现代密码学第二次作业

计34 2013011377 杨志灿

## 第一题

1. **SHA-1算法简述**

SHA-1杂凑算法以一长度小于264的01串（原文）为输入，输出是一个160位的杂凑值。

SHA-1先将原文分成512位一组，其中最后一组的后64位是原文的长度（位数）。剩余的不足部分使用100000…（1后若干个零）补齐。

分组后，SHA-1算法依次处理每个分组，迭代计算杂凑值，具体过程较复杂，可参考代码或FIPS PUB 180-4中的伪代码。

1. **寻找碰撞**

考虑到寻找到碰撞的两个信息最好可以转换为ASCII码显示，故不能采用前一次的输出作为下一次的输入的方法（不能保证输入的是可见字符）。故最终决定采用随机生成原文的方式寻找碰撞。

在随机生成输入文本的时候使用了一个小技巧，就是使用固定的初始化随机种子。这样只需存储是生成的第几个输入，就可以还原出文本而不需记录文本，可以大大节约内存空间的使用。

1. **工程概述**

SHA-1.h中实现了SHA-1算法，其中

uint32\_t**\*** SHA1**(**const char **\*** msg**,** bool print **=** **false)**

其中msg是字符数组格式的原文，print表示是否输出中间过程。输出5个32位整型数，共同组成160位杂凑值。

值得注意的是，本次实验的输入文本都比较短，只有一个分组，迭代次数为1，所以中间过程只有初始化和最终结果两个，而SHA-1算法的初始化状态是常数，就不在本文中赘述了。

uint32\_t**\*** SHA1**(**const bool **\*** msg**,** uint64\_t size**,** bool print **=** **false)**

其中msg是bool数组格式的原文，size表示原文的位数，print表示是否输出中间过程。输出5个32位整型数，共同组成160位杂凑值。

main.cpp是使用SHA-1算法计算文本的杂凑值，运算结果见第4节。

collision.cpp是寻找碰撞的程序，其中

void find\_collision**(**int len **=** 10**)**

find\_collision寻找杂凑值前50位相同的碰撞对，len表示随机文本的长度，寻找到碰撞对会输出碰撞对的生成编号。根据这两个生成编号，可以重新调用如下函数还原文本。

void print\_msg**(**unsigned int id**,** int len **=** 10**)**

其中id表示生成编号，len表示生成的文本长度。

1. **实验结果**

|  |  |
| --- | --- |
| Plaintext | SHA-1 |
| yangzhican2013011377 | 70d6cde6226cf95cf7a0e5b4a6eca6168e7b3cac |
| yezipeng2013011404 | 19f45aa3b28665b0fdbe3480408e3d0d581c2ff4 |
| D2/;q<IVQq | 6179f4612671b706258304541701437890943446 |
| e,N>rF 5es | 6179f4612671b15d162c5282f97a8ee4e222ed77 |

后两个文本是寻找到的长度为10的一对碰撞对，他们的杂凑值的前50位是相同的。

## 第二题

1. **AES加密算法简述**