Mobile Data Charging: New Attacks and Countermeasures

Chunyi Peng Chi-yu Li Guan-hua Tu Songwu Lu Lixia Zhang

Department of Computer Science, University of California, Los Angeles, CA 90095
{chunyip, lichiyu, ghtu, slu, lixia}@cs.ucla.edu

ABSTRACT

3G/4G cellular networks adopt usage-based charging. Mobile users are billed based on the traffic volume when accessing data service. In this work, we assess both this metered accounting architecture and application-specific charging policies by operators from the security perspective. We have identified loopholes in both, and discovered two effective attacks exploiting the loopholes. The "toll-free-data-access-attack" enables the attacker to access any data service for free. The "stealth-spam-attack" incurs any large traffic volume to the victim, while the victim may not be even aware of such spam traffic. Our experiments on two operational 3G networks have confirmed the feasibility and simplicity of such attacks. We also propose defense remedies.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General—*Security and protection*; C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design—*Wireless communication*

Keywords

Cellular Networks, Mobile Data Services, Accounting Attacks

1. INTRODUCTION

Wireless access to Internet data services is getting increasingly popular, thanks to the deployment of 3G/4G cellular networks. Statistics from OECD [29] shows that, 62% broadband users in the US have subscribed to wireless data plans, with 137M subscribers in June 2010. There are also 1.2B mobile Web users worldwide already [28]. The explosive growth of smartphones (e.g., iPhones and Android phones) will further accelerate this usage trend.

While users enjoy wireless data access, it does not come for free. Most 3G/4G operators bill the user based on the usage data volume[1]. This metered charging[2] is officially stipulated by the 3G/4G standards. Based on the standards, charging is performed inside the cellular network (CN) on a per-flow basis. Each flow is defined by the five-tuple (source-IP, destination-IP, source-port, destination-port, protocol) or its subset. Whenever a data flow is initiated with the phone, the traffic volume is recorded at the CN when data traverse the CN to reach the phone/server. Therefore, the CN performs accounting operations based on its observed traffic volume. Carriers can also define their flow-specific billing policy.

In this paper, we present the first work that critically assesses the vulnerability of 3G/4G charging system. We discover loopholes in its policy practice and weakness in its charging architecture. As a result, we identify two new types of attacks against the charging system. In the *toll-free data access attack*, attackers can access any data service from the mobile phone for any period of time free of charge. It exploits the policy loopholes when the operator allows for certain free service (e.g., DNS service). In the *stealth spam attack*, attackers will inject arbitrarily large volume of spam data into the victim device, even after the target device has terminated its data service, thus fully unaware of such a spam. This attack exploits the architecture weakness of not using feedback from the user when making charging decisions, as well as features of Internet instant messaging applications (e.g., Skype and Google Talk). Our prototypes show that both attacks are feasible and simple enough to launch over the operational 3G networks. Our experiments on two 3G networks from two US carriers show that, the undercharge or overcharge traffic volume can go unbounded.

Three main contributions of this work are as follows: (1) We report the first security analysis on the 3G/4G network charging system and identify its loopholes; (2) We describe two new types of attacks, i.e., "toll-free-data-attack" and "stealth spam attack," which exploit the identified loopholes to undermine the charging system; we also use real experiments over two operational 3G networks to validate the feasibility and simplicity of these attacks and their potential damage; (3) We articulate the root cause for the existence of these loopholes and propose effective solutions to eliminating them. In summary, our study shows that, a dependable, metered charging system requires concerted coordination among the mobile device, the network, and applications. Security mechanisms are needed to strengthen every part and the overall system.

The rest of the paper is organized as follows. Section 2 introduces the 3G/4G architecture and its data charging system. Section

CCS'12, October 16–18, 2012, Raleigh, North Carolina, USA.

[1]In fact, operators do not offer unlimited monthly data plans for smartphone users any more. Both AT&T and Verizon effectively ended such plans for new customers in 2011, and T-mobile limits the high-speed data volume in its so-called unlimited data plan.

[2]In this work, we use the words "charging" and "accounting" interchangeably.

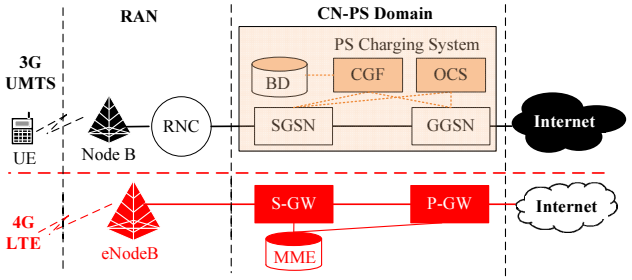


Figure 1: 3G/4G network architecture and charging components in PS domain.

3 analyzes the vulnerability of mobile data charging. Sections 4 and 5 describe the toll-free data access attack and stealth spam attack, as well as countermeasures, respectively. Section 6 compares with the related work, and Section 7 concludes the paper.

2. BACKGROUND

In this section, we introduce the charging architecture and process for mobile data services. Unlike the flat charging practice over the Internet, the current cellular network has been using usage-based charging for its data services. That is to say, the operator collects the actual usage volume over time for each user and imposes charges accordingly.

Broadly speaking, charging is performed on a per-connection basis. To communicate with a host on the Internet, the mobile device needs to first create a bearer service connection with the cellular network, which is further connected with the wired Internet. Once the connection is established, data packets are delivered. The connection has to traverse gateway-like devices (similar to routers in the Internet) in the cellular network core. These gateways then perform accounting operations by recording the data volume of those packets that traverse them, until the connection is completed.

Specifically, this accounting scheme has been shaped by both the standards and the operators' policy practice. The standards define the architecture and mechanisms, whereas the operators specify their own charging policies. Here, we mainly introduce the charging architecture and process in context of Universal Mobile Telecommunications System (UMTS), the most widely deployed 3G cellular network technology [19]. Note that, its charging mechanisms and operations are also applicable in 4G networks, e.g., Long Term Evolution (LTE) networks. For reference, Table 1 lists important acronyms used in this paper.

CDR	Charging Data Records
CN	Core Network
EH	External Host
FBC	Flow Based Charging
GGSN	Gateway GPRS Support Node
PDP	Packet Data Protocol
PS	Packet-Switched
SGSN	Serving GPRS Support Node
UE	User Equipment

Table 1: Table of important abbreviations and acronyms.

2.1 Data Charging Architecture

Figure 1 shows the overall 3G UMTS network architecture and charging system for data services. The UMTS network consists of the *Terrestrial Radio Access Network* (RAN) and the *core network* (CN). RAN provides wireless access to the mobile device (called User Equipment (UE)), and exchanges data session provisioning with the Packet-Switched (PS) core networks.

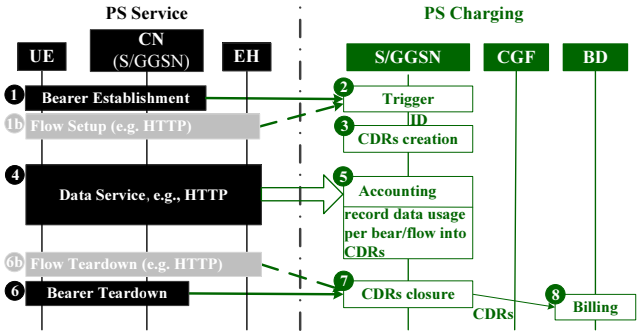


Figure 2: Charging procedures for a data service flow.

The major components of the PS core network are the *Serving GPRS Support Node* (SGSN) and the *Gateway GPRS Support Node* (GGSN). SGSN handles data packet delivery from and to the UEs within its geographical service area. GGSN acts as a router between the SGSN and the external wired Internet, and 'hides' the 3G UMTS infrastructure from the external network. In fact, SGSNs and GGSNs are the aforementioned gateway-like devices, recording data usage through them to perform charging functions.

Current cellular networks support both offline and online charging modes [17]. In addition to SGSN and GGSN, three more charging components work to support both modes: the *Billing Domain* (BD), the *Charging Gateway Function* (CGF), and the *Online Charging System* (OCS). In offline charging, data usage is collected during service provisioning in the form of *Charging Data Records* (CDRs), which are sent to the BD to generate data bills offline. The SGSN and GGSN are responsible for and generating CDRs. The CGF is used to validate CDRs from SGSNs/GGSNs and transfer CDRs to the BD. In online charging, mobile users have to pre-pay to obtain credits for data services in advance. The OCS authorizes whether or not users have enough credits so that GGSN/SGSN can proceed data services. GGSN/SGSN deducts data usage from the available credits and stops data services upon zero credit.

The charging subsystem for 4G LTE cellular network is almost identical to 3G UMTS. The major difference (also shown in Figure 1) is that, *Serving Gateway* (S-GW) and *Packet Data Network Gateway* (P-GW) [25] replace SGSN and GGSN to collect data usage and generate CDRs.

2.2 Data Charging Procedures

We next describe how mobile users are charged for data services through an example. Consider Alice is about to browse CNN news, thus starting a PS service (say, HTTP). Figure 2 illustrates the charging procedures (in the right) during the data service process (in the left), where the external host (EH) is `www.cnn.com`.

We first consider the offline charging mode. Initially, Alice has no available bearer service connection, which is used to carry one or multiple PS data services. She thus first establishes a bearer via Packet Data Protocol (PDP) Context Activation [15] (Step 1) where PDP contexts provide all the required information for IP packet data connections in cellular networks. Upon this activation, the UE is allowed to connect with the external data network through the SGSN and GGSN. This activation also triggers the charging procedure; the GGSN assigns a unique charging ID to the activated PDP context (Step 2). Using the charging ID, the SGSN and GGSN

start to create CDRs (Step 3), and are ready to record the upcoming data volume.

In addition to charging per bearer (PDP context), 3G operators also support charging per data flow, called as *Flow Based Charging* (FBC). FBC separates charging for different services (e.g., Web or VoIP) within the same PDP context [18]. The standard [16] specifies that one data flow is typically identified by five tuples: (1) source IP address, (2) source port number, (3) destination IP address, (4) destination port number, and (5) protocol ID of the protocol above IP, e.g., TCP or UDP. For example, a HTTP data flow can be represented by *(*, *, *, 80, TCP)*[3]. In this CNN case, FBC is triggered when Alice starts mobile Web browser and initiates a HTTP session (Step 1b).

Alice can read CNN news now. Both SGSN and GGSN route data packets between the UE and the external data network during the data service session (Step 4). In the meantime, the SGSN and GGSN record the traffic volume that *arrives* at them into corresponding CDRs (Step 5). They count the payload of GTP-U (*GPRS Tunneling Protocol- User Plane*) packets as data volume (see Figure 3); GTP-U delivers data within cellular networks and runs below the IP protocol.

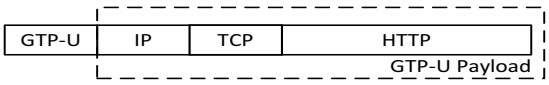


Figure 3: Example of of a GTP-U payload.

The accounting procedure (Step 5) lasts until this data service completes. It occurs when the UE tears down this bearer (Step 6) in bearer-based charging, or when Alice closes her HTTP session in flow-based charging (Step 6b). CDRs are subsequently closed and transferred to the BD (Step 7). Finally, the BD generates a billing item based on CDRs and assigns it to the proper user.

The online charging process is similar, though OCS participates in the triggering and accounting steps (Steps 2 and 5) by authenticating GGSN/SGSN to use user credits. There is also no need to send CDRs since the consumed credits are deducted during Step 5.

Regardless of the online/offline charging, the end goal is to ensure that the data usage recorded by the network is *indeed the same as the amount used (and wanted)* by a mobile device. The critical issue in mobile data charging is thus whether the accounting architecture and policy practice in cellular networks are secure enough to ensure proper billing. In this work, we seek to analyze such vulnerabilities, which malicious attackers can exploit to alter data usage record and make mobile users pay more (overcharging) or less (undercharging). We focus on the attack issues in offline charging, since the same issues also apply to online charging.

3. 3G ACCOUNTING VULNERABILITY

In this section, we provide an overview on the vulnerability issues we have identified in 3G data charging.

3.1 System Model

We focus on the security issues of the usage-based accounting rather than pricing, which sets the unit price for usage. In the typical scenario, a mobile user uses her/his smartphone to access the Internet data service via the 3G wireless network. Data communication is performed between the mobile phone and the Internet server/host. The study can be readily extended to the mobile to mobile communication setting.

As described in Section 2, the mobile user is charged for the data service (s)he uses. The operator records the data volume exchanged by the data service over time, and the user will pay based on the recorded usage volume. Specifically, the CN (more precisely, SGSN/GGSN) records the volume of the packets that traverse it and then charges the usage to the proper user via the mapping from the flow, the bearer to the corresponding UE and user, as illustrated in Figure 4.

[3]Each of the five tuples can be a wildcard.

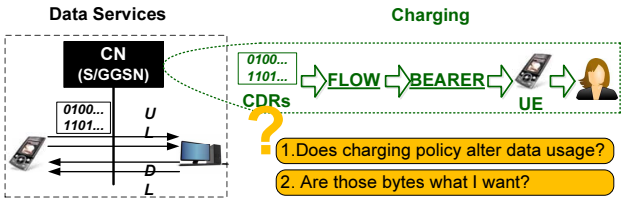


Figure 4: Issues in data charging practice.

We assume that the 3G charging subsystem is not compromised, i.e., all charging elements are operating properly (as described in Section 2). This implies that, the data usage records kept at SGSN/GGSN are not attacked, the mappings from CDRs to the flow, the bearer, and the mobile user are also intact. Moreover, user authentication within 3G/4G cellular networks works properly. Attackers cannot spoof other UE devices to access data services.

3.2 Two Achilles' Heels

Our study shows that 3G/4G accounting architecture and policy practice contain two loopholes, which can be exploited to launch charging-related attacks against the operator and mobile users.

The first relates to the charging policy that each 3G operator can define regarding what to be charged. Indeed, 3G/4G operators are allowed to adopt different charging policies. For example, they can charge Multimedia Messaging Service (MMS) service different from the common Internet data, or even provide free access for certain data services. The security implication is: Can the differential charging policy be exploited to alter the actual data usage? If two data services are charged differently, is it possible to fabricate the service type and masquerade as a cheaper one?

We study an extreme case of the above issue. Our findings in Section 4 show that, major US carriers usually offer free Domain Name Service (DNS) service, and all data usage associated with DNS service will be free. Our security question is: Given one type of free data service, is it possible to evade charges for other data services (e.g., standard Web browsing) as well? Our work confirms that it is indeed feasible. It is easy to exploit this loophole and launch an undercharging attack, during which the mobile user is charged for data volume smaller than its used amount, or even free of charge in the worst case. This attack defeats the fundamental principle of metered charging in all cellular networks.

The second loophole is rooted in the 3G/4G charging architecture. The core network records the packets, which traverse it and belong to specific flows, and charges the user accordingly. However, a question still remains: Is there any secure mechanism to verify with the user on whether the data would be indeed wanted by the user? What about those data bytes the attacker injects but mobile user never wants?

Our analysis in Section 5 shows that, current charging architecture lacks feedback mechanisms that allow the mobile user to explicitly express what packets are wanted or unwanted. Instead, operators decide on what packets are charged using their own rules. We further discover that, a mobile user is able to terminate the malicious (or suspicious) service on its application layer locally, but it cannot terminate the charging operations done at the carrier side.

Therefore, malicious attackers can inject spam packets and deceive the carriers to charge the mobile user for data volume larger than what it requests. Our experiments show that, the overcharging attack can be easily launched and there is no obvious upper bound on the overcharged volume. We note that, 3G/4G operators do provide security mechanisms via NAT and firewall. Consequently, a mobile device does not have a permanent and public IP address. It uses a private IP address and obtains temporary access to data services via NAT. The NAT-based operation ensures that the mobile user needs to initiate this service flow at the start. However, it is ineffective during the delivery process once the service flow starts. Consequently, it cannot shield incoming spam data when malicious hackers hijack the flow or when the victim later finds that (s)he is trapped.

3.3 Experimental Platform and Methodology

We design and conduct a series of experiments to examine security issues in 3G data charging. We now describe our platform and methods to obtain the data usage observed by the operator (V_{OP}) and the mobile phone (V_{UE}). The details of experiments are described in the followup sections.

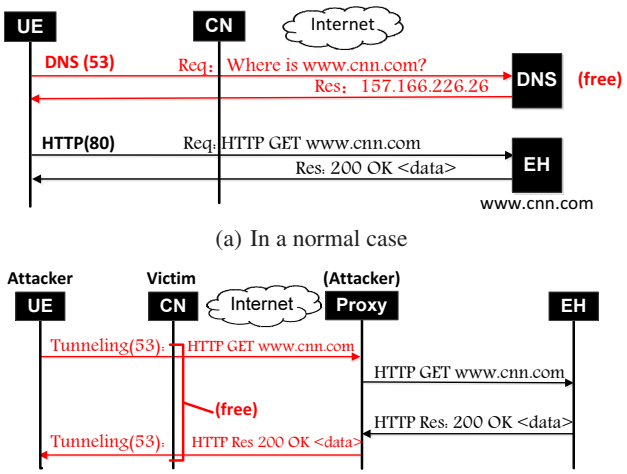
We run tests with two major mobile operators in the US, which together offer nationwide coverage for 102.3M users, thus claiming about 50% of US market. We denote them as Operator-I and Operator-II in this paper for privacy concerns. Our mobile devices use three Android phone models: HTC Desire, Samsung Galaxy S2, and Samsung Galaxy Note GT-N7000, running on Android 2.2, 2.3.4 and 2.3.6, respectively. Our experiments show that all the findings are phone platform independent. We use an ASUS EeeBox PC EB1501 desktop as the deployed host outside the cellular networks. It runs on an Intel Atom N330 1.6 GHz Dual Core processor and 1.5 GB DDR2 memory. This host acts as a content server (e.g., Web), proxy or attacker in various tests.

We use two methods to obtain data usage logged by operators. The first one is to dial a special number from the mobile phone to retrieve the remaining monthly data usage via a text message in a near real-time mode. Most operators support this Dial-In feature, e.g., via dialing #DATA for Verizon, *DATA# for AT&T, and #932# for T-Mobile in the US. By logging data usage before and after our experiment, we compute the usage volume observed by the operator during the experiment. The second method is to log onto the mobile carrier website and obtain itemized data usage records online. Based on the access availability, we choose the first method for Operator-I and the second for Operator-II. Both support 1 KB accuracy in their data usage report. Note that, data usage records only have timestamps. We use extra mechanisms to ensure that the usage record is exclusive to data services in our tests. We run factory reset first and disable "Background data" and "Auto-sync" features. We also use Wireshark [38], a monitoring tool to capture all-level packets to/from the phone to ensure clean environment.

To obtain data usage on mobile phones, we develop our own tool to use TrafficStats class interfaces [14] provided in Android SDK to collect network traffic statistics. We record the number of packets and bytes transmitted and received on all interfaces and on a per-application basis. We further use WireShark to log packet traces at our phones or deployed host if needed. We conduct each experiment for 5 –15 runs and average the results over these runs.

4. FREE MOBILE DATA ACCESS ATTACK

In this section, we report how attackers can obtain mobile data services for FREE. We find out that there exist loopholes in the current charging policy. Operators allow free data service for certain data flow, but do not enforce that the transmitted packets *indeed* belong to the designated free flow. Even worse, no effective mechanism is implemented to limit the traffic volume going through this free ride. Consequently, these loopholes can be exploited to enable any form of mobile data services for free. We use real experiments to examine security issues in the operators' charging practice, and describe three approaches to "free" data services. Finally, we make suggestions to fix this "bug."



(a) In a normal case

(b) Under a toll-free-data-attack

Figure 5: Web browsing in normal and attacked cases.

4.1 Loopholes in Charging Policy Practice

The 3G standards offer the operators flexibility to define their own charging policies. Unfortunately, their policies and implementations may contain serious flaws.

We use the example of Web browsing (`www.cnn.com`) to illustrate the vulnerabilities in the charging policy practice. Figure 5(a) illustrates typical steps for Web browsing. Upon receiving the target URL, the Web browser immediately initiates two actions. One is to send a DNS query to request the IP address for this URL. The other is to send a HTTP query to the Web server using the obtained IP address and receive a HTTP response. In mobile data charging, the above operations invoke two charging flows. One is the DNS query/response which goes through the CN to the DNS resolver or server. It is primarily carried by UDP on port 53, though TCP over port 53 is also allowed [32]. The other flow is for HTTP, which traverses the CN to enable communication between the UE and the Web server. It runs on TCP using port 80 (or other ports, e.g., 8080 or 443 for https). The CN records the data volume associated with each flow for billing.

Our study shows that, both operators tested in our experiments offer free DNS service. This policy practice makes sense, since DNS is considered a fundamental service for the Internet applications. Almost no Internet services can be initiated without DNS. DNS service is offered for free by many public DNS servers (e.g., Google, OpenDNS [4]). Operators thus have every reason not to charge DNS messages, to facilitate followup data usage by other Internet services. Therefore, free DNS service can be justified as a good (at least reasonable) policy.

However, our study shows that, there exist two loopholes to implement this free-DNS policy in reality. First, there is almost no enforcement mechanism to ensure that the packets going through this DNS-reserved port are indeed DNS messages (*free fake DNS loophole*). Second, there is no effective mechanism to limit the traf-

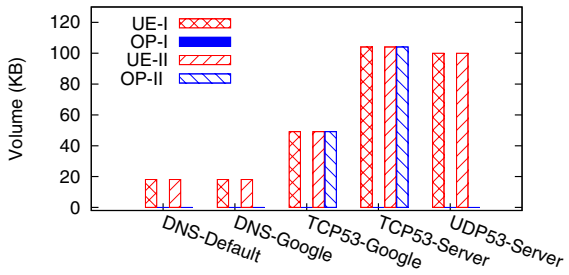


Figure 6: V_{UE} and V_{OP} in five DNS tests.

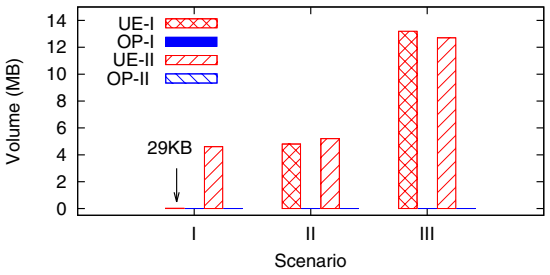


Figure 7: Feasibility test of free data services; $V_{OP} = 0$.

fic volume going through this port (*no volume-check loophole*). We next elaborate both using experiments.

• **Free fake DNS loophole** Our experiments show that, operators do not enforce free DNS service via the standard five-tuple flow ID (src IP, dest IP, src port, dest port, protocol). Instead, they use only the destination port (or plus protocol ID), thus exposing an obvious vulnerability.

We use experiments to verify whether the DNS service is free and what exact factors the free DNS service depends on in the operator's implementation. We conduct five experiments: (1) *DNS-Default*: the UE sends 100 DNS queries to the default DNS server provided by the operators; (2) *DNS-Google*: the UE sends the same DNS queries as (1) to a Google public DNS server (IP address: 8.8.8.8); (3) *TCP53-Google*: the UE sends the same DNS queries as (1) using TCP via port 53 to the above Google DNS server; (4) *TCP53-Server*: the UE sends 50 random packets to our own server using TCP via port 53, and require the server to return the received packets; each packet is 1KB, including IP/TCP headers; Source port number is randomly allocated; and (5) *UDP53-Server*: we repeat (4) but using UDP.

We conduct these experiments with two US major operators. We have purchased unlimited daily data plans from both operators and thus do not run into legal issues while testing free data services (the actual data usage is not counted by operators). We invalidate the hypothesis that the operator has no incentive to correctly report the traffic usage for users with unlimited access. To this end, we also use 200MB and 4GB data plans in free data service tests, and compare the results with using unlimited data plan. Results are consistent in all three plans. We further test different services (e.g., Web, YouTube, Gmail) using our unlimited data plans and verify that the data usage records at the UE and the operator are consistent. Figure 6 plots the data volume observed by the UE and two operators in all five cases. The results show that,

Operator-I: Packets via **port 53** are FREE Operator-II: Packets via **UDP + port 53** are FREE

Specifically, the UE sends and receives about 18.1 KB for 100 DNS queries and responses in both DNS-Default and DNS-Google tests. In the TCP53-Google test, the traffic volume rises to 48.1 KB due to TCP signaling overhead (SYNC, etc). In both TCP53-Server and UDP53-Server tests, the UE sends and receives 100 KB as expected. Operator-I charges for free (i.e., $V_{OP} = 0$) in all cases while Operator-II charges those TCP cases. From these results, we learn that the free DNS service is implemented by Operator-I using only one field in the flow ID (i.e., the destination port 53). In contrast, Operator-II enforces free DNS service using two tuples in the flow ID, i.e., UDP over destination port 53.

• **No volume-check loophole** Our study further shows that, there is no mechanism to limit the traffic volume going through this free-service port. To this end, we build our own server outside the cellular network that exchanges data services with mobile phones using UDP over port 53. We perform three experiments: (I) *Free-One*: the UE sends one request to our server to download a 5MB file; (II) *Free-Equal*: the UE uploads a 3MB file to our server, and requests to return the delivered packets; (III) *Free-Long*: the UE sends many small requests (100 B) to our server for an hour, each of which requests a 1KB response.

Figure 7 plots the data volume observed by the UE and both operators in the above three scenarios. It shows that, both operators can be exploited for free data services in all these scenarios, except that Operator-I does not allow unbounded traffic for one fake "DNS" request. The fake DNS message In the first test, Operator-I only allows to deliver 29 KB downlink data to the UE, while Operator-II delivers much larger file (up to 4 MB). We gauge that Operator-I might have enforced a checking mechanism to verify the size of the response message, in which a real DNS message size is typically bounded. However, this size checking can be easily bypassed. The UE simply sends out many small, dumb packets over this session, to increase the quota for downlink traffic. Then large downlink data can pass this checking. This has been validated in scenarios (II) and (III). In these tests, the gap between V_{UE} and the expected file size is mainly caused by unreliable transmission via UDP. These results demonstrate that free DNS service can be exploited to create any "free" data service.

4.2 Toll-Free Data Service Attack

We now show how to launch "free" mobile data access attack by using the above two loopholes. The key idea is to use a proxy server (placed outside the cellular network) to bridge the data access between the mobile phone and the Internet server. The communication between the proxy and the phone is carried out over the free channel (i.e., UDP or TCP over port 53, depending on the operator policy). We use "tunneling" between the UE and the proxy server. The proxy server relays packets on behalf of the UE. Free communication is thus extended to between the UE and an Internet host, while the 3G core network (CN) is the victim. Figure 5(b) illustrates the example of how Web browsing becomes free of charge. The process is similar to calling 800-voice hotlines, but for free data access. We thus name it as the "***toll-free-data-access-attack***."

We take three approaches when implementing the toll-free-data-attack. All show that, it is simple enough to obtain free mobile data access in reality. The first approach is to use a HTTP proxy running on port 53. It is easily done using available free proxy software such as FreeProxy [6]. The mobile Web browser is then configured to use the established HTTP proxy, as shown in Figure 8(a). This approach is easy to implement; no coding and hacking are needed. However, it only works for Web browsing and for Operator-I, which allows free TCP via port 53. To evaluate its effectiveness, we test two Web browsers – Mozilla Firefox and Opera Mobile [11], one

(a) HTTP proxy (b) Socks proxy (c) FDP proxy

Figure 8: Three approaches to "toll-free-data-access-attack."

hour each. We are able to use Operator-I network for free, while the actual data volume goes beyond 20 MB.

The second approach is to use a socks proxy. It works with various application protocols, e.g., HTTP, FTP, SMTP, POP3, NNTP, etc. Similarly, we deploy a socks proxy running on port 53. On the phone side, we install ProxyDroid [12] to enable socks proxy functionality. The phone configuration is shown in Figure 8(b). This method supports more applications without configuring each application individually. However, it still only applies to the TCP-53-free operators. We assess this attack with Operator-I using mobile applications, e.g., Web browsing, YouTube, Gmail, Google Map, Skype and FTP (via AndFTP [1]). The results show that, all services are free of charge except Skype voice call and FTP download. We figure out that, these two applications fail to go through the socks proxy; It is an implementation issue in ProxyDroid.

The third approach is to deploy a proxy server to enable "tunneling" between the phone and itself. To this end, we design a Free Data Protocol (FDP) to encapsulate data packets between the UE and the proxy into fake DNS messages, i.e., to carry packets in ANY-on-port-53 flows for Operator-I and UDP-on-port-53 flows for Operator-II. These messages are any data packets, not following DNS semantics. To bypass the limit of data volume for one fake DNS request (for Operator-I), FDP also periodically sends small KEEP-ALIVE messages from the UE side. The attacker enables the FDP at the UE and the proxy server. Note that, the DNS-tunneling idea is also used in the iodine [7] and NSTX [10] tools to enable Internet access over DNS. Moreover, the NSTX was used to demonstrates the similar idea for free Internet access with a toll-free Microsoft PPP dial-in number in Germany [8]. Both work in the wired Internet and free Internet access is available with specific DNS servers. In our experiments, we have built a simple prototype that revises applications to use FDP. We test our prototype with the revised HTTP and FTP applications working on top of FDP. Figure 8(c) captures the screen shot when visiting `www.cnn.com`. It shows that, data access is free for both operators while the actual data volume reaches 100 MB. Moreover, the upper limit of free traffic volume seems unbounded in our tests.

4.3 Suggestions to Fix the "Bug"

The simplest solution is to stop free DNS service or any other free data services that can go outside cellular networks. Fundamentally, for a metered charging service, people necessarily have incentives to exploit and abuse any transfer that is free. Therefore, the simplest, possibly also the best solution to abuse prevention is to eliminate the free services. Moreover, DNS traffic is negligible in normal cases; it should lead to no noticeable difference in most usage scenarios.

We also seek remedies to fix this bug while still retaining the free DNS service. For example, we have considered that the operator can provide quota for free DNS service. The DNS data usage beyond the quota will be still charged. Ideally, the quota should be assigned based on the average usage patterns. It can be a fixed amount or a percentage of the data usage. The challenge for this approach is how to set an appropriate quota. Some applications or services such as MobileMe [9] and DNSSEC [2] may heavily use DNS while others do not. The alternative approach is to enforce checking on the destination IP address of the DNS request. For example, free DNS services are only allowed when these messages go to designated or authenticated DNS resolvers or servers managed by carriers. However, it is still possible for attackers to deceive those resolvers/servers to forward fake "DNS" requests to a fake DNS server. The only difference is that the attack cost could be higher.

In the more general context, when the charging policy allows different unit-prices for diverse services (including free access to mobile Facebook [3] or a given Web site [5] in the extreme case), extra bullet-proof mechanisms are required; otherwise, the attacker always seeks to use the cheaper one. However, the deployment and operation of such security mechanisms will inevitably increase the cost of the carrier. Moreover, the security mechanism still needs to ensure itself to be secure in its design and operation. All these pose interesting research issues for the future.

5. STEALTH SPAM ATTACK

In this section, we describe the stealth spam attack, which is a new spam threat against mobile devices by exploiting the loopholes in current 3G/4G charging system. It stealthily injects a large volume of spam data, which the mobile device may not be even aware of (e.g., after the mobile device already closes the data session on its side). This incurs extra payment on the mobile user.

Stealth spam attack is different from conventional spam threats targeting mobile devices. Conventional spams include Email spam, SMS/MMS spam, junk image or video embedded in Web pages, etc. Users are typically aware of these annoying junk messages and may take actions to block them. In contrast, the stealth spam attack can be long lived, lasting several hours or more (observed in our experiments). The persistent spam session not only allows for the attacker to send a large volume of junk data, but also does it covertly. The users may be completely oblivious of such attacks.

5.1 Challenges and Opportunities

In practice, operators widely use NAT middlebox to handle IP address allocation of mobile devices [37]. Note that, attackers need to know the IP address of the phone when injecting spam data against the mobile device.

The deployment of NAT makes launching mobile spam attack a challenging task. Specifically, NAT offers two countermeasures against spam. First, it decouples network access from public reachability. The mobile UE is only allocated a private IP address (not reachable from the external network) when its bearer (i.e., PDP context) is activated. The UE is reachable from the public Internet only after NAT assigns it a translated IP address and a port number. This dynamic assignment only occurs when the UE initiates a data session (e.g., when starting a Google search or signing in Skype). Without the explicit activation from the UE side, data-charging operations never happen (as shown in the normal case of Figure 9). This tends to shield most conventional spam threats that send data to the UE via its IP address.

As the second countermeasure, operator's NAT boxes only grant temporary permissions for the traffic traversing the cellular core network. They only allow for the traffic to pass through within a provisional time window when the data session is alive. In the normal scenario, the charging time window ends when the UE terminates this data service. For example, mobile Web browser may immediately send a TCP FIN message to close the TCP connection, once the Web page is downloaded. This way, only within the

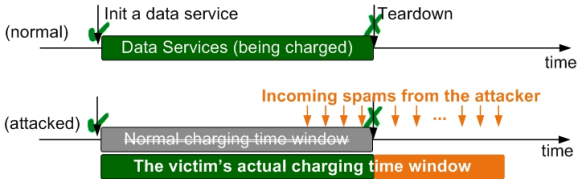


Figure 9: Illustration of stealth spam attack.

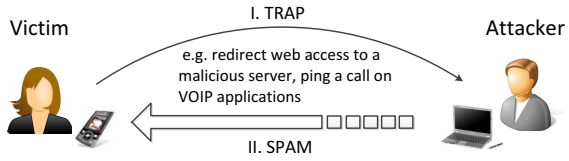


Figure 10: Steps to launch stealth spam attack.

given time window, those hosts, which know the access information (i.e., the translated IP address and the port number), are able to inject traffic to the UE. This window-controlled access also helps to protect the UE from spam threat. In addition, firewalls deployed by operators can also filter out spam.

On the other hand, the loopholes in the current 3G/4G charging system, as well as in applications, also create opportunities for stealth spam attack. Our analysis and experiments show that, there exist two loopholes in the current charging system. The first loophole is that,

> Data flow termination at the UE $\neq$ charging termination at the operator.

There exists inconsistency between the UE status and the operator's view on termination of a charging operation. When the user closes an application or an Internet service, (s)he thinks that the data flow is about to release and no more incoming traffic is allowed. However, the operator may view differently: This flow does not terminate as long as incoming packets belonging to this flow still arrive. The current 3G charging takes the operator's view. Therefore, charging can last much longer than expected. This occurs when the attacker starts this incoming spam before the normal teardown by the UE (shown in Figure 9)). We further find out that operators are unable to effectively stop data charging even when the UE explicitly sends teardown signals (e.g., in TCP). It is even worse for those UDP-based data service. The charging can last even longer once the spam starts; there is no sign for it to stop based on our experimental observation. We will elaborate them in next sections.

The second loophole is that,

> Initial authentication $\neq$ authentication during the whole data process.

All the authentication operations are performed at the start of the data flow (or when establishing the PDP context), but not when closing a flow. Therefore, the current charging procedure secures the initialization of the flow but not the whole process. Specifically, it cannot protect the data flow in the teardown process. The current design works for voice calls but not for data. Packet-switched IP data forwarding can push packets along different paths to reach the victim UE without prior consent, different from the circuit-switched fixed route for voice calls.

With these loopholes, stealth spam attack can be launched. Figure 10 shows two typical steps to launch this attack: trap and spam. First, it traps the UE to obtain its confidential access information

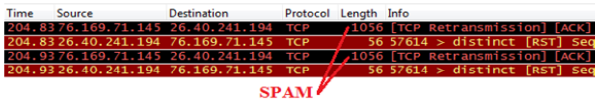


Figure 11: Wireshark traces at the victim even after the UE tear downs the TCP connection.

and flow permission to traverse the CN. The second step is to send junk packets. In the following, we describe how to implement them in several example scenarios and examine how badly it may hurt the victim.

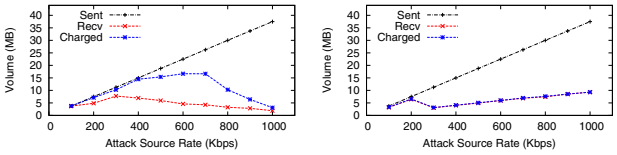
5.2 Spam Attack in TCP-based Services

We now describe how spam attack poses threats to those TCP-based services. Since TCP is a stateful procotol, we expect the spam to stop early once the UE application closes its TCP connection. Take Web browsing as an example. Once the Web page is fully retrieved, the Web browser may send a TCP FIN signal to the Web server and closes this TCP connection. Even though the Web server is malicious, the timeout mechanism also helps the UE to close this connection. The timer is typical set from tens of seconds to several minutes. However, our study has confirmed that the current charging practice contains loopholes. The operator may not stop charging, even when they can learn that the connection is closed by the UE.

In our experiments, we deploy a Web server as the attacker and modify its used TCP protocol. The spam attack starts when the UE clicks a malicious Web link and setups a TCP connection with the attacker. In the modified TCP, the normal TCP connection teardown procedure is disabled. This TCP will never send FIN or FIN-ACK signals like a normal TCP, upon receiving the teardown request from the UE. Once the UE is connected, the attacker immediately sends junk packets at a fixed rate for a given duration. To enable fixed-rate testing, we also disable TCP congestion control.

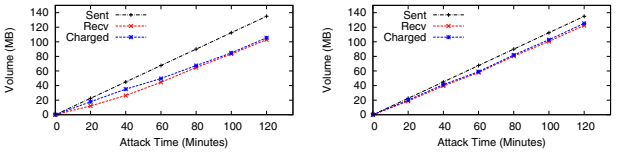
We first run experiments using various source rates for five minutes. Figure 12 plots the data volume increase due to this attack in both networks. It is observed that, as the incoming source rate grows beyond one threshold (about 400Kbps for Operator-I, 200Kbps for Operator-II), the attack seems to be blocked by the operator. The higher the source rate, the earlier the attack is blocked. For example, the spam is blocked in 24.7 seconds when the incoming rate reaches 1 Mbps for Operator-I while it gets blocked in 2 minutes for those attack at the source rates from 300 Kbps to 1 Mbps for Operator-II. This result implies that, operators do offer certain protection mechanism (e.g., blocking the TCP connection if it is too fast). However, these protection policies are operator specific. We find that, Operator-I may block the access to any data service while Operator-II only blocks this specific data service. We also observe that, the charging time window is not determined by the TCP connection status. When the UE closes this TCP connection, it sends TCP-RESET signals upon receiving spam packets. Figure 11 shows the Wireshark trace at the victim; TCP-RESET signals indicate that the UE aborts the connection and the 1056byte-packets are spam data units. The trace shows that, operators still allow the delivery of those spam packets and charge mobile users even when the UE TCP connection is closed.

We also test this attack at low source rate (150 Kbps) for various durations. Figure 13 plots the data volume increase in both operators. The low-rate attack can easily bypass the security check implemented by both operators. The attack can last for two hours; there is no sign to end during our experiments. The data volume incurred by this attack has exceeded 100 MB.



(a) Operator-I (b) Operator-II

Figure 12: Data volume caused by TCP-based stealth spam attacks under various source rates.



(a) Operator-I (b) Operator-II

Figure 13: Data volume caused by TCP-based stealth spam attacks for various durations.

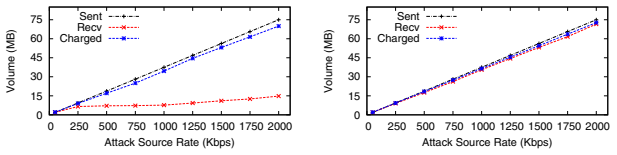
5.3 Spam Attack in UDP-based Services

We now describe stealth spam in UDP-based services. Since UDP is connectionless, it is even harder to decide when the UDP-based service ends and when the charging operation ends accordingly. The bad news is that, there is no clear protection mechanism for UDP-based service, while the operators at least use sort of abnormality-check for TCP-based sessions. The malicious attacker can launch stealth spam in UDP-based services by trapping the victim to open a UDP connection with itself. It may not be popular to use a malicious link to open UDP connection, we introduce to use two popular applications (e.g. VoIP and video streaming) to trap the victim and leak the access information.

• **Spam attack from your buddies** We demonstrate that the attacker can use VoIP service, including Skype and Google Talk, to construct the stealth spam. We use Skype as the example application. Skype is a globally used VoIP service and allows users to communicate with peers via voice, video, and instant messaging over the Internet [13]. Skype allows the buddies to communicate directly. A buddy on Skype has the chance to directly connect to the victim device without extra authentication.

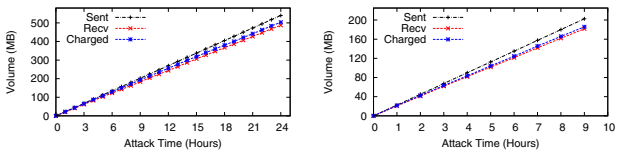
The first step to launch this attack is still to obtain the victim's confidential access information (i.e., translated IP address and port number) and its permission for this flow to traverse cellular networks (in the *trap* step). To this end, the attacker starts to make a call to this victim when he gets online using mobile phones. The attacker hangs up before the victim accepts the call, or even before the call rings at the victim side. This way, the victim may not be even unaware of this attempted call. During this process, the victim's Skype client performs two operations. First, it sends its access information to the attacker, which is proved in the attacker's Wireshark trace. In the meantime, it automatically notifies the operator that it accepts this flow, which subsequently grants the traffic flow from the spammer to traverse cellular networks. In the *spam* step, the attacker just sends junk UDP packets. The attacker can confirm that the victim is indeed a mobile user, based on the victim's translated IP address given by NAT. The operator-owned IP address block is readily known in advance. The spammer can also pick up the victim and launch operator-specific attacks.

We run experiments to validate this attack and verify whether extra checking mechanisms exist. We also vary the attack durations and incoming source rates in the tests. Figure 14 plots the overcharged volume versus different source rates during the five-



(a) Operator-I (b) Operator-II

Figure 14: Data volume caused by UDP-based (Skype) stealth spam attacks under various source rates.



(a) Operator-I (b) Operator-II

Figure 15: Data volume caused by UDP-based (Skype) stealth spam attacks for various durations.

minute Skype spam attack. The charging volume increase is in proportion to source rates. It implies that, operators do not enforce any security mechanism for UDP-based services. The spam volume can consequently grow much larger. We also make an interesting observation. In Operator-I, even though these packets may not be actually delivered to the UE (e.g., when the weak radio link cannot afford high-rate source), they are still charged by the operator. It shows that the operator might charge the mobile users based on the volume that arrive at them, not the one that they successfully delivery to the UE. Figure 15 plots the data volume caused by Skype stealth spam for various durations, with the source rate being 50kbps. It shows that the overcharge volume grows in proportion to the spam duration. There is no sign to end even when the attack has already lasted 24 hours for Operator-I (the overcharge volume reaches 500+ MB) during our experiments.

We also note that, the attack is still ongoing even after the victim signs out from Skype. The Wireshark trace at the victim side (see Figure 16) indicates that, spam packets still arrive at the UE and are charged by the operator after the UE logs out Skype. In the trace, the message of ICMP Port Unreachable shows that the UE has closed this application port after Skype logout.

In addition to Skype, this spam can be launched via Google Talk. The attacker also makes a call before the victim accepts it to trap the mobile user. The performance is similar; we omit it due to lack of space. Note that, the Skype/GTalk-based attack is a result of both 3G charging system vulnerability and Skype/GTalk implementation. The operator exposes the vulnerabilities at the first place, which still charges incoming spam packets that mobile application do not accept. The root cause is still that there is no feedback mechanism in the 3G charging system to tear down suspicious or malicious flows for mobile users. The Skype or other VoIP implementation (to release access information without explicit user confirmation) is exploited to mount this attack. Once you accept invitations from strangers or your buddies are compromised, you are vulnerable to this overcharging attack whenever you go online.

• **Spam attack in video streaming** Other channels exist to launch stealth spam attack in UDP-based services. Video streaming is another example. To trap the victim, the attacker can create a malicious link to redirect Web-browsing operations to start a realtime video streaming. For example, the victim may click one phishing link which redirects the victim's browser to:
rtsp://.*.1.204:554/trackID=5*,

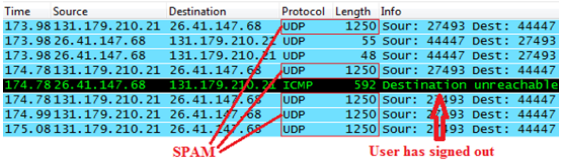


Figure 16: Wireshark traces at the victim after logout from Skype.

where RTSP is a network protocol to support video streaming [31]. Once the link is clicked, the victim automatically starts a new RTSP (over UDP) session running on port 554 and releases its confidential access information to the attacker. Once completed, the attacker blasts spam packets. We implement this attack and test it. We find that it performs similarly to Skype spam attack since both run on the top of UDP. We omit it due to lack of space. In both cases, UDP-based spam can inject an arbitrarily large volume of traffic and force the UE to pay more.

In summary, we have demonstrated that the stealthy spam attack is a real threat to mobile users. The attack is rooted in the inherent loopholes in the current charging architecture. Unless these loopholes are fixed, mobile users may always be victims when the stealth attack or more sophisticated attacks built on it are launched. On the other hand, the good news for mobile users is that, there is no obvious and strong incentive for attackers to launch such attacks now. Attackers cannot have immediate gains for themselves, unless an ill-intentioned operator contracts hackers to attack its own users for larger revenue gain or attack users in its competitor's network for unexpected user complaints, or a disgruntled attacker uses it to incur large monetary loss against his adversary. However, we quickly admit that incentive is an independent and interesting topic to study. Attackers may come up with unexpected incentives to launch more sophisticated attacks in this category in the future.

5.4 Remedy for Architecture Weakness

The fundamental problem underlying the stealth spam attack is that, there is no feedback mechanism from the UE to the carrier's charging system. So the operator cannot block unwelcome traffic based on the UE's feedback. This is an inherent design limitation in the current 3G/4G charging system. The IP-based push model makes spam attack easy. Any one can send to the UE without prior consent. Given the current architecture weakness, a viable charging system must have the following three components: (1) The mobile user himself must be aware of such potential attacks and apply precaution measures. He can simply limit the size of any automatic downloaded data (such as email fetching); (2) The UE must be able to detect unwanted traffic and send feedback. The current protocols at the network layer and the transport layer are designed with such feedback. However, many applications ignore unwanted data (e.g., Skype does so) in general. This has to be fixed to make them suitable to run over a metered charging service; (3) The carriers must take feedback from the UE to stop unwanted traffic.

Specifically, regarding the feedback mechanisms from the UE, we propose three solution options: *implicit-block*, *explicit-allow* and *explicit-stop*. The *implicit-block* solution is to enforce the CN components such as GGSN and NAT boxes. It uses implicit hints from the UE to justify whether the ongoing traffic is welcome or not. Once the traffic is unwanted, the CN blocks this flow and stops charging. The key issue is what messages can serve as hints on whether the UE's data packets are still wanted or not. For TCP-based service, TCP-RESET messages are sent from the UE if the the corresponding TCP connection is torn down earlier by the UE. Our study shows that, mobile Web browsers start to send TCP RESET messages upon receiving unintended TCP packets one-minute after they send the FIN signals. In case of UDP-based service, the UE responds a *ICMP Port Unreachable* message to the external sender upon receiving UDP packets on those closed ports. Therefore, messages of TCP-RESET and ICMP-Port-Unreachable can serve as the hints for the CN. To make correct decision, the CN can further exchange this information with the UE and seek confirmation from the mobile user. Using these implicit feedbacks, the CN should effectively disable the suspicious flows delivered to the UE. A downside of this solution is that, it takes effects only if the UE explicitly tear downs the service (e.g., quitting an application, terminating a TCP connection).

In the *explicit-allow* remedy, the UE explicitly specifies which packets are anticipated by adding/modifying the Packet Filters of Traffic Flow Template (TFT) associated with its PDP context. It can be done using *MS-Initiated PDP Context Modification Procedure* defined by 3GPP [15]. The attributes of the packet filter include [15]: (1) remote address; (2) local address; (3) protocol number, i.e., IPv4; (4) local port range; and (5) remote port range, etc. By adding packet filters, the UEs may not suffer from large spam attack when they are trapped or cheated to receive unexpected packets. One possible downside is that, it requires the UE to be fully aware of what it intends to send/receive. It requires detailed domain knowledge on various applications and services.

The *explicit-stop* solution is to provide explicit feedback from the UE to the carrier when closing some data services. Once the phone detects that there exists any malicious or suspicious flow, it immediately reports to the core network and asks to block such a flow. The spam flow can be detected by mobile anti-malware software, or identified by mobile applications or systems software (e.g., an exception is issued when the application layer or a lower layer in the protocol stack discards a large number of packets). Malicious attackers can also be detected through the collaboration of many phones [20]. This solution framework is flexible enough to integrate with different detection options. It also allows for the UE to stop data charging at any time, even when the UE was cheated or unaware of the attack at the start of the service. Its downside is that, current 3G/4G standards do not offer such mechanisms.

6. RELATED WORK

In recent years, security analysis on mobile devices has been an active research area (see [23, 24, 26, 36] for a few samples of the early work). Most of these studies focus on various types of mobile malware on various platforms of iOS, Android and Symbian, including virus [20, 26], spams such as SMS and making premium calls [23], DoS attacks [21, 27, 35], phishing [36], and privacy intrusion [24], etc. [33] has explored that unwanted traffic can cause large-scale wastage of logical resources in cellular networks. Our work uses real experiments to demonstrate that unwanted traffic can be cast to mobile victims and increase their payment. Certain types of these mobile malware such as viruses and SMS/MMS spams can also be used to incur overcharging attack as a byproduct. Despite these early efforts, security assessment of accounting system in the 3G/4G cellular networks remains a largely unaddressed topic. In this work, we provide the first experimental study that assesses the vulnerability, as well as new practical attacks, on the 3G/4G accounting system. We expose limitations in its charging architecture and loopholes in its policy practice. Both types of attacks described in this paper are also novel in 3G/4G security research.

Despite the popularity of 3G/4G data services, mobile data charging research (including pricing, accounting, billing) is still in its infancy. [22] provides a nice tutorial on pricing, charging, and billing methods for 3G systems up to 2005. [34] offers recent sur-

vey on pricing models, which are orthogonal to the accounting issue studied in this paper. [30] studies various cases of overcharging and undercharging in 3G networks but not from the security perspective. Finally, we note that several tools such as iodine, dns2tcp and NSTX [8] have been designed to circumvent data charging by wired Internet service providers. They are similar to our toll-free-data service approaches in principle, and we show that such ideas also work in wireless cellular networks.

7. CONCLUSION

The Internet is going wireless and mobile. Two driving forces for this trend have been the explosive growth of smartphones and the rapid deployment of 3G/4G infrastructure. Unlike the wired Internet, cellular networks have implemented usage-based charging, rather than the simpler flat-rate charging. The 3G/4G standards stipulate the accounting architecture, yet provide freedom for carriers to define their own charging policy. In this work, we conduct experiments on operational 3G networks to study the security implication of such an architecture and practice. We have discovered loopholes and showcased simple attacks, which are validated by experiments over two operational 3G networks.

Our study yields some insights. On the policy side, differential charging seems to be a popular practice for mobile data services. Given a metered charging system, people necessarily have incentives to exploit and abuse any transfer that is free. There is no simple, bullet-proof solution except eliminating the free service. In the more general problem setting, as long as differential charging exists among applications and services, attackers have incentives to abuse transfers that charge less. The free service simply exemplifies an extreme case. While the toll-free-data attack seems to be readily fixed, we believe that more fundamental issues need to be addressed in the long run. The current 3G/4G accounting architecture lacks proper validation and verification on the traversing traffic types and content, when offering differential charging for applications. The scalability of the associated security design also needs to be considered because of the increasing traffic diversity and volume, as well as the large user population. On the architecture side, the charging system records the data volume on behalf of users, but does not take any user feedback when making charging decisions. So the carrier cannot block unwelcome traffic by using feedback from users. The IP-based push model makes spam attack easier. Anyone can send to the UE without prior consent. Consequently, as confirmed by our experiments, victims may be charged for what they never anticipate, and attackers get data services they never pay.

Given the current architecture weakness, a dependable, usage-based charging system calls for concerted renovations among the network, the mobile device, and applications. The mobile user himself must be aware of such threats and apply precaution measures. The UE must be able to detect unwanted traffic and send feedback. Many applications lack such feedback mechanisms and simply ignore unwanted data, e.g., in the case of Skype. This must be fixed to make them suitable to run over a metered charging service. The operators must take feedback from the UE to stop unwanted traffic, and such feedback has to be carefully validated. The network also needs appropriate traffic validation and verification when making differential charging decisions for different applications and services. This work describes our current effort along this direction. We hope our preliminary study will stimulate further research on this important topic from both academia and industry.

Acknowledgment

We greatly appreciate the insightful comments and constructive feedback from the anonymous reviewers.

8. REFERENCES

[1] AndFTP - LYSESOFT, v2.9.8. http://www.lysesoft.com.
[2] DNSSEC. http://www.dnssec.net/.
[3] Fast and Free Facebook Mobile Access with 0.facebook.com. https://www.facebook.com/blog/blog.php?post=391295167130.
[4] Free Fast Public DNS Servers List. http://theos.in/windows-xp/free-fast-public-dns-server-list/.
[5] Free Gprs Mobile Tricks. http://darkwap.mobi/gprs-tricks/Free-Gprs-Mobile-Tricks.
[6] FreeProxy, v4.1. http://www.handcraftedsoftware.org/.
[7] Iodine. http://code.kryo.se/iodine/.
[8] IP Tunneling Through Nameservers. http://slashdot.org/story/00/09/10/2230242/ip-tunneling-through-nameservers.
[9] MobileMe. http://www.apple.com/mobileme/.
[10] NSTX. http://thomer.com/howtos/nstx.html.
[11] Opera Mobile, v12.0.0. http://www.opera.com/mobile/.
[12] ProxyDroid, v2.6.1. https://play.google.com/store/apps/details?id=org.proxydroid.
[13] Skype. http://www.skype.com.
[14] TrafficStats. http://developer.android.com.
[15] 3GPP. TS23.060: GPRS; Service description; Stage 2, Dec. 2006.
[16] 3GPP. TS23.125: Overall High Level Functionality and Architecture Impacts of Flow Based Charging, Mar 2006.
[17] 3GPP. TS32.240:Telecommunication management; Charging management; Charging architecture and principles, Sep. 2006.
[18] 3GPP. TS25.301: Radio Interface Protocol Architecture, 2008.
[19] G. America. Global 3G Deployments UMTS HSPA HSPA+, 2010.
[20] J. Cheng, S. H. Wong, H. Yang, and S. Lu. Smartsiren: virus detection and alert for smartphones. In *ACM MobiSys*, 2007.
[21] W. Enck, P. Traynor, P. McDaniel, and T. La Porta. Exploiting open functionality in sms-capable cellular networks. In *ACM CCS*, 2005.
[22] Z. Ezziane. Charging and Pricing Challenges for 3G systems. *IEEE Communications Surveys and Tutorials*, 7(1-4):58–68, 2005.
[23] A. P. Felt, M. Finifter, E. Chin, S. Hanna, and D. Wagner. A survey of mobile malware in the wild. In *Proceedings of SPSM'11*, 2011.
[24] C. Guo, H. Wang, and W. Zhu. Smartphone attacks and defenses. In *ACM HotNets-III*, 2004.
[25] H. Holma and A. Toskala. *LTE for UMTS: Evolution to LTE-Advanced*. Wiley, 2011.
[26] N. Leavitt. Mobile phones: The next frontier for hackers? *IEEE Computer*, 38(4):20–23, 2005.
[27] P. P. C. Lee, T. Bu, and T. Woo. On the Detection of Signaling DoS Attacks on 3G/WiMax Wireless Networks. *Computer Networks*, 53(15):2601–2616, Oct. 2009.
[28] mobiThinking. Global Mobile Statistics 2012. http://mobithinking.com/mobile-marketing-tools/latest-mobile-stats.
[29] OECD. Nearly Two-Thirds of US Broadband Subscribers are Wireless. http://www.websiteoptimization.com/bw/1012/.
[30] C. Peng, G. hua Tu, C. yu Li, and S. Lu. Can We Pay for What We Get in 3G Data Access? In *ACM MOBICOM*, 2012.
[31] RFC2326: Real Time Streaming Protocol (RTSP), 1998.
[32] RFC5966: DNS Transport over TCP - Implementation Requirements, 2010.
[33] F. Ricciato, P. Svoboda, E. Hasenleithner, and W. Fleischer. On the Impact of Unwanted Traffic onto a 3G Network. In *SecPerU'06*, 2006.
[34] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang. Pricing Data: A Look at Past Proposals, Current Plans, and Future Trends. *CoRR*, abs/1201.4197, 2012.
[35] 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 *ACM CCS*, 2009.
[36] D. S. Wallach. Smartphone security: Trends and predictions. In *Secure Application Development*, SecAppDev, 2011.
[37] Z. Wang, Z. Qian, Q. Xu, Z. Mao, and M. Zhang. An Untold Story of Middleboxes in Cellular Networks. In *SIGCOMM*, 2011.
[38] WireShark. http://www.wireshark.org/.