Final exam

Björn Bebensee (2019-21343)

November 28, 2019

Question 1

Given the plaintext p = "CRYPTOGRAPHY" and the corresponding ciphertext c = "XXSERJJLKWPF" we want to perform a known-plaintext attack on the Hill cipher with $m = 3$. The key is given by a matrix $K^{3\times 3}$. The encryption function is given as

$$KP = C \tag{1}$$

where M^{3xn} is the plaintext matrix. We obtain the plaintext matrix by encoding each value in p numerically (i.e. $A = 1, B = 2, C = 3$ and so on) and writing it as a matrix where each column corresponds to one block of words of length m. We do the same for ciphertext as well. Thus we get:

$$P = \begin{pmatrix} 2 & 15 & 6 & 15 \\ 17 & 19 & 17 & 7 \\ 24 & 14 & 0 & 24 \end{pmatrix}$$

$$C = \begin{pmatrix} 23 & 4 & 9 & 22 \\ 23 & 17 & 11 & 15 \\ 18 & 9 & 10 & 5 \end{pmatrix}$$

We can rearrange (1) to obtain the key as follows:

$$K = CP^{-1}$$

This requires us to invert P in $\mathbb{Z}_{26}$. To be able to compute the inverse we need to construct a square matrix M from using three of the columns of P. It is sufficient to have three linearly independent length m blocks in P that we can use in M to obtain the key using the inverse of M as $K = CM^{-1}$. However we

find that there is no set of three columns in P for which an inverse exists in $\mathbb{Z}_{26}$ as the columns of P are not linearly independent in $\mathbb{Z}_{26}$. Thus the plaintext p and its ciphertext c are not sufficient to obtain the key used. We would need a longer plaintext and ciphertext where the blocks are not linearly dependent on another in $\mathbb{Z}_{26}$.

Question 2

We define our cryptosystem $C = (\mathcal{P}, C, \mathcal{K}, e_j, d_j)$ with $\mathcal{P} = C = \mathcal{K} = \{1, 2, \ldots, n\}$ and $e_j(i) = A_{i,j}$ i.e. the encryption of a plaintext i with key j is exactly the i-th row of the j-th column of the encryption matrix $A^{n \times n}$. We construct this matrix such that each column of A is a different permutation of $\{1, ..., n\}$ (we do not allow identical columns) and more specifically such that

$$\forall i \in \mathcal{P}\ \forall j \in \mathcal{K}\ \forall k \in \mathcal{K} \text{ s.t. } k \neq j : A_{i,j} \neq A_{i,k}$$

i.e. no other column in A has the same value in the same row.

The Shannon Perfect Secrecy theorem states that a cryptosystem with $|\mathcal{K}| = |\mathcal{P}| = |C|$ has perfect secrecy if and only if each key is used with equal probability and

$$\forall x \in P\ \forall y \in C\ \exists k \in K : e_k(x) = y. \tag{2}$$

As $|\mathcal{K}| = |\mathcal{P}| = |C|$ and that each key is used with equal probability is given, we only need to show that our cryptosystem has a key for any $x \in P$ so that it is mapped to any $y \in C$ by the encryption function. By our construction, each column corresponds to a "key" j and each plaintext i is mapped to $A_{i,j}$ but as no value may occur twice in the same row of A we find that our encryption function maps each such plaintext x to any ciphertext $y = A_{i,j}$ exactly once for exactly one key j. Therefore (2) is satisfied and the Shannon Perfect Secrecy Theorem states that C has perfect secrecy.

Question 3

(a) Without loss of generality assume that the corrupted block is x_i for some $i \in \{1, \ldots, n\}$. We will denote the ciphertext of x_i as y_i. For the different block cipher modes of operation we then get the following results during encryption.

1. ECB: The message is divided into plaintext blocks of equal length each of which is encrypted individually with no relation between them. Thus, only the from x_i directly resulting ciphertext y_i is affected.

2. CFB: We see that the resulting ciphertext y_k during encryption for every block k depends not only on the plaintext x_k and the key but also the initialization vector (IV) and all previous plaintexts x_j with $j < k$. Thus if x_i is corrupted, all plaintexts x_j with $j > i$ are also corrupted.

3. CBC: Like in CFB the resulting ciphertext depends on all previous plaintexts and thus, if x_i is corrupted, then all ciphertexts x_j with $j > i$ will also be corrupted.

4. OFB: In OFB the resulting ciphertexts during encryption only depend on upon the IV, the key and their respective plaintexts and thus if a plaintext x_i is corrupted only the ciphertext y_i will be corrupted.

(b) Without loss of generality assume that the corrupted block is y_i for some $i \in \{1, \ldots, n\}$. We will denote the plaintext of y_i as x_i. For the different block cipher modes of operation we then get the following results during decryption.

1. ECB: Each ciphertext block is decrypted individually and thus only x_i will be decrypted incorrectly if y_i is corrupted.

2. CFB: During decryption the resulting plaintext in CFB depends on the key, the ciphertext y_i and the previous ciphertext y_{i-1}. Additioanlly, as there is no previous ciphertext for x_0, this plaintext block also depends on the IV. Thus, if y_i is corrupted then x_i and x_{i+1} will be decrypted incorrectly.

3. CBC: Each decrypted plaintext depends on the ciphertext and the previous block's ciphertext. Thus, if y_i is corrupted then x_i as well as x_{i+1} will be decrypted incorrectly.

4. OFB: Just like during encryption, the decryption process for a plaintext only requires the key, the ciphertext and the IV. Therefore, if a ciphertext y_i is corrupted then only x_i will be decrypted incorrectly.

Question 4

We can consider a substitution cipher as a map $f : X \to Y$ with $f(x,k) = x+k \mod 26$ and in this case $X = Y$. Since we are using a one-time pad each substitution is chosen randomly (i.e. with equal probabilities) and independently from other plaintexts. Thus, in order to encrypt a plaintext x we map it using $f(x,k)$ using a random k. This essentially corresponds to a random draw in the ciphertexts Y where we choose a random character $y \in Y$ which x will be mapped to. This random draw is a Bernoulli trial where we define a success as one $y_i \in C = \{y_1, y_2, y_3, y_4, y_5\}$ being the same as one letter $x \in M = \{x_1, x_2, x_3, x_4, x_5\}$, i.e. $x = y \mod 26$. Thus the probability of at least one success (at least one character in common) is given by:

$$\begin{aligned} P(X \geq 1) &= 1 - P(X = 0) \\ &= 1 - \binom{5}{0} \left(\frac{5}{26}\right)^0 \left(\frac{21}{26}\right)^5 \\ &\approx 0.1781 \end{aligned}$$

However, this computation is under the assumption that no character $x \in M$ appears twice. We can model drawing characters in M uniformly at random as a Bernoulli trial in the same way and obtain probabilities $P(X = 0), P(X = 1), P(X = 2), P(X = 3), P(X = 4), P(X = 5)$ and obtain final probability of two characters being the same as the weighted sum of probabilities that at least one character in C is common with M given that there are n identical characters in M.

Question 5

First we show that h is indeed preimage-resistant. Assume h is not preimage-resistant. Then it is possible to find $x = x_1 x_2$ with $x_1, x_2 \in \{0,1\}^m$ for a given $y \in \{0,1\}^m$ such that $h(x) = y$. But then it also follows that $h(x) = f(x_1 \oplus x_2) = y$ which implies that we can find a $c \in \{0,1\}^m$ for any given $y \in \{0,1\}^m$ with $f(c) = y$ by setting $c = x_1 \oplus x_2$. Then f cannot be preimage-resistant and since we have a contradiction the assumption that h is not preimage-resistant must be wrong. Thus h is preimage-resistant. □

We now show that h is **not** second-preimage-resistant. Consider $x = x_1 x_2$ with $x_1, x_2 \in \{0,1\}^m$. We can find a $x' \in \{0,1\}^{2m}$ with $h(x) = h(x')$ for any such $x = x_1 x_2$ by simply setting $x' = x_2 x_1$. Then it is easy to see that $h(x') =$

$h(x_2x_1) = f(x_2 \oplus x_1) = f(x_1 \oplus x_2) = h(x_1x_2) = h(x)$ with the commutativity of the $\oplus$ operation. Thus h is not second-preimage-resistant. □

Question 6

(a) We simply compute the encryption by first computing $n = p \cdot q = 199 \cdot 211 = 41989$ and then we can directly compute y as follows:

$$e_K(32767) = 32767 \cdot (32767 + 1357) \bmod 41989 = 16027$$

(b) Given $e_K(x) = x(x+B) \bmod n$ we want to compute the decryption operation as the reverse operation of $e_K(x)$ in mod n. We can expand as follows:

$$\begin{aligned} & x(x+B) = y \mod n \\ \equiv\ & x^2 + xB = y \mod n \\ \equiv\ & x^2 + xB - y = 0 \mod n \\ \equiv\ & x = -\frac{b}{2} \pm \sqrt{\frac{b^2}{4} + y} \mod n \end{aligned} \tag{3}$$

where x is the plaintext we want to obtain and we obtain equation (3) using the quadratic formula. The key idea is now that we can compute equation (3) mod p, mod q separately and then combine the two to obtain the result in n.

For mod p: In order to get rid of the fractions in the equation we rewrite and compute the inverse of 2 as $2^{-1} = 100 \bmod 199$ and the inverse of 4 as $4^{-1} = 50 \bmod 199$. By plugging into (3) we obtain:

$$\begin{aligned} x_p &= -\frac{b}{2} \pm \sqrt{\frac{b^2}{4} + y} \mod 199 \\ &= 1357 \cdot 100 \pm \sqrt{50 \cdot 1357^2 + 16027} \mod 199 \\ &= -1357 \cdot 100 \pm 86 \mod 199 \end{aligned}$$

and thus

$$x_p = 104 \vee x_p = 131. \tag{4}$$

For mod q: In order to get rid of the fractions in the equation we rewrite and compute the inverse of 2 as $2^{-1} = 106 \bmod 211$ and the inverse of 4 as

$4^{-1} = 53 \bmod 211$. By plugging into (3) we obtain:

$$\begin{aligned} x_q &= -\frac{b}{2} \pm \sqrt{\frac{b^2}{4} + y} \mod 211 \\ &= -1357 \cdot 106 \pm \sqrt{53 \cdot 1357^2 + 16027} \mod 211 \\ &= -1357 \cdot 106 \pm \sqrt{4} \end{aligned}$$

and thus

$$x_q = 58 \vee x_q = 62. \tag{5}$$

Now we can use the solutions in mod p, q to obtain the solution for x in mod n. We can obtain the coefficients a_p, a_q of Bézout's identity $a_p p + a_q q = \gcd(p, q)$ (for instance using the extended Euclidean algorithm):

$$-88 \cdot 199 + 83 \cdot 211 = 1$$

with $\gcd(p, q) = 1$ as p, q are prime. Then, the Chinese remainder theorem gives us the solution x in mod n using the coefficients a, b of Bézout's identity as follows:

$$x = x_q a_p p + x_p a_q q$$

and by using (4) and (5) we can obtain the four possible decryptions of the ciphertext y:

$$\begin{aligned} x_1 &= 58 \cdot -88 \cdot 199 + 104 \cdot 83 \cdot 211 = 7865 \mod 41989 \\ x_2 &= 58 \cdot -88 \cdot 199 + 131 \cdot 83 \cdot 211 = 18837 \mod 41989 \\ x_3 &= 62 \cdot -88 \cdot 199 + 104 \cdot 83 \cdot 211 = 21795 \mod 41989 \\ x_4 &= 62 \cdot -88 \cdot 199 + 131 \cdot 83 \cdot 211 = 32767 \mod 41989 \end{aligned}$$

We see that x_4 is our original plaintext. Used [2], [5] and [3] as well as [1] for calculations of extended gcd and square roots with respect to mod operation.

Question 7

In Shamir's threshold scheme the secret s is given by $P(0)$ of a randomly chosen polynomial P where P is of degree $k - 1$. Any k points uniquely

identify this polynomial P of degree $k-1$ so k is called the *threshold*. For the given $(4,7)$-Shamir threshold scheme we have the threshold $k=4$ so the polynomial P is of degree 3 and can be determined by any four (non-defective) shares. We can obtain P by Lagrange interpolation as follows.

$$P(x) = \sum_{j=0}^{k-1} y_j \ell_j(x) \tag{6}$$

where $\ell_j(x)$ is the so called Lagrange basis polynomial with:

$$\ell_j(x) = \prod_{\substack{0 \le m \le k \\ m \neq j}} \frac{x - x_m}{x_j - x_m} = \frac{x - x_0}{x_j - x_0} \cdots \frac{x - x_{j-1}}{x_j - x_{j-1}} \frac{x - x_{j+1}}{x_j - x_{j+1}} \cdots \frac{x - x_k}{x_j - x_k}$$

so for all $i \neq j$ the product ℓ_j will be 0 at $x = x_i$ for each $x_i \in X$ where X is the set of x in the k shares (x, y) which we use to determine the polynomial. Let $X = \{1, 2, 3, 4\}$. Thus we construct four Lagrange basis polynomials, one for each $x_i \in X$ such that the above condition is satisfied. For the chosen set X of shares $(1, 15), (2, 12), (3, 1), (4, 19)$ at $x = 1, x = 2, x = 3, x = 4$ we get:

$$\ell_1 = \frac{x-2}{1-2} \frac{x-3}{1-3} \frac{x-4}{1-4}$$
$$\ell_2 = \frac{x-1}{2-1} \frac{x-3}{2-3} \frac{x-4}{2-4}$$
$$\ell_3 = \frac{x-1}{3-1} \frac{x-2}{3-2} \frac{x-4}{3-4}$$
$$\ell_4 = \frac{x-1}{4-1} \frac{x-2}{4-2} \frac{x-3}{4-3}$$

and finally we obtain the polynomial P by plugging into (6)

$$P(x) = 15l_1 + 12l_2 + 1l_3 + 19l_4 \tag{7}$$

with respect to $\mathbb{Z}_{23}$.

(a) By plugging all shares into the polynomial P we can now verify the correctness (i.e. that we did not choose a defective share in X to compute the polynomial) and find the defective share. The defective share is exactly the only share (x, y) which yields a different $\hat{y} = P(x)$ with $y \neq \hat{y}$. We verify the correctness by checking that the output of the polynomial at x is exactly y for all the shares that we used to construct P. We see that $P(1) = 15$, verifying share $(1, 15)$,

$P(2) = 12$ verifying share $(2, 12)$, $P(3) = 1$ verifying share $(3, 1)$, $P(4) = 19$ verifying share $(4, 19)$ and thus our constructed polynomial. Furthermore we see that $P(5) = 103 = 11 \mod 23$ verifying $(5, 11)$ and $P(7) = 617 = 19 \mod 23$ verifying $(7, 19)$. However we find that $P(6) = 290 = 14 \mod 23$ and thus share $(6, 13)$ is defective.

(b) The correct value is given by $P(6) = 290 = 14 \mod 23$ and thus the correct share would be $(6, 14)$.

(c) The secret is given by $P(0) = 19 = s$.

(d) The minimum number of polynomial interpolations is given by the minimum number of points needed to exactly define a polynomial. For a polynomial of degree $k - 1$ we need at least k different points to uniquely identify the polynomial. Thus, for a (k, n)-Shamir threshold scheme we need at least k interpolations.

Used [4] in this question for a detailed explanation of how to do Lagrange interpolation.

Question 8

We may consider a lottery as a draw of a random number and participants in the lottery try to guess this number. The amounts of guesses per participants is the number of lottery "tickets", we assume $n = 1$. How to best design such a scheme heavily depends upon the cryptographic requirements one has for their use case. We will assume that a lottery should most importantly be *verifiable* for all participants and that each guess is a *commitment* and can't be changed later. One way to design such a scheme may be using an append-only data structure such as a blockchain. The organizer O chooses a winning number n and a key k randomly, computes a commitment c using a commitment function $c = C(n, k)$. The commitment is then made public to all participants on the blockchain. It should be signed so everyone can verify that it is indeed published by O. The used commitment scheme can be simply a keyed, collision-resistant and preimage-resistant hash function, then $C(n, k) = h(n||k)$. An alternative would be to use a commitment scheme based on the difficulty of the discrete logarithm such as the Pedersen commitment scheme. This commitment scheme is perfectly hiding which is of high importance so that participants can't guess the winning number. Participants will then commit their signed guesses to the blockchain as well. Finally, O makes n along with k

public on the blockchain so that everyone can verify the earlier commitment $c = C(n, k) = h(n||k)$. Furthermore, everyone can verify who the winner is as all the guesses are public and on the blockchain as well.

It is worth noting that this scheme does not guarantee anonymity and is mainly designed to make the lottery as transparent and verifiable to all participants as possible.

References

[1] Wolframalpha. http://wolframalpha.com/, 2019.

[2] Michael O Rabin. Digitalized signatures and public-key functions as intractable as factorization. Technical report, Massachusetts Inst of Tech Cambridge Lab for Computer Science, 1979.

[3] Wikipedia. Chinese remainder theorem. https://en.wikipedia.org/wiki/Chinese_remainder_theorem, 2019.

[4] Wikipedia. Lagrange polynomial. https://en.wikipedia.org/wiki/Lagrange_polynomial, 2019.

[5] Wikipedia. Rabin cryptosystem. https://en.wikipedia.org/wiki/Rabin_cryptosystem, 2019.