16-124-836
Marcel ZAUDER

1.1 Hiding the plaintext statistics of natural language

a.) *First text compression then monoalphabetic substitution. How secure is this method?*

Because the compressed text will almost have a uniform letter statistics an potential Eavesdropper cannot use the letter statistic of natural language. So there are **26!** ($\approx 4 * 10^{26}$) possible permutations for encrypting this compressed text with the monoalphabetic substitution method. For an attacker it is nearly impossible to check all possibilities and to reverse the compressing process every time - this would take too much time. So this method is pretty secure.

b.) *Homophonic substitution. How secure is this method?*

Because every symbol (for example numbers fromm 00 to 99) can be mapped onto the letters such that each of it has the same relative frequency of $\sim 0.01 = 1\%$ it is nearly impossible to decrypt the message without knowing the mapping tableux. The number of permutation assuming that each letter gets minimum one encryption symbol can be dexcribed with the following computation (for 100 symbols):

$$\begin{gathered} 26! \text{ (at least one symbol for each letter)} \\ 26^{74} \text{ (74 remaining symbols can be placed arbitrarily)} \\ \approx 2,058955 * 10^{131} \end{gathered}$$

So this method is even more secure than the first one!

c.) **Bonus question:** First compress then encrypt (or vice versa)?

Because the process of compressing can be easily reversed by anyone this step does not add any security for your encrypted text. So if you first encrypt your text and then compressing it, the step of compressing can be reversed and the attacker only has to identify the encrypting method to get the plaintext. This is much easier because the result will be a readable text. If you compress first and then encrypt it is much more difficult for the attacker to decrypt it because the compressed text is still not readable and so there is no easy method for checking the correctness of the decription. The attacker only gets the result if he reverses the compressing step.

1.2 Does the one-time pad leak information?

Because Alice removed 2 keys (the all-1 and all-0 key) we will end up with $2^\lambda - 2$ possible keys. Therefore the possibility of one key will be $\frac{1}{2^\lambda - 2}$. Because each key will have the same possibility (which also corresponds for the ciphertexts), the ciphertext distribution is uniform.

1.3 One-time pad using the same key

Alice is decoding two messages (m and m') with the same key so the following ciphertext are produced:

$$Enc(k, m) = k \oplus m = c$$
$$Enc(k, m') = k \oplus m' = c'$$

When Eve obtains both ciphertexts and knows that the same key is used the following conclusion can be made:

$$c \oplus c' = (k \oplus m) \oplus (k \oplus m') = (m \oplus k) \oplus (k \oplus m')$$
$$= m \oplus (k \oplus k) \oplus m' = m \oplus 0^\lambda \oplus m' = m \oplus m'$$

So Eve can calculate $m \oplus m'$!
With this Eve can compute information from these two ciphertexts which should not be possible. For example if $m_i \oplus m'_i = 0$, then Eve knows that the bits of m and m' are the same at this exact position i.

1.4 Attack at one-time pad

a.) *Show that a known-plaintext attack on OTP results in the attacker learning the key k.*

When an Eavesdropper knows the plaintext m and the created ciphertext Enc(k,m) = $k \oplus m$ = c it is possible for the attacker to calculate the key k.
We have to consider **4** cases to compute the key with the information of m and c:
CASE 1: if $m_i = 0$ and $c_i = 0$ then $k_i = 0$
CASE 2: if $m_i = 1$ and $c_i = 0$ then $k_i = 1$
CASE 3: if $m_i = 0$ and $c_i = 1$ then $k_i = 1$
CASE 4: if $m_i = 1$ and $c_i = 1$ then $k_i = 0$
Which is obviously the computation: $m \oplus c = k$

b.) *Can OTP be secure if it allows an attacker to recover the encryption key? Is this a contradiction to the security we showed for OTP? Explain.*

The OTP is not secure anymore because the attacker can decrypt every ciphertext which was encrypted with this OTP. Because we said that a OTP is only used once (as the name suggests) it does not matter if the key of this particular OTP is known because this key is not used anymore and no plaintext is encrypted with it. So the security of OTPs is still guaranteed even if the key can be recovered in the described action.