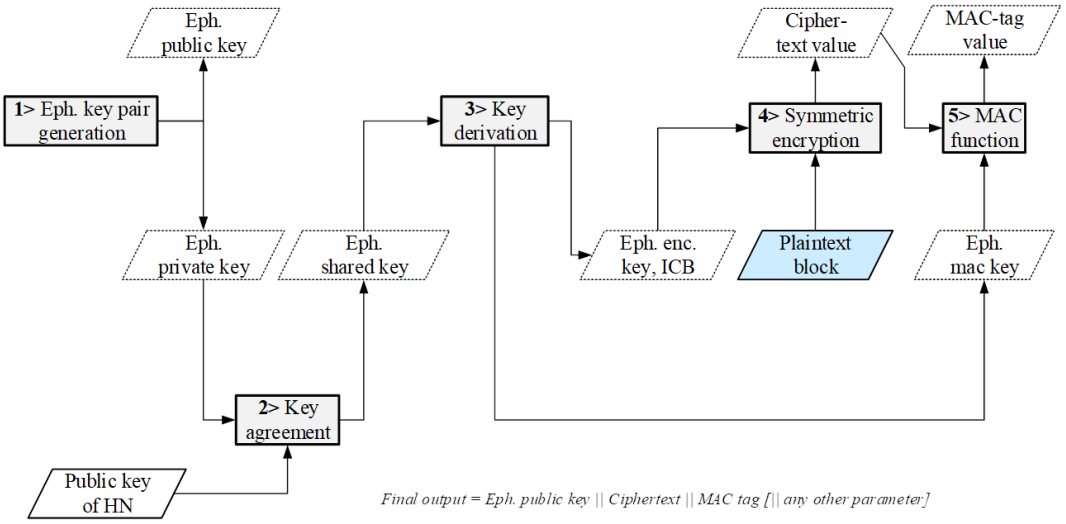
密钥完整生成、加密解密流程

1. 整体流程

1. 加密流程



1. Eph. key pair generation

   终端（UE）通过椭圆曲线加密方案生成一对密钥，其中私钥是一个256位的随机数k，由终端保管；公钥K是以该随机数对基元进行标量乘法，即K=k*G，生成的新的坐标。这里椭圆曲线的参数和基元G的选定是通信双方事先商量好的。基站的一对密钥生成方法类似。

2. Key agreement

   终端获得来自基站的公钥H，用标量乘法乘以自己的私钥k，即R=k*H，获得协商密钥Eph. shared key。协商密钥的位数为256位，尚不满足后续的需求。

3. Key derivation

   使用key derivation function，将输入的256位协商密钥变成128+128+256位。取高字节位（MSB）作为对称加密的密钥Eph. enc. key，低字节位（LSB）作为MAC的密钥Eph. mac key，而中间的128位作为偏置模块ICB。

4. Symmetric encryption

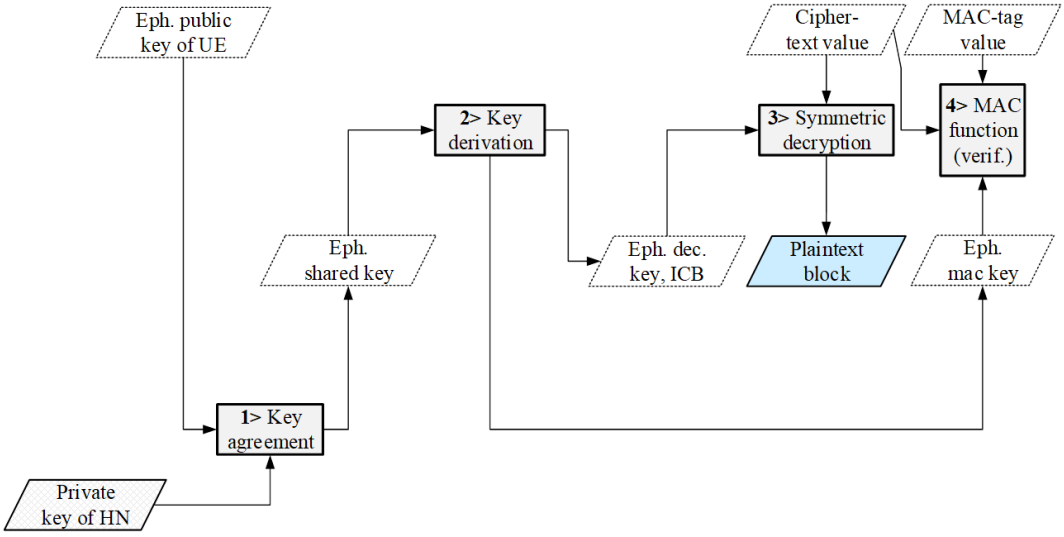
   使用密钥Eph. enc. key对明文Plaintext进行加密得到密文Ciphertext。这里的明文具体就是SUPI，密文具体就是SUCI。ICB是加密时的偏置项。

5. MAC function

   MAC全称Message Authentication Code，类似于数字证书中的标签。对得到的密文Ciphertext以Eph. mac key为密钥做SHA-256，取高64位得到标签MAC-tag value。

6. 最后的输出为终端公钥、密文、MAC-tag连接成的比特串，后面可以跟有可选的其他参数。

2. 解密流程



1. Key agreement

   基站获得来自终端消息，并根据字节分割为的终端公钥，SUCI，MAC-tag。将终端公钥K乘以基站的私钥h，即R'=h*K获得协商密钥Eph. shared key，和终端计算得到的协商密钥相同，即R=R'。

2. 后续步骤与加密流程完全相同。由于协商密钥Eph. shared key相同，在后续Key derivation过程中产生的Eph. enc. key、ICB、Eph. mac key也完全相同。只是MAC function在字节分割后最先执行，若生成的tag与消息中分割得到的tag不同则证明消息被篡改或者有误码，终止验证程序。若tag对应相同，则通过Symmetric decryption对Ciphertext解密得到Plaintext。

2. 技术细节

1. 备选方案

3GPP在33.501中规定了三种可用的加密方案：

> null-scheme　0x0;
>
> Profile　0x1;
>
> Profile　**0x2.**

之后0xC - 0xF 的值保留做运营商自己制定的规范方案。

传输数据尺寸为：

> **null-scheme　size of input, i.e., size of username used in case of NAI format or MSIN in case of IMSI;**
>
> **Profile　total of 256-bit public key, 64-bit MAC, plus size of input;**
>
> **Profile　total of 264-bit public key, 64-bit MAC, plus size of input.**

其中null-scheme即为透明传输，不对SUPI加密；

Profile方案如下：

> - **EC domain parameters　: Curve25519**
> - **EC Diffie-Hellman primitive　: X25519**
> - **point compression　: N/A**
> - **KDF　: ANSI-X9.63-KDF**
> - **Hash　: SHA-256**

- SharedInfo1 : $\overline{R}$
- MAC : HMAC–SHA-256
- mackeylen : 32 octets (256 bits)
- maclen : 8 octets (64 bits)
- SharedInfo2 : the empty string
- ENC : AES–128 in CTR mode
- enckeylen : 16 octets (128 bits)
- icblen : 16 octets (128 bits)
- backwards compatibility mode : false

Profile方案如下：

- EC domain parameters : secp256r1
- EC Diffie-Hellman primitive : Elliptic Curve Cofactor Diffie-Hellman Primitive
- point compression : true
- KDF : ANSI-X9.63-KDF
- Hash : SHA-256
- SharedInfo1 : $\overline{R}$
- MAC : HMAC–SHA-256
- mackeylen : 32 octets (256 bits)
- maclen : 8 octets (64 bits)
- SharedInfo2 : the empty string
- ENC : AES–128 in CTR mode
- enckeylen : 16 octets (128 bits)
- icblen : 16 octets (128 bits)
- backwards compatibility mode : false

具体细节见[1]。

2. **椭圆曲线加密**

这里不具体讨论椭圆曲线加密算法的理论基础，仅讨论方案。

> 数学基础见
>
> - 现代密码学中的数论基础知识梳理
> - 信息安全数学基础(数论)
> - 同余方程总结
> - 快速幂
>
> 加密原理见
>
> - 有趣的椭圆曲线加密
> - ECC椭圆曲线加密算法原理

Profile选择了Curve25519和X25519函数。

其中Curve25519是一条Montgomery曲线，参数如下：

> $v^2 = u^3 + A * u^2 + u$
>
> p $2^{255} - 19$
>
> A 486662
> order 2^{252} + 0x14def9dea2f79cd65812631a5cf5d3ed
> cofactor 8
> U(P) 9

> V(P)
> 14781619447589544791020593568409986887264606134616475288964881837755586237401
> The base point is u = 9, v = 14781619447589544791020593568409986887264606134616475288964881837755586237401.

X25519是对Curve25519上的点进行标量乘法的函数。其输入是<标量scalar，基元的u坐标>，输出是标量乘法产生的新的u坐标。由于所有运算在U域上，不需要point compression方案。

Profile选择了secp256r1曲线，这是一条Weierstrass曲线，参数如下：

> The verifiably random elliptic curve domain parameters over Fp secp256r1 are specified by the
> sextuple T = (p, a, b, G, n, h) where the finite field Fp is defined by:
>
> p = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFF
> = $2^{224}(2^{32} - 1) + 2^{192} + 2^{96} - 1$
> The curve E: $y^2 = x^3 + ax + b$ over F_p is defined by:
> a = FFFFFFFF 00000001 00000000 00000000 00000000 FFFFFFFF FFFFFFFF FFFFFFFC
> b = 5AC635D8 AA3A93E7 B3EBBD55 769886BC 651D06B0 CC53B0F6 3BCE3C3E 27D2604B
> E was chosen verifiably at random as specified in ANSI X9.62 [X9.62] from the seed:
> S = C49D3608 86E70493 6A6678E1 139D26B7 819F7E90
> The base point G in compressed form is:
> G = 03 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296
> and in uncompressed form is:
> G = 04 6B17D1F2 E12C4247 F8BCE6E5 63A440F2 77037D81 2DEB33A0 F4A13945 D898C296 4FE342E2 FE1A7F9B 8EE7EB4A 7C0F9E16 2BCE3357 6B315ECE CBB64068 37BF51F5
> Finally the order n of G and the cofactor are:
> n = FFFFFFFF 00000000 FFFFFFFF FFFFFFFF BCE6FAAD A7179E84 F3B9CAC2 FC632551
> h = 01

方案采用了Elliptic Curve Cofactor Diffie-Hellman Primitive，但由于协因子h=1，因此和其实Elliptic Curve Diffie-Hellman Primitive相同。注意到这里采取了点压缩方案，即将点的二维坐标原本需要32+32字节储存，通过压缩算法压缩为1+32字节。因此Profile传输的公钥会比Profile多一个字节。

3. **点压缩方案**

椭圆曲线上的任一仿射点(x, y)（非无穷远点）都可以压缩成利用其y坐标的最后一比特（记为y'）和x坐标来表示，即(x, y')，这就是点的压缩。反过来，利用(x, y')恢复y坐标，还原仿射点(x, y)的过程就称为点的解压缩。

其原理可以用简单的数论知识解释，即椭圆曲线上对应一个x值的y有且仅有两个，且满足$y_1+y_2=p$，显然y_1和y_2奇偶性不同。

> 具体细节见椭圆曲线点的压缩

4. **密码导出函数**

设置一个4字节的计数器counter，counter从00000001_{16}开始递增。

$$For \quad i=1 \quad to \quad \lceil keydatalen/hashlen \rceil :$$
$$K_i = HASH(Z||Counter||[SharedInfo])$$
$$K = K_1||K_2||\ldots||K_{\lceil keydatalen/hashlen \rceil}$$

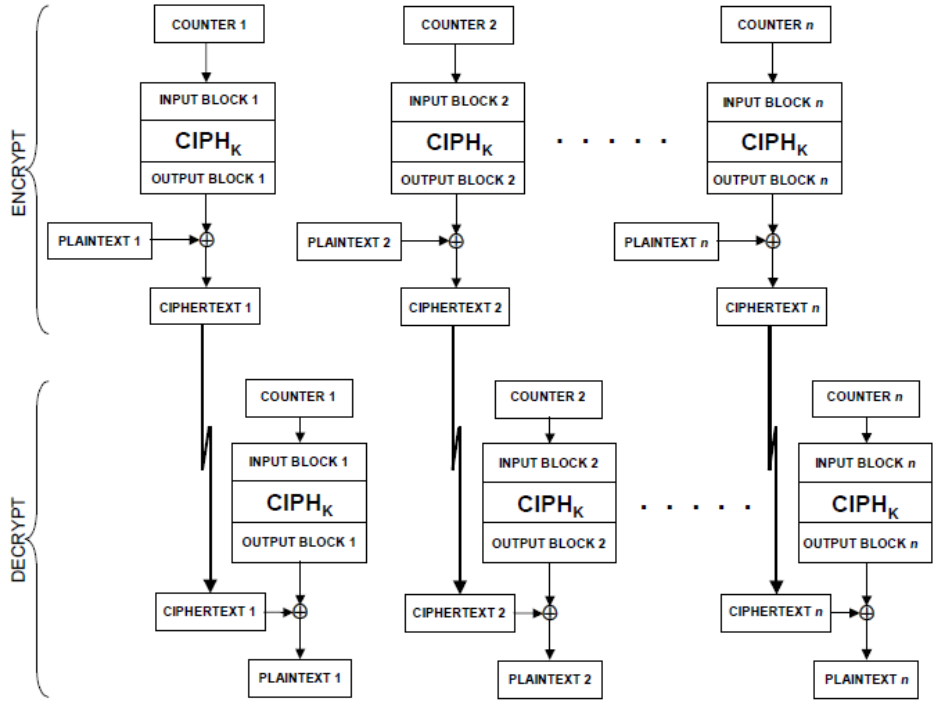
其中HASH函数设定为SHA-256。实际使用时keydatalen=512，hashlen=256,因此K由$K_1||K_2$组成。K_1又分为128位的对称加密的密钥Eph. enc. key，和128位的偏置模块ICB。

> **具体细节见[2]3.6节。**

5. **AES-128 CTR模式**

其中对称加密模块采用128位AES的CTR模式。

其原理图如下：

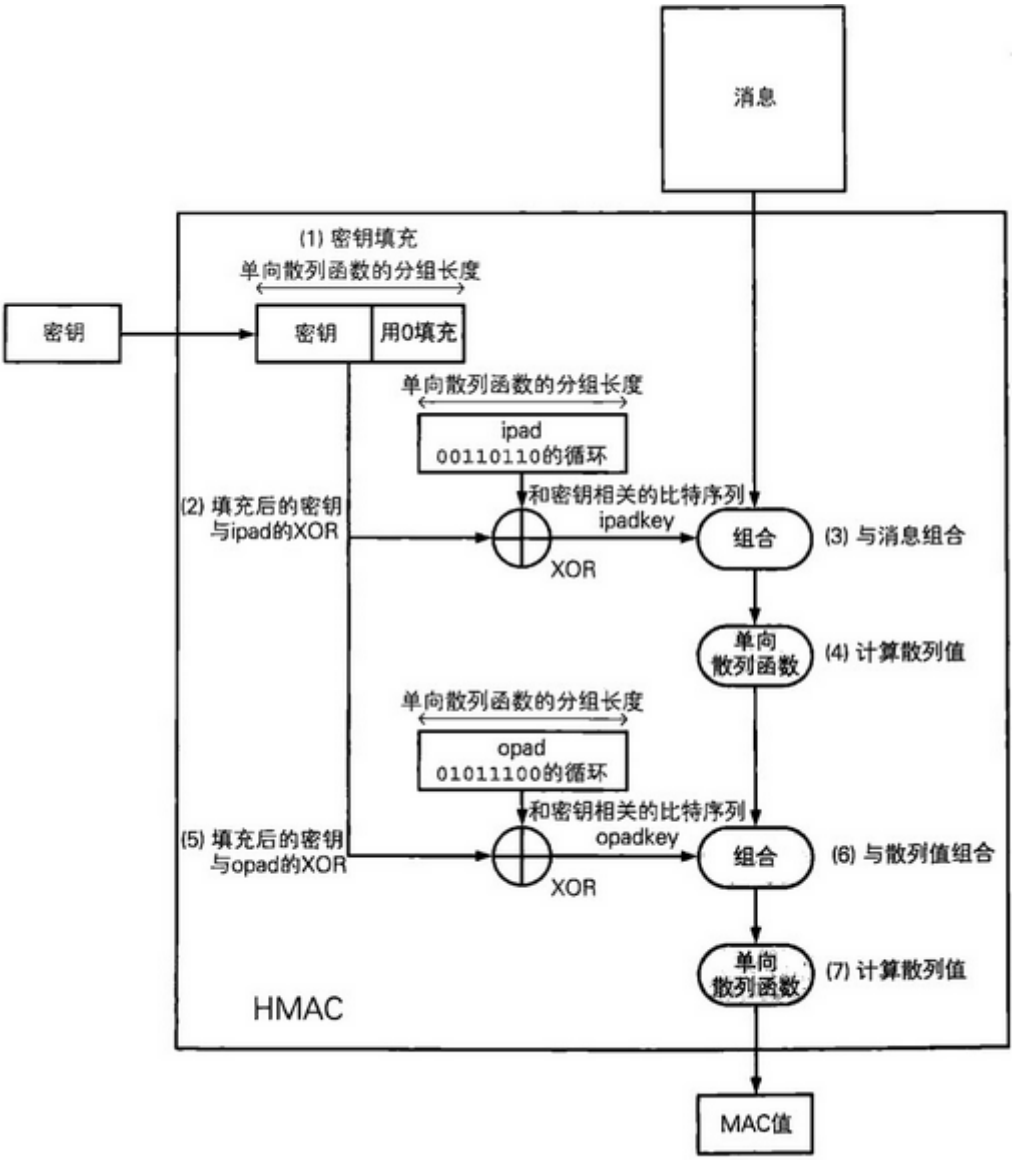


其中原文PLAINTEXT被分割128位一块；COUNTER是计数器模块，COUNTER1就是ICB(Initial Counter Block)，初始值起到偏置的作用，COUNTER每次加1。COUNTER用密钥函数$CIPH_K$加密后与原文块取异或得到密文块CIPHERTEXT。解码时步骤完全相同，两次异或运算重新得到明文。最后一个块中明文长度可能不足一个块长，设为u位，那么其与加密块的前u位取异或生成密文，后面多余的部分抛弃，因此明文长度和密文长度相同。

> **CIPH函数具体细节见[6]、[AES 加密算法的原理详解]**

6. **HMAC-SHA-256与SHA-256**

HMAC流程图如下：



使用单向散列函数实现消息认证码的例子（HMAC）

> HMAC具体细节见HMAC（Hash-based Message Authentication Code）实现原理

> SHA-256具体细节见SHA-2 安全散列算法2 算法详解

3. 函数框架

4. 参考文献

- [1] 3GPP TS 33.501 V15.3.1 (2018-12) Annex C (normative): Protection schemes for concealing the subscription permanent identifier
- [2] nistspecialpublication800-38a Recommendation for Block Cipher Modes of Operation
- [3] sec1-v2 SEC 1: Elliptic Curve Cryptography
- [4] sec2-v2 SEC 2: Recommended Elliptic Curve Domain Parameters
- [5] rfc7748 Elliptic Curves for Security
- [6] fips-197 ADVANCED ENCRYPTION STANDARD (AES)