

# Exercises for CO409 Cryptography Engineering

Exercises 54 and 55 are questions from past Exams of this Course.

**Exercise 1** Let  $x \oplus y$  denote the exclusive or defined over bit-vectors of length  $n > 0$ . Show that  $\oplus$  is

1. commutative: for all  $x$  and  $y$  we have  $x \oplus y$  equals  $y \oplus x$
2. associative: for all  $x, y, z$  we have that  $(x \oplus y) \oplus z$  equals  $x \oplus (y \oplus z)$
3. idempotent: for all  $x$  we have that  $x \oplus x = 0$  where  $0$  is the bit-vector of  $n$  zeros
4.  $0$  is a unit: for all  $x$  we have that  $0 \oplus x$  equals  $x \oplus 0$  which equals  $x$
5. for all  $x$  and  $y$  we have that  $(x \oplus y) \oplus y = x$
6. all equations  $a \oplus x = b$  have a unique solution  $x$

In particular,  $\oplus$  makes bit-vectors of length  $n$  into a commutative group with identity element  $0$  and where each element is its own inverse.

**Exercise 2** Let  $\mathbb{P} = \{a, b, c, d\}$ ,  $\mathbb{C} = \{1, 2, 3, 4\}$ , and  $\mathbb{K} = \{k_1, k_2, k_3\}$ . Let  $p(P = a) = 0.2$ ,  $p(P = b) = 0.3$ ,  $p(P = c) = 0.1$ , and  $p(P = d) = 0.4$ . Let  $p(K = k_1) = 0.3 = p(K = k_2)$  and  $p(K = k_3) = 0.4$ . Finally, the encryption  $e_k(c)$  for this is given in the table

	$a$	$b$	$c$	$d$
$k_1$	4	2	3	1
$k_2$	3	2	4	1
$k_3$	1	2	4	3

1. Compute  $p(C = c)$  for each  $c$  in  $\mathbb{C}$ .
2. Compute  $p(C = c \mid P = m)$  for each pair  $(c, m)$  in  $\mathbb{C} \times \mathbb{P}$ .

3. Compute  $p(P = a \mid C = 2)$  and  $p(P = c \mid C = 3)$ .
4. Explain why the above cryptosystem is not perfectly secure.

**Exercise 3** Recall the definition of perfect secrecy from our textbook: “for all plaintexts  $m$  and all ciphertexts  $c$ , we have that  $P(P = m \mid C = c)$  equals  $p(P = m)$ .”

Prove that this definition is equivalent to the one based on “for all plaintexts  $m$  and all ciphertexts  $c$ , we have that  $P(C = c \mid P = m)$  equals  $p(C = c)$ .”

**Exercise 4** For a perfectly secure cryptosystem, prove that we always have  $\#\mathbb{K} \geq \#\mathbb{C} \geq \#\mathbb{P}$ , where  $\#A$  denotes the number of elements in a finite set  $A$ .

**Exercise 5** Prove the second part of Theorem 9.4 in our textbook, Shannon’s Theorem: assume that

1. every key is used with equal probability  $1/\#\mathbb{K}$ , and
2. for each  $m$  in  $\mathbb{P}$  and  $c$  in  $\mathbb{C}$ , there is a unique  $k$  in  $\mathbb{K}$  such that  $e_k(m) = c$ .

Use these assumptions to prove that the cryptosystem has perfect secrecy.

**Exercise 6** Let us reconsider the one-time pad in the setting where Alice encrypts binary strings of length  $n > 1$  to be sent to Bob. Assume that Eve can intercept such encrypted messages but that Eve cannot control which messages Alice will encrypt. However, you may assume that Alice will encrypt more than one message with the same key  $k$ .

Explain what Eve can learn about two ciphertexts encrypted with the same key.

**Exercise 7** Show equation (10) in our textbook on page 172:

$$(8) \quad H(K \mid C) = H(K) + H(P) - H(C)$$

and discuss the significance of this equation in relation to cryptanalysis.

**Exercise 8** Recall the design of a symmetric key encryption schemes that relied on the use of a Feistel function for round-based encryption, with DES being a good example thereof.

Assume that encryption and decryption operate in three rounds, and show the correctness of this scheme: the decryption of ciphertext recovers correctly the plaintext.

**Exercise 9** Specify the decryption operation for the OFB mode of operation of a block cipher and show that it correctly decrypts ciphertext back to plaintext.

**Exercise 10** Specify the decryption operation for the CFB mode of operation of a block cipher and show that it correctly decrypts ciphertext back to plaintext.

**Exercise 11** In this exercise, assume the use of a block cipher  $e_k(\cdot)$  on a plaintext  $m$  with  $k$  data blocks  $m_1 m_2 \dots m_k$ , and the resulting ciphertext of  $k$  blocks  $c_1 c_2 \dots c_k$ .

1. Let one bit in block  $c_i$  of ciphertext  $c$  be flipped prior to decryption. Which portions of plaintext will be affected, and how, when decrypting  $c$  in ECB mode?
2. Repeat the previous exercise for CBC mode.
3. Suppose that a single bit is inserted in some block  $c_i$  of ciphertext and the last bit of ciphertext is deleted. Which portions of the plaintext will be affected, and how, when decrypting the modified ciphertext in ECB mode?

4. Repeat the previous exercise for CBC mode.

**Exercise 12** *A stream cipher is a device that generates a keystream  $k$  that is as long as the message  $m$ , and then encrypts and decrypts in the same manner as done with the one-time pad. However, the keystream is not completely random but generated from a seed and a state-based feedback loop (where the details vary from design to design and are rooted in mathematics). Assume that these designs are such that the keystream will eventually be periodic.*

*Informally discuss the advantages and disadvantages of stream ciphers in comparison to block ciphers, based on the above and your understanding of block ciphers.*

**Exercise 13** *Discuss the possible meanings of computationally infeasible along various dimensions such as mathematical or complexity-theoretic concepts, abilities and resources of an attacker, resiliency to future technological developments, etc.*

*For sake of concreteness, you may consider one or all of the three properties that cryptographically secure hash functions should possess.*

**Exercise 14** 1. *Assume that we have an oracle  $\mathcal{O}$  that, given  $y$  in the image of a hash function  $h$ , produces for us some legitimate input  $x$  of  $h$  such that  $h(x) = y$ . You may assume that calls to the oracle are efficient. Show that you can use this oracle to efficiently break second preimage resistance of  $h$ .*

2. *Show that the ability to break second preimage resistance can be used to also break collision resistance of a hash function.*

**Exercise 15** 1. *Show that the  $h$  in Algorithm 14.1 on page 276 of our textbook is not collision resistant. HINT: let  $s = 8$ ,  $n = 4$ , and consider  $m_1 = 010$  and as  $m_2$  a “padded” version of  $m_1$ .*

2. Assume that Algorithm 14.1 on page 276 is modified so that, before it divides the message into blocks it adds another block that encodes the length of the original, unpadded message in bits.

- (a) Show that this means that messages can have at most  $2^l$  bits.
- (b) Assume that function  $f$  is collision resistant. Use this to show that the collisions found for  $h$  in the previous item are no longer collisions.

**Exercise 16** We saw that one can build a MAC function out of a hash function  $h$ . Show that such a design

$$MAC_K(m) = h(m \parallel K)$$

is not secure, so appending a key to the message before hashing must not be used for computing MACs.

*Hint:* assume that hash function  $h$  is iterated, e.g. as in the Merkle-Damgard construction, and that  $h$  is not given in a length-strengthened version. Further assume that you can find two messages  $m_1 \neq m_2$  of equal length that give a collision of  $h$  (e.g. by relying on a birthday attack). Also assume that the key  $K$  has the same length as a data block for  $h$ . Now show that the MAC of  $m_1$  equals the MAC of  $m_2$ .

**Exercise 17** Discuss the benefits and pitfalls of using “padding method zero” and “padding method three” from page 285 in the textbook. You may also want to look up the term “reversible padding scheme” on the web to inform this discussion.

**Exercise 18** This exercise is about CBC-MAC. Let  $m = m_1m_2 \dots m_q$  be a message of  $q$  many data blocks, where  $m_q$  is padded as in padding method zero from page 285 in the textbook. Let

$$M = \text{CBC-MAC}_K(m)$$

Show that

$$M = \text{CBC-MAC}_K(m \parallel M \oplus m_1 \parallel m_2m_3 \dots m_q)$$

**Exercise 19** *This exercise is also about CBC-MAC. It shows that even the padding from padding method three can result in a length-extension attack.*

*Assume: the attacker has MAC  $M_1$  of message  $m_1$ , and has MAC  $M_2$  of message  $m_2$ , where both  $m_1$  and  $m_2$  are just a single data block.*

*Notation: we write  $\mathbb{P}(n)$  for a single data block that encodes in binary the natural number  $n$ ; this of course limits the size of  $n$  by the block size.*

*Now show the following:*

1.  $M_1$  equals  $e_K(e_K(m_1) \oplus \mathbb{P}(b))$  where  $b$  is the block size of the block cipher  $e_K(\cdot)$ .

2. Let  $m_3$  be a new data block. Then  $\text{CBC-MAC}_K(m_1 \parallel \mathbb{P}(b) \parallel m_3)$  equals

$$e_K(e_K(e_K(e_K(m_1) \oplus \mathbb{P}(b)) \oplus m_3) \oplus \mathbb{P}(3b))$$

3. Let the three-block message  $m$  be  $m_2 \parallel \mathbb{P}(b) \parallel m_3 \oplus M_1 \oplus M_2$ . Then we have

$$\text{CBC-MAC}_K(m) = \text{CBC-MAC}_K(m_1 \parallel \mathbb{P}(b) \parallel m_3)$$

**Exercise 20** 1. Describe how you generate a key pair for RSA encryption. In doing so, choose prime numbers  $p$  and  $q$  that are larger than 10 but smaller than 100, which is of course unrealistically small.

2. State which parts of this RSA setup should be secret, and explain why that is so.

3. State the set of plaintexts and the set of ciphertexts for the RSA key pair you computed in the first item above.

4. Then choose a plaintext, show details of its encryption, and explain why and how the corresponding ciphertext decrypts to the original plaintext.

**Exercise 21** Recall the definition of the Euler totient function  $\phi(n)$  for a natural number  $m \geq 2$ : it is defined as the size of the set  $\{a \mid 1 \leq a < m \mid \gcd(a, m) = 1\}$ :

$$\phi(m) = \#\{a \mid 1 \leq a < m \mid \gcd(a, m) = 1\}$$

For example, we have  $\phi(9) = 6$  as only the numbers 1, 2, 4, 5, 7, 8 from  $\{1, 2, \dots, 8\}$  are relative prime to 9.

1. Compute  $\phi(60)$ .
2. Let  $p$  and  $q$  be primes. Prove that  $\phi(p \cdot q)$  equals  $(p-1) \cdot (q-1)$ . (Note that we used this identity in the definition of RSA.)
3. Show that an attacker Eve can compute the private RSA key  $d$  if she knows the value of  $\phi(N)$ . This means that the security of RSA also depends on whether or not computing  $\phi(N)$  is a hard problem.

**Exercise 22** Let  $N \geq 2$  be a natural number and let  $\mathbb{Z}_N^*$  be the set of natural numbers  $a$  with  $1 \leq a < N$  such that  $\gcd(a, N) = 1$ .

1. Show that for all  $a$  in  $\mathbb{Z}_N^*$  there is some  $b$  in  $\mathbb{Z}_N^*$  such that  $a \cdot b = 1 \pmod N$ . (Hint: use the facts around the Extended Euclid algorithm for example.)
2. Assume that  $N$  equals  $p \cdot q$  for primes  $p$  and  $q$ . Show that  $\mathbb{Z}_N^*$  has  $\phi(N)$  many elements.
3. Use the fact shown in the previous item to prove that RSA decryption successfully recovers the plain text. (Hint: we basically did this in the notes already but we did not explain why)

$$(m^{(p-1)(q-1)})^s = 1 \pmod N$$

**Exercise 23** Let  $(e, N)$  and  $(d, N)$  be an RSA key pair. A message  $m$  is called *unconcealed* iff

$$m^e = m \pmod N$$

1. Show that there are unconcealed messages  $m$  that are unconcealed independent of the choice of  $N$ .
2. Assume that there are  $(\gcd(e-1, p-1)+1) \cdot (\gcd(e-1, q-1)+1)$  many unconcealed plain-text messages  $m$ . Discuss whether this is a concern for the security of RSA and, if so, how implementations can mitigate or prevent such security concerns.

**Exercise 24** *Prove that the RSA problem is no harder than the FACTORING problem. Concretely, consider an RSA system with a modulus  $N$  and a key pair for that modulus and then show the following:*

*“Knowing the factors  $p$  and  $q$  of the modulus  $N$ , allows an attacker to efficiently compute the private RSA key, and to decrypt all ciphertext of that RSA system.”*

**Exercise 25** *Consider an RSA key pair  $(N, e)$  and  $(N, d)$  where  $d$  is the secret key and  $N = p \cdot q$  for primes  $p$  and  $q$ . Below, we write  $c = m^e \bmod N$  to identify the ciphertext of  $m$ .*

*Let  $\text{parity}(c)$  denote the least significant bit of  $m$ . Let  $\text{half}(c)$  be 0 if  $0 \leq m \leq N/2$ , and set  $\text{half}(c) = 1$  otherwise.*

*Below you should complete staged exercises that will prove that any algorithm that computes  $\text{parity}(c)$  or  $\text{half}(c)$  from the ciphertext  $c$  of  $m$  can be used as an oracle to build an algorithm that computes  $m$  from its ciphertext:*

- 1. Show that  $\text{half}(c) = \text{parity}(c \cdot 2^e \bmod N)$ .*
- 2. Show that  $\text{parity}(c) = \text{half}(c \cdot (2^{-1})^e \bmod N)$ .*
- 3. Conclude that the computations of  $\text{parity}(c)$  and  $\text{half}(c)$  are polynomially equivalent: one can convert an algorithm for computing one of these into an algorithm for computing the other one – with polynomial overhead in that conversion.*
- 4. Let  $\mathcal{O}$  be an oracle such that  $\mathcal{O}(c)$  outputs  $\text{half}(c)$ . Design an algorithm that uses this oracle, has ciphertext  $c$  as input, and computes the corresponding plain-text  $m$  efficiently (assuming that calls to the oracle take constant time).*
- 5. Conclude that the security of RSA also depends on the hardness of computing the least significant bit of a plaintext, given its ciphertext.*



**Exercise 26** *More efficient RSA decryption and RSA signing:* Let  $(N, e)$  and  $(N, d)$  be an RSA key pair with a private key  $d$  and  $N = p \cdot q$  for primes  $p$  and  $q$ . The operation  $c^d \bmod N$  is of course used for either decryption or for signing a message, depending on whether the key pair is used as a cryptosystem or as a digital signature system.

Here is a technique that makes this operation more efficient and is therefore often implemented in practice. First, we precompute the values

$$d_p = d \bmod p-1 \qquad d_q = d \bmod q-1 \qquad q_{inv} = q^{-1} \bmod p$$

Then, to compute  $m = c^d \bmod N$  we compute

$$\begin{aligned} m_1 &= c^{d_p} \bmod p & m_2 &= c^{d_q} \bmod q \\ h &= q_{inv} \cdot (m_1 - m_2) \bmod p & m &= m_2 + h \cdot q \end{aligned}$$

1. Show that the definitions above indeed compute the plaintext  $m$  that satisfies  $m = c^d \bmod N$ .
2. Explain why this new computation of  $m$  may be more efficient than computing  $c^d \bmod N$ , keeping in mind that  $p$  and  $q$  would have at least 1024 bits.

**Exercise 27** *Illustrate the workings of the Diffie-Hellman key exchange protocol where  $p$  equals 701 and  $g$  equals 220:*

1. If Alice chooses  $a = 254$  at random, what concrete value will she send to Bob?
2. If Bob chooses  $b = 478$  at random, what concrete value will he send to Alice?
3. Given the choices of  $a$  and  $b$  above, what key will Alice and Bob agree upon at the end of the protocol?

**Background information:** The  $g$  above is such that it generates the entire group  $\mathbb{Z}_{701}^*$ . Such elements are called *primitive roots*. Whenever  $p$  is prime, the group  $\mathbb{Z}_p^*$  has primitive roots, and these can be computed efficiently. Please see

<http://www.bluetulip.org/programs/primitive.html>

for an online calculator of primitive roots for small prime values. For example, it informs us that there are 240 different primitive roots for  $p = 701$ , and it lists them all for us.

It turns out that the use of primitive roots leaks at least one bit of information. This is a reason, apart from efficiency, why one would like to pick a  $g$  that generates a large but proper subgroup of  $\mathbb{Z}_p^*$ . One technique is to use primes  $p$  of form  $2q + 1$  for some prime  $q$ . It is then not hard to find  $g$  whose subgroup has order  $q$ .

**Exercise 28** Assume that we use an RSA signature scheme based on a hash function  $h$  and key pairs  $(e, N)$  and  $(d, N)$  as follows: The signature of a message  $m$  is  $s = h(m)^d \bmod N$  and verifiers receive the pair  $(m, s)$ . Such a pair is verified in three steps:

- Compute  $h' = s^e \bmod N$ , an ‘encryption’
- Compute  $h(m)$  from  $m$
- Accept the signature if and only if  $h(m)$  equals  $h'$ .

Analyze what capabilities an attacker might have when the following properties of the used hash function  $h$  are not true:

1. preimage resistance
2. collision resistance
3. second preimage resistance

**Exercise 29** Let  $\mathcal{P} = \{A, B, C, D\}$  be a set of parties and let  $m(\Gamma)$  be the set of all subsets of  $\mathcal{P}$  is size 2. Assume that the secret  $s$  is a binary string of length  $l$ .

1. Compute shares  $s_P$  for each party  $P$  in  $\mathcal{P}$ , using the Ito-Nishizeki-Seito secret sharing scheme.
2. Informally explain why this scheme is information-theoretically secure for this example. Specify any assumptions needed for this argument.

**Exercise 30** Let  $\mathcal{P}$  be  $\{A, C, B, D\}$  and let

$$m(\Gamma) = \{\{A, C, D\}, \{A, B\}, \{B, C\}, \{B, D\}\}$$

Let  $s$  be a secret represented as a binary string of length  $l$ . Develop the Replicated Secret Sharing Scheme for this access structure:

1. Compute all maximally non-qualifying sets  $\mathcal{O}_i$ .
2. Compute the complements  $B_i$  of all sets in the previous item.
3. Explain how the secret is split up into shares  $s_i$ , in particular specify which strings are generated randomly.
4. Compute the shares for each party.
5. Explain why no information can be computed from a combination of the shares in the non-qualifying set  $\{A, D\}$ .

**Exercise 31** Consider the following set-up for the Reed-Solomon code:  $q$  equals 101,  $n$  equals 7,  $t$  equals 2, and  $X$  equals  $\{1, 2, 3, 4, 5, 6, 7\}$ .

1. Let  $f$  in  $\mathcal{P}$  be given by  $f(X) = 26 + 67X + 93X^2$ . Compute the Reed-Solomon code  $c$  for this  $f$  and the above setting.
2. Let  $Y = \{1, 4, 7\}$ . Let  $s_i = f(i)$  for all  $i$  in  $X$ . Explain in detail how you can recover from  $\{s_1, s_4, s_7\}$  the secret  $f(0) = 26$ .

**Exercise 32** Consider an arithmetic variant of the Ito-Nishizeki-Saito Secret Sharing Scheme: in the same manner as for that scheme, we can write a monotone access structure  $\Gamma$  over a set of parties  $\mathcal{P}$  as a Boolean formula  $\phi_{m(\Gamma)}$  that is in DNF form, contains no negation, and has parties from  $\mathcal{P}$  as atomic propositions.

Let  $m$  be a large integer such that the secret  $s$  is a number in  $\{0, 1, \dots, m\}$ . For each disjunct  $C_i$  of  $\phi_{m(\Gamma)}$ , we do the following. Let  $A_i^1, \dots, A_i^{k_i}$  be the parties that occur in  $C_i$  in that order. We randomly generate numbers  $a_i^j$  in  $\{0, 1, \dots, m\}$  for all  $1 \leq j < k_i$  and let  $a_i^j$  be the share of  $A_i^j$  for  $C_i$ . For  $A_i^{k_i}$  we set the share for  $C_i$  to be  $s - \sum_{l=1}^{k_i-1} a_i^l$ . The share of a party  $A$  in  $\mathcal{P}$  is the set of all shares that it has for clauses  $C_i$  in which  $A$  occurs.

In the exercises below, you can understand how this scheme works by means of a simple example. Let  $\mathcal{P} = \{A, B, C, D\}$  and

$$m(\Gamma) = \{\{A, B, D\}, \{A, C, D\}, \{B, C\}\}$$

Let the secret  $s$  be an integer with  $0 \leq s \leq m$  for a large positive integer  $m$ . Then clause  $C_1$  is  $A \wedge B \wedge D$ , clause  $C_2$  is  $A \wedge C \wedge D$ , and clause  $C_3$  is  $B \wedge C$ . The parties that occur in clause  $C_1$  are  $A$ ,  $B$ , and  $D$ . Etc.

1. Write down the shares for each party.
2. Show that all  $\mathcal{O}$  of  $\Gamma$  can recover the secret.
3. Explain why no  $\mathcal{O}$  with  $\mathcal{O} \notin \Gamma$  can recover the secret.

**Exercise 33** Let  $p$  be an odd prime (i.e.  $p \neq 2$ ),  $x$  a natural number with  $1 \leq x \leq p-1$ . Then  $x$  is a **quadratic residue modulo  $p$**  iff the equation  $y^2 = x \pmod{p}$  has a solution  $y$  in  $\mathbb{Z}_p$ .

Fermat's Little Theorem states that  $b^{p-1} = 1 \pmod{p}$  for prime  $p$  and all  $b$  in  $\mathbb{Z}_p^*$ . Use that theorem to prove that one can decide, in polynomial time, whether  $x$  is a quadratic residue modulo  $p$ :

1. Show that if  $x$  is a quadratic residue modulo  $p$ , then  $x^{(p-1)/2} = 1 \pmod{p}$ .
2. Conversely, show that if  $x^{(p-1)/2} = 1 \pmod{p}$ , then  $x$  is a quadratic residue modulo  $p$ .
3. Describe a polynomial-time decision procedure for quadratic residues modulo  $p$ .
4. Now suppose that, instead of a prime  $p$  we have  $N = p \cdot q$  for large primes  $p$  and  $q$  but where  $p$  and  $q$  are not known. Do you think that we can efficiently decide whether  $x$  is a quadratic residue modulo  $N$ ?

**Exercise 34 RSA Factoring and Computing Square roots.** Let  $N = p \cdot q$  for two large primes  $p$  and  $q$ , and let  $(N, e)$  and  $(N, d)$  be the public and private key, respectively. Consider the equation  $x^2 = 1 \pmod{N}$ .

1. Use the Chinese Remainder Theorem (CRT) to show that there are four solutions to this equation.
2. Explain that two of these four solutions are always the same values, these are called trivial solutions.
3. Suppose that  $x$  is a non-trivial solution for  $x^2 = 1 \pmod{N}$ . Show that you can now factor  $N$ , knowing the value of  $x$ .
4. Give an example of small values  $p$  and  $q$  that illustrates the three points above.
5. It can be shown that any oracle for computing a secret key  $d$  from a public key  $(N, e)$  can be used to compute (with a so called Las Vegas algorithm) a non-trivial square root of 1 modulo  $N$ . Discuss what ramifications this fact has for the usage of RSA.

**Exercise 35** Let  $N = p \cdot q$  for large primes  $p$  and  $q$ . Let  $m$  be a quadratic residue modulo  $N$ , i.e. the equation  $r^2 = m \pmod{N}$  has a solution in  $r$ . Assume that Peggy knows neither the factorization of  $N$  nor a square root of  $m$  modulo  $N$ . Let  $X = \mathbb{Z}_N^*$  and  $Y = QR(N)$ , the latter being the set of elements of  $\mathbb{Z}_N$  that are quadratic residues modulo  $N$ . We define a *bit commitment scheme* (a scheme that commits to a single bit  $b$ ) as follows:

$$f: \{0, 1\} \times X \rightarrow Y, \quad f(b, x) = m^b \cdot x^2 \pmod{N}$$

1. Show that  $f$  is well defined.
2. Show that the bit commitment scheme is information-theoretically concealing, and at best computationally binding.
3. Show that, if Peggy could compute squares roots of quadratic residues modulo  $N$ , then she could completely break the binding of this bit commitment scheme.

**Exercise 36** Consider the two graphs and an isomorphism  $f$  between them in Figure 1, taken from [en.wikipedia.org/wiki/Graph\\_isomorphism](https://en.wikipedia.org/wiki/Graph_isomorphism). In the notation of zero knowledge proofs, let  $G_1$  be the  $G$  from the figure,  $G_2$  the  $H$  from the figure, and  $\phi: G_1 \rightarrow G_2$  be the  $f$  from the figure.

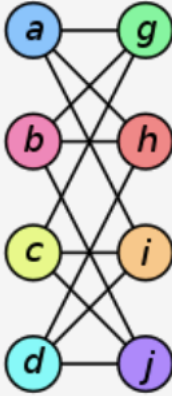
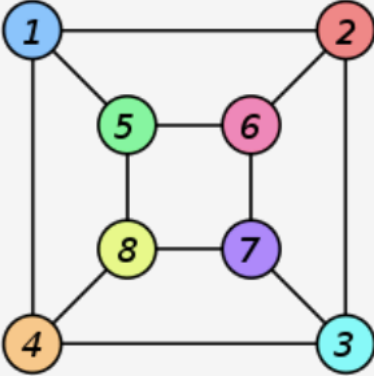
Graph G	Graph H	An isomorphism between G and H
		$f(a) = 1$ $f(b) = 6$ $f(c) = 8$ $f(d) = 3$ $f(g) = 5$ $f(h) = 2$ $f(i) = 4$ $f(j) = 7$

Figure 1: Two isomorphic graphs and an isomorphism between them

1. Describe a round of a zero knowledge proof in which Peggy wants to prove to Victor that she knows an isomorphism between  $G_1$  and  $G_2$ . That round should use the isomorphism from the figure and the challenge of Victor should be  $b = 1$ .
2. Show that Victor would accept the proof of the round in the above item.
3. Let the challenge of Victor now be  $b = 2$ . How does the round now differ, and why does Victor still accept the proof?

**Exercise 37** Show that the decision problem of “graph isomorphism” is in NP. That is to say, one can check efficiently whether a claimed isomorphism between two graphs is indeed an isomorphism.

**Exercise 38** Suppose that Peggy plays honestly in the protocol for zero knowledge proofs of graph isomorphism (in particular, she knows a graph isomorphism  $\phi: G_1 \rightarrow G_2$ ), and that she generates the same  $H$  on two different runs of this protocols.

Show that Victor can then learn the graph isomorphism between  $G_1$  and  $G_2$  that Peggy knows (and that forms the basis of her protocol moves).

**Exercise 39** A graph  $G = (V, E)$  is *k-colorable* for  $k > 1$  if there is a function  $\phi: V \rightarrow \{1, 2, \dots, k\}$  such that for all  $(v, w)$  in  $E$  we have  $\phi(v) \neq \phi(w)$ .

1. Consider the graph  $G$  in Figure 1. Show that  $G$  is 2-colorable.
2. Find a graph that is 3-colorable but not 2-colorable.

**Exercise 40** In the proof of Theorem 21.1, it speaks of the “obvious” simulation for the zero-knowledge proof of 3-colorability. Describe such a simulation for a round of that zero-knowledge proof.

**Exercise 41** Prove that the Chaum-Pederson Sigma Protocol satisfied soundness, assuming an honest verifier.

**Exercise 42** Consider the standard notation for the Sigma Protocol of the Pedersen Commitments, where Peggy wants to prove that  $y = g_1^{x_1} \cdot g_2^{x_2}$  where  $g_1$  and  $g_2$  are publicly known element of order  $q$  in a group  $G$  of prime order  $q$ , and  $x_1$  and  $x_2$  are known to Peggy:

$$\begin{aligned} R(x, (k_1, k_2)) &:= (r_1, r_2) \\ S(e, (x_1, x_2), (k_1, k_2)) &:= (s_1, s_2) \\ V((r_1, r_2), e, (s_1, s_2)) &:= \text{true} \Leftrightarrow (g_1^{s_1} \cdot g_2^{s_2} = y^e \cdot r_1 \cdot r_2) \\ S'(e, (s_1, s_2)) &:= (r_1, g_1^{s_1} \cdot g_2^{s_2} \cdot y^{-e} \cdot r_1^{-1} \pmod q) \end{aligned}$$

where  $r_i = g_i^{k_i}$  for random  $k_i$  in  $\mathbb{Z}_q$ ,  $s_i = k_i + e \cdot x_i \pmod q$ , and  $r_1$  is a random element of  $G$ .

1. Prove completeness of this protocol.
2. Assume an honest verifier. Further assume that Peggy does not cheat in two runs of this protocol in which her commitments are the same, the challenges are different, but Victor accepts both runs. Show that Victor can now extract all secret information.
3. Why does the argument from the previous item *not* show the special soundness property?
4. Prove zero knowledge of this protocol.

1. Peggy chooses a random  $v$  in  $\mathbb{Z}_N^*$ , computes  $y = v^2 \bmod N$ , and sends  $y$  to Victor as her commitment.
2. Victor chooses a random bit  $b$  from  $\{0, 1\}$  and sends this to Peggy as his challenge.
3. Peggy computes  $z = u^b \cdot v \bmod N$  where  $u$  is a square root of  $x$  modulo  $N$ , and sends  $z$  to Victor as response.

Figure 2: A round in this zero knowledge proof that Peggy knows a quadratic residue modulo  $N$ .

**Exercise 43** Let  $N = p \cdot q$  for large primes  $p$  and  $q$ . Assume that Peggy does not know the factorization of  $N$ . But Peggy knows that  $x$  is a quadratic residue modulo  $N$ , so she knows  $u$  with  $u^2 = x \bmod N$ . She wishes to construct a zero knowledge proof to convince Victor that  $x$  is a quadratic residue modulo  $N$ , without revealing anything about  $x$  to Victor.

The protocol proceeds in  $\lfloor \log_2 N \rfloor$  rounds. In each round the steps from Figure 2 are executed.

Victor then verifies that  $z^2 = x^b \cdot y \bmod N$  and, if so, accepts that round. As always, Victor accepts the proof if he accepts all  $\lfloor \log_2 N \rfloor$  rounds.

1. Show that this protocol is complete.
2. Show that this protocol is sound. Specifically, show that Peggy's probability of deceiving Victor in all  $\lfloor \log_2 N \rfloor$  rounds is  $1/N$ .
3. Show that the protocol has the zero-knowledge property.

**Exercise 44** Consider the protocol we discussed that a voter (as Peggy) uses to prove to Victor that she submitted a vote  $x$  in  $\{-1, 1\}$  but where Victor cannot learn the value of  $x$ .

1. Show that, if Peggy plays honestly in the protocol, then Victor will accept the protocol run.
2. Discuss in what sense this protocol proves that Peggy knows  $\text{dlog}_g(B_a(x) \cdot h)$  or knows  $\text{dlog}_g(B_a(x) \cdot h^{-1})$ , and why this would demonstrate to Victor that she has committed to some  $x$  in  $\{-1, 1\}$ .



3. If Peggy committed to some  $x$  not in  $\{-1, 1\}$ , she will find it hard to respond to Victor's challenge so that Victor will accept the protocol run.
4. The protocol reveals no information to any party as to the exact value of Peggy's commitment, except that it came from  $\{-1, 1\}$ .

**Exercise 45** A Boolean circuit may also contain NOT gates. The garbled tables we defined for gates with two input wires need to be adjusted to work for NOT gates, which have only one input wire.

1. Define how to garble the table for a NOT gate with input wire  $w_i$  and output wire  $w_j$ .
2. Illustrate your garbled table for the case when party A uses  $\rho_i = 0$  and  $\rho_j = 1$ .
3. What potential problems do you see in using NOT gates in secure multi party computations? Can you suggest how to address some of these problems?

**Exercise 46** The protocol for secure two-party computation with garbled Boolean circuits uses oblivious transfer protocols so that party B learns key halves for each of her input wires. We also saw that this protocol always puts party B into the possession of two key halves (and no more) for each garbled table in the circuit.

1. Discuss why it may be a problem if party B does not know which of the four cipher-texts these two key halves correspond to.
2. Explain why the protocol allows party B to uniquely identify the correct ciphertext amongst the four candidates for the next decryption in the secure two-party computation.

**Exercise 47** Consider the Boolean function

$$f(\{w_1, w_4\}, \{w_2, w_3\}) = (w_1 \wedge w_2) \oplus \neg(w_3 \vee w_4) \quad (1)$$

where  $\{w_1, w_4\}$  is the private input of party A and  $\{w_2, w_3\}$  is the private input of party B.

1. Draw a Boolean circuit that implements function  $f$  above and only uses gates for AND, NOR, and XOR. Make sure that all wires have names.

2. Suppose that party  $A$  generates random keys  $\rho_1 = \rho_2 = \rho_5 = 0$  and  $\rho_3 = \rho_4 = \rho_6 = \rho_7 = 1$  and that the input values  $v_i$  for input wires  $w_i$  are  $v_1 = v_4 = 0$  and  $v_2 = v_3 = 1$ .

(a) Write down the corresponding garbled circuit in the style seen in Figure 22.2 on page 443 of our textbook.

(b) Show the progress of the secure Two-Party Computation, in the style seen in Figure 22.3 on page 444 of our textbook.

**Exercise 48** Consider a finite field  $\mathbb{F}_q$  where  $q = p^k$  for some prime  $p \geq 2$  is prime. The *characteristic* of  $\mathbb{F}_p$  is the minimal  $n$  such that  $\sum_{i=1}^n 1 = 0$ . One can show that the characteristic of  $\mathbb{F}_q$  is  $p$ .

Now suppose that  $p > 2$ . Show that all Boolean Circuits can be faithfully encoded as polynomials over  $\mathbb{F}_q$ :

1. First show that all Boolean operators can be represented using only constants 0, 1, the exclusive-or  $\oplus$ , and the AND operator denoted here as  $\odot$ .
2. Show how one can encode  $\oplus$  faithfully as a polynomial in two variables over  $\mathbb{F}_q$ .
3. Show how one can encode  $\odot$  faithfully as a polynomial in two variables over  $\mathbb{F}_q$ .

**Exercise 49** Consider the elliptic curve

$$E : Y^2 = X^3 + X + 3$$

and the finite field  $K = \mathbb{F}_7$ .

1. Show that  $(4, 1)$  is in  $E(\mathbb{F}_7)$ .
2. Show that  $(4, 6)$  is in  $E(\mathbb{F}_7)$ .
3. Compute  $(4, 1) + (4, 6)$ .
4. Compute  $(4, 1) + (6, 6)$ .
5. Compute  $4 \star (4, 1)$ .

**Exercise 50** Let  $K$  be a finite field and  $E$  an elliptic curve in the short Weierstrass form

$$E : Y^2 = X^3 + aX + b$$

for constants  $a$  and  $b$ . Let  $+$  denote the point addition defined through the chord-tangent process. Assume that there is an implementation of this addition operation  $+$  and of  $\lambda P \in E(K) : 2 \star P$ , that is of  $\lambda P \in E(K) : P + P$ . Let  $k$  be an  $l$ -bit natural number  $> 2$  with binary representation

$$k = \sum_{j=0}^{l-1} k_j \cdot 2^j \quad (k_j \in \{0, 1\})$$

Write an algorithm that takes such a vector  $(k_j)_{j=0}^{l-1}$  and a point  $P$  in  $E(K)$  as input, outputs  $k \star P$ , and has running time logarithmic in  $k$ .

**Exercise 51** Consider an elliptic curve

$$E : Y^2Z = X^3 + XZ^2 + 3Z^3$$

in the projective plane, and a short Weierstrass form elliptic curve

$$E : Y^2 = X^3 + X + 3$$

1. Let  $K$  be the finite field  $\mathbb{F}_7$ .

(a) Show that  $(1, 5, 2)$  is in  $E(\mathbb{F}_7)$ .

(b) Show that  $(1/2, 5/2)$  is in  $E'(\mathbb{F}_7)$ .

2. Show that for all fields  $K$  and for all  $(X, Y, Z) \in \mathbb{P}^2(K)$  with  $Z \neq 0$  we have that  $(X, Y, Z) \in E(K)$  implies that  $(X/Z, Y/Z) \in E'(K)$ . That is to say, we can transform points on  $E(K)$  to points on  $E'(K)$ .

**Exercise 52** Recall the Replicated Secret Sharing Scheme.

1. Find an example of a monotone access structure  $\Gamma$  over a set of parties  $\mathcal{P}$  such that

$$\mathcal{B} = \{\mathcal{P} \setminus \mathcal{O} \mid \mathcal{O} \text{ maximally non-qualifying}\}$$

is disjoint from  $m(\Gamma)$ , the set of minimal elements of  $\Gamma$  with respect to subset inclusion.

2. Find an example for  $\mathcal{B} \cap m(\Gamma) = \emptyset$  in which  $\mathcal{B}$  equals the set of maximally non-qualifying sets.
3. Find an example for  $\mathcal{B} \cap m(\Gamma) = \emptyset$  in which  $\mathcal{B}$  has less elements than  $m(\Gamma)$ .

**Exercise 53** Let  $p = 2q + 1$  be a safe prime, and so  $q$  is prime as well. Let  $g \in \mathbb{Z}_p^*$  be an element such that  $g \not\equiv 1 \pmod{p}$  but where  $g$  is a quadratic residue modulo  $p$ .

Show that the subgroup generated by  $g$  in the multiplicative group  $\mathbb{Z}_p^*$  has order (i.e. size)  $q$ .

**Exercise 54 (Q1 of 409 Exam in 2016, Shannon's notion of perfect secrecy)**

Let  $\mathbb{P} = \{a, b, c, d, e\}$ ,  $\mathbb{C} = \{1, 2, 3, 4, 5\}$ , and  $\mathbb{K} = \{k_1, k_2, k_3, k_4, k_5\}$ . The probability distribution  $p(P = p)$  for plain-texts  $p$  is given by

	$a$	$b$	$c$	$d$	$e$
$p(P = p)$	0.31	0.15	0.21	0.06	0.27

Let  $p(K = k_i) = 0.2$  for all  $1 \leq i \leq 5$ . Finally, the encryption  $e_k(m)$  for this cryptosystem is given in the table

	$a$	$b$	$c$	$d$	$e$
$k_1$	4	2	1	3	5
$k_2$	5	1	4	2	3
$k_3$	1	3	2	5	4
$k_4$	2	4	5	1	3
$k_5$	2	5	3	4	1

1. Compute  $p(C = 3)$  and  $p(C = 5)$ .
2. Compute  $p(C = 3 \mid P = a)$  and  $p(P = c \mid C = 3)$ .
3. Explain why the above cryptosystem is not perfectly secure in the sense of Shannon.
4. The table for  $e_k(m)$  above has 25 entries from  $\mathbb{C}$ . Without changing  $\mathbb{P}$ ,  $\mathbb{C}$ , and  $\mathbb{K}$ , make a minimal number of changes to these 25 entries so that the resulting cryptosystem is perfectly secure in the sense of Shannon.

5. Suppose you need to engineer a perfectly secure cryptosystem for GlobalCorp, for communications between its headquarter in Singapore and its British office.

Briefly discuss what cryptosystem you would choose, and sketch what security and implementation issues such a choice would entail.

**Exercise 55 (Past Exam Question)** Let  $p$  be a large prime of  $\geq 1024$  bits, and also a safe prime, i.e.  $p = 2 \cdot q + 1$  where  $q$  is a prime as well. Let  $G$  be the abelian group with elements in  $\{1, 2, \dots, p-1\}$ , group operation  $x \cdot y \bmod p$ , and unit element 1. Let  $\mathbb{P} = \mathbb{C} = \{1, 2, \dots, p-1\}$ .

Below are the first three steps of a protocol in which Alice wants to send a message  $m \in \mathbb{P}$  to Bob in encrypted form over an insecure communication channel:

- (S1) Alice generates a random natural number  $x_A$  co-prime to  $p-1$  and sends Bob  $a$ , defined as  $m^{x_A} \bmod p$ .
- (S2) Bob generates a random natural number  $x_B$  co-prime to  $p-1$  and sends Alice  $b$ , defined as  $a^{x_B} \bmod p$ .
- (S3) Alice computes  $x_A^{-1} \bmod p-1$  and sends Bob  $a'$ , defined as  $b^{x_A^{-1}} \bmod p$ .

1. Briefly explain why Alice and Bob are able to perform the computations of these three steps (S1)-(S3), and efficiently so.
2. Specify what Bob does in the last step (S4) to recover the message  $m$ .
3. Assume that Alice and Bob follow steps (S1)-(S4) and messages are not tampered with in transit. Explain why your step (S4) correctly recovers  $m$ .
4. Is this protocol subject to the man-in-the-middle attack? If not, explain why not. If it is, describe such an attack and any assumptions that it requires to succeed.
5. Considering that  $p$  is a safe prime, explain why  $\mathbb{P}' = \{2, 3, \dots, p-2\}$  is a better set of plaintexts for the above protocol. **Hint:** Lagrange's Theorem says that the size of any sub-group  $H$  of group  $G$  must be a divisor of the size of  $G$ .