高等学校网络空间安全专业“十三五”规划教材

应用密码学实验

主　编　张　薇　吴旭光
副主编　魏悦川　朱率率
苏　阳　刘龙飞

西安电子科技大学出版社
http://www.xduph.com

高等学校网络空间安全专业“十三五”规划教材

应用密码学实验

主　编　张　薇　吴旭光
副主编　魏悦川　朱率率
　　苏　阳　刘龙飞

西安电子科技大学出版社

内 容 简 介

本书针对信息安全相关专业“密码学”课程，介绍了课程中涉及的大部分算法及其C/C++语言实现。全书包括八个实验，内容涵盖古典密码、密码学的数学基础、分组密码、流密码、公钥密码、散列函数、数字签名、同态密码及TFHE方案的实现等密码学知识。其中，实验一至七为大部分密码学教材中包含的内容，要求学生必须掌握；实验八为拓展实验，供学有余力的学生自学。每个实验都介绍了算法的相关知识点和编程实现时的难点，并给出了算法源代码。

本书可供高等院校信息安全、计算机、通信等专业的学生使用，也可供信息安全领域的技术人员参考。

图书在版编目(CIP)数据

应用密码学实验/张薇，吴旭光主编. —西安：西安电子科技大学出版社，2019.1
ISBN 978-7-5606-5165-1

Ⅰ. ① 应… Ⅱ. ① 张… ② 吴… Ⅲ. ① 密码学—高等学校—教材
Ⅳ. ① TN918.1

中国版本图书馆CIP数据核字(2019)第001442号

策划编辑 陈 婷
责任编辑 师马玮 陈 婷
出版发行 西安电子科技大学出版社(西安市太白南路2号)
电 话 (029)88242885 88201467 邮 编 710071
网 址 www.xduph.com 电子邮箱 xdupfxb001@163.com
经 销 新华书店
印刷单位 陕西天意印务有限责任公司
版 次 2019年1月第1版 2019年1月第1次印刷
开 本 787毫米×1092毫米 1/16 印张 9.5
字 数 219千字
印 数 1～3000册
定 价 25.00元
ISBN 978-7-5606-5165-1/TN

XDUP 5467001-1

高等学校网络空间安全专业“十三五”规划教材
编审专家委员会名单

前　言

在信息社会、互联网＋的时代背景下，密码早已从军事应用走向全社会，成为保护商业信息、网络交易和个人隐私的必备工具。了解一定的密码学知识，理解经典密码算法的原理和实现，是现代社会中人们需要掌握的一项重要技能。

本书按照密码系统的传统分类方式组织内容，介绍了多种密码算法及其实现。实验一介绍古典密码，包括单表代替、多表代替、置换密码。实验二介绍密码学的数学基础，包括模幂运算、欧几里得算法以及素数的检测。实验三介绍分组密码，包括 DES、AES 以及国产商用分组密码标准 SMS4 的原理与实现。实验四介绍流密码，包括产生随机数的线性同余发生器和 BBS 随机数发生器、LFSR、流密码的加/解密过程以及 RC4 密码算法。实验五介绍公钥密码，包括 DH 协议、RSA 密码、ElGamal 加密体制、椭圆曲线密码等经典密码算法。实验六介绍散列函数，主要是 SHA 系列算法的实现。实验七介绍数字签名，包括 RSA 签名和 DSA 签名。实验八介绍同态密码及 TFHE 方案的实现。附录介绍了算术运算库 GMP 的安装、配置与使用。

本书是在作者多年教学实践的基础上编写而成的。为了配合课堂教学，书中挑选出密码中的重要算法进行详细讨论并编程实现。实验一、二由张薇编写，实验三由魏悦川编写，实验四由苏阳编写，实验五、六、七由吴旭光和朱率率合作完成，实验八由刘龙飞编写。全书内容严谨、语言精练，既可作为实验教材，也可作为工程实践的参考书独立使用。

本书提供相关程序代码，需要者可登录出版社网站（http://www.xduph.com）免费下载。

本书在编写过程中得到了武警工程大学密码工程学院领导的大力支持，以及西安电子科技大学马建峰教授、沈玉龙教授和李兴华教授的鼓励和帮助，在此表示感谢！

作　者

2018 年 9 月

目　录

实验一　古典密码

1.1　单表代替

一、实验目的

通过实验熟练掌握凯撒密码原理，编程实现加密算法，提高C++程序设计能力，掌握穷举破译的方法。

二、实验要求

(1) 输入任意的一段明文，对其加密并输出密文。

(2) 输入一段密文，利用穷举法进行唯密文攻击，输出密钥。

(3) 要求有对应的程序调试记录和验证记录。

三、实验内容

(一) 凯撒密码的加解密

1. 知识点

凯撒密码是一种典型的单表代替密码技术，其加密方法如下：

$$\text{密文} = \text{明文} + \text{密钥} \mod 26$$

解密方法如下：

$$\text{明文} = \text{密文} - \text{密钥} \mod 26$$

2. 程序代码

程序功能：从文件 plaintext.txt 中读明文，用密钥 key 加密，密文保存于字符数组 ciphertext中。

注意：(1) 因为明文长度是可变的，所以在输入时用字符串指针存储明文。

(2) 由于凯撒密码对明文中的空格、标点等不做任何处理，因此在实际编程时把明文中的非字母符号直接复制到密文中。

程序代码如下：

```
#include <iostream.h>
#include <stdio.h>
#include <string.h>
```

```
int main(int argc, char *argv[])
{
    FILE *file1;
    char *message,plaintext\[50],ciphertext\[50];
    int i,lengthofmessage,key=7;

    file1=fopen("plaintext.txt","r");
    fgets(message,50,file1);
    printf("length of plaintext: %d", strlen(message));
    lengthofmessage=strlen(message);
    strcpy(plaintext,message);
    printf("\nplaintext is:");
    for (i=0;i<lengthofmessage;i++)  printf("%c",plaintext\[i]);
    printf("\nPlease input the key:");
    scanf("%d",key);
    i=0;
    while (i<lengthofmessage)
    {
        if (plaintext[i]>='A'&&plaintext[i]<='Z')
            ciphertext[i]='A'+(plaintext[i]-'A'+key)%26;
        else if (plaintext[i]>='a'&&plaintext[i]<='z')
            ciphertext[i]='a'+(plaintext[i]-'a'+key)%26;
        else ciphertext[i]=plaintext[i];
        i++;
    }
    ciphertext[i]='\0';
    printf("\n");
    printf("ciphertext:");
    i=0;
    for (i=0;i<=lengthofmessage;i++)  printf("%c",ciphertext[i]);
    fclose(file1);
    getchar();
    return 0;
}
```

3. 运行结果

输入密钥 key 为 7，明文为“This is a test.”，则输出密文如下：

Aopz pz h alza.

程序运行结果如图 1-1 所示。

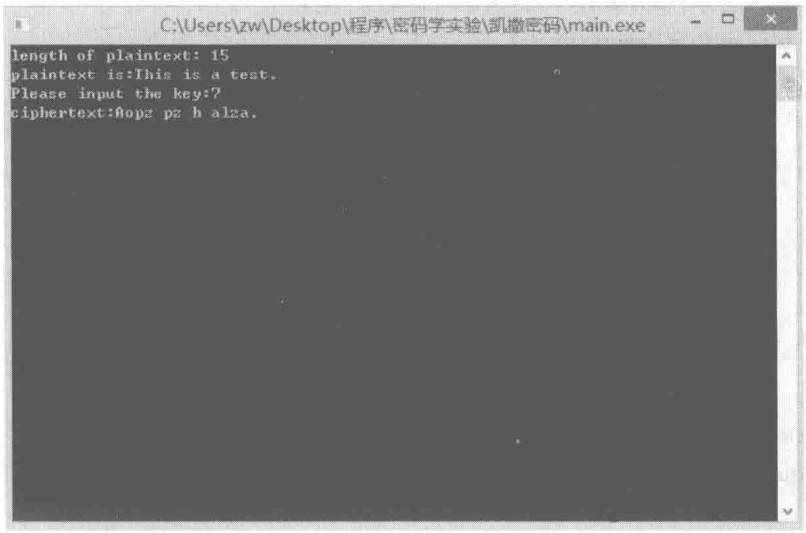

图 1-1　凯撒密码加密实验结果

（二）凯撒密码的穷举破译

1. 知识点

凯撒密码可能的密钥有 26 个。所谓穷举破译，是指用所有可能的密钥尝试解密，直到找出正确的密钥和明文。穷举破译是一种唯密文攻击，任意给定一段密文，利用穷举法找出所用的密钥，最多需要尝试 26 次。

2. 程序代码

程序功能：对于给定的密文，用密钥空间中的所有密钥逐个尝试解密并输出结果，通过观察明文语义完成破译。

注意：每次解密前要重置保存明文的数组，否则后面的解密结果都会追加到前一次的迭代结果中。

程序代码如下：

```
#include <iostream.h>
#include <string.h>
int main(int argc, char *argv[])
{
  char message[]="GUVF VF ZL FRPERG ZRFFNTR";
  char translated[50];
  int length,key,i;
  lenth=strlen(message);
  printf("length of message:%d",length);
  printf("\n");
  key=0;
  for (key=0;key<26;key++)
  {
      printf("\ntranslation attempt %d:",key);
```

```
        for (i=0;i<length;i++) translated[i]='0';
        for (i=0;i<length;i++)
        {
            if (message[i]>='A'&&message[i]<='Z')
                translated[i]='A'+((message[i]-'A'-key+26)%26);
            else if (message[i]>='a'&&message[i]<='z')
                translated[i]='a'+((message[i]-'a'-key+26)%26);
            else translated[i]=message[i];
            printf("%c",translated[i]);
        }
    }
    getchar();
    return 0;
}
```

3. 运行结果

对于密文“GUVF VF ZL FRPERG ZRFFNTR”进行暴力破解，程序运行结果如图 1-2 所示。

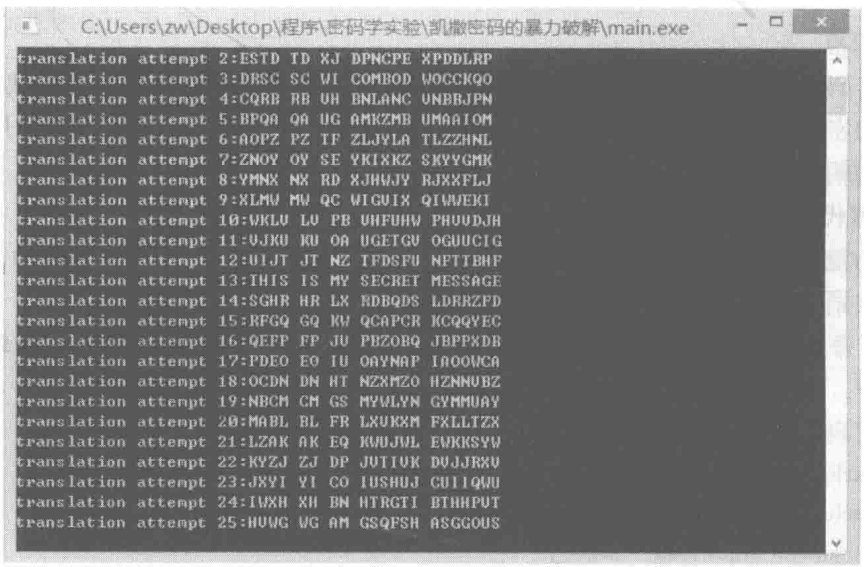

图 1-2　凯撒密码的暴力破解实验结果

观察发现只有当密钥为 13 时，解密结果才是一段有意义的话，所以可断定密钥 key=13。

1.2　多表代替

多表代替是指使用多个不同的代替表对明文加密。最典型的一种多表代替加密技术是维吉尼亚密码，它共有 26 个代替表，每个代替表都是字母表循环左移产生的新表，在实际加密时，使用一个密钥字符串来控制代替表的使用。如设密钥为“cipher”，则需要轮流使

用以“c”、“i”、“p”、“h”、“e”、“r”开头的代替表，加密完 6 个明文字符后，再重新循环使用这些代替表。

一、实验目的

通过实验熟练掌握多表代替密码的实现方法，特别是进一步熟悉 C 语言中的字符串操作，提高程序设计能力。

二、实验要求

编程实现维吉尼亚密码，程序功能：从文件中读取明文并加密，将密文保存到另一个 txt 文件中。

三、实验内容

1. 维吉尼亚密码的程序设计

程序中用字符串数组 key 保存密钥，也可直接输入密钥，再利用密钥对应的代替表对一段明文加密。

注意： 由于维吉尼亚密码的代替表是固定的，变化的是密钥，因此每次输入密钥时，需计算密钥长度，并根据密钥选择代替表。

2. 算法模块

1）密钥的处理

可预设密钥为“cipher”，长度为 6；也可直接输入密钥(以字母形式或数字形式)，再计算密钥长度，放在整型变量 period 中。

2）读取明文并预处理

从文件 plaintext.txt 中读取明文，取出所有字母，放入数组 plaintext[]中；统计明文中的字母个数，放入整型变量 lengthofmessage 中。

注意： 串拷贝操作 strcpy(plaintext,message)一定要在读完文件后马上执行拷贝，否则明文会出错。

3）加密

维吉尼亚密码的加密与凯撒密码几乎完全相同，只是加密每个字符时所使用的密钥在变化。用语句：

```
ciphertext[i]='a'+(plaintext[i]-'a'+key[i%6])%26
```

即可实现加密。

3. 程序代码

```
#include <iostream.h>
#include <string.h>
int main(int argc, char *argv[])
{
    FILE *file1;
    int period;
    int i,j,lengthofmessage,numofletters,numkey[6];
    char key[6];              //="cipher"
```

```
char * message,plaintext[50],ciphertext[50];
strcpy(key, "cipher");
* message=NULL;
file1=fopen("plaintext.txt","r");
fgets(message,50,file1);
fclose(file1);
strcpy(plaintext,message);

printf("length of plaintext: %d", strlen(message));
lengthofmessage=strlen(message);

printf("\nplaintext is:");
for (i=0;i<lengthofmessage;i++)  printf("%c",plaintext[i]);
period=6;
printf("\nKey:");
for (i=0;i<period;i++)
{
    numkey[i]=key[i]-'a';
    printf("%d  ",numkey[i]);
}
j=0;
numofletters=0;  //由于只对字母加密，因此需要统计当前字母是明文所有字母中的
                 //第几位，再用这个数字模 6 得到的值决定使用密钥的哪一位
for (i=0;i<lengthofmessage;i++)
{
    if ((plaintext[i]>='A')&&(plaintext[i]<='Z'))
    {
        numofletters++;
        j=(numofletters-1)%6;
        ciphertext[i]='A'+(plaintext[i]-'A'+numkey[j])%26;
    }
    else {
        if ((plaintext[i]>='a')&&(plaintext[i]<='z'))
        {
            numofletters++;
            j=(numofletters-1)%6;
            ciphertext[i]='a'+(plaintext[i]-'a'+numkey[j])%26;
        }
        else ciphertext[i]=plaintext[i];
    }
}
printf("\nnumofletters=%d",numofletters);
printf("\nCiphertext is:");
```

```
    for (i=0;i<lengthofmessage;i++)    printf("%c",ciphertext[i]);
    return 0;
}
```

4. 运行结果

程序运行结果如图 1-3 所示。

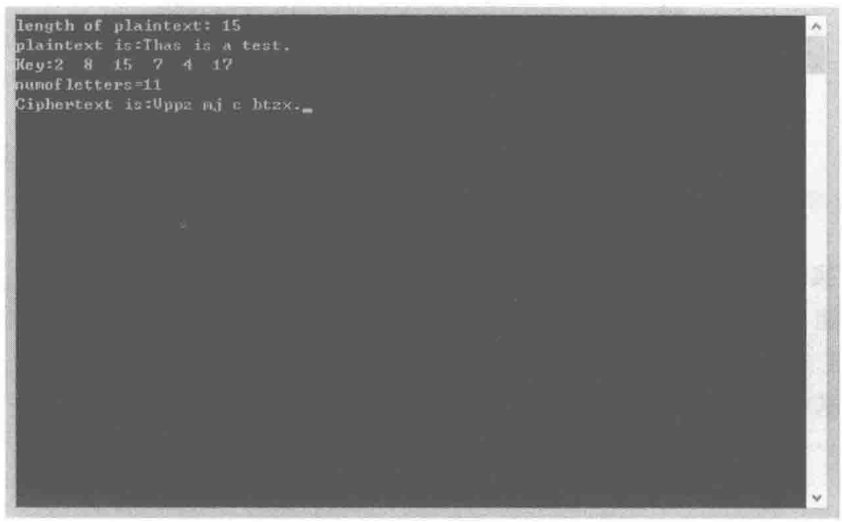

图 1-3　维吉尼亚密码实验结果

1.3　置换密码

数学上的置换是指对 n 个符号进行重新排列，而置换密码是把明文重新排列。对明文进行重排的方式有很多，比如倒排、天书以及栅栏密码，都属于简单的重排。

一、实验目的

理解置换密码的加密方式，熟悉 C 语言文件操作。

二、实验要求

编程实现数据加密标准 DES 中的初始置换，用其加密 64 比特的明文。

三、实验内容

1. 知识点

一般意义上的置换密码用一个表格来表示置换规则。假设每次将 n 个符号看做一组进行置换，则需要构造 n 个位置的全排列：$(i_1 i_2 \cdots i_n)$，这表示把明文中的第 i_1 个符号取出来，作为密文的第一个符号，明文的第 i_2 个作为密文的第 2 个……

解密时仍需查表：1 出现在第几位，就把密文中第几个符号作为明文的第 1 个；也可以先写出逆置换表，然后按照与加密完全相同的算法来实现。

这里我们令 $n=64$，构造一个置换密码，使用的置换表是数据加密标准 DES 中的初始

置换 IP，见图 1-4。

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

图 1-4　DES 中的初始置换 IP

2. 算法实现

程序功能：输入 64 比特明文(也可从文件中读取)，利用上述置换表进行重排。

注意：定义一个整型数组 key[64]，保存上述置换表。

加密的基本语句十分简单：

```
for (i=0;i<64;i++)
{
    j=key[i];
    ciphertext[i]=plaintext[j];
}
```

3. 程序代码

```
#include <iostream.h>
#include <string.h>
int main(int argc, char *argv[])
{
    FILE *file1;
    char plaintext[64],ciphertext[64];
    char *message;
    int i,j;
    int
    key[64]={58,50,42,34,26,18,10,2,60,52,44,36,28,20,12,4,62,54,46,38,30,22,14,
             6,64,56,48,40,32,24,16,8,57,49,41,33,25,17,9,1,59,51,43,35,27,19,
             11,3,61,53,45,37,29,21,13,5,63,55,47,39,31,23,15,7};

    file1=fopen("plaintext.txt","r");
    fgets(plaintext,65,file1);
    fclose(file1);

    printf("\nPlaintext: ");
    for (i=0;i<64;i++)  printf("%c ",plaintext[i]);
    for (i=0;i<64;i++)
```

```
    {
        j=key[i-1];
        ciphertext[i]=plaintext[j];
    }
    printf("\n Ciphertext:");
    for (i=0;i<64;i++)   printf("%c ",ciphertext[i]);
    getchar();
    return 0;
}
```

4. 运行结果

从文件“plaintext.txt”中读取 64 比特明文，加密后输出密文，并将明文和密文都显示出来。

运行结果如图 1-5 所示。

图 1-5　置换结果

实验二　密码学的数学基础

2.1　模幂运算

模幂运算是对给定的整数 p、n、a，计算 $a^n \bmod p$，这个运算在密码学中应用极为普遍，RSA、ElGamal、DH 交换等重要密码方案中都涉及模幂运算。

一、实验目的

通过实验熟练掌握模幂运算的计算方法，提高 C++程序设计能力。

二、实验要求

(1) 输入任意的整数 p、n、a，计算 $a^n \bmod p$。

(2) 有对应的程序调试记录和验证记录。

三、实验内容

1. 算法原理

快速实现模幂运算的基本原理是模重复平方计算法，其理论基础是模运算的基本性质，即相乘与求模两个运算是可交换的。

$$r \equiv (ab) \pmod m \equiv (a \pmod m)(b \pmod m) \pmod m$$

算法步骤：

(1) 将 n 表示为二进制，$n = n_r n_{r-1} \cdots n_1 n_0$，用一个数组保存 n 的各个数位，n 共有 $r+1$ 位。

(2) 循环计算(用数组 b 保存所有中间结果)：先循环计算一系列结果：$a^2 \bmod p$，$a^4 \bmod p$，$a^8 \bmod p$，…，再把指数 n 的二进制表示中取值为 1 的位对应的 a 的幂相乘，即可得到最终结果。

2. 程序代码

```
#include <iostream.h>
#include <math.h>
int main(int argc, char *argv[])
{
  int n,a,p;
  int nn[30],aa[30],bb[30];
  cout<<"Please input a and n:";
  cin>>a;
  cin>>n;
  cout<<"Input a prime, p ";
```