1st Implementation Project
MC889/MO421 – Introduction to Cryptography
Prof. Diego de Freitas Aranha
2015/1

1 Introduction

This assignment consists in implementing an automated attack against an insecure instance of the One-Time Pad (OTP) cipher, given two challenge ciphertexts using the same key.

2 Method

The OTP cipher is a stream cipher, where the key material has the same length as the input message, attaining the perfect secrecy property. OTP is extremely hard to use securely in practice, because key material needs to be random and can never be repeated.

For this assignment, we will consider as the alphabet of definition the Latin alphabet and punctuation marks represented in the ASCII encoding. The plaintext messages were written in the English language.

3 Material

To understand how encryption works in this domain, a simple program implementing OTP is provided. The source code of this program is available as `otp.c` and serves as illustration. The program was used to generate the challenge ciphertexts and was compiled as:

```
$ gcc otp.c -o otp
```

The 205-byte key was extracted from the GNU/LINUX `/dev/urandom` pseudo-random number generator as follows:

```
$ dd if=/dev/urandom of=key.txt bs=205 count=1
```

Encryption was performed as follows for the two plaintext messages:

```
$ ./otp plaintext.txt ciphertext.txt key.txt
```

Decryption works in the same way.

4 Objective

The objective is to implement a program capable of recovering the plaintext messages and the encryption key corresponding to the provided challenge ciphertexts. You can use any approach you may find suitable, such as frequency analysis or dictionary-based *cribbing*. Breaking OTP sometimes requires manual intervention. Thus, the program can receive limited manual assistance for directing the automatic method.

The cryptanalisys tool can receive as input the challenge ciphertexts and produce the original plaintext messages and the encryption key. Recovered plaintext should be identical to the original messages.

For verification, the challenge ciphertexts have the following hash values:

```
$ sha256sum challenge1.txt challenge2.txt
e4e5ce118fb465349ca94cc2935497528a06fd9cbfb74665c216d3118cb63342  challenge1.txt
9eacadf8023abfdd4c50e4b9ce577b47233d6fe646133d13163e9f4a28b0457a  challenge2.txt
```

5 Scoring

The cryptanalysis tool should be submmited in both binary and source code forms by private e-mail until April 07, 2015. Please including compile and usage instructions and annotate your message subject with either `[MC889]` or `[MO421]`. The program should be followed by a short report describing the results. Graduate students will receive more scrutiny in the evaluation of their reports. A single **bonus point** in the first written test will be assigned to the fastest program able to automatically recover the message in its entirety.