**现代密码学**

**实验一**

1. Coursera Dan Boneh Week 1 Program Assignment

Many Time Pad

1. PA1 option

Write a program that allows you to "crack" ciphertexts generated using a Vigenere-like cipher, where byte-wise XOR is used instead of addition modulo 26.

1. http://www.cryptopals.com/sets/1
2. Convert hex to base64
3. Fixed XOR
4. Single-byte XOR cipher
5. Detect single-character XOR
6. Implement repeating-key XOR
7. Break repeating-key XOR
8. MTC3 Cracking SHA1-Hashed Passwords

<https://www.mysterytwisterc3.org/en/challenges/level-2/cracking-sha1-hashed-passwords>

def hamming\_distance(s1, s2): dis = 0 for i in range(min(len(s1), len(s2))): b = bin(ord(s1[i]) ^ ord(s2[i])) dis += b.count('1') return dis def guess\_keysize(string): keys = [] for keysize in range(2, 40): blocks = [] count = 0 dis = 0 for i in range(0, len(string), keysize): count += 1 blocks.append(string[i:i+keysize]) if count == 4: break #选取四个块，两两组合求汉明距离 pairs = itertools.combinations(blocks, 2) for (x, y) in pairs: dis += hamming\_distance(x, y) ndis = dis / keysize key = {'keysize': keysize, 'distance': ndis} keys.append(key) return sorted(keys, key=lambda c:c['distance'])[0:3] def guess\_key(keysize, string): key = '' for i in range(keysize): now\_str = '' #获取每个块相同位置的字符 for index, ch in enumerate(string): if index % keysize == i: now\_str += ch key += chr(traversal\_singlebyte(now\_str)['key']) return key def break\_repeatingkey\_xor(string): keysizes = guess\_keysize(string) candidate = [] plains = [] for keysize in keysizes: key = guess\_key(keysize['keysize'], string) #二元组：重复密钥异或解密明文，对应密钥key plains.append((hex\_to\_str(repeatingkey\_xor(string, key)), key)) return sorted(plains, key=lambda c:get\_score(c[0]))[-1] f = open('challenge6.txt', 'r') s = f.read() string=base64.b64decode(s) res=break\_repeatingkey\_xor(string) print 'plaintext: \n'+res[0] print 'key: \n'+res[1]  
  
本文来源：[码农网](https://www.codercto.com/)  
本文链接：https://www.codercto.com/a/43657.html