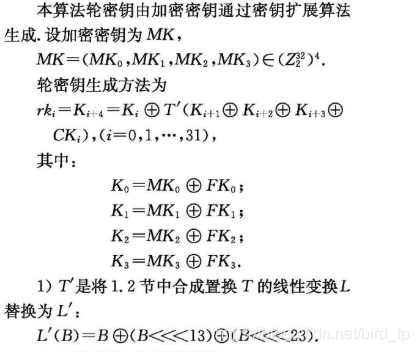
1. **sm4算法详解**

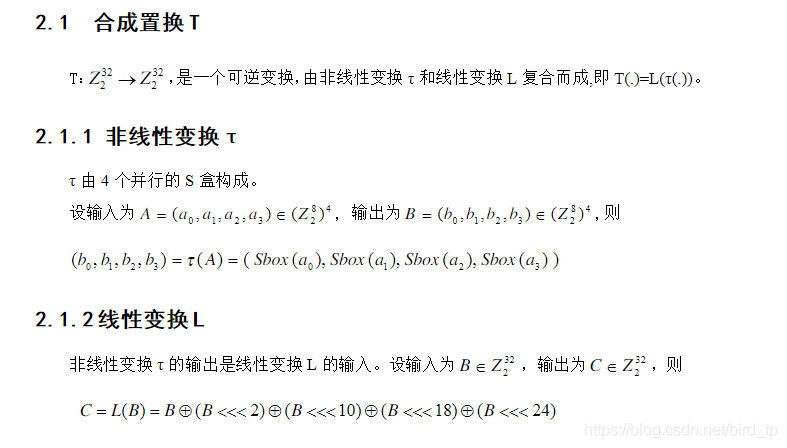
**具体的算法可以查看同文件夹下的网页文件，非常详细，这里提供一个思路和极大特征**

**128位密钥-->密钥扩展算法-->合成置换T函数-->32轮轮函数F**

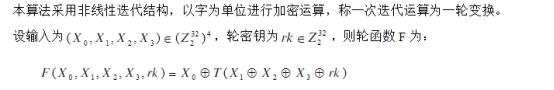
**密钥扩展算法**



**合成置换T函数**



**32轮轮函数F**



其中注意：L和L’函数中是<<<也即循环移位而非单向移位也即

(v3 >> 9) | (v3 << 23)) ^ v3 ^ ((v3 >> 19) | (v3 << 13)和

((v3 >> 8) | (v3 << 24)) ^ ((v3 >> 14) | (v3 << 18)) ^ ((v3 >> 22) | (v3 << 10)) ^ v3 ^ ((v3 >> 30) | (4 \* v3));

**特征**

**1.FK**系统参数FK的取值固定

FK0=(A3B1BAC6)，FK1=(56AA3350)，FK2=(677D9197)，FK3=(B27022DC)

**2.CK**固定参数CK的取值

00070e15, 1c232a31, 383f464d, 545b6269,

70777e85, 8c939aa1, a8afb6bd, c4cbd2d9,

e0e7eef5,  fc030a11,  181f262d, 343b4249,

50575e65, 6c737a81, 888f969d, a4abb2b9，

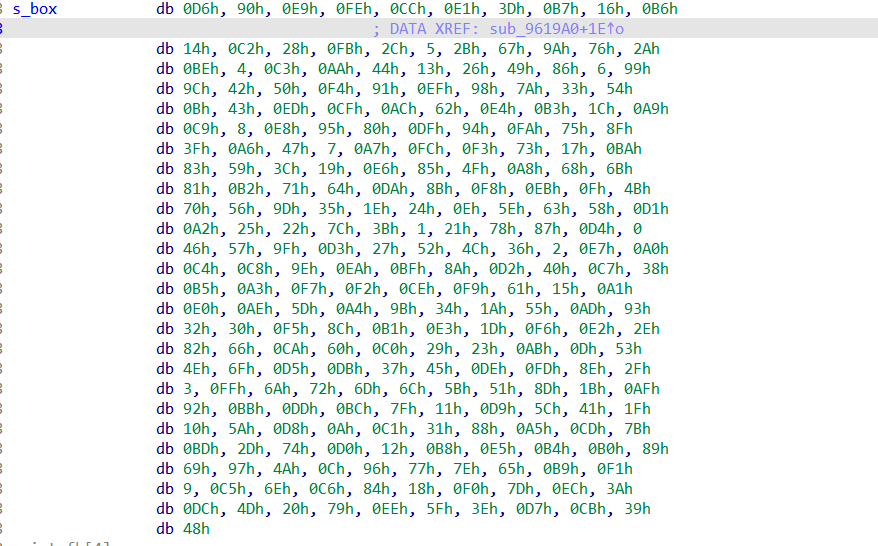
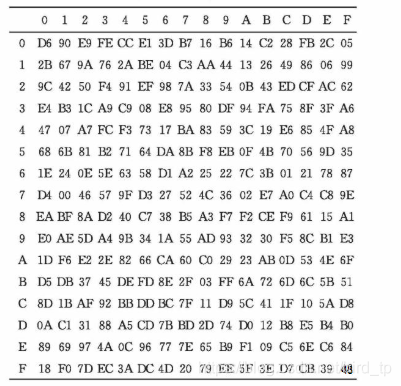
c0c7ced5,  dce3eaf1,  f8ff060d,  141b2229,

30373e45, 4c535a61, 686f767d, 848b9299，

a0a7aeb5, bcc3cad1,  d8dfe6ed, f4fb0209,

 10171e25, 2c333a41, 484f565d, 646b7279

**3.S盒固定**



**4.特殊的循环移位函数**

(v3 >> 9) | (v3 << 23)) ^ v3 ^ ((v3 >> 19) | (v3 << 13)和

((v3 >> 8) | (v3 << 24)) ^ ((v3 >> 14) | (v3 << 18)) ^ ((v3 >> 22) | (v3 << 10)) ^ v3 ^ ((v3 >> 30) | (4 \* v3));

**异常处理函数解析**

**具体内容见网页，以下仅列出问题涉及到的两个函数**

**至于hook部分已有了解不再赘述**

**SetUnhandledExceptionFilter**()



即当异常发生后无处理函数即

**AddVectoredExceptionHandler**()

注册VEH异常

按照程序执行逻辑即先执行hook函数，并注册了VEH异常，然后程序自动引发异常，执行VEH注册函数，又通过**SetUnhandledExceptionFilter**()继续通过异常执行sm4密钥扩展函数

另外main函数里有base64大小写换表操作和sm4的32轮轮函数，最终在异常内通过两两换位和base64魔改（= -> !和向左循环移位24位）后进行比较