Cryptography and Security — Final Exam
Solution

Serge Vaudenay

28.1.2019

- duration: 3h
- no documents allowed, except one 2-sided sheet of handwritten notes
- a pocket calculator is allowed
- communication devices are not allowed
- the exam invigilators will **not** answer any technical question during the exam
- readability and style of writing will be part of the grade

The exam grade follows a linear scale in which each question has the same weight.

1 The Mersenne Cryptosystem

The following exercise is inspired from A New Public-Key Cryptosystem via Mersenne Numbers *by Aggarwal, Joux, Prakash, and Santha, published in the proceedings of CRYPTO 2018.*

In what follows, p denotes a prime number of form $p = 2^n - 1$. It is called a *Mersenne prime*. Elements in $\mathbf{Z}_p$ are represented by numbers between 0 and $p-1$. Given $x \in \mathbf{Z}_p$, $W(x)$ denotes the number of 1's when writing the element x in binary.

Q.1 For all $x \in \mathbf{Z}_p^*$, prove that $W(-x \bmod p) = n - W(x)$.

The number p consists of n bits all equal to 1, when written in binary. Hence, $p - x$ is the same operation as flipping all the n bits of x individually. When $x > 0$, $-x \bmod p$ is $p - x$. Hence, $W(x) + W(-x \bmod p)$ will add all the n bits of 1 which are in p, i.e. $W(x) + W(-x \bmod p) = n$.

Q.2 For all $x, y \in \mathbf{Z}_p$, prove that $W(x + y \bmod p) \le W(x) + W(y)$.
HINT: first consider $y = 1$, then $W(y) = 1$, then proceed by induction.

We have $0 \leq x < p$, so there must be at least one 0 bit in the binary representation of x. We write $x = x' \| 0 \| 11 \cdots 1$ in binary, where the number of consecutive 1 in the least significant bits is i. For $y = 1$, computing $x + y \bmod n$ results in $x' \| 1 \| 00 \cdot 0$ in binary. Hence, $W(x + 1 \bmod p) = W(x') + 1 = W(x) + 1 - i \leq W(x) + 1$. This shows the $y = 1$ case.
Multiplying by 2 modulo p consists of rotating all bits circularly. Hence,

$$W(z2^i \bmod p) = W(z)$$

for all z and i.
For $W(y) = 1$, we have $y = 2^i$. Thanks to the previous observation,

$$W(x + y \bmod p) = W((x + y)2^{-i} \bmod p) = W(x2^{-i} + 1 \bmod p)$$

Due to the $y = 1$ case, we obtain

$$W(x + y \bmod p) \leq W(x2^{-i} \bmod p) + 1 = W(x) + 1$$

This shows the $W(y) = 1$ case.
We proceed by induction on $W(y)$. This is quite trivial for $W(y) = 0$ as it means $y = 0$. We have proven it for $W(y) = 1$.
Assuming this is true for $W(y) - 1$, we let i be the index of one bit of y equal to 1 and we write $y = y' + 2^i$. Clearly, $W(y') = W(y) - 1$. We have $W(x + y' \bmod p) \leq W(x) + W(y) - 1$ by induction. Then,

$$W(x+y \bmod p) = W((x+y' \bmod p)+2^i \bmod p) \leq W(x+y' \bmod p)+1 \leq W(x)+W(y)$$

Q.3 For all $x, y \in \mathbf{Z}_p$, prove that $W(x \times y \bmod p) \leq W(x) \times W(y)$.
HINT: use binary and show $W(x2^j \bmod p) = W(x)$.

We have already shown $W(x2^j \bmod p) = W(x)$ in the previous question.
We write $y = \sum_{i=1}^{W(y)} 2^{j_i}$. We have

$$x \times y \equiv \sum_{i=1}^{W(y)} x2^{j_i} \pmod{p}$$

We have $W(x2^{j_i} \bmod p) = W(x)$. Hence, $W(x \times y \bmod p) \leq W(x) + \cdots + W(x)$ ($W(y)$ times).

Q.4 In what follows, h denotes a positive integer such that $4h^2 < n$.
After the parameters n, p, and h are set up, we define the following algorithms:

$\mathsf{Gen}(n, p, h)$:
1: pick $F, G \in \mathbf{Z}_p$ random such that $W(F) = W(G) = h$
2: set $\mathsf{pk} = \frac{F}{G} \bmod p$ and $\mathsf{sk} = G$
3: output pk and sk

$\mathsf{Enc}(\mathsf{pk}, b)$:

4: pick $A, B \in \mathbf{Z}_p$ random such that $W(A) = W(B) = h$

5: set $\mathsf{ct} = \left((-1)^b \times (A \times \mathsf{pk} + B)\right) \bmod p$

6: output ct

where b is a plaintext from the space $\{0, 1\}$ (i.e. we encrypt only one bit).

Design a decryption algorithm and prove it is correct.

We define

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$*:*

1: compute $x = \mathsf{sk} \times \mathsf{ct} \bmod p$

2: compute $\mathsf{pt} = 1_{W(x) > \frac{n}{2}}$

3: output pt

Indeed, if the computation is done correctly, we have

$$x \equiv G \times (-1)^b \times (A \times \mathsf{pk} + B) \equiv (-1)^b \times (A \times F + B \times G) \pmod{p}$$

We have $W(A \times F + B \times G \bmod p) \leq 2h^2 < \frac{n}{2}$*. So, if* $b = 0$*, we obtain* $W(x) \leq 2h^2$*. For* $b = 1$ *and* $x \neq 0$*, we obtain* $W(x) \geq n - 2h^2 > \frac{n}{2}$*. To show that decryption is perfectly correct, it remains to show that* $x = 0$ *cannot happen.*
If $x = 0$*, then* $A \times F \equiv -B \times G$*. We have* $B \times G \bmod p \neq 0$*, so* $W(-B \times G \bmod p) = n - W(B \times G \bmod p) \geq n - h^2$*. We also have* $W(A \times F \bmod p) \leq h^2$*. Since* $h^2 < n - h^2$*, we cannot have* $A \times F \equiv -B \times G$*. Therefore,* $x \neq 0$*.*

Q.5 As a toy example, take $n = 17$, $p = 131\,071$, $h = 2$. Generate a key pair using $F = 2^{14} + 2^2$ and $G = 2^{10} + 2^6$. Then, encrypt $b = 1$ using $A = 2^{11} + 2^5$ and $B = 2^9 + 2^2$. Detail the computations and give, pk, sk, ct.

HINT1: for people who have a 4-operation calculator: $a \times 2^n + b \equiv a + b \pmod{2^n - 1}$.

HINT2: by thinking of how multiplication by 2 works modulo p, find a trick to perform the division by 2.

HINT3: $\frac{1}{17} \bmod p = 123\,361$.

We have $2^n \equiv 1$ so $a \times 2^n + b \equiv a + b$, modulo $2^n - 1$. Hence, after an integer multiplication, we can write a number in the form $a \times 2^n + b$ and perform a modulo p reduction easily, by just adding a and b then iterating if it is still larger than p.
Multiplication by 2 is just a circular bit rotation to the left. Hence, division by 2 is a circular bit rotation to the right. To divide a number by 2, it is easy if it is even. If x is odd, we can just compute $\frac{x-1}{2} + \frac{1}{2}$. Given that $\frac{1}{2} \equiv 2^{n-1} \pmod p$, we obtain

$$\frac{x}{2} \equiv \frac{x-1}{2} + 2^{n-1} \pmod p$$

To invert a number, we could use the Extended Euclid Algorithm. Here, we give as a hint that $\frac{1}{17} \bmod p = 123\,361$.
We have $\mathsf{sk} = 2^{10} + 2^6 = 1\,088$,

$$\begin{aligned}
\mathsf{pk} &= \frac{2^{14}+2^2}{2^{10}+2^6} \bmod p \\
&= \frac{2^{12}+1}{2^4 \times 17} \bmod p \\
&= \frac{4\,097 \times \frac{1}{17}}{2^4} \bmod p \\
&= \frac{4\,097 \times 123\,361}{2^4} \bmod p \\
&= \frac{505\,410\,017}{2^4} \bmod p \\
&= \frac{3\,855 \times 2^{17} + 127\,457}{2^4} \bmod p \\
&= \frac{3\,855 + 127\,457}{2^4} \bmod p \\
&= \frac{131\,312}{2^4} \bmod p \\
&= \frac{1 \times 2^{17} + 240}{2^4} \bmod p \\
&= \frac{241}{2^4} \bmod p \\
&= \frac{240}{2^4} + \frac{2^{16}}{2^3} \bmod p \\
&= 15 + 8\,192 \bmod p \\
&= 8\,207 \\
&= \texttt{0x200f}
\end{aligned}$$

and

$$\mathsf{ct} = -((2^{11}+2^5)\times 8\,207+(2^9+2^2)) \bmod p = -(130+31\,716) \bmod p = 99\,225 = \texttt{18399}$$

We can check that

$$\mathsf{ct} \times \mathsf{sk} \bmod p = 85\,367 = \texttt{0x14d77}$$

has weight 11 so decrypts to 1.

2 Collision Attack on CBC Mode

The following exercise is inspired from On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN *by Bhargavan and Leurent, published in the proceedings of ACM CCS 2016.*

We consider TLS using a block cipher with n-bit message blocks in CBC mode. The goal of this exercise is to develop message recovery attacks, or at least to recover a sensitive part of a partially-known plaintext.

Q.1 Given 2^d independent and uniformly distributed random variables $X_1, \ldots, X_{2^d}$ with values in $\{0,1\}^n$, what is the expected number of pairs (i,j) with $i < j$ such that $X_i = X_j$?

We have $\binom{2^d}{2} = 2^{d-1}(2^d - 1) \approx 2^{2d-1}$ possible pairs. Each satisfies $X_i = X_j$ with probability 2^{-n}. Hence, the expected number of pairs is roughly 2^{2d-n-1}.

Q.2 Given 2^s independent and uniformly distributed random variables $X_1, \ldots, X_{2^s}$ and 2^t independent and uniformly distributed random variables $Y_1, \ldots, Y_{2^t}$, all with values in $\{0,1\}^n$, what is the expected number of pairs (i,j) such that $X_i = Y_j$?

We have 2^{s+t} possible pairs, so 2^{s+t-n} expected pairs with collision.

Q.3 Consider a list of plaintexts of 2^d blocks in total. We assume that all blocks can be split into three categories: blocks which are already known by the adversary (we denote by α the fraction of blocks in this category), blocks which are privacy-sensitive thus an interested target for the adversary (we denote by β the fraction of blocks in this category), and other blocks which are unknown but uninteresting to recover (within a fraction $1 - \alpha - \beta$). All ciphertext blocks are known by the adversary.
Assuming that the inputs of the block cipher are independent and uniform, design an attack which recovers some privacy-sensitive blocks. How large must 2^d be in order for the expected number of recovered sensitive blocks to be 1? Compute the data complexity 2^d in terms of n, α, and β.
HINT: encryption uses the CBC mode.

We denote by Z the ciphertext blocks. If the adversary observes a collision $Z_i = Z_j$, then he deduces $Z_{i-1} \oplus Z_{j-1} = X_i \oplus X_j$ due to the CBC structure. If i is the index of a known block and j is the index of a sensitive one, he deduces X_j because he knows X_i. Here, we match $\alpha 2^d$ ciphertext blocks with index of a known plaintext block with $\beta 2^d$ ciphertext blocks with index of a sensitive block. Thanks to the previous question, the expected number of pairs is $\alpha\beta 2^{2d-n}$. Hence, we take $2^d = 2^{n/2}/\sqrt{\alpha\beta}$ to obtain one.

Q.4 Assuming that the encryption key changes every 2^r blocks, adapt the previous attack and estimate its data complexity. Application: how much data do we need for $n = 64$, $\alpha = \beta = \frac{1}{2}$, $r = \frac{n}{2}$?

The previous attack can only use $2^d = 2^r$ blocks. We repeat it $2^{n-2r}/(\alpha\beta)$ times to obtain one interesting collision. Hence, the data needed consists of $2^{n-r}/(\alpha\beta)$ blocks. For the proposed parameters, this is 2^{34} blocks of 64 bits, i.e. 128GB.

Q.5 We now assume that a plaintext of 2^u blocks is encrypted many times (with a random IV). We assume that all blocks but k sensitive ones are known by the adversary and that $k \ll 2^u$. However, the purpose is now to recover *all* sensitive blocks. Estimate the data complexity (in blocks) in terms of n, u, and k.

We take $\alpha \approx 1$ and $\beta = k2^{-u}$. Given one particular sensitive block, the probability of not recovering it after D encryptions of the same message is roughly

$$(1 - 2^{-n})^{D^2 2^u} \approx e^{-D^2 2^{u-n}}$$

Thus, the probability to recover all blocks is roughly

$$\left(1 - e^{-D^2 2^{u-n}}\right)^k \approx e^{-e^{(\ln k) - D^2 2^{u-n}}}$$

Hence, we should use $D = \sqrt{\ln k}\, 2^{\frac{n-u}{2}}$. The data complexity is thus of $\sqrt{\ln k}\, 2^{\frac{n+u}{2}}$ blocks.

3 PKC vs KEM vs KA

In this exercise, we compare *Public-Key Cryptosystems* (PKC), *Key Encapsulation Mechanisms* (KEM), and non-interactive *Key Agreement* schemes (KA). We formalize the interface for each of the three primitives:

PKC

- $\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
- $\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
- $\mathsf{Enc}(\mathsf{pk}, \mathsf{pt}) \xrightarrow{\$} \mathsf{ct}$
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow \mathsf{pt}/\bot$

KEM

- $\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
- $\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
- $\mathsf{Enc}(\mathsf{pk}) \xrightarrow{\$} (K, \mathsf{ct})$
- $\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow K/\bot$

KA

- $\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
- $\mathsf{Gen}_A(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_A, \mathsf{sk}_A)$
- $\mathsf{Gen}_B(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_B, \mathsf{sk}_B)$
- $\mathsf{KA}_A(\mathsf{sk}_A, \mathsf{pk}_B) \rightarrow K/\bot$
- $\mathsf{KA}_B(\mathsf{sk}_B, \mathsf{pk}_A) \rightarrow K/\bot$

The notation $\xrightarrow{\$}$ means that the function is probabilistic while $\rightarrow$ is for deterministic ones. The notation $K/\bot$ means that either some K or an error message $\bot$ is returned.

Q.1 Define the correctness notion for *each* of the three primitives.

Correctness implies that for all random coins, the following experiments always return 1:

PKC

$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
pick pt *at random*
$\mathsf{Enc}(\mathsf{pk}, \mathsf{pt}) \xrightarrow{\$} \mathsf{ct}$
$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow x$
return $1_{x=\mathsf{pt}}$

KEM

$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
$\mathsf{Enc}(\mathsf{pk}) \xrightarrow{\$} (K, \mathsf{ct})$
$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct}) \rightarrow x$
return $1_{x=K}$

KA

$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}_A(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_A, \mathsf{sk}_A)$
$\mathsf{Gen}_B(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_B, \mathsf{sk}_B)$
$\mathsf{KA}_A(\mathsf{sk}_A, \mathsf{pk}_B) \rightarrow K$
$\mathsf{KA}_B(\mathsf{sk}_B, \mathsf{pk}_A) \rightarrow x$
return $1_{x=K}$

Q.2 The INDCPA security notion was defined for PKC in the course. We make a slight change and give a new definition: A PKC is (t, ε)-INDCPAror-secure if for all probabilistic adversary $\mathcal{A}$ limited to a time complexity of t, we have

$$\Pr[x = 1|b = 0] - \Pr[x = 1|b = 1] \leq \varepsilon$$

where b is an input bit and x is the output of the following procedure, and the probability is over all probabilistic operations:

input b
$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
pick coins at random
$\mathcal{A}(\mathsf{pp}, \mathsf{pk}; \mathsf{coins}) \rightarrow \mathsf{pt}_0$
pick pt_1 at random, of same length at pt_0
$\mathsf{Enc}(\mathsf{pk}, \mathsf{pt}_b) \xrightarrow{\$} \mathsf{ct}$
$\mathcal{A}(\mathsf{pp}, \mathsf{pk}, \mathsf{ct}; \mathsf{coins}) \rightarrow x$
return x

What was changed, compared to the INDCPA definition from the course?

Discuss on the importance of the change.

In the definition from the course, the adversary chooses both pt_0 *and* pt_1*. Here, the adversary chooses only* pt_0 *while* pt_1 *is random and unknown to the adversary.*
We could prove that both security notions are equivalent as follows. (Students are not expected to answer this.) Indeed, an adversary in the ror *game can be transformed into an adversary in the previous definition (just generate* pt_1 *with fresh coins). The converse is less direct but still easy: given an adversary generating* $(\mathsf{pt}_0, \mathsf{pt}_1)$*, we can pick a random bit* a *and issue* pt_a *to produce as an output. The final* x *from the old adversary is returned as the output* $y = x \oplus a$ *by the new adversary by XORing with* a*. We have* $\Pr[y = b|b = 1] = \Pr[x = a \oplus 1|b = 1] = \frac{1}{2}$ *because the computation of* x *does not use* a *at all, so is independent. Hence, the advantage of the new adversary is*

$$\Pr[y = 1|b = 0] - \Pr[y = 1|b = 1] = 1 - 2\Pr[y = b] = \frac{1}{2} - \Pr[y = b|b = 0]$$

We have that

$$1 - 2\Pr[y = b|b = 0] = \Pr[x = 1|a = 0, b = 0] - \Pr[x = 1|a = 1, b = 0]$$

which is the advantage of the old adversary. Hence, the new adversary has an advantage which is half of the old one.

Q.3 We define the KEM security as follows. A KEM is (t, ε)-$\mathsf{INDCPAror}$-secure if for all probabilistic adversary $\mathcal{A}$ limited to a time complexity of t, we have

$$\Pr[x = 1|b = 0] - \Pr[x = 1|b = 1] \leq \varepsilon$$

where b is an input bit and x is the output of the following procedure, and the probability is over all random coins:

input b
$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}, \mathsf{sk})$
$\mathsf{Enc}(\mathsf{pk}) \xrightarrow{\$} (K_0, \mathsf{ct})$
pick K_1 at random of same length as K_0
$\mathcal{A}(\mathsf{pp}, \mathsf{pk}, \mathsf{ct}, K_b) \xrightarrow{\$} x$
return x

Given a PKC, construct a KEM.

Prove that if the PKC is correct, then the KEM is correct.

Prove that there exists a constant τ such that for all t and ε, if the PKC is (t, ε)-$\mathsf{INDCPAror}$-secure, then the KEM is $(t - \tau, \varepsilon)$-$\mathsf{INDCPAror}$-secure.

We define Setup, Gen, *and* Dec *the same as in the PKC. Then, we define*

KEM.Enc(pk):

pick K at random
PKC.Enc(pk, K) $\xrightarrow{\$}$ ct
return (K, ct)

If we write the correctness experiment for KEM, so with KEM.Enc, *and if we expand* KEM.Enc *as the two lines of code above, we obtain exactly the correctness experiment for PKC with* PKC.Enc. *Hence, the two experiments are indeed the same. When fed with the same random source, they produce the same output. So, if PKC is correct, it always returns 1. Therefore, the KEM is correct.*

We consider an adversary $\mathcal{A}$ against the KEM and we define an adversary $\mathcal{B}$ against the PKC.

$\mathcal{B}$(pp, pk, ct; coins):

1: pick pt at random using the first coins in coins and remove them from coins
2: **if** input ct is not present **then**
3: **return** pt
4: **else**
5: $\mathcal{A}$(pp, pk, ct, pt; coins) $\rightarrow x$
6: **return** x
7: **end if**

The complexity of $\mathcal{B}$ is the complexity of $\mathcal{A}$ plus a small overhead τ for all steps but Step 5. If $\mathcal{A}$ has complexity bounded by $t - \tau$, then $\mathcal{B}$ has complexity bounded by t.

If $b = 0$, we can clearly see that the INDCPAror *game against PKC with $\mathcal{B}$ is exactly the* INDCPAror *game against KEM with $\mathcal{A}$. So, $\Pr[x = 1|b = 0]$ is the same. If now $b = 1$, we compare the* INDCPAror *game against PKC with $\mathcal{B}$ (left) is exactly the* INDCPAror *game against KEM with $\mathcal{A}$ (right):*

Setup $\xrightarrow{\$}$ pp	Setup $\xrightarrow{\$}$ pp
Gen(pp) $\xrightarrow{\$}$ (pk, sk)	Gen(pp) $\xrightarrow{\$}$ (pk, sk)
pick coins at random	pick K_0 at random
pick pt at random using the first coins in coins and remove them from coins	Enc(pk, K_0) $\xrightarrow{\$}$ ct
pick pt_1 at random, of same length at pt	pick K_1 at random of same length as K_0
Enc(pk, pt_1) $\xrightarrow{\$}$ ct	$\mathcal{A}$(pp, pk, ct, K_1) $\xrightarrow{\$} x$
$\mathcal{A}$(pp, pk, ct, pt; coins) $\rightarrow x$	**return** x
return x	

We can see that K_0 plays the role of pt_1 *and that K_1 plays the role of* pt. *There is an invertible mapping of the random source from left to right making $\Pr[x = 1|b = 1]$ the same. Hence, $\Pr[x = 1|b = 0] - \Pr[x = 1|b = 1]$ is the same. So, the KEM is secure as well.*

Q.4 Propose a definition for the INDCPAror-security of KA. Given a correct KA, construct a correct KEM.

Show that with the same method as in the previous question, we prove that there exists a constant τ such that for all t and ε, if the KA is (t, ε)-$\mathsf{INDCPAror}$-secure, then the KEM is $(t - \tau, \varepsilon)$-$\mathsf{INDCPAror}$-secure.

We define it for KA as follows. A KA is (t, ε)-INDCPA-*secure if for all probabilistic adversary* $\mathcal{A}$ *limited to a time complexity of* t*, we have*

$$\Pr[x = 1 | b = 0] - \Pr[x = 1 | b = 1] \leq \varepsilon$$

where b *is an input bit and* x *is the output of the following procedure, and the probability is over all random coins:*

input b
$\mathsf{Setup} \xrightarrow{\$} \mathsf{pp}$
$\mathsf{Gen}_A(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_A, \mathsf{sk}_A)$
$\mathsf{Gen}_B(\mathsf{pp}) \xrightarrow{\$} (\mathsf{pk}_B, \mathsf{sk}_B)$
$\mathsf{KA}_A(\mathsf{sk}_A, \mathsf{pk}_B) \to K_0$
pick K_1 *at random*
$\mathcal{A}(\mathsf{pp}, \mathsf{pk}_A, \mathsf{pk}_B, K_b) \xrightarrow{\$} x$
return x

We define Setup *the same as in the KA. Then, we define* $\mathsf{Gen} = \mathsf{Gen}_B$ *and*

$\mathsf{Enc}(\mathsf{pk})$*:*

$\mathsf{Gen}_A \xrightarrow{\$} (\mathsf{ct}, \mathsf{sk}_A)$
$\mathsf{KA}_A(\mathsf{sk}_A, \mathsf{pk}) \to K$
return (K, ct)

$\mathsf{Dec}(\mathsf{sk}, \mathsf{ct})$*:*

$\mathsf{KA}_B(\mathsf{sk}, \mathsf{ct}) \to K$
return K

The same proof as in the previous question shows that if the KA is correct, then the KEM is correct, and if the KA is secure, then the KEM is secure.