonstrate that our scheme doubles the cost for DoS attackers.

Key words: Paging, DoS Attacks, General Page Message, Quick Paging

1 Introduction

The biggest advantage of mobile networks over wired networks is mobility, which allows users to access the network from different locations. Mobile networks achieve mobility through *Macro-mobility management* and *Micro-mobility management*. The former ensures that mobile terminals (e.g., cell phones and laptops with wireless cards) are addressable when they roam between different domains, while the latter manages the mobile terminal's movement between access points or base stations within the same domain. While implementations vary across mobile networks, macro-mobility management always requires roaming users to notify the network each time they arrive at a new domain. By contrast, a similar scheme, which would require terminals to update their locations every time they move to a new access point or base station, is impractical for micro-mobility management for several reasons. First, location updates would be much more frequent than in macro-mobility management, which would consume significant

wireless bandwidth and mobile terminal power. Second, to trace their own locations, terminals must continuously monitor the beacon or pilot channel of base stations, which would drain their batteries even faster.

Paging is a critical mechanism to improve the bandwidth and terminal energy efficiency in micro-mobility management. The designer divides the network into *paging areas* and requires terminals to notify the network about their location only when they enter a new paging area. Each paging area is usually large enough so that location updates are infrequent even for highly mobile terminals. Meanwhile, terminals monitor the network at longer intervals and enter the idle mode in between. When an incoming call arrives, the network controller broadcasts a page message to the entire paging area. If the terminal is located in the paging area, it responds to acquire a traffic channel. As an efficient location management scheme, paging has been widely adopted in mobile networks, including cellular communication systems (GSM[1], W-CDMA and CDMA2000[2]), WiMax[3] and Mobile IP systems[4].

There are a pair of low-bandwidth channels in a cellular network used for location management. The downlink channel, often referred to as the *paging channel*, is used for paging while the uplink channel (the *access channel*) is used for location updates. To lower the bandwidth requirement on the access channel, we desire larger paging areas; however, larger paging areas would increase the load on the paging channel since all the users in the same paging area share the same paging channel. Concentrated flash crowds could even lead to temporary paging channel overload or saturation. This tradeoff between bandwidth requirement and paging area size is known as the *paging efficiency problem*.

The recent convergence of wireless voice and data networks exacerbates this problem. Besides incoming voice calls, incoming short messages (SMSs) and data packets may also increase the load on the paging channel. This provides attackers with an opportunity to launch a DoS attack on wireless networks from the Internet, possibly with very low cost. For example, Serror et al. showed how to saturate the paging channel of a cellular network by sending data packets from the Internet at a very low cost [5], and Enck et al. showed how to disrupt a cellular network in a major city by sending SMS messages of a sufficient rate [6].

If we improve paging efficiency, we could not only accommodate flash crowds but also mitigate DoS attacks. Previous approaches focused on reducing the number of paging requests (e.g., by predicting terminals' locations [7]). In this paper, we take a different approach by increasing the number of paging requests that the paging channel can carry. A page message contains the identifiers of all the paged mobile terminals. Our key insight is that the shorter the lengths of these identifiers are, the more terminals a single page message can page. Towards this goal, we propose a series of methods for shortening terminal IDs by leveraging the knowledge of the population size in a paging area, by grouping terminals based on paging channel slots, and by using special Bloom filters in quick paging. When applying these methods to a CDMA2000 system, we are able to reduce the length of each terminal ID from 34 bits to 7 bits, which we shall show doubles the number of terminals that one page message can contain.

Since our scheme only shortens terminal IDs, it has no adverse effect on paging performance (e.g., paging latency, missed page rate, false paging) and requires little change to the paging protocols.

The rest of the paper is organized as follows: We describe the paging channel operation and page message format in Section 2, and show the importance of improving paging efficiency. We present the optimization schemes for reducing the terminal ID length in Section 3. We then evaluate the scheme using an experiment on a real cellular system and a simulation tool and illustrate the results in Section 4. After comparing our scheme with several related works in Section 5, we conclude in Section 6.

2 Paging Channel Operation

In this section, we first describe the paging channel operation and the page message format in the context of a cellular network using CDMA2000 technology. To show that our scheme is not limited to cellular systems, we explore the page operations in other mobile systems and discuss their differences from cellular systems.

2.1 Paging Channel Operation

A mobile network needs to track the location of the mobile terminals so that it can deliver data and voice calls to their intended recipients. A simple solution would be to require mobile terminals to report their locations through *location updates* whenever their locations change. However, since mobile devices are typically resource (e.g., battery power) constrained, requiring them to remain in the "active state" just to report their locations would be inefficient. Thus, mobile operators typically divide their networks into *location areas*, and mobile terminals report their locations only when they enter a new location area. When a new call or data packet arrives, the network *pages* the recipient mobile terminal in the location area. Therefore, *paging* and location updates are key components in mobility management in a mobile network.

CDMA2000 networks have a dedicated channel, the *paging channel*, that delivers page messages to mobile terminals [1]. Mobile terminals monitor this channel through a Time Division Multiple Access (TDMA) scheme. The network divides a paging cycle (either 2.56 or 5.12 seconds) into *slots* (either 32 or 64 slots, respectively). Thus, a mobile terminal stays in the *idle mode*, when the power consumption is minimal, most of the time and wakes up only during its assigned slot (whose duration is 80 ms) to determine whether it has been paged. The network assigns a mobile terminal to a slot based on the terminal's International Mobile Station Identifier (IMSI). Figure 1 illustrates the structure of 32-slot paging channel.

[1] A CDMA system can be configured to have at most seven paging channels

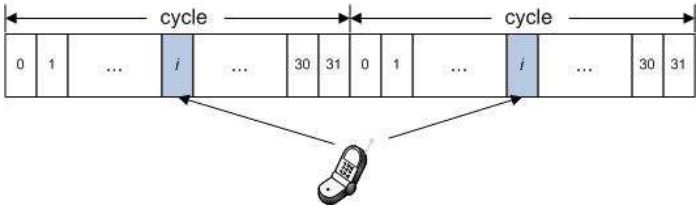


Fig. 1. An example structure of a paging channel [5]. A mobile terminal is in idle mode except during the i^{th} slot, when it wakes up and monitors the paging channel.

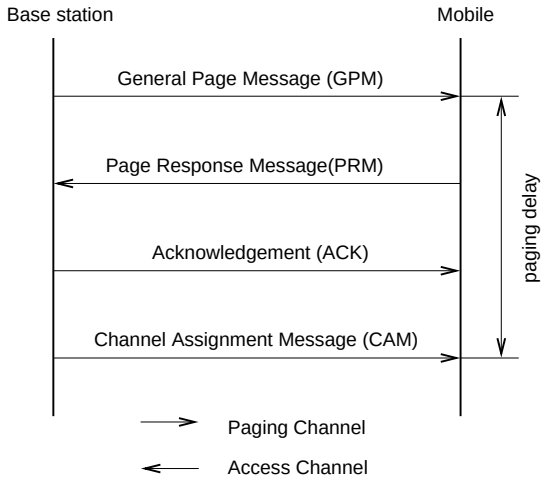


Fig. 2. Messages exchanged between a base station and a mobile terminal during a paging operation.

When an incoming call (or downlink data packet) arrives, the mobile switching center (MSC) broadcasts a General Page Message (GPM) to the location area (also known as the *paging area*) during the recipient mobile terminal's assigned slot. When the mobile terminal finds that it has been paged, it will respond through the associated base station with a Page Response Message (PRM). Then, the MSC sends an acknowledgment message (ACK) and a Channel Assignment Message (CAM) to the mobile terminal. The mobile terminal responds to the CAM with an ACK to establish a connection with the base station over the assigned *traffic channel*. Both the ACKs and the CAMs are sometime referred as *non-slotted* messages, since they can be sent during any slot (After sending the PRM, the terminal leaves the slotted mode and monitors all the slots). On the other hand, GPMs are *slotted* messages, which must be sent during the recipient mobile terminal's assigned slot. Figure 2 shows the messages exchanged between the base station and a mobile terminal during the paging process when an incoming call arrives. After this step, the terminal communicates with the base station on the assigned traffic channel only.

The number of slotted messages in the paging channel greatly exceeds that of non-slotted messages. When the MSC sends a slotted message, it does not know the location of the recipient, so it has to broadcast the page message to all the base stations in the paging area. By contrast, it needs to send the subsequent non-slotted messages to only the base station as determined from the PRM. Besides slotted messages and non-slotted messages, the paging channel is also used for transmitting system parameters, which are sometimes regarded as overhead traffic in the paging channel. These messages occupy the paging channel based on the paging load. The more GPMs there are, the fewer overhead messages will be sent. Typically, overhead traffic takes at least 25% of the capacity.

2.2 Paging Message Format

In the CDMA2000 standard, the specification for the GPM format is highly flexible, so GPMs in different systems may vary in length and pattern. Figure 3 shows an actual GPM that we captured from a mobile device in a live commercial CDMA2000 network. We can see that a GPM is composed of a header, several page records, padding, and the CRC.

Page records comprise the main body of a page message. Among all the fields in a page record, the terminal identifier field plays a critical role in the paging operation. This field may use many identifier types, indicated by the *PAGE_CLASS* and *PAGE_SUBCLASS* fields, listed in the CDMA2000 specification [8]. In our example, the terminal's IMSI_S (a 10-digit number derived from the terminal's International Mobile Subscriber Identity (IMSI)) is used as the identifier. In a GPM, the IMSI_S is encoded into a 34-bit string. Other optional identifier types have similar length (e.g., a TMSI is 32 bits) or longer.

Another important field in page records is the *SERVICE_OPTION*. It informs the terminal of the type of the incoming call. This is important since the message after the GPM could be of different service types. For example, the GPM is always followed by Channel Assignment Message(CAM) in the case of a voice call, but the CAM can be replaced by a Data Burst Message (DBM) in the case of text service.

2.3 Paging Operation in Other Mobile Networks

IEEE802.16 systems such as WiMax [9] or Hpi [10] do not use separated slotted channels in their paging operations. When a WiMax terminal is not engaged in communication, it enters the *idle mode*, which works in four stages: *idle mode initialization*, *idle mode entry*, *idle mode operation* and *idle mode exit*. Idle mode can be initiated either by the mobile terminal or the base station. When the mobile terminal initiates the idle mode, it sends a deregistration request message (DREG-REQ); when the base station initiates the idle mode, it sends a DREG-CMD message to the terminal, which responds with a DREG-REQ message. The base station then notifies the paging controller of the terminal's service information. The paging controller decides the PAGING_CYCLE, PAGING_OFFSET,

Field	value	length(bit)
General Paging Message		
Message Header		
MSG_LENGTH	0000xxxx	8
MSG_ID	010001	6
CONFIG_MSG_SEQ	000011	6
ACC_MSG_SEQ	011101	6
CLASS_0_DONE	1	1
CLASS_1_DONE	1	1
TMSI_DONE	1	1
ORDERED_TMSIS	0	1
BROADCAST_DONE	1	1
RESERVED	0000	4
ADD_LENGTH	000	3
Mobile Station 1		
PAGE_CLASS	00	2
PAGE_SUBCLASS	00	2
MSG_SEQ	100	3
IMSI_S	(xxx) xxx-xxxx	34
SDU_INCLUDED	1	1
SERVICE_OPTION	xx	16
Mobile Station 2		
...		
Message end Padding		
PDU_PADDING	0000	4
CRC		30

Fig. 3. The format of an actual General Paging Message. A 34-bit IMSI_S is used as the terminal identifier, and the length of each page record is 58 bits, while the header and the tail are 38 and 30 bits long, respectively.

and *Paging Listen Interval* (PLI) parameters and sends them to the terminal in a DREG-CMD message via the base station. Then, the terminal wakes up during the Paging Listen Interval periodically to check for a MOB-PAG-ADV message (the GPM's counterpart in a WiMax system). The message consists of a 48-bit MAC header, several page group IDs, several page records, and padding. The length of each page record is 32 bits. A paged terminal is identified with a hash value of its 24-bit MAC address.

Mobile IP systems also propose a paging operation [11, 12, 13, 4, 14]. In most of these schemes, a terminal's home IP address is used as its identifier in the page message, so the identifier is 32 bits in IPv4 systems and 128 bits in IPv6 systems.

2.4 Paging Channel Overload Problem

Recall that the size of a paging area determines how often mobile terminals send location updates. The smaller a paging area is (i.e., containing fewer cells), the more frequently a terminal with high mobility needs to send location updates,

which consumes more power and generates more traffic on the access (uplink signaling) channel, which is also a low-bandwidth channel like the paging channel. To avoid this adverse effect, in current cellular networks, a paging area usually consists of hundreds of cells.

Equation 1 calculates the maximum number of terminals that can be paged in each slot per paging area, where we assume that the bandwidth of the paging channel is 9600bps, the duration of a paging slot is 0.08s, the overhead traffic occupies 25% of the channel capacity.[2], the length of the page message header is 38, the length of the CRC value is 30, and the length of each page record is 58.

$$N_{max} = \lfloor \frac{9600 \times 0.08 \times (1 - 0.25) - 38 - 30}{58} \rfloor = 8 \quad (1)$$

Equation 1 shows that the call arrival rate to a paging area is limited to 100 per second (8 calls / 0.08 second). Given the size of a typical paging area, this maximum call arrival rate is acceptable when only voice calls are paged. However, when the network provides more and more text and data services, this upper bound makes the paging operation an essential bottleneck.

Worse yet, the paging channel has become an ideal target of Denial of Service (DoS) attacks on the cellular network. [5] described such an attack by flooding the network with UDP packets. When a mobile terminal establishes a wireless data connection with the network, it acquires an IP address. The network reserves the address for the terminal until the terminal disconnects from the network, even when it is in the idle mode. The DoS vulnerability lies within the fact that when a data packet arrives at the mobile network, the recipient mobile terminal needs paged. Since it is relatively easy to find the IP subnets assigned to a mobile service provider, an attacker on the Internet can flood these IPs with UDP packets to trigger a flood of page messages within the mobile network. The authors explored the feasibility of this attack by conducting experiments on a live commercial CDMA2000 network. Due to legal and ethical constraints, the goal of the experiments was only to increase the paging channel load by 10%. The authors predicated that the performance of the network would degrade further if they had increased the attack load or if the attacks had been carried out in a busy area.

3 Efficient Encoding in Page Records

To mitigate paging channel overload, we wish that a page record can carry more terminal IDs. However, the length of a page record is determined by its slot duration and the paging channel bandwidth, both of which are constrained by system configurations and physical limitations. Instead, we investigate how to fit more terminal IDs into existing page records.

[2] 25% is a common overhead load. When the overhead traffic load is less than 25%, we occasionally observe GPMs with 9 records in real paging data.

The CDMA2000 specification, for example, supports different types of terminal IDs [2], but most of them are longer than 30 bits. IMSI_S, one of the most commonly used terminal ID, is 34 bits, and a TMSI is 32 bits. Typically, terminal IDs account for more than half of a page record's size. Therefore, they are a good target for optimization. Moreover, terminals IDs are universal in all paging systems, while other fields in page records are system specific.

For convenience, we describe our scheme for efficiently encoding terminals IDs in the context of a CDMA2000 system, although the principle applies to other mobile networks, such as WiMax and Mobile IP. Using a series of techniques, we are able to reduce terminal IDs from 34 bits down to only 7 bits, as described in detail below.

3.1 Approaches

Optimization using Knowledge about Population Size in a Paging Area One reason why the IMSI_S is long is that it is globally unique. However, the paging operation only needs to differentiate between terminals in the same paging area. Therefore, as the first step, we replace the globally unique IMSI_S with a locally unique identifier. As we observed from a commercial CDMA2000 system, the number of terminals in a single paging area, including the most populated areas such as Manhattan, does not exceeded one million. This indicates that 20 bits suffice for locally unique IDs.

Optimization using the Slotted Nature of the Paging Channel Section 2.4 showed that a cellular network divides the paging channel into 32 or 64 slots. Each terminal wakes up in only one slot (calculated based on its IMSI) in the paging cycle to listen to the page message. In other words, terminals in a paging area are divided into distinctive *slot groups* by their slot numbers. Since a terminal only listens to one slot, their local IDs need to be unique only within each slot group. A typical CDMA 2000 system has 64 slots. If all the terminals in a paging area are evenly divided into slot groups, no slot group should contain more than $2^{20}/64 = 2^{14}$ terminals. Therefore, we can reduce the length of local IDs further to 14 bits.

Optimization using the Quick Paging Mechanism Finally, we decrease the length of the local IDs even further by using the Quick Paging channel. Quick Paging is a standardized operation adopted by most mobile networks to reduce terminals' wakeup time to improve their power efficiency. Similar to the Paging Channel operation, the Quick Paging channel is also divided into slots. In fact, a terminal's quick paging channel slot occurs exactly 100ms earlier than its paging channel slot. The purpose of Quick Paging is to convey *"paging indicator bits"* to help terminals pre-determine whether they are paged. Towards this goal, each quick paging slot is divided into four frames, and each frame carries a sequence of indicator bits. Each terminal has two indicator bits. The system calculates the positions of these two bits in the quick paging frames by feeding the terminal's IMSI into two hash functions. The standard requires that these two indicator

bits occur in either the first and third frames, or the second and fourth frames (so that a terminal needs to wake up in only half of the frames). If a terminal detects that either one of its indicator bits is not set, it is not paged and therefore will stay idle in the coming paging slot; otherwise, it might be paged, so it will wake up in the coming paging slot. Quick paging increases the wakeup duration of the paged terminals by half, but decreases the wakeup duration of unpaged terminals by at least half (because the terminal only wakes up in two of the four frames of the quick paging slot). Since typically only a small fraction of terminals are paged, quick paging reduces the overall wakeup time of all terminals.

The Quick Paging operation uses a special Bloom filter. Due to the false positives inherent in Bloom filters, quick paging cannot replace the paging operation. However, we can take advantage of quick paging to reduce the length of local IDs further. Since quick paging instructs only a very small fraction of terminals to wake up and listen to their paging slots, the local IDs need to differentiate only between the terminals that are truly paged and those that are not paged but whose paging indicator bits are set due to the inaccuracy of the Bloom filter.

As mentioned earlier, the first indicator bit of a terminal must be in either the first or second frame. In our reference CDMA system, the quick paging channel operates at full speed (9600 bps) and each frame is 20ms, so there are $9600 \times 0.02 \times 2 = 384$ bits in the first two frames. The CDMA2000 specification uses several bits in these frames as broadcast bits so the total number of bits used as paging indicators in the first two frames is 368. Since we only need to differentiate between the terminals whose first indicator bits are at the same location in the first two quick paging frames, we can reduce the local ID space further. Assuming that the locations of the first indicator bits of all terminals are evenly distributed, we can reduce the local ID space by $368 \approx 2^8$. As a result, we would need only $14 - 8 = 6$ bits to represent each local ID. We discuss our scheme below.

If no first indicator bits of the paged terminals share the same location, we can order the local IDs in the page record by the order of their corresponding first indicator bits in the quick page frames. For example, if a terminal's first indicator bit is the i_{th} set bit in the quick page frames, the terminal will check the i_{th} local ID in the page record (to see if it is really paged or if its first indicator bits are set merely due to Bloom filter inaccuracy).[3]

However, the above solution would not work when multiple terminals are paged but their first indicator bits share the same location in the quick paging frames. To solve this problem, in the page record, we group terminals by the

[3] A subtle complication occurs when the first indicator bit of a mobile terminal is in the second quick page frame. In this case, since the mobile terminal does not listen to the first quick page frame, it does not know how many bits are set there, so it does not know the position of its page record in the page frame. We can solve this problem by a simply trick: rather than calculating its position from the beginning of the frame, the above terminal should calculate its position from the end of the frame. For example, if a mobile terminal's first indicator bit is the i_{th} set bit from the end of the second quick page frame, it should check the i_{th} page record from the end of the page frame for its local ID.

locations of their first indicator bits, and prepend a *group bit* to each local ID. We set the group bit of the first terminal in a group to 1, and the group bits of all the other terminals in the same group to 0. For example, in Figure 4 the first indicator bits of both Terminal 1 and 4 are at the same position in the first quick paging frame. Therefore, in the page record, the group bit of Terminal 1 is 1 since it is the first terminal in this group, and the group bit of Terminal 4 is 0 since it is not the first terminal in this group.

We summarize the paging operation from a terminal's perspective. When a terminal joins a paging area, the network assigns a 6-bit ID to the terminal. The terminal then calculates the position of its slot in the page message and the positions of its first and second indicator bits in the quick page message. In each paging cycle, the terminal wakes up to listen to two of the four frames in its slot in the quick page message. If both the first and second indicator bits are set, the terminal wakes up to listen to its paging channel slot to receive the page record. Using the method described in Section 3.1, the terminal compares its local ID with the corresponding one in the page record. If they match, the terminal is paged.

3.2 Bandwidth Gain

For our reference CDMA2000 system, our scheme reduces the length of local IDs from 34 bits down to 7 bits, and the length of each pag/doubing records from 58 bits to 31 bits. After applying our scheme, the maximum number of page records per slot increases from 8 to 16(Figure 5).

3.3 Implementation Requirements

Implementing our scheme is straightforward. It requires only the following modifications to the existing paging operation.

- **Local ID management by Paging Controller** The paging controller (PC) maintains all the local IDs of terminals in the paging area. When a terminal arrives, the PC searches for an unused local ID and assigns it to the terminal. When an incoming call for the terminal arrives, the PC constructs a GPM using the local ID. When the terminal leaves, the PC reclaims the local ID. Given the high computational power of the paging controller, such management overhead is negligible.
- **Local ID transfer** Our scheme requires that the paging controller sends the local ID to the terminal. The controller can do this during user registration. Since the local ID is only several bits, the overhead is negligible.
  To determine the length of local IDs, the paging controller must estimate the maximum number of terminals in the same paging area. If the controller finds this estimate insufficient, it may increase the length of local IDs and broadcast the new length to all the terminals in a configuration message.

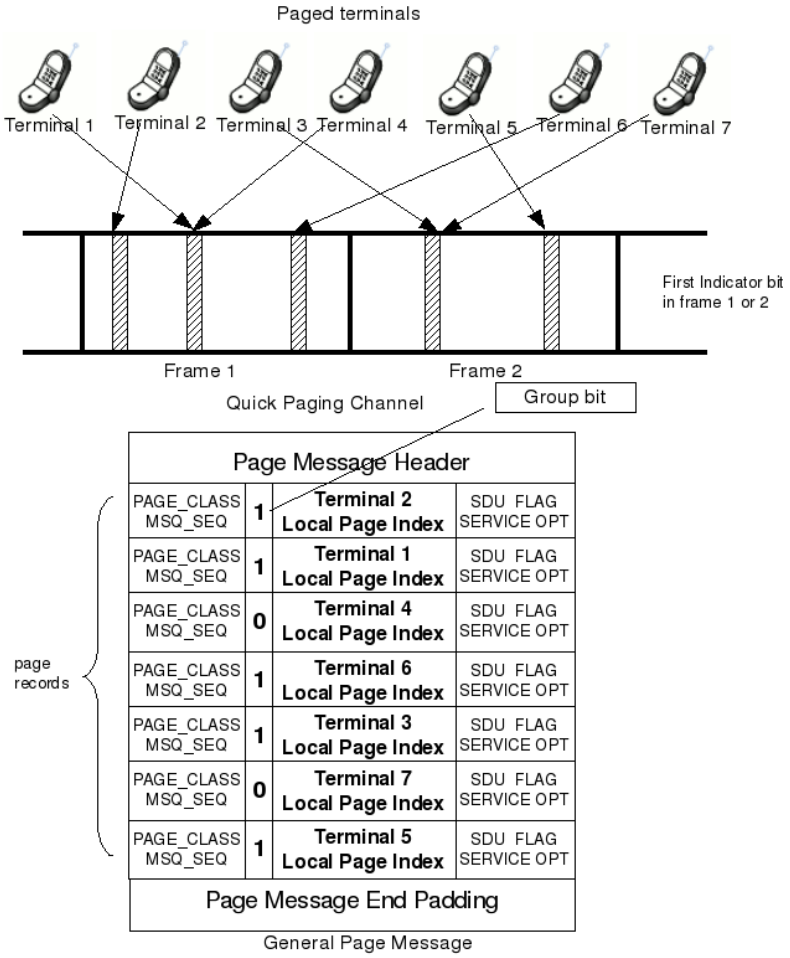


Fig. 4. An example of an optimized GPM based on quick paging. Seven paged terminals are ordered by the position of their corresponding first indicator bits. Those with the same first indicator bits are grouped by a group bit.

	Without our scheme	With our scheme
Terminal identifier length	34	7
Page record length	58	31
Maximum page records per each slot	**8**	**16**

Fig. 5. Our scheme doubles the maximum page records per slot.

– **Terminal modification** Our scheme requires slight modification to the paging module in terminals. Note that our scheme does not change the protocol messages; rather it merely changes the algorithm by which a terminal searches for its local ID in the page record.

3.4 Advantages

Simplicity . Our scheme does not cause any adverse effects, such as paging latency, false paging, and missed paging, that other schemes often suffer from. Our scheme is compatible with and complementary to many other schemes, such as the ones based on location prediction [7].

Versatility . Besides cellular networks, our scheme applies to many other mobile networks. WiMax, for example, can also benefit from this page message optimization, although it uses a very different paging operation. Instead of using slots to determine terminals' wakeup time, the base station and the mobile station in WiMax negotiate the numerical values of the *PAGING_CYCLE*, *PAGING_OFFSET* and *PAGING_LISTEN_INTERVAL* parameters through the *DREG_CMD* and *DREG_REQ* message pair. Such mechanism invalidates our scheme where it groups terminals by their wake up slots. However, [9] proposes to group WiMax terminals by aligning their *PAGING_OFFSET* so that their page messages can be merged into one message. The essence of the scheme is to borrow the concept of slots from cellular networks. This proposal makes our idea of using a shorter identifier local to each slot feasible again. Furthermore, [15] and [16] propose a quick paging channel for IEEE802.16, making our entire scheme applicable to WiMax systems. In WiMax's page message, *MOB-PAG-ADV*, the mobile identifier, is a 24-bit hash value of the MAC address, so false paging is inevitable. Our scheme, by contrast, can eliminate the unnecessary false paging in WiMax.

4 Evaluation

We evaluate the effectiveness of our scheme on mitigating DoS attacks and on increasing the capacity of the paging channel. Since modifying a commercial cellular system and launching a full-fledged DoS attack are prohibited, we demonstrate the performance of our scheme using real paging data collected from a live cellular network as well as using a simulation tool.

4.1 Evaluation based on partial DoS attack on live cellular network

One advantage of our scheme is that it does not change existing paging protocols, as our scheme merely changes the terminal IDs inside the GPM. Therefore, we can use paging data measured on a real paging system to infer the performance of our scheme (such as its impact on reducing channel utilization) with one exception: During high paging load, the paging controller without applying our scheme may not be able to page all the requested terminals in a slot, so it will page some of these terminals in the next paging cycle instead. Since our scheme allows the paging controller to fit more terminal IDs into one page message, it will eliminate some or all of these delays. In this case, the terminals paged

in each paging cycle would be different if the paging system had adopted our scheme.

Based on the above observation, we recreated the partial DoS attack experiment described in [5]. We captured GPMs over an CDMA2000 interface. We then launched a partial DoS attack by injecting UDP packets from the Internet to data users of the cellular network. Using the captured GPMs, we calculated the utilization of the paging channel by GPMs. To infer the channel load when our scheme is applied, we only need to calculate the length of the GPMs under our scheme, if there were no or negligible paging delays indicated by our captured GPMs. To verify this assumption, we examined the captured GPMs and found only three GPMs (out of more than 20,000 GPMs) that contained the maximum number of paging records (which indicates *potential* paging delays). This validates our assumption that paging delays occurred rarely in the captured GPMs.

Figure 6 depicts the utilization of paging channel by GPM under three different situations. For legibility, we have smoothed the curves using the Exponential Moving Weighted Average (EMWA) algorithm. Before the attack, the average utilization by GPMs in the measured system was 18.1%[4]. The utilization went up to 23.2% during the attack. If the system deployed our scheme, the average utilization would be 14.2% before the attack (not shown in the figure for legibility), and 16.8% after the attack.

As another measurement of the effectiveness of our scheme, we quantified the resources that an attacker must acquire to saturate the paging channel. Since overhead messages occupy at least 25% of the paging channel capacity, an attacker only needs to saturate the remaining 75% of the paging channel. We calculated how many page records the attacker must trigger. We assume that all the messages other than GPMs remain unchanged after the attack begins. Figure 7 shows that our scheme almost doubles the efforts of the attacker to completely saturate the paging channel.

4.2 Simulating a paging system

In Section 4.1, we examined the effect of our scheme using page messages measured on a live cellular network. In this section, we use simulation to study our scheme under different conditions of the paging system. We simulate the paging channel *at a base station* as a queueing system. There are two main types of messages in a paging system, slotted messages and non-slotted messages. Slotted messages need to be sent during their assigned slots in the paging channel while non-slotted messages can be sent at any time. Non-slotted messages arrive only after the paged terminal moves into the cell (by contrast, a page message is used to locate a terminal in the paging area and hence is not necessarily associated

[4] The paging channel utilization is calculated as the total bits of GPM during a certain time period, divided by the product of the length of the time period and the channel capacity (e.g., 9600bps).

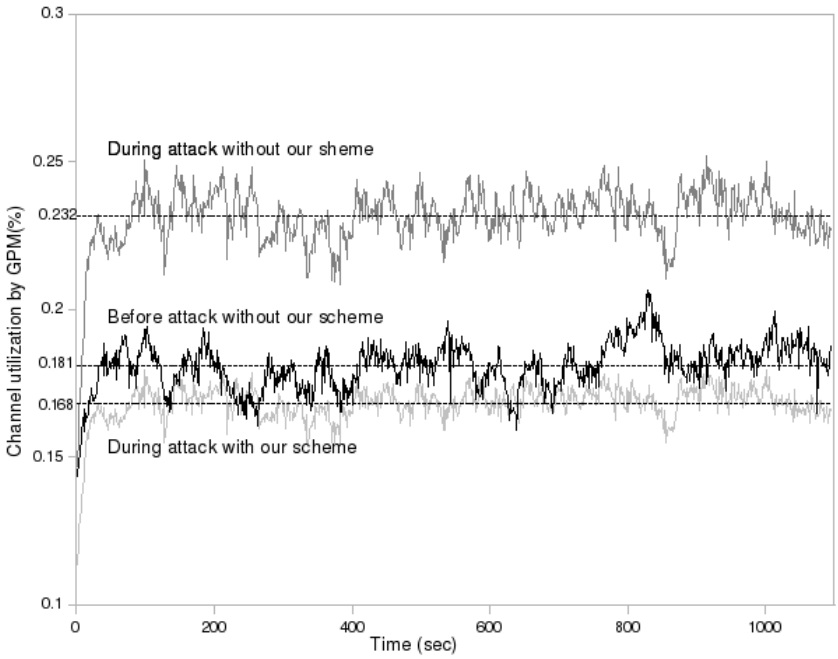


Fig. 6. Paging channel utilization during the attack without our scheme (top), before the attack without our scheme (middle), and during attack with our scheme (bottom).

with *this* base station). Therefore, the arrival process of the non-slotted messages is equivalent to the service process of the slotted messages multiplied by a factor of p, where p, the paging success factor, is the probability that a mobile terminal is located with a given base station and is inversely proportional to the size of the paging area. We assume that there is no delay between the time when a slotted message is served and the time it triggers a non-slotted message. We also assume that slotted messages initially arrive according to a Poisson process. This is a common assumption for modelling the arrival of events such as calls in phone systems. Figure 8 illustrates this queueing system.

We simulate such a paging system with a paging cycle divided into 64 slots. Hence we have 64 *slotted* queues and 1 *non-slotted* queue, as shown in Figure 8. The simulation program has three main modules, namely, the arrival, slot, and server modules, which we describe in detail below.

Arrival. The arrival module generates slotted messages according to a Poisson process. Then it randomly assigns them to one of the 64 slotted queues.

Slot. The slot module implements the schedule in which the slotted queues are served. Each slotted queue is served in a time division multiplexing manner. Specifically, the slot module calls the server module on each slotted queue in a

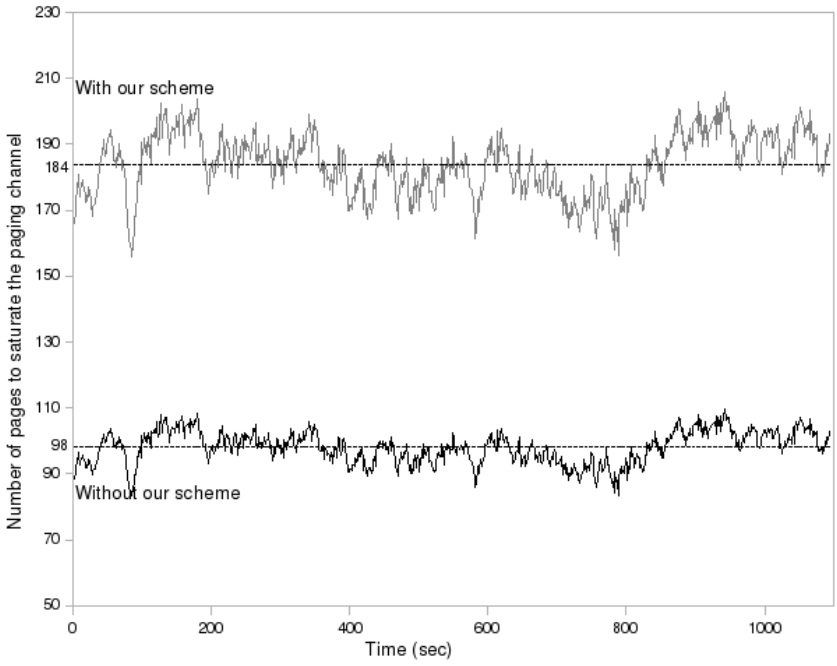


Fig. 7. Number of page records required to completely saturate the channel with and without our scheme

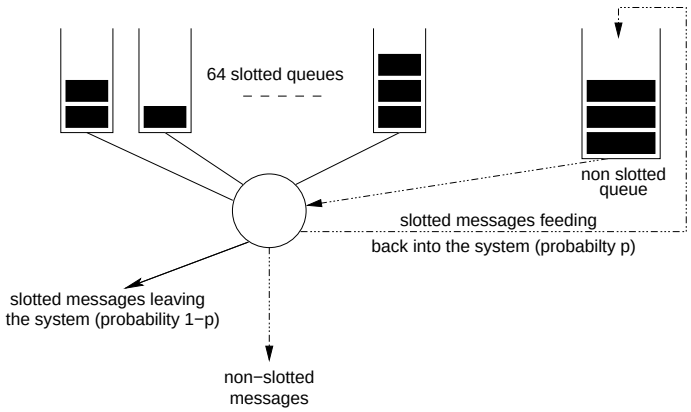
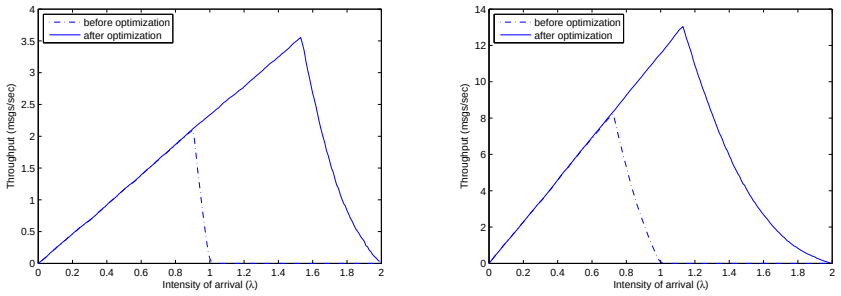


Fig. 8. A queueing system representation of a paging system. There are 64 slotted queues and 1 non-slotted queue. Slotted messages initially arrive according to a Poisson process. For each slotted message, the system generates a non-slotted message with probability p.



(a) $p = 0.01$ (large paging area) (b) $p = 0.05$ (small paging area)

Fig. 9. Throughput of non-slotted messages under different arrival rates λ. A small value of p (e.g., 0.01) represents a large paging area while a large value of p (e.g., 0.05) represents a small paging area.

round-robin schedule. The slot module allows the service of each slotted queue for a fixed duration of $D = 0.08$ seconds (i.e., the slot duration).

Server. The server module dequeues messages (i.e., sending messages) as well as generates non-slotted messages. When invoked by the slot module, it builds a GPM by dequeuing messages up to the maximum capacity (N_{max}). In more detail, it computes the total capacity of the paging channel, subtracts the length of the GPM header, and subtracts up to ($N_{max} \times r_s$) bits where r_s is the length of each page record. Before applying our scheme, N_{max} is only 8 and r_s is 58 bits. After our scheme is applied, N_{max} becomes 16 and r_s becomes 31 bits. For each new page record, the module generates, with a probability p, a subsequent non-slotted message and inserts it into the non-slotted queue.

If the slotted queue has spare bandwidth (i.e., the slotted queue contains fewer than N_{max} messages), the non-slotted queue is serviced for the remainder of the slot duration. To do this, the simulator builds a non-slotted message by subtracting the length of a non-slotted message header followed by the length of a non-slotted message record multiplied by as many non-slotted messages as can be dequeued (sent) during the remainder of the slot duration. If there are insufficient messages in the non-slotted queue, the server sits idle during the rest of the slot duration.

A sent non-slotted message indicates a successfully established call so we use the number of serviced non-slotted messages to calculate the system throughput. We plot the throughput of non-slotted messages with increasing arrival rate (λ) in Figure 9. We find that in both cases, applying our scheme allows the throughput to be sustained for up to about twice the peak arrival rate that the current scheme can sustain. This can be attributed to the increase in N_{max} after applying our scheme.

We simulated the paging system for $p = 0.01$ and $p = 0.05$. Since p represents the success rate of the paging algorithm, (e.g., location management scheme), it is inversely proportional to the size of the paging area. A small value of p (e.g.,

0.01) represents a relatively large paging area, while a large value of p (e.g., 0.05) indicates a small paging area. From the data we captured in our experiments (Section 4.1), we observed that p was less than 5%.

Paging delay is the amount of time that it takes to establish a connection between the initiating terminal and the target terminal. It is mainly caused by paging channel overload. Figure 10 shows the average paging delay before and after applying our scheme. Again, we find that our scheme can sustain up to twice the slotted message arrival intensity that the current scheme can before paging delay grows exponentially. Note that the paging delay is roughly the same regardless of p since the number of non-slotted messages in the paging channel only affects the paging delay when λ is small. However, the paging delay is also small when λ is small.

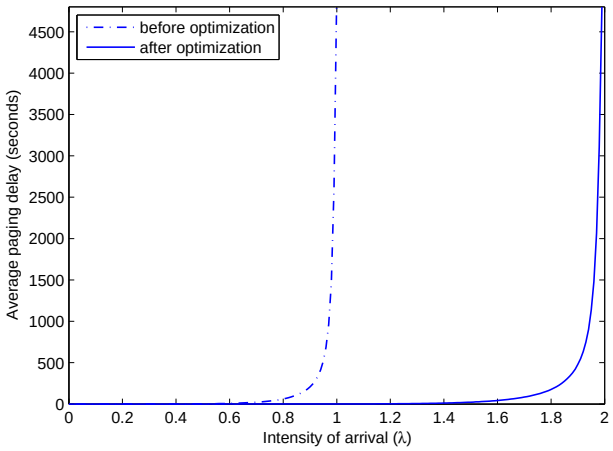


Fig. 10. The average paging delay on different slotted message arrival rates, before and after applying our scheme.

Section 4.1 demonstrated that our scheme would force the attacker to spend more resources before he could overload the paging channel (i.e., the attacker would need to generate more calls). Figure 11 shows that our scheme doubles the number of slotted messages required for saturating the paging channel.

5 Related Work

With the ongoing convergence of wireless voice and data networks, denial of service (DoS) attacks on the paging channel of wireless networks have attracted a lot of attention. Enck et al. presented a denial-of-service attack by sending a sufficient number of SMS messages per second to a range of cellular phones in the same area [6]. An attacker would need only a single computer with a

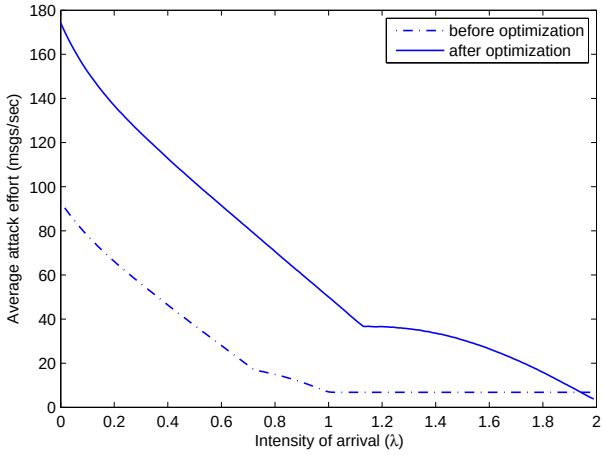


Fig. 11. The average number of slotted messages needed to saturate the paging channel (i.e., the attacker's effort).

broadband network access to disrupt a network in a major city by saturating control channels shared between voice calls and SMSs. Serror et al. provided experimental evidence of the impact of an attack on the paging channel. They injected UDP packets from the Internet to data users in a cellular network to cause increased load on the paging channel [5]. By improving paging efficiency, we aim at mitigating these attacks. Our approach is complementary and orthogonal to other attack detection [17] and mitigation techniques.

Researchers have proposed many solutions to enhance paging efficiency by improving location update schemes. The underlying idea is to increase the estimation precision of the mobile stations' location by exploiting their mobility patterns. [18] provides a good survey of early work. Some of them, such as time-based, distance-based and zone-based, have been adopted in the standard. Other proposed schemes include *movement based*, *cost based* and *velocity based*. One popular approach is profile based location management: each mobile station has a profile that helps the paging controller to predict the paging area. This idea is based on the observation that each individual user has her own mobility pattern. Researches in this area mainly focus on how to establish user's profile. E.g., [7] mined the call history of the users to build their mobility profiles.

To improve paging efficiency, location management schemes break the paging process into two or more stages. In the first stage, the network sends the paging message to a predicted small subset of cells in the paging area. If the mobile terminal does not respond, the network then sends the message to a larger set of cells. If the prediction is accurate, the average number of paged cells is expected to be much smaller; however, if the prediction is wrong, these approaches cause paging latency. The average paging latency in a normal paging operation is half of the paging cycle (2.56s). Each additional phase will add 5.12s to it. Our

scheme, by comparison, does not increase the paging latency, since it does not break paging into stages.

Some researchers have explored paging message optimization. [19] proposed a Bloom filter to map multiple page records to one fixed length bitmap. Quick paging, described in Section 3, also uses a special form of Bloom filter. The main difference between them is that the number of hash functions in [19] is dynamically calculated and is transferred as a parameter of each page message. While achieving high paging capacity in certain situations, this Bloom filter based paging system suffers from excessive false page rates and hence low battery efficiency. By contrast, our scheme causes no false page. Another problem with this approach is that it removed single paging records from paging messages; therefore, useful information previously piggybacked with the paging records, such as the Service Option field, was no longer available to terminals.

[9] aims at improving the paging efficiency of WiMax networks. Based on the observation that two individual MOB-PAG-ADV messages use more bandwidth than one MOB-PAG-ADV message with two records, it grouped multiple mobile station records into one MOB-PAG-ADV message to reduce the overhead and improve the paging efficiency. This solution is specific to WiMax. By contrast, our scheme applies to almost all mobile networks that require paging.

6 Conclusion

We propose a novel approach to improve paging efficiency and to mitigate DoS attacks on the paging channel. We describe a series of mechanisms for efficiently encoding terminal identifiers in page messages to increase the paging channel capacity. For instance, we can shorten the terminal identifier in a CDMA2000 General Page Message from its current length of 34 bits down to 7 bits. We evaluated our scheme using data measured on a live cellular network and using simulation. The results indicate that our scheme can significantly increase the paging throughput and the cost to the attackers, thereby mitigating DoS attacks on the paging channel. Our scheme is simple and is straightforward to implement. It does not incur any adverse effect, such as paging delay, false paging, and higher missed paging rate, that other schemes often suffer from. Furthermore, it is compatible with location-based paging efficiency improving schemes. Although we describe our scheme in the context of cellular networks, the scheme applies to other mobile networks such as WiMax.

Acknowledgment

This paper is based upon work supported by the National Science Foundation under Grant Nos. 0644450 and 0520320 and by a generous gift from Sprint. We thank Jean Bolot, Prasant Mohapatra and Sridhar Machiraju for their valuable comments.

References

1. Mobile radio interface layer 3 specification, December 2008. 3GPP TS24008.
2. Upper layer (layer 3) signaling standard for cdma2000 spread spectrum systems, September 2005. 3GPP2 C.S0005-D.
3. Air interface for fixed and mobile broadband wireless access systems, February 2006. IEEE Std 802.16eTM-2005.
4. H. Haverinen. J. Malinen. Mobile ip regional paging, june 2000. draft-haverinen-mobileip-reg-paging-00.txt.
5. Jérémy Serror, Hui Zang, and Jean C. Bolot. Impact of paging channel overloads or attacks on a cellular network. In *WiSe '06: Proceedings of the 5th ACM workshop on Wireless security*, pages 75–84, New York, NY, USA, 2006. ACM.
6. William Enck, Patrick Traynor, Patrick McDaniel, and Thomas La Porta. Exploiting open functionality in sms-capable cellular networks. In *CCS '05: Proceedings of the 12th ACM conference on Computer and communications security*, pages 393–404, New York, NY, USA, 2005. ACM.
7. Hui Zang and Jean C. Bolot. Mining call and mobility data to improve paging efficiency in cellular networks. In *MobiCom '07: Proceedings of the 13th annual ACM international conference on Mobile computing and networking*, pages 123–134, New York, NY, USA, 2007. ACM.
8. Signaling link access control (lac) standard for cdma1998 spread spectrum systems, September 2005. 3GPP2 C.S0004-D.
9. S. Mohanty, M. Venkatachalam, and X. Yang. A novel algorithm for efficient paging in mobile wimax. *Mobile WiMAX Symposium, 2007. IEEE*, pages 48–53, March 2007.
10. Hyun Suk Roh and Sang ho Lee. Paging scheme for high-speed portable internet (hpi) system. *Advanced Communication Technology, 2006. ICACT 2006. The 8th International Conference*, 3:4 pp.–1732, Feb. 2006.
11. J. Kempf. Dormant mode host alerting ("ip paging") problem statement, june 2001. rfc3132.
12. R. Ramjee, K. Varadhan, L. Salgarelli, S.R. Thuel, Shie-Yuan Wang, and T. La Porta. Hawaii: a domain-based approach for supporting mobility in wide-area wireless networks. *Networking, IEEE/ACM Transactions on*, 10(3):396–410, Jun 2002.
13. A.T. Campbell, J. Gomez, and A.G. Valko. An overview of cellular ip. *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 606–610 vol.2, 1999.
14. Xiaowei Zhang, Javier Gomez Castellanos, and Andrew T. Campbell. P-mip: paging extensions for mobile ip. *Mob. Netw. Appl.*, 7(2):127–141, 2002.
15. Shantidev Mohanty, Muthaiah Venkatachalam, Shailender Timiri, and Sassan Ahmadi. Proposal for ieee 802.16m quick paging channel design, Jul 2008.
16. Havish Koorapaty and Per Ernstrm. Quick paging signal for ieee 802.16e, May 2008.
17. Patrick Traynor, Patrick McDaniel, and Thomas La Porta. On attack causality in internet-connected cellular networks. In *Proceedings of 16th USENIX Security Symposium on USENIX Security Symposium*, 2007.
18. I.F. Akyildiz and S.M. Ho. On location management for personal communications networks. *Communications Magazine, IEEE*, 34(9):138–145, Sep 1996.
19. Pars Mutaf and Claude Castelluccia. Hash-based paging and location update using bloom filters: a paging algorithm that is best suitable for ipv6. *Mob. Netw. Appl.*, 9(6):627–631, 2004.