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: __(a)__ and block encryption. Block encryption is done by setting a block cipher into a mode, e.g., Electronic Codebook mode, or __(b)__. When you instantiate a block cipher with a fixed key, it becomes a __(c)__, therefore it has an inverse. You can also use block ciphers to create MAC-functions, for instance, __(d)__.

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

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

   Solution:

   (a) stream encryption;

   (b) Cipher Block Chaining; (Counter and Output Feedback modes are stream encryption modes)

   (c) permutation;

   (d) CBC-MAC; (/C-MAC)

   (e) Merkle-Damgård;

   (f) length-extension;

   (g) permutation;

   (h) (the) rate r; (/capacity c)

   (i) (the) capacity c; (/rate r)

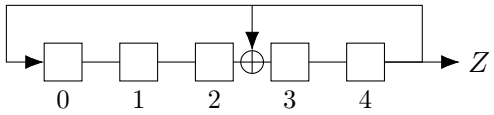
   (j) half the capacity. ($/c/2$)

   [[Grading Instruction:

Grading (total 10):
1 for each correct notion

   ]]

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

Solution:

(a) The key (initial state) consists of 5 bits, the asked security strength is 5 bits.

(b) We iterate the LFSR with an initial state `abcde`, substituting as we go:

$\boxed{0}$	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	$\boxed{4}$	$\boxed{Z}$	
a	b	c	d	e	0	$e = 0$
0	a	b	c	d	0	$d = 0$
0	0	a	b	c	1	$c = 1$
1	0	0	$a+1$	b	1	$b = 1$
1	1	0	1	$a+1$	0	$a = 1$
0	1	1	0	1	1	

Hence, the current state s^t at $t = 5$ is `01101`.

(c) To find the next five output bits, `10010` we continue iterating the LFSR:

$\boxed{0}$	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	$\boxed{4}$	$\boxed{Z}$
0	1	1	0	1	1
1	0	1	0	0	0
0	1	0	1	0	0
0	0	1	0	1	1
1	0	0	0	0	0

OR:

Using the alternate current state, we find `01011`:

$\boxed{0}$	$\boxed{1}$	$\boxed{2}$	$\boxed{3}$	$\boxed{4}$	$\boxed{Z}$
1	0	0	1	0	0
0	1	0	0	1	1
1	0	1	1	0	0
0	1	0	1	1	1
1	0	1	1	1	1

[[Grading Instruction:

Grading (total 9):	
aspect:	points
(a)	**2**
answer	1
explanation	1
(b)	**4**
answer	2
computation	2
(c)	**3**
answer	1
computation	2

]]

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

Solution:

(a) The tag is generated by the MAC function. You get $T = \mathrm{B}_K(\overline{m_1})\oplus\mathrm{B}_K(m_3)\|\,\mathrm{B}_K(\overline{1}\oplus m_3) = \mathrm{B}_K(\overline{m_1}) \oplus \mathrm{B}_K(m_3)\|\,\mathrm{B}_K(m_3)$. Therefore $t_1 = \mathrm{B}_K(\overline{m_1}) \oplus \mathrm{B}_K(m_3)$ and $t_2 = \mathrm{B}_K(m_3)$.

(b) By the hint we can find $t_1 \oplus t_2 = \mathrm{B}_K(\overline{m_1})$. By setting $m_2' = m_1$ and $m_3 = 0^n$, we get $t_2' = \mathrm{B}_K(\overline{m_1})$. Then we have $t_1' = \mathrm{B}_K(\overline{m_1'}) \oplus \mathrm{B}_K(0)$. We can then take $m_1' = 1^n$ to have a verification query that works with tag $t_1'\|t_2'$.

Verification query. We query the verification oracle with $(1^n\|m_1\|0^n, 0^n\|t_1 \oplus t_2)$.

Proof: We have

$$\begin{aligned}\mathrm{MAC}_K(1^n\|m_1\|0^n) &= \mathrm{B}_K(\overline{1^n}) \oplus \mathrm{B}_K(0^n)\|\,\mathrm{B}_K(\overline{m_1} \oplus 0^n)\\ &= \mathrm{B}_K(0^n) \oplus \mathrm{B}_K(0^n)\|\,\mathrm{B}_K(\overline{m_1})\\ &= 0^n\|t_1 \oplus t_2\end{aligned}$$

as required.

OR:

By the hint, we can find $t_1 \oplus t_2 = \mathrm{B}_K(\overline{m_1})$. By setting $m_2' = 1$ and $m_3' = \overline{m_1}$ we get $t_2' = \mathrm{B}_K(\overline{1^n} \oplus \overline{m_1}) = \mathrm{B}_K(\overline{m_1})$. Then we have $t_1' = \mathrm{B}_K(\overline{m_1'}) \oplus \mathrm{B}_K(\overline{m_1})$. We can then take $m_1' = \overline{m_3}$ to have a verification query that works with tag $t_1'\|t_2'$.

Verification query. We query the verification oracle with $(\overline{m_3}\|1^n\|\overline{m_1}, t_1\|t_1 \oplus t_2)$.

Proof: We have

$$\begin{aligned}\mathrm{MAC}_K(\overline{m_3}\|1^n\|\overline{m_1}) &= \mathrm{B}_K(\overline{\overline{m_3}}) \oplus \mathrm{B}_K(\overline{m_1})\|\,\mathrm{B}_K(\overline{1^n} \oplus \overline{m_1})\\ &= \mathrm{B}_K(m_3) \oplus \mathrm{B}_K(\overline{m_1})\|\,\mathrm{B}_K(\overline{m_1})\\ &= t_1\|t_1 \oplus t_2\end{aligned}$$

as required.

[[Grading Instruction:

Grading (total 9):	
aspect	points
(a)	**2**
each	1
(b)	**7**
Verification query	3
Working out hint	1
Giving message	1
Giving tag	1
Proof/Explanation	4

]]

4. (**12 points**) **Distinguishing ECB mode.** Consider ECB mode instantiated with AES, denoted as ECB[AES$_K$]. We consider the ideal version of ECB[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 ECB[AES$_K$] from a $v\mathcal{RP}^*$. 4 pt

(b) Give the probability that you draw the conclusion that it is ECB[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 ECB[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

Solution:

(a) Query: We query a plaintext of the form $P\|P$ where $|P| = 128$, to obtain $C_1\|C_2\|C_3$, since padding is applied to $P\|P$.

Decision: If $C_1 = C_2$, the distinguisher decides that it is speaking with ECB[AES$_K$], otherwise, the $v\mathcal{RP}^*$.

Proof: Since ECB[AES$_K$] works on blocks and is deterministic, if the first two plaintext blocks are equal, then the first two ciphertext blocks will be equal.

For the $v\mathcal{RP}^*$ this is not the case, as it is random on the full input.

Other distinguishers may also be just as good. For example, querying $P\|\operatorname{unpad}(P)$, where only $M = 2$ changes in (d).

(b) The probability that $C_1 = C_2$ is 2^{-128} since there are 2^{128} possibilities for C_2, while only one of them is equal to C_1.

(c) The advantage for the distinguisher can be computed as

$$\mathrm{Adv}_{\mathcal{D}} = |\Pr(\mathcal{D} = 1 \mid \mathrm{ECB}) - \Pr(\mathcal{D} = 1 \mid v\mathcal{RP}^*)| = |1 - 2^{-128}| = 1 - 2^{-128}$$

(d) Since the number of blocks queried online (M) is 3, and we do no offline computation, the security strength is upper bounded by $s = \lg(\frac{3}{1-2^{-128}}) = \lg(3) - \lg(1 - 2^{-128}) \approx 1 - 0 = 1$, so around 1 bit of security. (Or at most 2 bits of security.)

[[Grading Instruction:

Grading (total 12):	
aspect:	points
(a)	**4**
query	1
decision	1
proof	2
(b)	**3**
answer	1
explanation	2
(c)	**2**
answer	1
advantage formula right	1
computation	1
(d)	**3**
answer	1
explanation M	1
explanation Adv	1

]]

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

Solution:

(a) (S)PRP is to distinguish the block cipher keyed with a fixed and unknown key from a random permutation. The adversary knows the specification of the block cipher and can make encryption queries to the block cipher/permutation and make computations. In SPRP she can also make decryption queries.

(b) PRP is appropriate in modes that do not use the inverse because an adversary that is attacking a mode that does not use the inverse cannot exploit weaknesses due to inverse block cipher calls. In modes that do use the inverse an adversary may be able to exploit such weaknesses and hence SPRP is appropriate.

(c) You encrypt some plaintext m to a ciphertext c with an online query. Then do offline calls to EMAES with plaintext m and key guesses until c is obtained. Then you conclude that the current key is the key that is used.

(d) After N attempts we have success probability $p = 2^{-150}N$ and so $(N + M)/P = (N + 1)/N2^{-150} \approx 2^{150}$ so the security strength is $\lg(2^{150}) = 150$ bits.

(e) You find the key with exhaustive key search in the case of a block cipher. In the case of a random permutation it is likely there is no key, but it is possible. If there is no key, it is a random permutation. Otherwise, you do a single additional online query: the block cipher behaves according to EMAES specifications and the random permutation does not.

(f) You encrypt some plaintext m to the corresponding ciphertexts c_i for all the d instances. This has online cost $M = d$. Then do offline calls to EMAES with plaintext m and key guesses until a ciphertext is obtained that is equal to one of the c_i values. Then you conclude that the current key is the corresponding key. The success probability for a single key guess is $d2^{-150}$ and for N key guesses approximately $Nd2^{-150}$. The security strength is therefore $\lg(d+N)/Nd2^{-150}) = 150 + \lg((1+d/N)/d)$. As N grows, this converges to $150 - \lg(d)$. Filling in $d = 2^{10}$ gives a security strength of about 140 bits. Answers that give that expression will be considered correct.

(g) $s = 150 - \lg(d)$, so for $s = 128$ we get $128 = 150 - \lg(d)$ giving $d = 2^{150-128} = 2^{22}$ so approximately 4 million.

[[Grading Instruction:

Grading (total 10):	
aspect:	points
(a)	**2**
random permutation	1
decryption queries	1
(b)	**1**
modes with no decr.	1
(c)	**2**
1 online query	1
offline queries till success	1
(d)	**1**
value	1
(e)	**1**
explanation	1
(f)	**2**
explanation	1
value	1
(g)	**1**
value	1

]]

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 __(a)__ groups. For schemes based on the hardness of __(b)__, 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 __(c)__.

To obtain 128 bits of security, order q needs to be roughly __(d)__ 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 ___(e)___, ___(f)___ and/or Pohlig-Hellman. For multiplicative modular groups in particular, an attacker can also use ___(g)___ 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 ___(h)___ for encryption, which is IND-CPA secure, and ___(i)___ for signatures, which is secure against forgeries. Both these versions employ a hash function. If these hash functions were to be replaced by a ___(j)___, then these schemes would be secure.

Solution:

(a) elliptic-curve

(b) finding discrete logarithms / solving the computational/decisional Diffie-Hellman problem

(c) two (large) primes (numbers)

(d) 256

(e) baby-step giant-step / Pollard's ρ algorithm

(f) Pollard's ρ algorithm / baby-step giant-step

(g) Index calculus

(h) RSA-KEM / RSA-OAEP

(i) FDH-RSA

(j) (truncated) random oracle

[[Grading Instruction:

Grading (total 10):
1 for each correct notion

]]

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

Solution:

(a) We compute $18^3 + 18 + 5 \equiv -5 \cdot 25 \equiv -10 \equiv 13 \pmod{23}$, while $6^2 = 36 \equiv 13 \pmod{23}$.

(b) We compute the following:

$$\lambda = \frac{3x^2 + A}{2y} = \frac{3 \cdot 18^2 + 1}{2 \cdot 6} \equiv \frac{75 + 1}{12} \equiv \frac{7}{12} \equiv 7 \cdot 2 \equiv 14 \pmod{23}$$

$$x_{[2]Q} = \lambda^2 - 2 \cdot x_Q = 14^2 - 2 \cdot 18 \equiv 81 + 10 \equiv 22 \pmod{23}$$

$$y_{[2]Q} = -y_Q + \lambda(x_Q - x_{[2]Q}) = -6 + 14(18 - 22) \equiv 17 + 14 \cdot -4 \equiv 17 - 56 \equiv 7 \pmod{23}$$

Hence $[2]Q = (22, 7)$.

(c) By Lagrange's Theorem, we know that the order of a point $R \in \mathcal{E}(\mathbb{F}_{23})$ must be a divisor of 22. So the possible orders for R are $1, 2, 11, 22$.

(d) By the fact that Q does not generate the entire group, the order of Q is not 22. Since $Q \neq \mathcal{O}$, we find that the order of Q is not 1. Since $[2]Q \neq \mathcal{O}$, we find that the order of Q is not 2, hence the order of Q is 11.

OR:

By the fact that Q does not generate the entire group, the order of Q is not 22. Since $Q \neq \mathcal{O}$, we find that the order of Q is not 1. Since $y_Q \neq 0$, we do not have $Q = -Q$, hence the order of Q is not 2. Therefore the order of Q is 11.

(e) Since we can take $Z \neq 0$ as we choose, we find $Q_{\mathbb{P}} = (18 : 6 : 1)$.

(f) We compute $6^{-1} \equiv 4 \pmod{23}$. Then $\pi = (x, y) = (18 \cdot 4, 15 \cdot 4) = (3, 14)$.

[[Grading Instruction:

Grading (total 15):	
aspect:	points
(a)	**1**
(b)	**4**
λ, x_3, y_3 each	1
conclusion	1
(c)	**2**
mentioning divisors	1
correct divisors	1
(d)	**3**
not 1 (or 22)	1
not 2	1
conclusion	1
(e)	**2**
correct point (can vary)	1 or 2
explanation	1 or 0
(f)	**3**
computation 6^{-1}	1
computation/explanation	1
answer	1

]]

8. (**10 points**) **Discrete logarithm.** Compute $x = \mathrm{dlog}_4(8)$ 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.]

Solution:
Pollard-ρ: For this method, we construct the following table where we have $a_i = g^{b_i} \cdot h^{c_i}$ following the function

$$(a_{i+1}, b_{i+1}, c_{i+1}) = \begin{cases} (a_i \cdot g, b_i + 1, c_i) & \text{if } a_i \equiv 1 \pmod 3; \\ (a_i \cdot h, b_i, c_i + 1) & \text{if } a_i \equiv 2 \pmod 3; \\ (a_i^2, 2b_i, 2c_i) & \text{if } a_i \equiv 0 \pmod 3. \end{cases}$$

In this case, we have $g = 4$ and $h = 8$. Assume moreover that we start with $b_0 = 1$ and $c_0 = 0$.

i	0	1	2	3	4
a_i	4	16	18	2	16
b_i	1	2	3	6	6
c_i	0	0	0	0	1

We see that $a_1 = a_4$ which implies $g^2 \cdot h^0 \equiv g^6 \cdot h^1 \pmod{23}$. This translates to $4^2 \equiv 4^6 \cdot 8^1 \pmod{23}$, which in implies that $4^{-4} \equiv 8 \pmod{23}$. Since 4 has order 11, we have that

$$x \equiv -4 \equiv 7 \bmod 11.$$

Hence $x = \mathrm{dlog}_4(8) = 7$.

Baby-step giant-step: We have $q = 11$, so $m = \lfloor \sqrt{q} \rfloor = 3$. We get the tables with the Baby-Steps and Giant-Steps respectively:

i	0	1	2	3
g^i	1	4	16	18

j	0	1	2
$h \cdot g_0^j$	8	3	4

Note that in the Giant-Steps table, $g_0 := 4^{-3} \equiv 18^{-1} \equiv (-5)^{-1} \equiv 9 \pmod{23}$. Alternatively: $g_0 = 4^{-3} \equiv 6^3 = 36 \cdot 6 \equiv -10 \cdot 6 = -60 \equiv 9 \pmod{23}$.

So we find that the solution is $x = 1 + 2 \cdot 3 = 7$.

[[Grading Instruction:

Grading (total 10):	
Pollard's ρ:	points
Correctly stating/referring to the function for $(a_{i+1}, b_{i+1}, c_{i+1})$	**2**
Giving the correct table using the previous function until collision of a_i	**5**
Correct method of calculating $x = \mathrm{dlog}_4(8)$ from the table	**2**
Correct answer of $\mathrm{dlog}_4(8)$	**1**
	OR
Baby-Step Giant-Step:	points
Concluding that $m = \lfloor\sqrt{11}\rfloor = 3$	**1**
Calculating $g_0 \equiv 4^{-3} \equiv 9 \pmod{23}$ correctly	**2**
Giving correct Baby-Step table	**2**
Giving correct Giant-Step table	**3**
Correct calculation of $x = \mathrm{dlog}_4(8)$	**2**

]]

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

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

Solution:

(a) The key encapsulation mechanism (KEM) based on ElGamal.

(b) There are several advantages:

- No encoding issues
- More efficient due to the use of symmetric-key encryption
- IND-CPA secure
- Messages that are longer than one block can be securely encrypted
- More efficient key generation

(c) It is (a variation on) Schnorr's signature scheme.

(d) Soundness and zero-knowledgeness. Soundness protects against the generation of forgeries, as it ensures that someone who doesn't know the secret key cannot cheat the protocol and generate a valid transcript. (Honest-verifier) zero-knowledgeness ensures that attackers cannot learn anything about the secret key by observing the transcripts. In this way, the secret key remains hidden, and thus, the attacker cannot generate forgeries.

(e) First, we have that $M' = M$, because

$$\begin{aligned} K' &= \mathrm{h}_2(\text{“KDF”}; (A')^b) = \mathrm{h}_2(\text{“KDF”}; (g^{a'})^b) = \mathrm{h}_2(\text{“KDF”}; (g^{a'b}) \\ &= \mathrm{h}_2(\text{“KDF”}; (g^b)^{a'}) = \mathrm{h}_2(\text{“KDF”}; B^{a'}) = K \\ M' &= \mathrm{Dec}_{K'}(CT) = \mathrm{Dec}_K(CT) = \mathrm{Dec}_K(\mathrm{Enc}_K(M)) = M. \end{aligned}$$

Then, verification passes, because

$$\begin{aligned} c' &= \mathrm{h}_2(p; g; A; V; \mathrm{Bob}; M') = \mathrm{h}_2(p; g; A; V; \mathrm{Bob}; M) = c \\ g^r A^{c'} &= g^r A^c = g^{v-ca}(g^a)^c = g^{v-ca} g^{ca} = g^{v-ca+ca} = g^v = V. \end{aligned}$$

(f) In verifying the signature, Charlie computes the challenge c' as $\mathrm{h}_2(p; g; A; V; \mathrm{Charlie}; M)$ which is unlikely to be equal to $c = \mathrm{h}_2(p; g; A; V; \mathrm{Bob}; M)$. As such, it is unlikely that $g^r A^c = g^r A^{c'}$ holds, and thus the verification check passes.

[[Grading Instruction:

Grading (total 15):	
aspect:	points
(a)	**2**
(b)	**2**
if the advantage is minor	only 1
(c)	**2**
(d)	**4**
soundness	1
explanation	1
(honest-verifier) zero-knowledgeness	1
explanation	1
completeness	-1
(e)	**3**
$K = K'$	1
$M = M'$	0.5
$c = c'$	0.5
$g^r A^c = V$	1
(f)	**2**
challenge different for Charlie	1
$c \neq c'$ and thus $V \neq g^r A^{c'}$	1

Note: Final number of points is rounded up.

]]

Formula sheet of Introduction to Cryptography

1 Mathematical concepts

1.1 Euler's totient function

Let $n > 1$ be an integer such that $n = \prod_i p_i^{k_i}$, where p_i are distinct prime numbers and $k_i > 0$. Then $\varphi(n)$ is computed as

$$\varphi(n) = \varphi\left(\prod_i p_i^{k_i}\right) = \prod_i \varphi(p_i^{k_i}) = \prod_i (p^{k_i} - p^{k_i - 1}).$$

1.2 Square-and-multiply algorithm

Data: Integers a, d, n
Result: x with $x \equiv a^d \pmod{n}$
1. Write $d = (d_{k-1}d_{k-2}\cdots d_1 d_0)_2$;
2. $x \leftarrow 1$;
3. **for** $i = k-1$ *to* 0 **do**
   $x \leftarrow x^2 \bmod n$;
   if $d_i = 1$ **then**
      $x \leftarrow ax \bmod n$;
   end
end
4. **return** x.

1.3 CRT, specifically for RSA

Suppose that we want to solve a system of modular equations like

$$\begin{cases} x \equiv a_0 & \pmod{p}; \\ x \equiv a_1 & \pmod{q}. \end{cases}$$

A solution is $x = u_0 a_0 + u_1 a_1 \pmod{n}$, where $u_0 = (q^{-1} \bmod p) \cdot q$ and $u_1 = (p^{-1} \bmod q) \cdot p$.

Garner's method:

A solution is $x = a_1 + q \cdot ((a_0 - a_1 \bmod p) \cdot (q^{-1} \bmod p) \bmod p)$.

2 Security strength

(Computational) Security strength:

A cryptographic scheme offers security strength s if there are no attacks with $(M+N)/p < 2^s$ with N and M the adversary's (offline and online) resources and p the success probability.

Advantage:

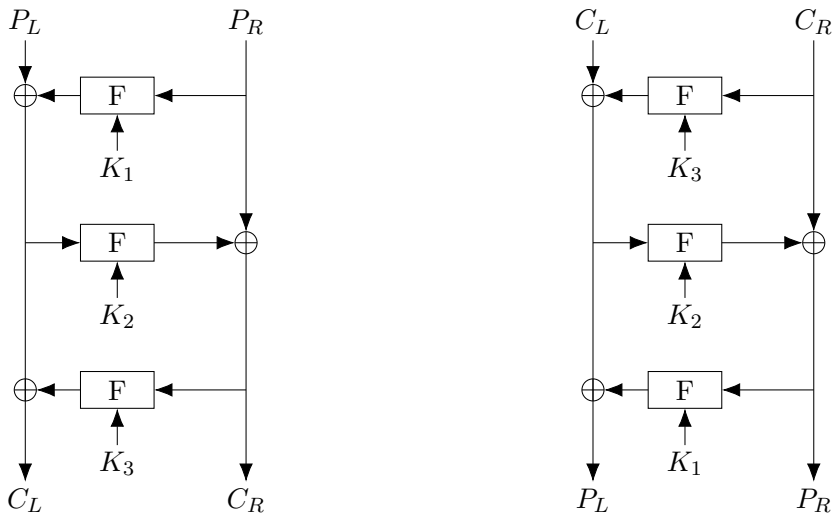
The advantage of distinguishing a stream cipher SC from a random oracle $\mathcal{RO}$ is given by: $\mathrm{Adv}_{\mathcal{A}} = |\Pr(\mathcal{A} = 1 \mid \mathrm{SC}_K) - \Pr(\mathcal{A} = 1 \mid \mathcal{RO})|$

(Decisional) Security strength:

A cryptographic scheme offers security strength s if there are no attacks with $(M+N)/\mathrm{Adv} < 2^s$ with N and M the adversary's (offline and online) resources and Adv the advantage of the adversary.

3 Symmetric cryptography

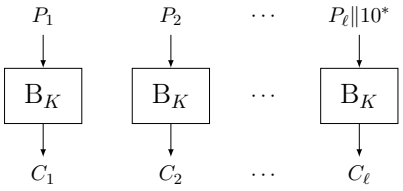
3.1 Feistel structure



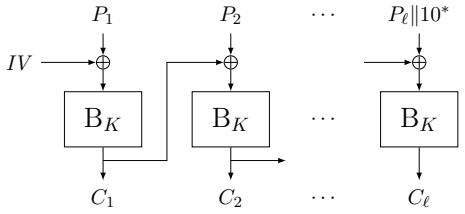
(a) Encryption (b) Decryption

Figure 1: Three-round Feistel structure.

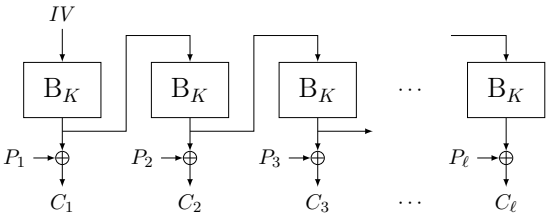
3.2 Block cipher modes



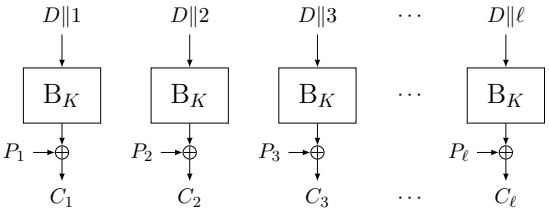
(a) Electronic codebook mode



(b) Cipher block chaining mode



(a) Output Feedback mode



(b) Counter mode

3.3 Hash function constructions

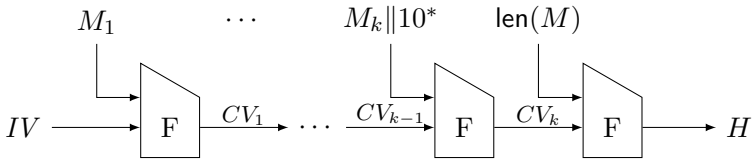


Figure 4: Merkle-Damgård construction for hash functions.

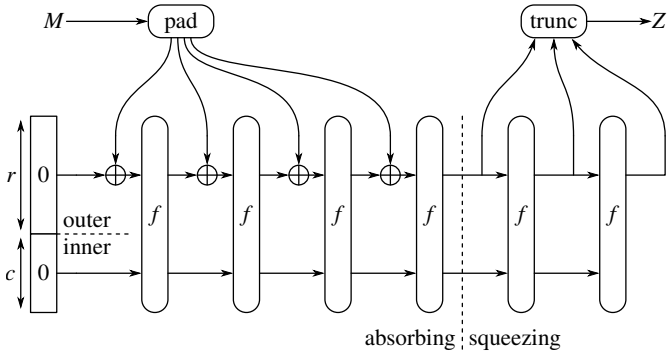


Figure 5: Sponge function.

4 Public-key cryptography

4.1 Key agreement schemes

4.1.1 Textbook (Merkle-)Diffie-Hellman key agreement

Alice		Bob
p, g, q		p, g, q
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		$b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$
$A \leftarrow g^a$		$B \leftarrow g^b$
	$\xrightarrow{\text{Alice},A}$	
	$\xleftarrow{\text{Bob},B}$	
$K_{A,B} \leftarrow B^a$		$K_{B,A} \leftarrow A^b$

4.2 Encryption schemes

4.2.1 ElGamal encryption scheme

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{(C,A)}$	$M \leftarrow C \times A^{q-b}$

4.2.2 Textbook RSA encryption scheme

Bob		Alice
Alice's public key (n, e)		Alice's private key (n, d)
$c \leftarrow m^e \bmod n$	$\xrightarrow{c}$	$m \leftarrow c^d \bmod n$

4.3 Key encapsulation mechanisms (KEM)

4.3.1 KEM from ElGamal

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$K \leftarrow \mathrm{h}(\text{“KDF”}; B^a)$		
$CT \leftarrow \mathrm{Enc}_K(m)$	$\xrightarrow{(A,CT)}$	$K \leftarrow \mathrm{h}(\text{“KDF”}; A^b)$
		$m \leftarrow \mathrm{Dec}_K(CT)$

4.3.2 KEM from RSA

Bob has Alice's public key (n, e)		Alice with private key (n, d)
$r \xleftarrow{\$} \mathbb{Z}/n\mathbb{Z}$		
$c \leftarrow r^e \bmod n$		
$K \leftarrow \mathrm{h}(\text{“KDF”}; r)$		
$CT \leftarrow \mathrm{Enc}_K(m)$	$\xrightarrow{(c,CT)}$	$r \leftarrow c^d \bmod n$
		$K \leftarrow \mathrm{h}(\text{“KDF”}; r)$
		$m \leftarrow \mathrm{Dec}_K(CT)$

4.4 Authentication protocols

4.4.1 Chaum-Evertse-van de Graaf (CEG) protocol

Alice		Bob
p, g, q, A, a		p, g, q (Alice: A)
$v \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$V \leftarrow g^v$	$\xrightarrow{\text{Alice},V}$	$c \xleftarrow{\$} \{0, 1\}$
	$\xleftarrow{c}$	
$r \leftarrow v - ca$	$\xrightarrow{r}$	$V \stackrel{?}{=} g^r A^c$

4.4.2 Schnorr's authentication protocol

Alice		Bob
p, g, q, A, a		p, g, q (Alice: A)
$v \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$V \leftarrow g^v$	$\xrightarrow{\text{Alice},V}$	$c \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$
	$\xleftarrow{c}$	
$r \leftarrow v - ca$	$\xrightarrow{r}$	$V \stackrel{?}{=} g^r A^c$

4.5 Signature schemes

4.5.1 Schnorr's signature scheme

Alice		Bob
p, g, q, A, a		p, g, q (Alice: A)
$v \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$V \leftarrow g^v$		
$c \leftarrow \mathrm{h}(p; g; A; V; m)$		
$r \leftarrow v - ca$	$\xrightarrow{\text{Alice},m,(r,V)}$	$c \leftarrow \mathrm{h}(p; g; A; V; m)$
		$V \stackrel{?}{=} g^r A^c$

4.5.2 Full-domain hash RSA signatures

Alice with private key (n, d)		Bob with Alice's public key (n, e)
$H \leftarrow \mathrm{h}(m)$		
$s \leftarrow H^d \bmod n$	$\xrightarrow{\text{Alice}, m, s}$	$H \leftarrow \mathrm{h}(m)$
		$H \stackrel{?}{=} s^e \bmod n$

4.5.3 Security notions

Discrete log (DL) problem:
Let $a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ and $A \leftarrow g^a$. Given $\langle g \rangle$ and A, determine a.

Computational Diffie-Hellman (CDH) problem:
Let $a, b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, $A \leftarrow g^a$ and $B \leftarrow g^b$. Given $\langle g \rangle$ and A, B, determine g^{ab}.

Decisional Diffie-Hellman (DDH) problem:
Let $a, b, c \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, and $A \leftarrow g^a$, and $B \leftarrow g^b$. With probability $\frac{1}{2}$, set $C \leftarrow g^c$, and otherwise $C \leftarrow g^{ab}$. Given $\langle g \rangle$ and A, B, C, determine whether $C = g^{ab}$ holds.

Advantage:

The advantage of an adversary on the decisional Diffie-Hellman problem is given by:

$$\mathrm{Adv}_{\mathcal{A}} = |\Pr(\mathcal{A} = 1 \mid C = g^{ab}) - \Pr(\mathcal{A} = 1 \mid C = g^c)|$$

IND-CPA security:

Challenger		Adversary
$p, g, (q)$		$p, g, (q)$
$b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$PK \leftarrow g^b$	$\xrightarrow{PK}$	Repeat: $\mathrm{Enc}_{PK}(M)$
	$\xleftarrow{M_0, M_1}$	M_0, M_1 messages
$i \xleftarrow{\$} \{0, 1\}$		
$CT \leftarrow \mathrm{Enc}_{PK}(M_i)$	$\xrightarrow{CT}$	Repeat: $\mathrm{Enc}_{PK}(M)$

4.6 Elliptic curves

4.6.1 Addition formulas for Weierstrass curves over prime fields

An elliptic curve (in short Weierstrass form) is the set of points in $\mathbb{F}_p^2$ that satisfy

$$\mathcal{E}\colon y^2 = x^3 + ax + b, \qquad (a, b \in \mathbb{F}_p)$$

together with the point at infinity $\mathcal{O}$.

If points $P = (x_1, y_1)$, $Q = (x_2, y_2)$ are on curve $\mathcal{E}$, then we can compute their sum, $R = (x_3, y_3)$, algebraically as follows:

	$P = -Q$	$P \neq \pm Q$	$P = Q$
$R =$	$\mathcal{O}$	$\lambda = \frac{y_1 - y_2}{x_1 - x_2}$ $x_3 = \lambda^2 - x_1 - x_2$ $y_3 = -y_1 + \lambda(x_1 - x_3)$	$\lambda = \frac{3x_1^2 + a}{2y_1}$ $x_3 = \lambda^2 - 2x_1$ $y_3 = -y_1 + \lambda(x_1 - x_3)$

4.6.2 Projective coordinates

We can convert any point $(X : Y : Z)$ with $Z \neq 0$ to affine coordinates, as (XZ^{-1}, YZ^{-1}). The homogeneous elliptic curve has the form

$$Y^2 Z = X^3 + aXZ^2 + bZ^3.$$

The curve's point at infinity is $\mathcal{O} = (0 : 1 : 0)$.

4.7 Attacks on the discrete logarithm problem

We use multiplicative notation in the following. In additive notation, multiplications are replaced by additions and exponentiations by scalar multiplications.

4.7.1 Baby-step giant-step algorithm

Data: Group elements g, h and table size m
Result: Integer a such that $h = g^a$
1. $q \leftarrow \#\langle g \rangle$;
2. **for** $i = 0$ *to* m **do**
   $b_i \leftarrow g^i$;
end
3. $j \leftarrow 0$;
4. **repeat** $c_j \leftarrow h \cdot g^{-m \cdot j}$;
until $c_j = b_i$;
return $i + m \cdot j$.

4.7.2 Pollard's ρ algorithm

Let p be a prime number such that $g \in (\mathbb{Z}/p\mathbb{Z})^*$ has order q. We want to compute $x = \mathrm{dlog}_g(h)$ for some $h \in \langle g \rangle$. We take as starting point $(g, 1, 0)$ and as our function:

$$(a_{i+1}, b_{i+1}, c_{i+1}) = \begin{cases} (a_i \cdot g, b_i + 1, c_i) & \text{if } a_i \equiv 1 \pmod 3; \\ (a_i \cdot h, b_i, c_i + 1) & \text{if } a_i \equiv 2 \pmod 3; \\ (a_i^2, 2b_i, 2c_i) & \text{if } a_i \equiv 0 \pmod 3. \end{cases}$$

Note that we take computations on a_i modulo p, and computations on b_i and c_i modulo q.

When we find $i \neq j$ with $a_i = a_j$, then we have

$$g^{b_i} h^{c_i} \equiv g^{b_j} h^{c_j} \pmod p,$$

so we get

$$g^{b_i - b_j} \equiv h^{c_j - c_i} \equiv g^{x(c_j - c_i)} \pmod{p}.$$

We then find x by solving $b_i - b_j \equiv x(c_j - c_i)$ modulo q.