RC4算法

简介

在[密码学](https://baike.baidu.com/item/%E5%AF%86%E7%A0%81%E5%AD%A6/480001" \t "/home/lilin/文档\\x/_blank)中，RC4（来自Rivest Cipher 4的缩写）是一种[流加密](https://baike.baidu.com/item/%E6%B5%81%E5%8A%A0%E5%AF%86" \t "/home/lilin/文档\\x/_blank)算法，[密钥](https://baike.baidu.com/item/%E5%AF%86%E9%92%A5/101144" \t "/home/lilin/文档\\x/_blank)长度可变。它加解密使用相同的密钥，因此也属于[对称加密算法](https://baike.baidu.com/item/%E5%AF%B9%E7%A7%B0%E5%8A%A0%E5%AF%86%E7%AE%97%E6%B3%95/211953" \t "/home/lilin/文档\\x/_blank)。RC4是[有线等效加密](https://baike.baidu.com/item/%E6%9C%89%E7%BA%BF%E7%AD%89%E6%95%88%E5%8A%A0%E5%AF%86" \t "/home/lilin/文档\\x/_blank)（WEP）中采用的加密算法，也曾经是[TLS](https://baike.baidu.com/item/TLS" \t "/home/lilin/文档\\x/_blank)可采用的算法之一.

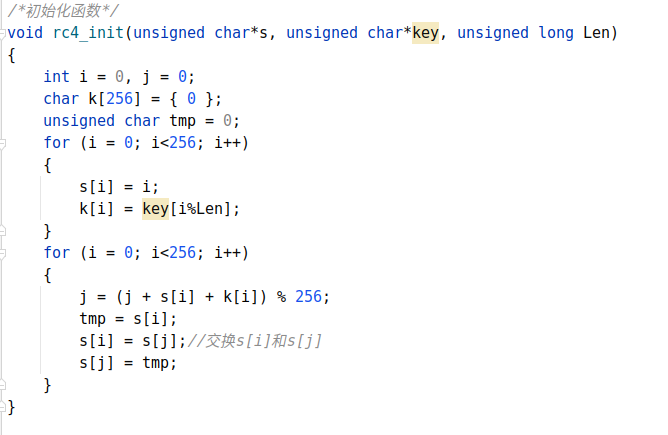
原理

[RC4算法](https://baike.baidu.com/item/RC4%E7%AE%97%E6%B3%95" \t "/home/lilin/文档\\x/_blank)的原理很简单，包括初始化算法（KSA）和[伪随机](https://baike.baidu.com/item/%E4%BC%AA%E9%9A%8F%E6%9C%BA" \t "/home/lilin/文档\\x/_blank)子密码生成算法（PRGA)两大部分。假设S-box的长度为256，密钥长度为Len。先来看看算法的初始化部分（用C代码表示）： [1]

其中，参数1是一个256长度的char型[数组](https://baike.baidu.com/item/%E6%95%B0%E7%BB%84" \t "/home/lilin/文档\\x/_blank)，定义为: unsigned char sBox[256];

参数2是密钥，其内容可以随便定义：char key[256];

参数3是密钥的长度，Len = strlen(key);



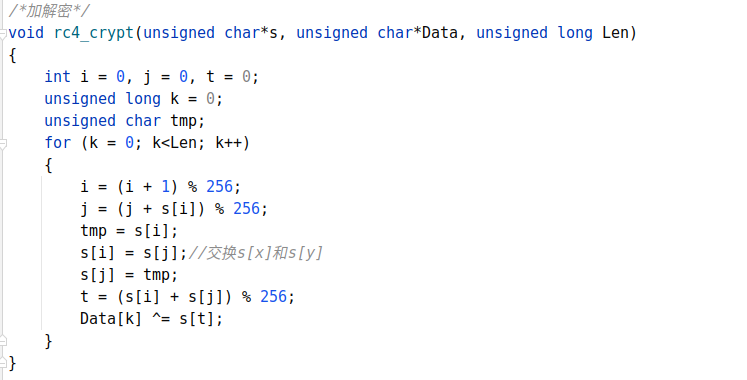
在初始化的过程中，[密钥](https://baike.baidu.com/item/%E5%AF%86%E9%92%A5" \t "/home/lilin/文档\\x/_blank)的主要功能是将S-box搅乱，i确保S-box的每个元素都得到处理，j保证S-box的搅乱是随机的。而不同的S-box在经过[伪随机](https://baike.baidu.com/item/%E4%BC%AA%E9%9A%8F%E6%9C%BA" \t "/home/lilin/文档\\x/_blank)子密码生成算法的处理后可以得到不同的子密钥序列，将S-box和明文进行xor运算，得到密文，解密过程也完全相同。

再来看看算法的加密部分（用C代码表示）：

其中，参数1是上边rc4\_init函数中，被搅乱的S-box;

参数2是需要加密的数据data;

参数3是data的长度.



最后，在main函数中，调用顺序如下：



安全性

由于[RC4算法](https://baike.baidu.com/item/RC4%E7%AE%97%E6%B3%95/9686396" \t "/home/lilin/文档\\x/_blank)加密是采用的xor，所以，一旦子密钥序列出现了重复，密文就有可能被破解。关于如何破解xor加密，请参看Bruce Schneier的Applied Cryptography一书的1.4节Simple XOR，在此我就不细说了。那么，RC4算法生成的子[密钥](https://baike.baidu.com/item/%E5%AF%86%E9%92%A5" \t "/home/lilin/文档\\x/_blank)序列是否会出现重复呢？由于存在部分弱密钥，使得子密钥序列在不到100万字节内就发生了完全的重复，如果是部分重复，则可能在不到10万字节内就能发生重复，因此，推荐在使用RC4算法时，必须对加密密钥进行测试，判断其是否为弱密钥。其不足主要体现于，在无线网络中IV（初始化向量）不变性漏洞。

而且，根据分析结果，没有任何的分析对于密钥长度达到128位的RC4有效，所以，RC4是最安全的[加密算法](https://baike.baidu.com/item/%E5%8A%A0%E5%AF%86%E7%AE%97%E6%B3%95" \t "/home/lilin/文档\\x/_blank)之一，大家可以放心使用！

分布式代码管理网站Github从2015年1月5日将停止对RC4的支持，RC4作为一种老旧的验证和加密算法易于受到黑客攻击。这意味着，用户在使用Windows XP系统上的IE浏览器时将无法进入github.com网站。

实现程序

完整程序可见项目工程文件夹RC4

#include<stdio.h>  
#include<string.h>  
typedef unsigned longULONG;  
  
*/\*初始化函数\*/*void rc4\_init(unsigned char\*s, unsigned char\*key, unsigned long Len)  
{  
 int i = 0, j = 0;  
 char k[256] = { 0 };  
 unsigned char tmp = 0;  
 for (i = 0; i<256; i++)  
 {  
 s[i] = i;  
 k[i] = key[i%Len];  
 }  
 for (i = 0; i<256; i++)  
 {  
 j = (j + s[i] + k[i]) % 256;  
 tmp = s[i];  
 s[i] = s[j];*//交换s[i]和s[j]* s[j] = tmp;  
 }  
}  
  
  
*/\*加解密\*/*void rc4\_crypt(unsigned char\*s, unsigned char\*Data, unsigned long Len)  
{  
 int i = 0, j = 0, t = 0;  
 unsigned long k = 0;  
 unsigned char tmp;  
 for (k = 0; k<Len; k++)  
 {  
 i = (i + 1) % 256;  
 j = (j + s[i]) % 256;  
 tmp = s[i];  
 s[i] = s[j];*//交换s[x]和s[y]* s[j] = tmp;  
 t = (s[i] + s[j]) % 256;  
 Data[k] ^= s[t];  
 }  
}  
  
int main()  
{  
 unsigned char s[256] = { 0 }, s2[256] = { 0 };*//S-box* char key[256] = { "justfortest" };  
 char pData[512] = "这是明文:今天的天气很好";  
 unsigned long len = strlen(pData);  
 int i;  
  
 printf("pData=%s\n", pData);  
 printf("key=%s,length=%d\n\n", key, strlen(key));  
 rc4\_init(s, (unsigned char\*)key, strlen(key));*//已经完成了初始化* printf("完成对S[i]的初始化，如下：\n\n");  
 for (i = 0; i<256; i++)  
 {  
 printf("%02X", s[i]);  
 if (i && (i + 1) % 16 == 0)putchar('\n');  
 }  
 printf("\n\n");  
 for (i = 0; i<256; i++)*//用s2[i]暂时保留经过初始化的s[i]，很重要的！！！* {  
 s2[i] = s[i];  
 }  
 printf("已经初始化，现在加密:\n\n");  
 rc4\_crypt(s, (unsigned char\*)pData, len);*//加密* printf("pData=%s\n\n", pData);  
 printf("已经加密，现在解密:\n\n");  
 *//rc4\_init(s,(unsignedchar\*)key,strlen(key));//初始化密钥* rc4\_crypt(s2, (unsigned char\*)pData, len);*//解密* printf("pData=%s\n\n", pData);  
 return 0;  
}

运行结果

