

Security Coursework

Part A

To brute force the key the algorithm must generate all possible permutations of the key, and use the generated key to attempt to decrypt the ciphertext. As the plaintext is valid English the algorithm uses a regular expression to check if the decrypted plaintext only contains alphanumeric, punctuation, and space characters. If the regular expression matches we have found the correct key.

As the key only uses lower case letters (a – z), the key space can be calculated using:

$$\text{Key Space} = 26^l$$

Where l is the length of the key.

When I run the algorithm on my computer I get an average of 8250 attempts per second. Knowing this, we can calculate the time taken to attempt every key.

Key Length	Key Space	Time Taken
4	456,976	~ 55 seconds
5	11,881,376	~ 24 minutes
6	308,915,776	> 10 hours

As the key length increases, the key space and time taken to attempt all possible keys grows exponentially.

Part B

I expect this attack to work as we can retrieve the keystream which was used to encrypt the plaintext.

If we have C_i as byte i of the ciphertext, P_i as byte i of the plaintext, and $KS_i(k)$ as byte i of the keystream generated by the key, k , where n, m are the start and end bytes of the ciphertext to replace and $n \leq i \leq m$ then:

$$C_i = P_i \oplus KS_i(k)$$

Thus, if we know both the plaintext and ciphertext we can recover the keystream by:

$$KS_i(k) = P_i \oplus C_i$$

We can then use to the replacement plaintext, P' , to generate the replacement ciphertext, C' :

$$C'_i = P'_i \oplus KS_i(k)$$

We then replace the C_{n-m} with C' .

Without the original plaintext, we wouldn't be able to recover the keystream, and thus would be unable to attack in this way.

Part C

Similarly to part B, we use the plaintext and ciphertext to recover the keystream used to encrypt the plaintext:

$$KS_i(k) = P_i \oplus C_i$$

We are now able to XOR the recovered keystream with the second ciphertext C' , to recover the second plaintext P' :

$$P'_i = C'_i \oplus KS_i(k)$$

However, we are unable to recover the key, k , that was used to generate the keystream.

If S' is equal to the size in bytes of P' , and S is equal to the size in bytes of P , and the case were $S < S'$, we would only be able to recover the first S bytes of P' .