Introduction to Cryptography: exam

8 July, 2021

Instructions: You can score a maximum of 100 points and you have 180 minutes to solve all nine problems. Each question indicates how many points it is worth. You are **not** allowed to use any books/slides/notes/*etc.*, nor a smart phone or any device. Please write clearly and **explain your answers**. Each exercise is independent and they can be solved in the order of your choice. The formula sheet can be found attached at the end of the exam. Do not forget to **put your name and student number on each sheet.**

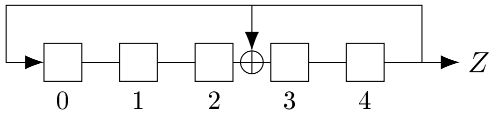
1. (**10 points**) **Symmetric cryptography fill-in question.** In the following text, fill in the missing notions. There is no explanation necessary.

   In symmetric cryptography, there are two approaches for encryption: $\underline{\quad(a)\quad}$ and block encryption. Block encryption is done by setting a block cipher into a mode, e.g., Electronic Codebook mode, or $\underline{\quad(b)\quad}$. When you instantiate a block cipher with a fixed key, it becomes a $\underline{\quad(c)\quad}$, therefore it has an inverse. You can also use block ciphers to create MAC-functions, for instance, $\underline{\quad(d)\quad}$.

   In symmetric cryptography, hash functions are often used. Classical hash functions like MD5, SHA-1 and SHA-2 are constructed by using the $\underline{\quad(e)\quad}$ construction. These hash functions using the (e)-construction can be differentiated from their ideal by exploiting the $\underline{\quad(f)\quad}$ property.

   The newer hash function SHA-3 makes use of the sponge construction. The sponge construction uses a $\underline{\quad(g)\quad}$ as a primitive. The sponge has several parameters, the output length n, the permutation length b, $\underline{\quad(h)\quad}$ and $\underline{\quad(i)\quad}$. The security strength of a good sponge-based hash function against preimage attacks is the minimum of the output length n and $\underline{\quad(j)\quad}$.

2. (**9 points**) **LFSR security.** Consider the following linear feedback shift register. We view it as a stream cipher, where the key is the initial state of the LFSR.



   (a) Give the upper bound for the security strength when only considering exhaustive key search. 2 pt

   (b) Consider the output stream $Z = (z_0, z_1, z_2, z_3, z_4) = \texttt{00110}$. Determine the current state (i.e., the state s^t at $t = 5$) of the LFSR. 4 pt

   (c) Give the next five output bits, i.e., the output bits z_5, z_6, z_7, z_8, z_9. [If you did not manage to solve (b), you can try to solve (c) assuming the answer to (b) would be $s^5 = \texttt{10010}$.] 3 pt

3. **(9 points) Message authentication code.** In this question, we are confronted with bad MAC design. We have an n-bit key K. Let $\mathrm{B}_K \colon \{0,1\}^n \to \{0,1\}^n$ be a PRP secure block cipher. Consider messages m of length $3n$ bits. Write $m = m_1 \| m_2 \| m_3$ where $|m_1| = |m_2| = |m_3| = n$. The symbol $\|$ denotes concatenation of strings. Our MAC function is defined as:

$$\mathrm{MAC}_K(m) = \mathrm{B}_K(\overline{m_1}) \oplus \mathrm{B}_K(m_3) \| \mathrm{B}_K(\overline{m_2} \oplus m_3).$$

   Consider a generation query of the form $m_1 \| 1^n \| m_3$, that provides you with the tag $T = t_1 \| t_2$.

(a) Give explicit expressions for t_1 and t_2. [Hint: Your expressions will likely use B_K.] 2 pt

(b) Show how you can make a forgery for this MAC function without making any other generation queries. [Show that a verification query on your forgery outputs that the tag is valid. Hint: You can distill more information from the generation query than just the values t_1 and t_2.] 7 pt

4. (**12 points**) **Distinguishing ECB mode.** Consider ECB mode instantiated with AES, denoted as $\mathrm{ECB}[\mathrm{AES}_K]$. We consider the ideal version of $\mathrm{ECB}[\mathrm{AES}_K]$ to be a scheme that uses a variable-length random permutation $v\mathcal{RP}$. This variable-length random permutation is a collection of random permutations for each input length ℓ. The ideal scheme is denoted as $v\mathcal{RP}^*$, which pads the input before running it through $v\mathcal{RP}$:

> The $v\mathcal{RP}^*$ performs the following steps for some input that is queried:
>
> - It is padded with 10^*-padding to obtain a message of length 128ℓ, for some ℓ.
> - The result is input for a random permutation on 128ℓ bits.
> - The final result is the outcome of your query.

(a) Give a distinguisher $\mathcal{D}$ that distinguishes $\mathrm{ECB}[\mathrm{AES}_K]$ from a $v\mathcal{RP}^*$. 4 pt

(b) Give the probability that you draw the conclusion that it is $\mathrm{ECB}[\mathrm{AES}_K]$, but in actuality, it is $v\mathcal{RP}^*$. 3 pt

(c) Give the advantage of your distinguisher. 2 pt

(d) Compute the security strength of $\mathrm{ECB}[\mathrm{AES}_K]$ considering your distinguisher. [The unit of computation e.g., for M and N, is expressed in the number of 128-bit blocks handled by the device. The costs for checking whether blocks have certain value can be seen as negligible.] 3 pt

5. (**10 points**) EMAES**, a new great block cipher! But how secure is it?**

In 1977, NIST standardized DES, and in 2001, they standardized its successor AES, so it looks like they will publish a successor to AES in 2025, and that would logically be called EMAES (an Even More Advanced Encryption Standard). We make some speculation on what that cipher will look like and ask you about its security strength against some attacks. We assume the following about EMAES:

- Its block length is 256 bits.
- Its key length is 150 bits.
- Its security goal is to be PRP secure. It is not designed to be SPRP secure.

(a) PRP and SPRP security are about distinguishing a block cipher from an ideal counterpart. Give that ideal counterpart and explain the difference between PRP and SPRP security. 2 pt

(b) Explain why PRP security is appropriate in counter mode and SPRP security in CBC mode. 1 pt

(c) How can you attack EMAES with exhaustive key search? Explain the attack mentioning online encryption queries and offline encryption queries. 2 pt

(d) Give the security strength of EMAES against exhaustive key search. 1 pt

(e) Explain how you can create a distinguisher for PRP security with the exhaustive key search attack. 1 pt

(f) Consider a *multi-target* attack: suppose we have d encryption devices, which implement EMAES with each a different secret key. The goal of the attacker is to find one of these keys. Give the security strength of EMAES against such an attack for $d = 1024 = 2^{10}$. 2 pt

(g) Give the maximum value of d for which EMAES still offers 128 bits of security against a multi-target attack with d targets. 1 pt

6. (**10 points**) **Public-key fill-in question.** In the following text, fill in the missing notions. There is no explanation necessary.

In this course, we have treated two types of groups used for public-key cryptography: (multiplicative) modular groups, and $\underline{\quad\text{(a)}\quad}$ groups. For schemes based on the hardness of $\underline{\quad\text{(b)}\quad}$, e.g., (Merkle-)Diffie-Hellman, subgroups of multiplicative modular groups such as $(\mathbb{Z}/p\mathbb{Z})^*$ (where p is a large prime modulus) and (a) are used, and these need to have a large prime order q. For schemes based on the hardness of factoring large numbers, e.g., (textbook) RSA encryption, the modulus n of the multiplicative modular group $(\mathbb{Z}/n\mathbb{Z})^*$ needs to be a product of $\underline{\quad\text{(c)}\quad}$.

To obtain 128 bits of security, order q needs to be roughly $\underline{\quad\text{(d)}\quad}$ bits long. To break protocols such as the (Merkle-)Diffie-Hellman key agreement protocol, which security depends on the hardness of (b), an attacker can try to compute the private key corresponding to a given public key. To do this, the attacker can use generic attacks such as $\underline{\quad\text{(e)}\quad}$, $\underline{\quad\text{(f)}\quad}$ and/or Pohlig-Hellman. For multiplicative modular groups in particular, an attacker can also use $\underline{\quad\text{(g)}\quad}$ attacks, which are the reason that p needs to be much larger than q.

While the security of the textbook RSA encryption and signature schemes depend on the hardness of factoring, they are not secure in practice. Instead, one can use slightly altered versions of these schemes, e.g., use $\underline{\quad\text{(h)}\quad}$ for encryption, which is IND-CPA secure, and $\underline{\quad\text{(i)}\quad}$ for signatures, which is secure against forgeries. Both these versions employ a hash function. If these hash functions were to be replaced by a $\underline{\quad\text{(j)}\quad}$, then these schemes would be secure.

7. (**15 points**) **Elliptic curves.** Consider the elliptic curve $\mathcal{E} : y^2 = x^3 + x + 5$ over $\mathbb{F}_{23}$, such that $\#\mathcal{E}(\mathbb{F}_{23}) = 22$. (Remark: $22 = 2 \cdot 11$.)

(a) Show that $Q := (18, 6)$ lies on the curve $\mathcal{E}$. 1 pt

(b) Compute $[2]Q$. 4 pt

(c) What are the possible orders of a point R in $\mathcal{E}(\mathbb{F}_{23})$? [Hint: Lagrange's Theorem.] 2 pt

(d) Consider $Q = (18, 6)$, that is not a generator of the *entire* group $\mathcal{E}(\mathbb{F}_{23})$. What is the order of Q? 3 pt

(e) Give homogeneous projective coordinates for Q. 2 pt

(f) Consider the point $\pi_{\mathbb{P}} = (18 : 15 : 6)$ in homogeneous projective coordinates. Give the point $\pi_{\mathbb{P}}$ in affine coordinates as $\pi = (x, y)$. 3 pt

8. (**10 points**) **Discrete logarithm.** Compute $x = \text{dlog}_4(8)$ in $(\mathbb{Z}/23\mathbb{Z})^*$ (i.e., compute the smallest positive integer x such that $4^x \equiv 8 \pmod{23}$). You should solve this by using Pollard's ρ algorithm, or the baby-step giant-step (BSGS) algorithm. For the BSGS algorithm, you should take $m = \lfloor\sqrt{q}\rfloor$. [Hint: The order q of 4 in $(\mathbb{Z}/23\mathbb{Z})^*$ is 11.]

9. **(15 points) A composite protocol.** Let p be a large prime number of 3072 bits, and let $g \in (\mathbb{Z}/p\mathbb{Z})^*$ be an element of order q, where q is a large prime number of 256 bits. We assume that $\langle g \rangle$ provides 128 bits of security with respect to the decisional Diffie-Hellman problem. Let $\mathrm{h}_1 \colon \{0,1\}^* \to \{0,1\}^{128}$ and $\mathrm{h}_2 \colon \{0,1\}^* \to (\mathbb{Z}/p\mathbb{Z})^*$ be two hash functions, and let Enc_K and Dec_K denote the encryption and decryption algorithms, respectively, with key K of some symmetric encryption scheme. Consider the protocol in Figure 1, which is a composite of two protocols or schemes that we have treated in the course.

Alice		Bob
$p, g, q, A = g^a, a,$ (Bob: B)		$p, g, q, B = g^b, b,$ (Alice: A)
$v \overset{\$}{\leftarrow} \mathbb{Z}/q\mathbb{Z},\ V \leftarrow g^v$		
$a' \overset{\$}{\leftarrow} \mathbb{Z}/q\mathbb{Z},\ A' \leftarrow g^{a'}$		
$K \leftarrow \mathrm{h}_1(\text{“KDF”}; B^{a'})$		
$CT \leftarrow \mathrm{Enc}_K(M)$		
$c \leftarrow \mathrm{h}_2(p; g; A; V; \mathrm{Bob}; M)$		
$r \leftarrow v - ca$		
	$\xrightarrow{\text{Alice}, (CT, A'), (V, r)}$	$K' \leftarrow \mathrm{h}_1(\text{“KDF”}; (A')^b)$
		$M' \leftarrow \mathrm{Dec}_{K'}(CT)$
		$c' \leftarrow \mathrm{h}_2(p; g; A; V; \mathrm{Bob}; M')$
		$V \overset{?}{=} g^r A^{c'}$

Figure 1: The composite protocol

(a) Which key establishment protocol or public-key encryption scheme is used in the composite protocol? 2 pt

(b) Give at least one advantage of the protocol or scheme in (a) compared to textbook RSA encryption. 2 pt

(c) Which signature scheme is used in the composite protocol? 2 pt

(d) In the course, we have treated three security properties related to authentication protocols: completeness, soundness and (honest-verifier) zero-knowledgeness. Explain which of these three properties protect the protocol or scheme in (c) against forgery attacks. 4 pt

(e) Show that the scheme is correct, i.e., that indeed $V = g^r A^{c'}$ holds if Bob performs the steps as expected. [Hint: First show that $M' = M$ holds.] 3 pt

(f) Suppose that Bob has received $(CT, A'), (V, r)$ from Alice, and that message M is retrieved by decrypting (CT, A'). Now, he wants to send message M to Charlie with the composite protocol, pretending that it came from Alice. For encryption (i.e., with the protocol or scheme in (a)), Bob uses the (authenticated) public key of Charlie: $B' = g^{b'}$, resulting in the ciphertext (CT', A''). Explain why Bob cannot send the resulting $(CT', A''), (V, r)$ to Charlie, and successfully pretend that it came from Alice. [Hint: Does the equation on the right-hand side of the protocol verify correctly?] 2 pt