初始数据：

50020232A00000033301010162141573129000251 65F0000020000000000000700010 3A020000 10A0100000019D03845AFB0D773000000010000000000000000015600000000000156111226994909329 87C00000203A02000300921000000000000000000010000000000100000000000000000

卡号：6214157312900025165

卡序列号：00

ATC：0002

74 位数据源：

写卡充值脚本：

72199F18040000000 1861004DA9F790A000000010000F5AB5B1300

主密钥：

WK 索引 00

MDK: A46B717318EED28A2B5DB38FCC28FE6E

MDK-ENC: 6AAF8070BE0185BD78B114CB64CF476A

MDK-MAC: CFE31C8FEC62792BCB1A5D41A2442166

KMU: 027B7367D6A36C6C51F2A07D091E4D91

1、 计算 ARPC 值

X：=(ARC||‘00’||‘00’||‘00’||‘00’||‘00’||‘00’)。

2) 计算 Y：=ARQC⊕X。

3) 计算 ARPC

基于 128 位分组加密算法获得 16 字节 ARPC

ARPC：=ALG（SKAC）[Y||‘00’||‘00’||‘00’||‘00’||‘00’||‘00||‘00’||‘00]

1.1 用 MDK、Pan、序列号生成卡片分散密钥

这一方式以主账号（PAN）和主账号序列号（如果主账号序列号不存在，则用一个字节“00”代替）的最右 16 个数字作为输入数据，以及 16 字节的发卡行主密钥 IMK 作为输入，生成 16 字节的 IC 卡子密钥 MK 作为输出：

1) 如果主账号和主账号序列号 X 的长度小于 16 个数字，X 右对齐，在最左端填充十六进制的“0”

以获得 8 字节的 Y。如果 X 的长度至少有 16 个数字，那么 Y 由 X 的最右边的 16 个数字组成。

2) 计算 2 个 8 字节的数字

得：5731290002516500

金融 IC 卡是基于 128 位分组加密算法的计算方法

Z：=ALG(MDK)[Y||(Y⊕(‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’))]

[Y||(Y⊕(‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’||‘FF’))]=5731290002516500A8CED6FFFDAE9AFF；

得到卡片分散密钥：9D70763D423A2A0023265A2AAF503084

1.2 计算卡片过程密钥

第一步：卡片/发卡行决定是使用 MAC 密钥还是数据加密密钥来进行所选择的算法处理。

第二步：将当前的 ATC 在其左边用十六进制数字’0’填充到 8 个字节记为数据源 A，将当前的 ATC 异或

十六进制值 FFFF 后在其左边用十六进制数字“0”填充到 8 个字节记为数据源 B，将数据源 A 和数据源 B 串

联，用选定的密钥对该数据作如图 10 所示的运算产生过程密钥。

Z：=ALG(Key)[[‘00’||‘00’||‘00’||‘00’||‘00’||‘00’||ATC||‘00’||‘00’||‘00’||‘00’||‘00’||‘00’||(ATC ‘FFFF’)] ⊕ ‘FFFF’)]

[[‘00’||‘00’||‘00’||‘00’||‘00’||‘00’||ATC||‘00’||‘00’||‘00’||‘00’||‘00’||‘00’||(ATC ‘FFFF’)] ⊕ ‘FFFF’)]= 00000000000000020000000000FFFD；

卡片过程密钥：BA72C2B2F314489013DD121A61715D52

1、3 计算 X：

4 位授权响应码+12 为 0 填充的，3030000000000000

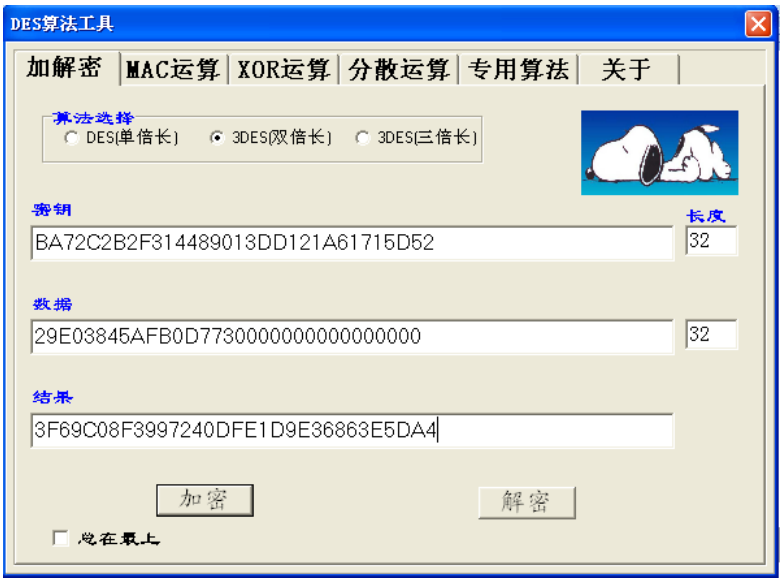
1.4 计算 Y：

ARQC 异或授权响应吗填充 X：

得到 Y：=29E03845AFB0D773

1.5 计算 ARPC

先把 Y:用右 0 填充到 16 个自己做为加密数据；过程密钥做位密钥加密得到 16 个字节 AQPC；



16 字节 ARPC =3F69C08F3997240DFE1D9E36863E5DA4，取前 8 个字节做为发送给 IC 卡 ARPC：3F69C08F3997240D

2.计算 MAC

命令中需要加密的数据加密以后再计算 MAC。MAC 使用对称密钥算法计算的，步骤如下：

步骤 1：初始值为 8 字节全零（此步骤可省略）；

步骤 2：下列数据按顺序排列得到一个数据块 D：

——CLA、INS、P1、P2 和 Lc（Lc 的长度包括 MAC 的长度）；

——ATC（对于发卡行脚本处理，此 ATC 在请求中报文中上送）；

——应用密文（对于发卡行脚本处理，此应用密文通常是 ARQC，或 AAC，在请求报文中上送）；

——命令数据域中的明文或密文数据（如果存在）。

步骤 3：将上述数据块 D 分成 8 字节长的数据块 D1、D2、D3…最后一块数据块的字节长度为 1 到 8；

步骤 4：如果最后一块数据块的长度为 8 字节，后面补 8 字节数据块：80 00 00 00 00 00 00 00，

执行步骤 5；

如果最后一块数据块的长度小于 8 字节，后面补一个字节 80，如果长度到 8 字节，执行

步骤 5。如果仍然不够 8 字节，补 00 直到 8 字节；

步骤 5：用 MAC 过程密钥对数据块进行加密。MAC 过程密钥的生成见 C.4；

图 C.1 是使用 MAC 过程密钥 A 和 B 生成 MAC 的流程图。

步骤 6：MAC 的计算结果为 8 字节，从最左边的字节开始取 4 字节。

2.1 计算安全报文鉴别（MAC）过程密钥

和 1.2 流程一样，只是使用的 MDK-MAC 密钥分散卡片 MAC 子密钥；

得到 MAC 过程密钥 822492068CD484E4E4A5B17CF57ACD43

F5AB5B13922A9512 取左边四个 4 字节 MAC 即为：F5AB5B13

3．计算 ARQC

4. 加密密码数据