Project 01
Prof. Diego de Freitas Aranha
MO421 - 2017/1

Pedro Gabriel Calixto Mendonça - RA 118363

1 Introduction

The project 01 reports the method used for breaking a Two-Time Pad cipher. The attack was done using two ciphertexts generated by two different plaintexts using the same One-Time Pad (OTP) cipher. The original messages are written in English.

The OTP is a stream cipher, in which the key has the same length of the message being sent. The OTP performs a XOR operation with every bit of the message and the key. If the sender does not repeat the same key for various messages and the key is generated using random bits (such that a frequency analysis cannot be performed as an attack), the perfect secrecy can be guaranteed.

As it will be seen, using the same OTP cipher for different messages is insecure and can be broken using a technique called *scrib dragging*.

2 Method

When the OTP cipher is used on two different messages, a XOR operation between the two ciphertexts equals a XOR between the two original messages. Being c_i the ciphertexts, m_i the plaintexts k the key and $\oplus$ the XOR operation:

$$c_1 \oplus c_2 = (m_1 \oplus k) \oplus (m_2 \oplus k)$$
$$c_1 \oplus c_2 = m_1 \oplus m_2$$

This can be achieved because the XOR operation is commutative and the XOR of two equal variables equals 0 ($k \oplus k = 0$).

Using the property above, a technique called *scrib dragging* can be used. It consists in using a dictionary of possible words for the plaintexts, and XOR'ing it against the $c_1 \oplus c_2$ ($c_1 \oplus c_2$ is called c_{xor} in this report). A word is guessed (*e.g.*, "the", the most common word in English), and if the XOR of the word and the subsection of c_{xor} with the same length of the word returns a word that is valid in the English language, the word used is probably a word in one of the plaintexts. This can be seen below in the example.

$$m_1[1..3] = "the", m_2[1..3] = "and"$$

$$c_{xor}[1..3] \oplus "the" = (c_1[1..3] \oplus c_2[1..3]) \oplus "the"$$
$$c_{xor}[1..3] \oplus "the" = (m_1[1..3] \oplus c_2[1..3]) \oplus "the"$$
$$c_{xor}[1..3] \oplus "the" = ("the" \oplus "and") \oplus "the"$$
$$c_{xor}[1..3] \oplus "the" = "and"$$

So, if the $c_{xor} \oplus word$ results in a possible English word, the next positions of c_{xor} can be tested against the dictionary of English words until the plaintexts completion.

This is the general idea for *crib dragging* and the pseudocode used is below.

Algorithm 1 Pseudocode used for crib dragging using two ciphertexts

1: **procedure** CribDrag(cipherxor, cipherlen, dictionary, plain1, plain2, index1, index2)
2: **if** length of $plain1 \leqslant cipherlen$ **then return** plain1, plain2
3: **if** last word in $plain1$ is present in $dictionary$ **then**
4: $possibleWords \leftarrow$ words in dictionary starting with plain1 last word
5: **else**
6: $possibleWords \leftarrow dictionary$
7: **for all** $word$ in $possibleWords$ **do**
8: $word \leftarrow word +$ ' '
9: $cribxor \leftarrow (plain1 + \text{word}) \oplus cipherxor$
10: **if** $cribxor$ is a valid text according to the dictionary **then**
11: $newplain1, newplain2 =$ CribDrag($cipherxor, cipherlen, dictionary, cribxor, plain1 + word, index2 +$ length of word, $index1 +$ length of word) ▷ Recursively calls CribDrag for the next cipherxor positions
12: **if** $newplain1$ is not empty and $newplain2$ is not empty **then**
13: **return** newplain1, newplain2 ▷ If the CribDrag returned plaintexts are not empty, they are valid and returned
14: **return** empty, empty ▷ If it reaches the end, the plaintexts could not be found

3 Conclusion

The dictionary used is a list of the 10,000 most common English words in order of frequency, as determined by n-gram frequency analysis of the Google's Trillion Word Corpus, found on GitHub. Sixteen words were added to the dictionary, among *entirety*, *Snowden*, *U.S.*, *liberties*, *secretly*, etc. They were added according to the iterations of the program when these words were obvious ones and were not in the dictionary.

The procedures and algorithm described in 2 were used against the two given ciphertexts, and the discovered original messages are:

```
I can't in good conscience allow the U.S. government ultimately simple conclusion:
the U.S. government had built a system that has as its with this massive surveillance
machine they're secretly buildi
```

```
Taken in its entirety, the Snowden archive led to an to destroy privacy, internet
freedom and basic liberties for people around the world goal the complete elimination
of electronic privacy worldwide.
```

The first plaintext has an incomplete ending because its ciphertext had a length greater than the other ciphertext, and the original message can only be found for the same length of characters. Still, only 3 characters were not discovered, which probably are "ng.". For the same reason, 201 bits of the key were discovered, for it would need 205-bit ciphertexts length to discover the remaining key bits.