Full name: _____

Student ID: _____

- This exam is made of 13 pages and **10 questions**.
- You work individually on your answers. With the submission of your answers, you declare that this is your **individual work** and you did not ask any other person for assistance. You neither discussed with fellow students nor asked in an internet forum or chat for help.
- Explain and **motivate** all your answers.
- You have **180 minutes** to do the exam (+45 if you are entitled for extra time).
- Make sure your **handwriting is clear** and legible. Answer the questions in the spaces provided on the question sheets. If you run out of room for an answer, continue on the back of the page.
- This is a **closed-book** exam. As such, you are *not* allowed to use any book, notes, cheatsheet, phone or laptop.
- You can use a **non-programmable calculator** to double-check your calculations.

Exercise 1 – Perfect Secrecy [10 points]

Consider the cipher where $\mathcal{M} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_4 = \{0, 1, 2, 3\}$, defined by the following encryption map:

$$e_k(m) = (3m + k) \mod 4$$

for all $m \in \mathcal{M}$ and $k \in \mathcal{K}$.

(a) **[2 points]** Draw the encryption table of this cipher.

(a) **[4 points]** Assuming plaintext and key distributions to be uniform, does this cipher satisfy perfect secrecy? Motivate your answer.

(b) **[4 points]** Suppose that we modify the encryption map as follows:

$$e'_k(m) = (2m + k) \mod 4$$

Is this modified cipher perfectly secret? Motivate your answer.

Solution

(a) Given the following mapping:

$$m_0 = c_0 = k_0 = 0$$
$$m_1 = c_1 = k_1 = 1$$
$$m_2 = c_2 = k_2 = 2$$
$$m_3 = c_3 = k_3 = 3$$

The encryption table of e_k is the following:

	k_0	k_1	k_2	k_3
m_0	0	1	2	3
m_1	3	0	1	2
m_2	2	3	0	1
m_3	1	2	3	0

(b) Yes, the cipher is perfectly secret, because by Shannon's theorem one can see from the encryption table that for each pair (m_i, c_j) There is exactly one key k_l mapping m_i to c_j. Equivalently, the cipher is perfectly secret because the encryption table of e_k is a Latin square of order 4.

(c) No, the cipher is no longer perfectly secret. This can be easily seen by writing the full encryption table:

	k_0	k_1	k_2	k_3
m_0	0	1	2	3
m_1	2	3	0	1
m_2	0	1	2	3
m_3	2	3	0	1

In particular, since the first two rows are repeated, the encryption table is not a Latin square, and thus the cipher cannot satisfy perfect secrecy.

Exercise 2 – PRG and PRF [10 points]

(a) **[3 points]** Let G be a PRG with expansion factor ℓ such that $\ell(n) > n$ for all $n \in \mathbb{N}$, and consider this definition for G':

$$G'(s) = G(s)||G(s)_R \ ,$$

where $G(s)_R$ denotes the string $G(s)$ in reverse order. Is G' a PRG? Motivate your answer. It is not necessary to provide a full proof by reduction or a worked out counterexample, but you should at least provide the intuition behind your answer (for example: if you think that G' is not a PRG, then you should describe a strategy for a distinguisher that gives a non-negligible advantage).

(b) **[3 points]** An *involution* is a function $I : \{0,1\}^n \to \{0,1\}^n$ such that $I(I(x)) = x$ for all $x \in \{0,1\}^n$. Let $\mathbb{F} = \{f_s\}$ be a length-preserving family of functions $f_s : \{0,1\}^{|s|} \to \{0,1\}^{|s|}$ such that f_s is an involution for all $s \in \{0,1\}^*$. Show that f_s cannot be a PRP, by describing a successful strategy for a distinguisher.

(c) **[4 points]** Suppose now that we want to determine if the family $\mathbb{F} = \{f_s\}$ of point (b) above is indistinguishable from a *random involution*, rather than from a generic random permutation as usually done with a PRP. Give a definition of *Pseudorandom Involution* (PRI) for this specific case. In particular, show how the interaction between the distinguisher and the oracle should be modified compared to the usual definition of PRP/PRF.

Solution

(a) Although the expansion factor condition is satisfied (since $\ell'(n) = 2\ell(n) > n$ for all $n \in \mathbb{N}$), G' is not a PRG, under any definition of G. When given a string $x \in \{0,1\}^{\ell'(n)}$, a strategy for the distinguisher is to check whether the right half of x is the reverse of the left half. If so, then the distinguisher outputs 1 (the string has been generated by G'), otherwise it outputs 0. The probability that a uniformly random string $x \leftarrow \{0,1\}^{\ell'(n)}$ has the right half equal to the reverse of its left half is $2^{-\ell(n)}$. Therefore, the distinguisher can successfully distinguish strings generated by G' with a probability of $1 - 2^{-\ell(n)}$, which is a non-negligible function of n.

(b) A successful strategy for the distinguisher is the following:

1. Initialize a counter c to 0, and query the oracle on a random input $x \in \{0,1\}^n$, receiving the output $y \in \{0,1\}^n$ as an answer.
2. Query the oracle on the received output y, receiving the new output $z \in \{0,1\}^n$ as an answer
3. If $z = x$, then increase c by 1
4. Repeat steps 1-3 for a number of times which is polynomial in n, say $q(n)$ if by the end of this iteration, $c = q(n)$, then output 1 (i.e. the distinguisher believes that it is interacting with an oracle of f_s). Otherwise, output 0.

It can be seen that this is a PPT strategy for the attacker, since we are making $2 \cdot q(n)$ queries to the oracle, which is still a polynomial in n. If the distinguisher is interacting with an oracle of f_s, then it will correctly output 1 with probability 1. The probability that a random permutation behaves as an involution on a random subset of inputs is negligible in n (details of the calculations are omitted, as they were not requested in the exercise).

(c) The distinguishing game must be modified in the following way: *the distinguisher can ask a polynomial number of queries to the oracle, but it is forbidden for the distinguisher to query any of the outputs received from the oracle.* In this way, the strategy described above cannot be applied anymore. The definition thus becomes:

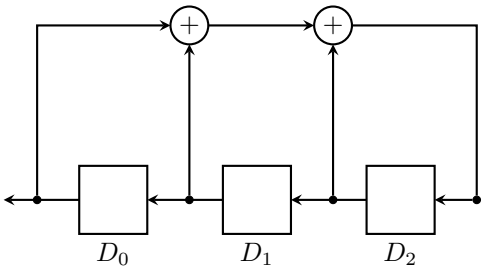
Definition 1. *A length-preserving family of involutions* $F : \{0,1\}^*\{0,1\}^* \to \{0,1\}^*$ *is a pseudo-random involution if, for all PPT distinguishers* D*, there exists a negligible function* $\mathsf{negl} : \mathbb{N} \to \mathbb{R}$ *such that:*

$$|Pr(D^{F_s(\cdot)}(1^n) = 1) - Pr(D^{I(\cdot)}(1^n) = 1) \leq \mathsf{negl}(n)$$

where the key $s \leftarrow \{0,1\}^n$ *is sampled with uniform probability from the set of all bitstrings of length* n*, and* $I(\cdot)$ *is sampled with uniform probability from the set* $\mathcal{I} = \{I : \{0,1\}^n \to \{0,1\}^n \ : I(I(x)) = x \ \forall x \in \{0,1\}^n\}$ *of all involutions of* n *bits.*

Exercise 3 – LFSR, Boolean Functions and Stream Ciphers

[**10 points**] Consider the following LFSR:



(a) [**3 points**] Is this a maximum-length LFSR? Draw the transition graph where the nodes represent the states of the registers.

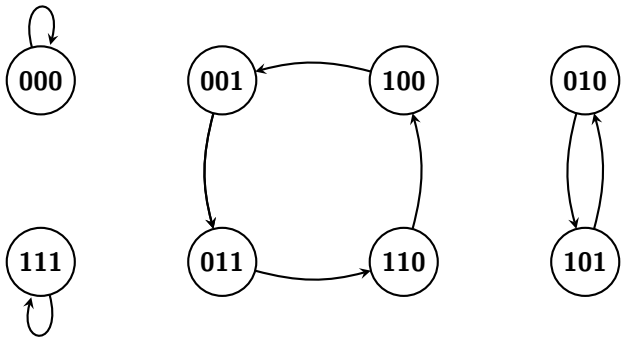
(b) [**3 points**] Let $f : \mathbb{F}_2^3 \to \mathbb{F}_2$ be the 3-variable Boolean function defined as follows: for each $x = (x_1, x_2, x_3) \in \mathbb{F}_2^3$, $f(x) = x_1 \oplus 1$ if $x_2 = x_3 = 0$; otherwise, $f(x) = x_1$. Write the truth table of f. Is f balanced?

(c) [**2 points**] Compute the Walsh coefficient $W_f(1, 0, 0)$ of the function f defined in (b). Based only on the value of this coefficient, what is the nonlinearity of f?

(d) [**2 points**] Compute the first 3 bits of the *filter* generator defined by the LFSR of point (a), filtered with the Boolean function f defined in point (b), starting from the state $(0, 1, 1)$.

Solution

(a) No, this is not a maximum-length LFSR. The transition graph, reported below, is composed of one cycle of length 4, one of length 2, and two fixed points (000 and 111).



(b) The truth table of $f : \mathbb{F}_2^3 \to \mathbb{F}_2$ is the following:

x_1	x_2	x_3	$f(x_1, x_2, x_3)$
0	0	0	1
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

The function is balanced, since the output column is composed of four ones and four zeros.

(c) The Walsh coefficient of f for the linear function defined by $(1, 0, 0)$ is:

$$\begin{aligned}
W_f(1,0,0) &= \sum_{x \in \mathbb{F}_2^3} (-1)^{f(x) \oplus (1,0,0) \cdot x} = \\
&= (-1)^{f(0,0,0) \oplus (1,0,0) \cdot (0,0,0)} + (-1)^{f(0,0,1) \oplus (1,0,0) \cdot (0,0,1)} + \\
&+ (-1)^{f(0,1,0) \oplus (1,0,0) \cdot (0,1,0)} + (-1)^{f(0,1,1) \oplus (1,0,0) \cdot (0,1,1)} + \\
&+ (-1)^{f(1,0,0) \oplus (1,0,0) \cdot (1,0,0)} + (-1)^{f(1,0,1) \oplus (1,0,0) \cdot (1,0,1)} + \\
&+ (-1)^{f(1,1,0) \oplus (1,0,0) \cdot (1,1,0)} + (-1)^{f(1,1,1) \oplus (1,0,0) \cdot (1,1,1)} = \\
&= (-1)^{1 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + (-1)^{0 \oplus 0} + \\
&+ (-1)^{0 \oplus 1} + (-1)^{1 \oplus 1} + (-1)^{1 \oplus 1} + (-1)^{1 \oplus 1} = \\
&= -1 + 1 + 1 + 1 - 1 + 1 + 1 + 1 = 4
\end{aligned}$$

Thus, considering this as the maximum coefficient in absolute value, the nonlinearity of f is

$$nl(f) = 2^{n-1} - \frac{1}{2} W_f(1,0,0) = 4 - 2 = 2$$

(c) The filter generator taps, at each stage, all registers of the LFSR, and consider them as input to the filter function (defined by f in this case). Starting from $(0, 1, 1)$, the first 3 bits z_1, z_2, z_3 given in output by the filter generator are thus:

$$\begin{aligned}
z_1 &= f(0,1,1) = 0 \\
z_2 &= f(1,1,0) = 1 \\
z_3 &= f(1,0,0) = 0
\end{aligned}$$

Exercise 4 – Block Cipher Modes of Operation

[10 points]

(a) **[4 points]** Describe the ECB and CBC modes of operation for block ciphers, and provide an overview of their pros and cons.

(b) **[3 points]** Briefly describe the Feistel paradigm for designing a block cipher.

(c) **[3 points]** DES is the most famous example of a block cipher designed with the Feistel paradigm. Explain why we should not use it anymore today. Could the use of CBC mode improve the security properties of DES?

Solution

(a) Given a block cipher with encryption function Enc_k and a plaintext message $m = m_1 \parallel \cdots \parallel m_N$ (with m_i being blocks of suitable size), the ECB mode of operation encrypts m as $c = c_1 \parallel \cdots \parallel c_N$, where $c_i = \mathsf{Enc}_k(m_i)$ for $i = 1, \ldots, N$. While, for the CBC mode of operation, we have $c_i = \mathsf{Enc}_k(m_i \oplus c_{i-1})$, where $c_0 = IV$ is a given initialization vector.

ECB:

- PRO: efficient, since it can be performed in parallel over the blocks.
- CON: as two identical plaintext blocks are encrypted to the same ciphertext block, it leaks patterns in data.

CBC:

- PRO: chaining the blocks during encryption allows for some form of diffusion.
- CON: since the encryption of a block depends on the outcome of the encryption of the previous block, it cannot be parallelized.

(b) The Feistel paradigm employs a round function that is structured as follows: first, the state of the cipher $x \in \{0,1\}^{2n}$ at round $i > 0$ is divided into a left half $L_i \in \{0,1\}^n$ and a right half $R_i \in \{0,1\}^n$. Then, the new left and right half L_{i+1} and R_{i+1} are defined as follows:

$$\begin{aligned} L_{i+1} &= R_i \\ R_{i+1} &= L_i \oplus S_{k_i}(R_i) \end{aligned}$$

where $S_{k_i} : \{0,1\}^n \to \{0,1\}^n$ is a $n \times n$ S-box, keyed with the round key k_i. Decryption works symmetrically by using the properties of the XOR operation:

$$\begin{aligned} R_i &= L_{i+1} \\ L_i &= R_{i+1} \oplus S_{k_i}(R_i) \end{aligned}$$

In particular, the Feistel paradigm does not require S to be invertible.

(c) While no substantial cryptanalytic attacks have been found so far on the structure of DES, this block cipher should not be used anymore today because of its small key length (56 bits), which is now amenable to bruteforce attacks. Plugging DES into a CBC mode of operation would not get around this issue, as the key size is a parameter of the block cipher itself.

Exercise 5 – Message Authentication Codes and Authenticated Encryption

[10 points]

(a) **[5 points]** Alice and Bob use an unconditionally secure MAC where $\mathcal{M} = \{0, 1, \cdots, 12\}$, $\mathcal{A} = \mathbb{Z}_{17}$ and $\mathcal{K} = \mathbb{Z}_{17} \times \mathbb{Z}_{17}$. The tag generation algorithm is defined as usual for all keys $k = (i, j) \in \mathcal{K}$ and messages $m \in \mathcal{M}$ as:

$$\mathsf{Mac}_k(m) = (i + mj) \mod 17 \ .$$

Oscar discovered that Alice and Bob determine the next key to be used in their MAC using the following method: given the key $k = (i, j)$ used to authenticate a message m, the next message m' is authenticated with the key $k' = (i', j') = (i + 1, j + 1)$. Suppose now that Oscar observes the two consecutive message/tag pairs $(8, 3)$ and $(10, 7)$. What are the keys k and k'? Explain in detail how Oscar can retrieve them, assuming that the tag generation algorithm is public.

(b) **[3 points]** Turn the following informal description of a MAC scheme into a formal description, then prove its insecurity: to authenticate a message, we cut it in two halves, XOR together the two halves and then apply a pseudorandom function to the result of the XOR.

(c) **[2 points]** What does an authenticated encryption scheme add beyond CCA-security?

Solution.

(a) Using the definition of $\mathsf{Mac}_k(m)$ and the fact that if $k = (i, j)$ then $k' = (i + 1, j + 1)$, we get the following system of two equations modulo 17:

$$\begin{cases} i + 8j & \mod 17 = 3 \\ (i + 1) + 10(j + 1) & \mod 17 = 7 \end{cases}$$

We have to solve for i, j. Subtracting the first equation from the second one, we obtain:

$$2j + 11 \mod 17 = 4 \ ,$$

which is satisfied for $j = 5$. Substituting this value back in the first equation, we get:

$$i + 40 \mod 17 = 3 \ ,$$

which is satisfied for $i = 14$. Thus, the first key is $k = (14, 5)$, while the second key is $k' = (15, 6)$

(b) The MAC scheme can be formally defined as follows: given a pseudorandom function $F_k : \{0,1\}^n \to \{0,1\}^n$ and a message $m \in \{0,1\}^n$ (where $n = 2t$ is an even number), and denoting by m_L, m_R respectively the left and the right half of m (each of length $t \in \mathbb{N}$), the authentication tag a is computed as $\mathsf{Mac}_k(m) = F_k(m_L \oplus m_R)$. A possible forgery attack is the following: once a pair (m, a) has been observed, a is also a valid tag for the message $m' = m_R || m_L$, that is, the original message m with the two halves inverted. This is due to the fact that, since the XOR is commutative, the argument to the pseudorandom function F_k does not change if we invert the order of the summands.

(c) By definition, beside CCA-security, an authenticated encryption (AE) also provides unforgeable encryptions.

Exercise 6 – Hash Functions [10 points]

(a) **[3 points]** The hash function MD5 (which has been deprecated since at least 2008) has a digest length of 128 bits. How many input messages do we approximately need to hash before finding a collision with a probability of at least 50%? Would it be good enough for today's standards, in terms of security from birthday attacks?

(b) **[4 points]** Let $H : \{0,1\}^* \to \{0,1\}^n$ be a collision-resistant hash function. We define the new hash function H' as follows: given a message m, we parse it in two halves, i.e. $m = m_0||m_1$, then we compute $H'(m) = H(m_0) \oplus H(m_1)$. Is H' also collision-resistant? If not, show how an attacker can produce a collision.

(c) **[3 points]** Briefly describe the random oracle model, with its pros and cons. Name one concrete example where the random oracle model is necessary to prove the security of a construction.

Solution.

(a) Using the birthday bound, we would need to hash approximately $2^{\frac{n}{2}}$ messages before finding a collision with a probability of at least 50%, where n is the digest length, so 128. Thus, we would need to hash 2^{64} messages. This is too low by today's security standard, where usually we require a security level of at least 2^{80}.

(b) H' is not collision resistant, since an attacker can easily produce a collision by taking the two messages $m = m_0||m_1$ and $m' = m_1||m_0$. Similarly to the insecurity of the MAC scheme of Exercise 5, the XOR is commutative, which means that if we swap the two halves of the digests and XOR them together, we obtain the same digest.

(c) The random oracle model is an idealized model for hash functions, which is useful to prove the security of other cryptographic constructions, where the collision-resistance property of hash functions from the standard model is not sufficient. In essence, we assume the hash function H to behave as an oracle such that, every time that an input is submitted to it, a uniformly random string is given as an output.

The main advantage of this model is that if a construction has been proven secure under the assumption that H is a RO, and later weaknesses are found in H, one does not have to throw away the entire construction, but just replace H with another hash function that is assumed to behave as a RO. The main disadvantage is that there is no formal justification that a security proof in the RO model guarantees security in a real-world deployment of a cryptographic construction.

An example of a concrete construction whose security relies on the RO model is RSA-OAEP (Optimal Asymmetric Encryption Padding), which gives a CCA-secure instantiation of RSA under the assumption that the two underlying hash functions are modelled as random oracles.

Exercise 7 – Primality Testing and Group Generators [10 points]

(a) **[4 points]** Find a generator for the multiplicative group $\mathbb{Z}_{13}^*$. Motivate your answer by showing how to check the order of the generator.

(b) **[6 points]** Consider the number $p = 67$. How many witness for the primality of p do we need to know that p is composite with a probability less than 5%? Further, perform the Fermat's primality test on p for one witness a of your choice, showing the calculation in detail with the square-and-multiply algorithm.

Solution.

(a) Given $p = 13$, the divisors of $p - 1 = 12$ are $1, 2, 3, 4, 6$ and 12. Thus, by Lagrange theorem any element in $\mathbb{Z}_{13}^*$ has one of these orders. To find a generator of this group, we have to find an element of order 12. The element $g = 2$ has this property: in fact, we have that

$$\begin{aligned} 2^1 &\mod 13 = 2 \neq 1 \\ 2^2 &\mod 13 = 4 \neq 1 \\ 2^3 &\mod 13 = 8 \neq 1 \\ 2^4 &\mod 13 = 3 \neq 1 \\ 2^6 &\mod 13 = 12 \neq 1 \end{aligned}$$

Thus, since the order of 2 is not any of the proper divisors of 12, by Lagrange's theorem its order must be 12.

(b) The probability that p is composite after finding k witnesses for its primality is $\frac{1}{2^k}$. Thus, if we want our probability to be less than 5%, we have to find $k = 5$ witnesses, since $\frac{1}{2^5} = 0.03125 < 0.05$.

Let us take $a = 2$. To verify that this is a witness for the primality of 67, we apply the square-and-multiply algorithm to compute the congruence $a^{p-1} \equiv 1 \mod p$, where $p = 67$. The binary expansion of $p - 1 = 66$ is 1000010. Setting $c = 1$, this yields the following calculations:

b=1 so $c = c^2 \cdot a \pmod p = 1^2 \cdot 2 \pmod{67} = 2$

b=0 so $c = c^2 \pmod p = 2^2 \pmod{67} = 4$

b=0 so $c = c^2 \pmod p = 4^2 \pmod{67} = 16$

b=0 so $c = c^2 \pmod p = 16^2 \pmod{67} = 55$

b=0 so $c = c^2 \pmod p = 55^2 \pmod{67} = 10$

b=1 so $c = c^2 \cdot a \pmod p = 10^2 \cdot 2 \pmod{67} = 66$

b=0 so $c = c^2 \pmod p = 66^2 \pmod{67} = 1$

Exercise 8 – RSA [10 points]

(a) **[3 points]** Given an RSA setup with public modulus n, public exponent e, and private exponent d, describe how to sign a message $m \in \mathbb{Z}_n$ and how to later verify the produced signature. In particular, describe what elements are needed by the signer and by the verifier in order to sign a message and verify a signature, respectively.

(b) **[4 points]** Given primes $p = 5$ and $q = 11$, public modulus $n = pq$, and public exponent $e = 3$, sign the message $m = 5$. Show all intermediate calculation steps.

(c) **[3 points]** In RSA, $e = 3$ is a common choice for the public exponent. Another common value is $e = 65537$. Explain why these are commonly used values (Hint: think about how we perform the encryption/verification operation from an algorithmic point of view).

Solution.

(a) To sign a message m, the signer needs access to the private key. The signature is then $s = m^d \pmod n$. To verify the signature s for a message m, the verifier needs access to the public key. The verifier accepts the signature if and only if $s^e \pmod n$ is equal to m. Whoever has access to the private key, can sign any message.

(b) We compute the modulus as $n = pq = 55$, then $\phi(n) = (p-1)(q-1) = 40$, and the private exponent as $d = e^{-1} \pmod{\phi(n)} = 3^{-1} \pmod{40}$. To compute this inverse, we can use the extended Euclidean algorithm: since we know that $\gcd(e, \phi(n)) = 1$, that means there exist d, a such that $ed + a\phi(n) = 1$, thus $ed - 1 = a\phi(n)$, so $\phi(n) \mid ed - 1$, that means $ed = 1 \pmod{\phi(n)}$, and by definition d is the inverse of e modulo $\phi(n)$. We find d (and a) by performing the extended Euclidean algorithm:

$$\mathbf{40} = 13 \cdot \mathbf{3} + \mathbf{1} \qquad \text{[eq1]}$$

from which we immediately get that

$$\mathbf{1} = \mathbf{40} - 13 \cdot \mathbf{3} \qquad \text{[isolating } \mathbf{1} \text{ in eq1]}$$

So $d = -13 = 27 \pmod{\phi(n)}$. Now, we can sign the message $m = 5$, just by computing $s = m^d \pmod n = 5^{27} \pmod{55}$. To do this, we can use the square and multiply algorithm. We write 27 in binary as 11011, we initialize $c = 1$ then by starting from the MOST significant bit, at each position, we square c and, if the bit is 1, we also multiply it by m, so:

b=1 so $c = c^2 m \pmod n = 1^2 \cdot 5 \pmod{55} = 5$

b=1 so $c = c^2 m \pmod n = 5^2 \cdot 5 \pmod{55} = 15$

b=0 so $c = c^2 \pmod n = 15^2 \pmod{55} = 5$

b=1 so $c = c^2 m \pmod n = 5^2 \cdot 5 \pmod{55} = 15$

b=1 so $c = c^2 m \pmod n = 15^2 \cdot 5 \pmod{55} = 25$

So the corresponding signature is $s = 25$.

(c) The reason why 3 and 65537 are commonly used values is related to the efficiency of the square-and-multiply algorithm: in particular, we perform a square and multiply operation only when the corresponding bit in the exponent is set to 1; otherwise, we only perform squaring. The binary expansions of 3 and 65537 are respectively 011 and 10000000000000001. Thus, in each case only two square and multiply operations are done, which means that encryption (or signature verification) can be performed very efficiently.

Exercise 9 – Diffie-Hellman Key Agreement [10 points]

We consider the following variant of the Diffie-Hellman Key Agreement protocol. Given a finite cyclic group G (where the DLOG is assumed to be hard) and a generator g, Alice picks a random exponent a uniformly at random, computes $A = g^a$ and sends it to Bob. Bob picks a random exponent b uniformly at random, computes $B = g^b$ and sends it to Alice. Now, Alice computes the final key as $k_A = B \cdot g^a$, while Bob computes the final key as $k_B = A \cdot g^b$.

(a) **[4 points]** Is the protocol correct? In other words, are Alice and Bob generating the same key?

(b) **[6 points]** Discuss the security of this Diffie-Hellman variant.

Solution.

(a) Yes, this variant of DH is correct since $k_A = B \cdot g^a = g^b \cdot g^a = g^{b+a} = g^{a+b} = g^a \cdot g^b = A \cdot g^b = k_B$.

(b) This variant of DH is not secure, since any eavesdropper who can access the public quantities A and B can simply generate the common key as $k = A \cdot B = g^a \cdot g^b = g^{a+b}$.

Exercise 10 – Schnorr Identification Scheme and Signatures [10 points]

(a) [**6 points**] Given $p = 17$ and the generator $g = 3$ for $\mathbb{Z}_{17}^*$, Alice wants to convince Bob that she knows the DLOG of $y = 5$. Alice thus sends to Bob the value $t = 13$ as a commitment, while Bob sends back to Alice the challenge $c = 2$. Finally, Alice replies to Bob with $s = 14$. Does Bob believe Alice?

(b) [**4 points**] Briefly explain how the Schnorr identification scheme used above can be turned into a (non-interactive) signature scheme. In particular, what additional assumption do we need beside the conjectured hardness of the discrete logarithm problem to prove the security of the signature scheme?

Solution.

(a) Bob computes:

- $g^s \pmod p = 3^{14} \pmod{17} = 2$
- $t \cdot y^c \pmod p = 13 \cdot 5^2 \pmod{17} = 2$

since the values match, Bob believes Alice.

(b) The Schnorr identification scheme (or any interactive identification scheme, in general) can be turned into a non-interactive signature scheme by applying the Fiat-Shamir transform. The signature of a message $m \in \{0,1\}^*$ works as follows:

1. Compute $I := g^k \mod q$, where $k \leftarrow \mathbb{Z}_q$
2. Compute $r := H(I, m)$
3. Compute $s := [rx + k \mod q]$
4. Output the signature (r, s)

Where H is a hash function. The verification, on the other hand, of a message m together with its signature (r, s) works as follows:

1. Compute $I := g^s \cdot y^{-r}$
2. Check if $H(I, m) = r$. If so, the signature is accepted, otherwise, it is rejected.

In particular, the security of the scheme relies on the fact that H is modelled as a random oracle.