文章编号:1002-1175(2013)02-0272-06

# 基于动态密钥的 Android 短信加密方案*

李 昭[1,2],王跃武[2†],雷灵光[1,2],张中文[1,2]

(1 中国科学院研究生院,北京 100049;2 中国科学院信息工程研究所信息安全国家重点实验室,北京 100093)

(2012 年 1 月 11 日收稿;2012 年 4 月 16 日收修改稿)

**Li Z, Wang Y W, Lei L G, et al. Android SMS encryption scheme based on dynamic key[J]. Journal of Graduate University of Chinese Academy of Sciences,2013,30(2):272-277.**

**摘 要** 在动态口令的基础上,提出一种基于动态密钥的短信加密方案.该方案通信交互较少,动态加密密钥增加了密码分析攻击的难度,动态认证密钥可以抵抗重放攻击.整个方案全部采用散列和对称加密算法,更适用于资源受限的移动通信系统.在当前主流的手机操作系统 Android 平台上,完整地实现了该方案.实验结果显示,该方案具有良好的实现性能.

**关键词** 短信加密;动态密钥;密钥分配;Android 平台

**中图分类号**:TP309 **文献标识码**:A **doi**:10.7523/j.issn.1002-1175.2013.02.020

## Android SMS encryption scheme based on dynamic key

LI Zhao[1,2], WANG Yue-Wu[2], LEI Ling-Guang[1,2], ZHANG Zhong-Wen[1,2]

(1 *Graduate University, Chinese Academy of Sciences, Beijing* 100049, *China*; 2 *State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing* 100093, *China*)

**Abstract** Asynchrony of short messages(SMS) brings inconvenience to the key distribution. We propose an SMS encryption system based on the dynamic key. The dynamic encryption key increases difficulty in cryptanalysis. The dynamic authentication key resists the replay attack. The scheme uses only hash function and symmetric encryption which are more suitable to the resource-limited mobile system. Experimental results on Android platform show good realization performance of the scheme.

**Key words** SMS encryption; dynamic key; key distribution; Android platform

目前,短信已经成为人们在日常生活和工作中交流沟通的重要通信方式之一.随着短信使用的深入,人们对其加密功能的需求日益迫切.与语音业务和大多数网络业务不同,短信业务是一种异步的通信方式.这就给其加密功能的密钥协商机制实现提出了挑战.已有的短信加密技术密钥分配机制的实现多建立在公钥体制之上[1-6].这些短信加密方案相对于短信业务本身消耗了较多的计算和通信资源,不利于其在手机系统中的部署和应用.为此有必要提出和实现一种轻量级的具有足够安全性的适用于手机系统的短信加密方案.

* 国家自然科学基金(61003273)和信息安全国家重点实验室自主研究课题(2010-13)资助

† 通信作者,E-mail:wangyuewu@iie.ac.cn

借鉴动态口令认证技术[7-8],本文提出一种短信加密密钥分配方案,并在当前流行的 Android 手机系统上实现了一套完整的短信加密系统,验证了该方案的有效性. 动态口令通过在认证端和服务器端同步地进行口令更新,实现认证口令的实时变化,使得每次的认证口令都不一样,从而将敌手攻击时间有效地压缩,提高认证的安全性. 据此原理,本文提出短信的收发双方通过通信运营商网络系统时间进行时间同步,维持短信加密密钥的实时更新,并引入时间延时误差容忍机制. 使得每次短信加密都使用不同加密密钥,从而大大提高了短信加密系统的安全性,同时避免了复杂的协商过程. 因此该方案更加适用于异步通信的短信加密应用. 动态密钥的生成,除通信双方必须保持时间同步外,他们还需要维护一个共享的初始种子. 由安全的散列算法的不可逆性和散列输出动态密钥的实时变化性,保证该初始种子可以在一段时间内重复使用,并且定期采用多种方式进行分配更新. 本文采用有中心的密钥分配架构[9-12]实现该初始种子的分配和定期更新,在系统实现上为满足用户不同的需求也实现了 DH[13]临时协商和手动设置初始种子的密钥分配方案.

# 1　动态密钥原理

## 1.1　手机短信传输的特性

短信通信是一种类似邮件的异步通信方式. 在发送短信之前,发送方不要求对方是可以接通的. 因此在短信发送之前进行实时的临时密钥协商是不可取的. 所以大多数语音加密采用的临时密钥协商方案[14-15]不适用于短信加密系统. 此外短消息的计费方式也使得临时协商密钥的方案不适用于短信加密. 语音通信按通信时长计费,密钥协商的时长相对于通话时长要少很多. 而短信计费是按照发送短信的条数计费,为了发送一条加密短信而使用数条短信进行密钥协商将会大大增加加密功能的成本. 同时也使得为发一条短信而要等多条临时密钥协商交互短信过程结束,造成很大的时间延迟. 虽然采用预埋密钥的方法,可以避免密钥协商的复杂交互过程,但是却会造成多条短信密文采用同一个密钥加密,大大降低加密系统的安全性. 如何在尽可能少的密钥协商交互条件下,实现较高安全性的短信加密系统,是本文要解决的核心问题.

## 1.2　短信 PDU 协议及加密短信协议栈

对于移动端之间的通信,只需要调用 Android 系统 API,加密数据的收发由 Android 系统自动处理. 同时也可以调用 Android 系统的 API 拿到来信的号码和发信的时间戳. 但是密钥分配服务器的架构为计算机控制无线 modem 进行短信收发. 这样服务器端计算机与 modem 通信的 AT 指令,PDU 数据报文[16]的组织和收发就需要编程实现. PDU 协议与其他通信协议一样由协议头和协议体(短信内容)两部分组成. 协议头的 SCTS 值是发送者将短信刚发到短信息中心时打的时间戳. 这对动态密钥时间同步机制非常有利. 协议体部分在国内主要有 7 bit、UCS2 两种编码方式. 7 bit 编码发送纯普通字符,UCS2 编码可以发送汉字. 7 bit 协议的数据部分可发送 140 个 ASC 字符. 本文协议设计以尽量减少通信量为原则,密钥分发协议中的每条短信通信量在 60 个 ASC 字符左右,所以一条普通短信足够分发密钥使用. 而公钥类短信加密方案通常需要一个证书分发机制,所以需要使用长短信来分发证书.

安全协议部分是顶层的应用协议. 短信 PDU 属于数据传输协议,所以可以利用该传输协议提供的收/发号码和短消息中心时间戳 SCTS 为安全协议的功能服务,从而减少安全协议层的消息量. 接收方可以利用该时间戳进行认证和解密密钥的计算. 安全协议层对 PDU 层是透明的,传输的加密内容是 PDU 层的数据,所以运营商并不知道是在传输加密消息. 这使得系统的搭建更加灵活方便,并且使加密方案与运营商无关,提高了方案的兼容性.

## 1.3　基于动态密钥的短信加密

当前绝大多数的智能手机由应用部分(AP)和基带部分(BP)组成. AP 通过串口向 BP 发送 AT 指令对 BP 进行控制. 短信数据就是作为 AT 指令的数据由 AP 发送给 BP,进而在移动通信网络中进行传输. 短信加密不对发送短信的 AT 指令进行任何改动,而仅仅对短信的数据部分进行改动,即在 AP 中通过动态密钥机制对短信进行加密后,传送给 BP 进行发送. 终端数据库中保存一对初始种子,分别用来加密和认证. 短信加密软件将当前通信运营商网络系统时间提取出来,使用终端保存的用来加密的初始种子和该时间通过散列算法进行加密密钥的动态生成,然后调用加

密算法对编辑的短信数据进行加密. 同时通过另一份用来认证的初始种子和该时间通过散列算法进行认证密钥的动态生成,然后调用 HMAC 算法生成消息认证码附于加密密文之前. 之后 AP 通过 AT 指令,将加密后的整个 mac 和密文作为数据,进行正常的短信传输. 在接收端 AP 接到短信密文后,从短信 PDU 报文头中提取对方本地通信运营商网络时间戳. 用其保存的一对加密和认证初始种子和该时间戳散列生成动态的认证密钥和解密密钥,进行短信数据的认证、解密. 动态认证密钥机制可以有效地抵抗重放攻击. 动态加密密钥机制增加了加密密钥的不确定性,从而增加了敌手的攻击难度.

在动态密钥生成中通信双方需要共享一对加密和认证的初始种子. 为了进一步提高加密系统的安全性和方便信任关系的建立和管理,需要对共享初始种子的建立、更新等进行有效的管理. 为此,方案引入了初始种子分配中心. 该中心通过短信与通信双方进行联系,分配和管理生成动态密钥的初始种子. 中心与终端之间的管理短信采用特定的协议格式,以完成管理任务,并且这些短信在终端初始注册时生成的共享秘密密钥保护下进行通信.

短信加密时间变量使用的是通信运营商系统时间. 短信发到运营商本地短消息服务中心后,短消息服务中心为该条短信 PDU 报文头打上时间戳. 该时间戳是短消息中心收到该条短信的时间戳. 接收方利用该时间戳作为解密时间变量解密. 短信从发送端发出到本地短消息中心接收之间有一定的延时,造成该时间戳与加密时间不一致. 该延时随着移动端处理能力的变化和移动端在小区漫游的位置变化而变化. 不同通信运营商网络系统时钟之间也可能存在时间误差. 因而进行动态密钥加密通信时,解密方要有时间误差的容错机制. 如果解密过程中接收方通过尝试多个时间点发现动态密钥的生成时间与短信时间戳存在时间误差即时间漂移,那么接收方记录下该时间漂移量作为过往时间漂移量. 下次再进行加密通信的时候,接收方以短消息中心时间戳加上数据库中存储的过往时间漂移量为修正后的时间变量值,并以该时间为中心开始计算动态密钥并尝试认证和解密. 每次在该时间中心基础上加入一个时间漂移. 其尝试的时间漂移量的顺序为 0,1, -1,2, -2,…,7, -7 s. 即误差容忍阈值为 7 s,该阈值由后文的试验数据确定. 如果产生了新的时间漂移量,那么就将其与过往时间漂移量相加,保存作为新的过往时间漂移量. 该方法实现了时间漂移的自动修正和修复功能. 一般情况下时间误差不会超过预先设定的阈值. 本文实验结果最大时间误差为 ±4 s,实际使用该系统可以另行测定. 接收方验证,首先通过多次尝试临近的时间点计算认证密钥进而计算 mac 值与接收到的 mac 比对. 若有一个 mac 值比对一致则认证通过. 认证通过即可使用该时间点生成解密密钥解密消息. 时间漂移量超出时间冗余阈值的概率极小,认证比对不通过一般情况下是敌手的攻击行为造成.

## 2 系统设计

### 2.1 基于动态密钥的短信加密系统架构

参与通信的双方 A、B 要在初始种子管理中心 S 进行注册. 通过注册,中心与通信参与者共享了一个秘密. 如果中心在移动服务提供商管理之下,则该注册过程可以与移动通信服务的注册过程相结合,例如共享的秘密可以是 SIM 卡的认证密钥等. 本方案采用注册时随机生成共享秘密的方式,在初始种子管理中心和通信终端之间建立一个随机的共享秘密. 为进一步利用动态密钥的原理提高系统安全性,初始种子管理中心会定期发送管理消息,进行共享秘密更新管理. 共享秘密的更新在前一个共享秘密的保护下进行,是一个从 S 到通信终端的单向操作. 由于新旧共享秘密之间存在联系,从理论上不能彻底防止共享秘密泄露的威胁,但是可以在一定程度上延缓敌手攻击时间,减少敌手收集到的有效明密文对. 确认共享秘密泄露后,通信终端必须到 S 重新注册.

### 2.2 信息格式

中心服务器向移动端发送的短信有应答初始种子和共享秘密管理两类. 所以设服务器发往移动端的协议标识符为 *flag*. *flag* = 0 表示共享秘密管理信息,*flag* = 1 表示应答初始种子信息. 移动端发往服务器的短信只有请求初始种子一种类型. 定义各信息类型短信息格式如下.

#### 2.2.1 共享秘密信息管理

为增强系统的安全性,初始种子管理中心定期更换与各个通信终端的共享秘密,更换信息在原共享秘密的保护之下进行,具体格式如下:

$$message = mac + flag + \mathrm{Enc}(dynKey, newSecret + newSecretForMac), \quad (1)$$

$$dynKey = \mathrm{Hash}(oldSecret + times),$$

$$dynKeyForMac = \mathrm{Hash}(oldSecretForMac + times),$$

$$mac = \mathrm{Hash}(flag + \mathrm{Enc}(dynKey, newSecret + newSecretForMac) + dynKeyForMac). \quad (2)$$

用于动态生成加密密钥的原共享秘密 *oldSecret* 与信息发送时间 *times* 相结合,在 Hash 算法的作用下,生成 *dynKey*,用对称加密算法 Enc 对新的共享秘密 *newSecret* 加密. 用于动态生成认证密钥的原共享秘密 *oldSecretForMac* 与信息发送时间 *times* 相结合,在 Hash 算法的作用下,生成 *dynKeyForMac*. 由公式(2)所示,消息内容 *flag* + Enc(*dynKey*, *newSecret* + *newSecretForMac*) 和认证密钥 *dynKeyForMac* 通过散列算法生成 *mac* 值. 标记 *flag* = 0 表示信息类型为共享秘密管理信息,以区别于其他信息. 通信终端收到 *message* 后,可以获得新的 *newSecret* 和 *newSecretForMac* 值. 之后初始种子管理中心和终端同时将 *newSecret* 代替 *oldSecret*, 将 *newSecretForMac* 代替 *oldSecretForMac*.

### 2.2.2　初始种子管理信息格式

通信终端发起一个加密短信通信时会首先查看有没有保存与通信目标方共享的有效初始种子,如果有,则可以生成动态密钥,进行短信加密通信. 如果没有,则需要向初始种子管理中心申请初始种子,消息格式如下:

$$message = mac + dest, \quad (3)$$

$$dynKeyForMac = \mathrm{Hash}(secretForMac + times),$$

$$mac = \mathrm{Hash}(dest + dynKeyForMac). \quad (4)$$

*dest* 表示通信的对方标识, *secretForMac* 是通信终端与初始种子管理中心共享的秘密. *times* 表示信息发送的时刻. 在 Hash 算法的作用下得到散列值 *mac*. 初始种子管理中心根据 *mac* 可以验证请求消息的真实性和新鲜性. 收到 *message* 后,中心生成初始种子,并赋予该种子一个有效期. 到有效期时,中心会自动更新初始种子. 初始种子请求回应和更新都采用下面的信息格式,发送给加密通信申请者和目标方,消息格式如下:

$$message = mac + flag + dest + \mathrm{Enc}(dynKey, initialSeed + initialSeedForMac), \quad (5)$$

$$dynKey = \mathrm{Hash}(secret + times),$$

$$dynKeyForMac = \mathrm{Hash}(secretForMac + times),$$

$$mac = \mathrm{Hash}(flag1 + dest + \mathrm{Enc}(dynKey, initialSeed + initialSeedForMac) + dynKeyForMac). \quad (6)$$

*flag* = 1 表示该消息是初始种子请求的回应消息(如 2.2.1 节所述,*flag* = 0 为更新共享秘密 *secret* 和 *secretForMac* 的共享秘密管理消息,此时不涉及通信对端,(5)、(6)式中没有 *dest* 项),或者初始种子的更新消息,*dest* 为对端的标识,对于短信加密通信的接收者来说,就是该种子申请者的标识,*initialSeed* 和 *initialSeedForMac* 为初始种子分别在通信中用来生成动态密钥和动态 MAC,初始种子的值由中心生成,Enc 代表对称加密算法,其余符号同前. 通信终端接收到消息后,提取出 *initialSeed* 和 *initialSeedForMac* 安全地保存在通信终端,通信双发建立起安全连接. 必要的情况下,任一方可以主动地通过中心更新 *initialSeed* 和 *initialSeedForMac*.

### 2.2.3　加密短消息的信息格式

获得初始种子后,发送方 A 采用动态密钥的方式进行加密,信息格式如下:

$$message = mac + \mathrm{Enc}(dynKey, Msg), \quad (7)$$

$$dynKey = \mathrm{Hash}(initialSeed + times),$$

$$dynKeyForMac = \mathrm{Hash}(initialSeedForMac + times),$$

$$mac = \mathrm{Hash}(\mathrm{Enc}(dynKey, Msg) + dynKeyForMac). \quad (8)$$

*Msg* 为发送的短消息,以 *dynKey* 为通信密钥,采用加密算法 Enc 对 *Msg* 进行加密. *dynKeyForMac* 为认证密钥,其余各符号与前相同. B 收到消息后,由于可以从短消息报文头中提取到时间戳且已知 *initialSeed* 和 *initialSeedForMac*,所以可计算出 *dynKey* 和 *mac* 值,对消息进行认证和解密. 由于采用动态密钥的方式,不同时刻消息的密钥均不同,增加了系统的安全性. 同时时间因素引入 *dynKeyForMac* 中,进而引入 *mac* 中,对于 *mac* 值的计算发挥了新鲜值的作用.

## 2.3　关键参数确定

### 2.3.1　*secret*、*secretForMac*、*initialSeed*、*initialSeedForMac*、*dynKey*、*dynKeyForMac* 和 *mac* 值的长度

考虑到手机系统计算资源受限的问题,短信

加密算法 Enc 采用密钥位数为 128 bit 的对称加密算法,具体算法可协商. 所以 2.2.3 节式(8)中加密密钥 *dynKey* 的长度定为 128 bit. 因此,为计算方便,初始种子 *initialSeed* 的长度定为 128 bit,即 128 bit 的随机串.

*mac* 的构造采用 HMAC 体制,即用带密钥的散列函数构造 *mac*. 具体的散列算法可协商. HMAC 的安全性由密钥长度 $k$ 和 MAC 长度 $n$ 共同决定. 认证密钥包括种子 *SecretForMac*、*initialSeedForMac* 和由种子生成的的动态密钥 *dynKeyForMac*,敌手对 HMAC 的攻击代价为 $N = \min(2^k, 2^n)$. $N$ 取大于 128 位的值被认为是安全的. 如前,密钥 *SecretForMac*、*initialSeedForMac*、*dynKeyForMac* 确定为 128 bit,所以 *mac* 的散列值也定为 128 bit.

### 2.3.2 时间变量参数格式

时间变量精度选取过高,将会给动态密钥生成的容错机制带来负面影响,若包含精度过低的部分,在初始种子的有效期内,这些值不会发生改变,会造成不必要的计算量浪费并带来影响安全性的冗余度. 在设计中,将初始种子的有效期定为 30 d,即 30 d 后,中心将会自动更新初始种子. 本文定义动态密钥和 *mac* 值的生成中的时间变量 *times* 的格式为日、时、分、秒. 时钟同步系统误差冗余度阈值的设置既要保证区间大小足够充分,使得解密不会因为正常的时间误差导致失败. 冗余度也不能设置过大,过大的冗余度给敌手更大的攻击空间. 本文结合大量实验测得的实际时间延迟误差将实际数据误差冗余阈值设为 ±7 s.

## 3 实验

我们在多普达 A6388 手机,搭载通用 Android1.6 系统的平台上实现了该终端短信加密系统. 初始种子管理中心由通用计算机加无线 Modem 共同组成. 系统采用英特尔双核酷睿 2 处理器,主频 2.66 GHz,内存 2.00 GB 的 PC 加西门子 MC37i GSM 模块共同组成. 操作系统为 32 位 Win7 系统,数据库为 SQLserver 2008. 通用计算机和 Modem 之间通过串口连接. 运行在通用计算机上的管理中心程序采用 AT 指令对无线 Modem 进行操控.

如果动态密钥时间同步系统不可靠,时间漂移量尝试过多,则会造成较大的处理延时. 并且如果尝试次数超过最大容忍阈值( ±7 s)则会导致解密失败. 实验通过统计解密尝试的时间漂移次数来证明系统时钟同步机制的可靠性和系统总体性能的稳定性.

更新初始种子和更新共享秘密的算法是相同的. 由表 1、表 2 可知,移动端尝试时间漂移次数平均不大于 5 次,原始实验数据最大一次时间漂移尝试次数为 8 次即误差 ±4 s. 由此可见时钟同步系统误差冗余阈值设置为 ±7 s 是可靠的.

**表 1 解密来信实验数据统计**

**Table 1 Experimental data of decryption of the incoming SMS**

| 实验次数 | 成功次数 | 验证 mac 尝试时间漂移平均次数 |
|---|---|---|
| 30 | 30 | 4.40 |

**表 2 更新初始种子实验数据统计**

**Table 2 Experimental data of seed updating**

| 实验次数 | 成功次数 | 验证 mac 尝试时间漂移平均次数 |
|---|---|---|
| 30 | 30 | 1.93 |

信息加密耗时见表 3. 加密包括动态密钥的生成、AES 加密、SM3 散列算法计算 mac 值和信息组包的全部过程,解密为加密过程的逆过程. 解密过程耗时见表 4.

**表 3 加密短信息耗时**

**Table 3 Consuming time of encryption of SMS**

| 加密消息量 | 实验次数 | 成功次数 | 加密信息平均耗时/ms |
|---|---|---|---|
| 30 个汉字 | 30 | 30 | 9.12 |
| 100 个汉字 | 30 | 30 | 13.06 |

**表 4 解密短信息耗时**

**Table 4 Consuming time of decryption of SMS**

| 解密出的明文量 | 实验次数 | 成功次数 | 解密信息平均耗时/ms |
|---|---|---|---|
| 30 个汉字 | 30 | 30 | 15.67 |
| 100 个汉字 | 30 | 30 | 20.88 |

从表 3、表 4 我们看到所有密码运算平均耗时均在 18 ms 左右,偶然数据最大不超过 100 ms. 这表明对称加密系统是高效的,能够满足移动通信终端的性能需求.

## 4 总结与展望

本文针对短信通信的异步交互特点,在动态口令技术的基础上,提出了一种基于动态密钥的短信加密方案. 该方案可以在尽量少的通信交互

的条件下,实现高安全性的密钥协商效果,解决了短信加密中密钥分配的困难. 本文在当前主流平台上对方案进行了具体实现,并通过实际实验对方案的性能进行了分析,结果显示该方案具有良好性能,可以满足实际需要.

移动终端需要保存大量的秘密信息. 这些信息的安全存储直接关系到系统的安全性. 目前我们主要采用 Android 已有的安全机制进行安全存储. 在后续的研究中,本文将进一步对 Android 系统的安全机制进行研究,对其安全存储进行加固,提高系统安全性.

## 参考文献


[1] Agoyi M, Seral D. SMS security: an asymmetric encryption approach [C] // 2010 Sixth International Conference on Wireless and Mobile Communications. 2010, 8(10): 448-452.

[2] Hasan S, Al-bakri M L, Kiah M. A novel peer-to-peer SMS security solution using a hybrid technique of NTRU and AES-Rijndael[J]. Scientific Research and Essays, 2010, 5(22): 3455-3466.

[3] Hasan S, Al-Bakri M L, Kiah M, et al. Securing peer-to-peer mobile communications usingpublic key cryptography: new security strategy[J]. International Journal of the Physical Sciences, 2011, 6(4): 930-938.

[4] Hassinen M, Markovski S. Secure SMS messaging using Quasigroup encryption and Java SMS API[C] // SPLST'03. Finland, 2003.

[5] Hassinen M. Java based public key infrastructure for SMS messaging, information and communication technologies[C] // ICTTA'06. 2006.

[6] Buckingham S. Shortmessage peer to peer protocol specification[EB]. SMPP Developers Forum. 1999.

[7] Das M L, Saxena A, Gulati V P. A dynamic ID-based remote user authentication scheme [J]. IEEE Transactions on Consumer Electronics, 2004, 50(2): 629-631.

[8] Raihi D M, Bellare M, Naccache D, et al. An HMAC-based one-time password algorithm[S]. RFC 4226, HOTP, 2005.

[9] Popek G, Kline C. Encryption and secure computer networks [R]. ACM Computing Surveys, 1979.

[10] Needham R, Schroeder M. Using encryption for authentication in large networks of computers [R]. Communications of the ACM, December 1978.

[11] Denning D. Timesstamps in key distribution protocols[R]. Communication of the ACM, Augest 1981.

[12] Kehne A, Schonwalder J, Langendorfer H. A nonce-based protocol for multiple authentications[R]. Operating Systems Review, 1992.

[13] Diffie W, Hellman M. Multiuser cryptographic techniques [C] // AFIPS Conference Proceedings. 1976, 45: 109-112.

[14] Secure Voice GSM. Are your cell phone conversations secure? [EB/OL]. [2011-11-30]. http: // www. securevoicegsm. com.

[15] Rohde & Schwarz. TopSec mobile: secure voice encryption for smartphones and laptops[EB/OL]. [2011-11-30]. http: // www2. rohde-schwarz. com/product/TopSec% 20Mobile. html.

[16] GSM Technical Specification. Digital cellular telecommunications system (Phase 2+): Alphabets and language-specific information (GSM 03.38) [EB/OL]. [2011-11-15]. http: // www. dreamfabrie. com/sms/.