流加密算法

原理

流加密算法（Stream Cipher），是一种加密方式，它通过将明文与一个随机生成的密钥流进行逐位或逐字节异或操作来产生密文。

基本操作

密钥流生成器（Key Stream Generator）：流加密算法的核心是密钥流生成器，它使用一个短的密钥（种子密钥）
来生成一个长的密钥流。这个密钥流决定了加密的随机性和强度。

异或操作（XOR Operation）：明文数据与密钥流进行异或操作，生成密文。解密时，密文与同一密钥流再次进行
异或操作，恢复出明文。

RC4算法介绍

原理

在密码学中，RC4（来自Rivest Cipher 4的缩写）是一种串流加密算法，密钥长度可变。他加密解密使用
相同的密钥，因此也被称为“流加密算法”。RC4由伪随机数生成器和异或运算组成。RC4的密钥长度可变，
范围是[1,255]。RC4一个字节一个字节地加解密。给定一个密钥，伪随机数生成器接受密钥并产生一个S盒。
S盒用来加密数据，而且在加密过程中S盒会变化。由于异或运算的对合性，RC4加密解密使用同一套算法。

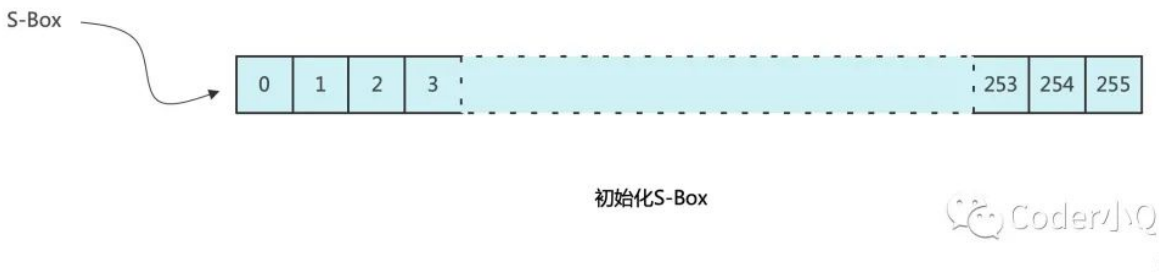
基本操作

1、初始化S-Box
2、KSA过程（置乱刚刚初始化完成的S表）
   - 初始化密钥（可无可有）
   - 置乱过程（KSA）
3、PRGA过程（生成密钥流，用于与明文进行异或生成密文）

RC4算法刨析

第一步：初始化S-Box

具体过程如下图所示，其实就是从0~255填充满大小为256的数组。



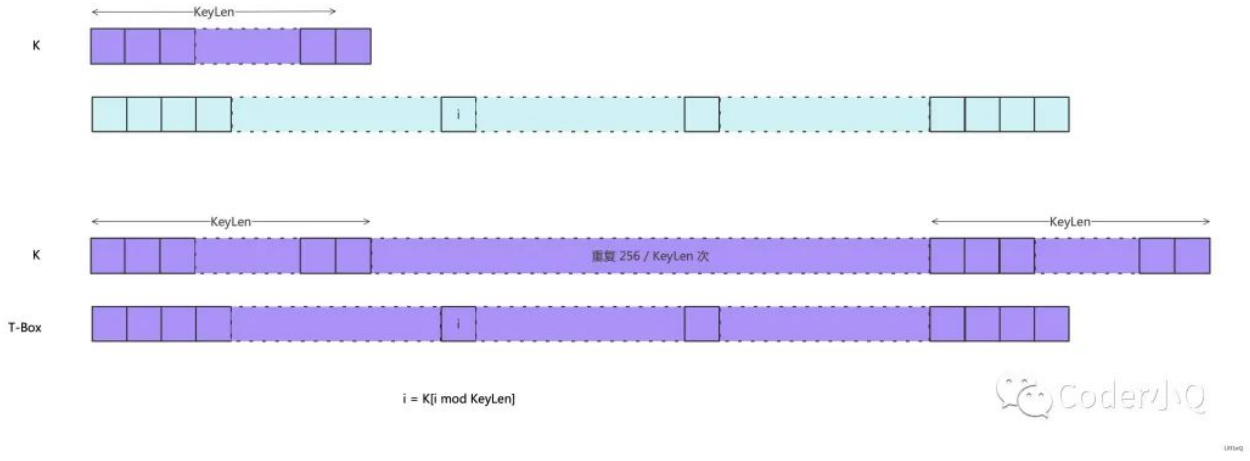
初始化S-Box

第二步：KSA过程

初始化密钥

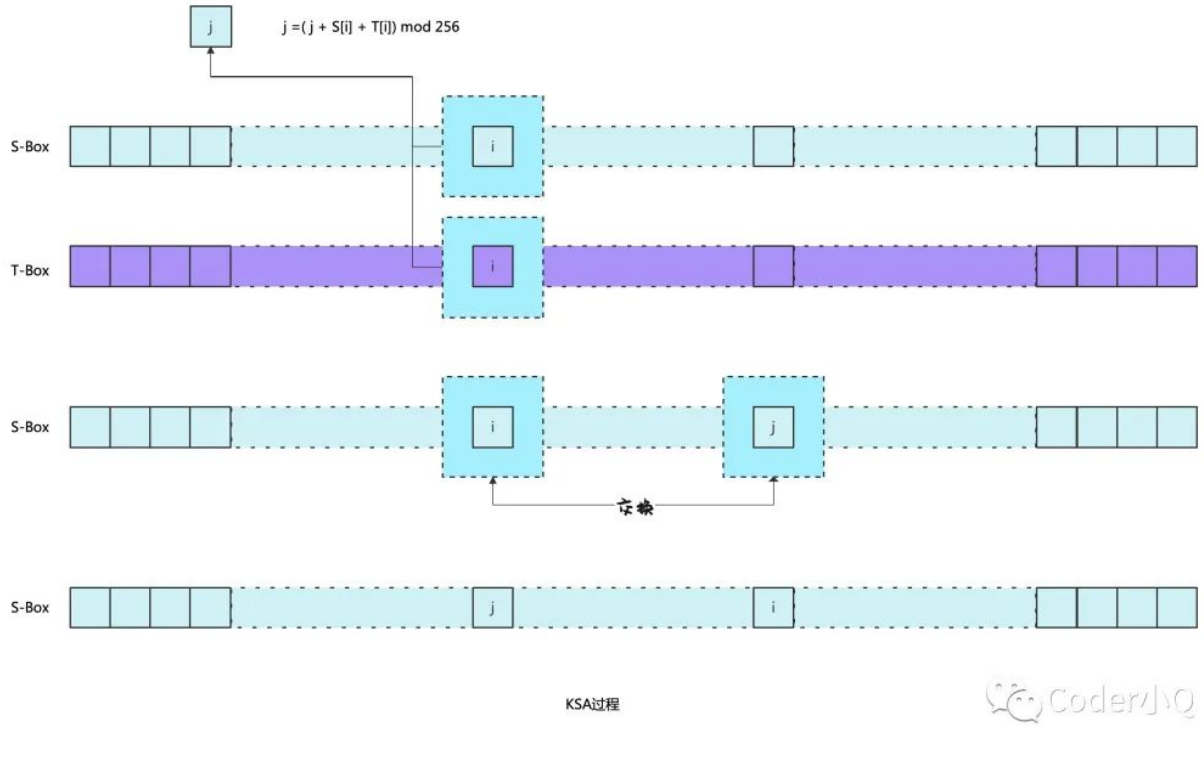
上面我们有了最初的S-Box，那么对于KSA的核心作用呢，实际上是通过密钥来置乱初始的向量，这个初始向量是一个固定值，从0~255来填充满S-Box。

> 这一步呢其实在编码过程当中是可有可无的，如果我们不生成T-Box，可以在编码过程当中模KeyLen来实现，如果使用T-Box，那么我们在编码过程当中需要模256了，这里展示使用T-Box的情况，和文章最开头的流程图保持一致。对于T-Box来说，其实就是密钥循环复制m次，使得循环之后充满256长度的数组，从这里可以直观的看出，对于RC4来说，它的密钥长度大小最大是256。

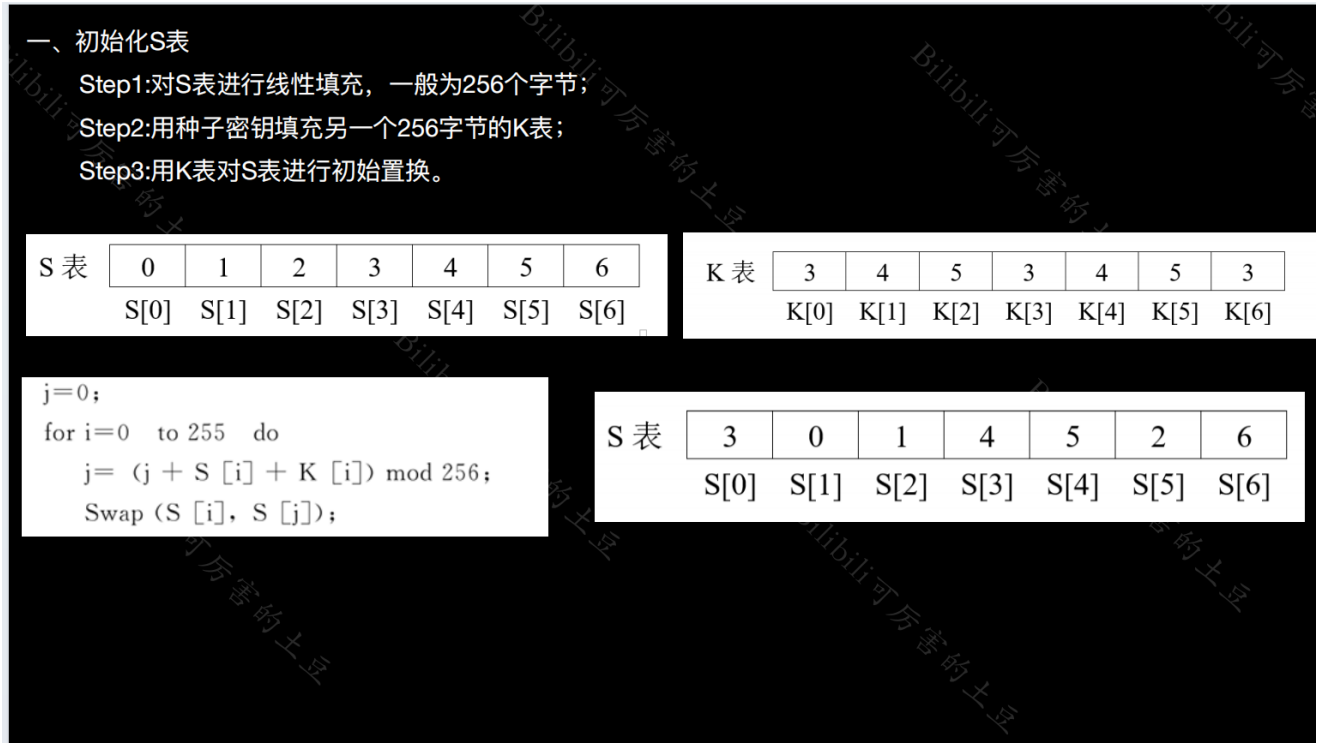


i = K[i mod KeyLen]

置乱过程(KSA)

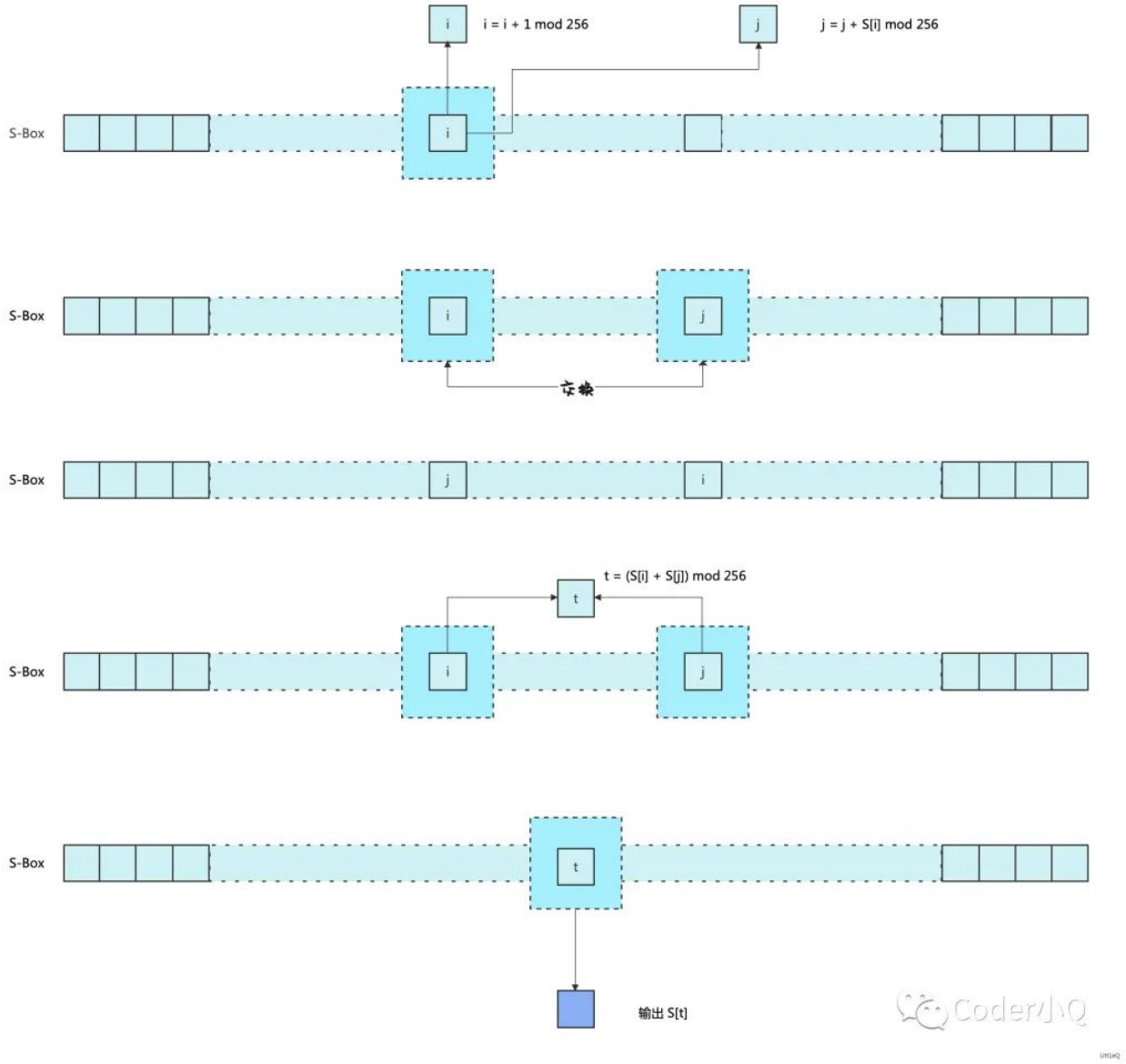


KSA整体流程图



第三步：PRGA过程

这个过程是整个RC4算法的核心，通过这个过程生成我们需要的PRNG序列，具体过程如下图所示。



PRGA整体流程图

二、密钥流的生成（为每个待加密的字节生成一个伪随机数，用来异或）。

```
i,j=0;
for r=0 to len do  //r为明文长度，r字节
  i=(i+1) mod 256;
  j=(j+S[i])mod 256;
  swap(S[i],S[j]);
  t=(S[i]+S[j])mod 256;
  k[r]=S[t];
```

S 表

3	0	1	4	5	2	6
S[0]	S[1]	S[2]	S[3]	S[4]	S[5]	S[6]

$i = (i+1) \bmod 7 = 0+1=1;$

$j = (j+S[i]) \bmod 7 = 0+S[1] = 0+0=0;$

$Swap(S[0], S[1])$

$t = S[0]+S[1] \bmod 7 = 3;$

$S[3] = 4;$

$K[0] = S[3] = 4$

解决逆向中对称加密算法的小技巧

使用工具：

LazyIDA：具体可以看我的bilibili上的视频教程，提供插件下载。

主要作用：
自动重定位跳转
数据格式转换
粘贴数据到指定内存
DUMP指定内存到文件
修改内存数据

具体思路

首先我们知道流加密算法的加密和解密过程是相同的，我们输入的明文会被加密为密文，那么如果我们输入
的是密文那么得到是不是就是我们要的明文，所以我们要想办法获得密文，一般逆向过程是，我们输入字符串
然后程序进行加密，得到加密后的密文，与正确的flag密文进行比较，如果相同，那么则正确。

那么我们可以使用LazyIDA中带有提取特定数据格式和修改内存数据的功能，通过提取内存中的密文，然后作为
输入，进行解密，得到flag，因为提取的密文可能会带有不可见字符，所以需要LazyIDA插件。

具体可以看我的bilibili上的视频教程，有例题讲解。

bilibili视频教程

十分钟带你解决逆向中对称加密算法（RC40)