Exercise Sheet 1
Solutions

Prof. Kenny Paterson, F. Günther
M. Backendal, M. Filić

Group Discussion (Basics). Take some time to think about and discuss:

(a) What, for you, is the most important application of cryptography?

> **Solution.** There is obviously no definitive answer to this question, but positions may range from securing communication, to protecting privacy, to enabling online business, to shielding military operations, to ... — what you consider important or valuable applications depends on your personal metrics and values. We will revisit this question towards the end of this course.

(b) What's the difference between *cryptographic design* and *cryptanalysis*?

> **Solution.** Cryptographic design is about achieving certain goals by cryptographic means; the modern approach is to formally prove that the desired goals are indeed achieved.
> Cryptanalysis is the study of how to break cryptographic schemes / violate the claimed goals (not necessarily by causing a complete break; already notably weakening a scheme is a cryptanalytic success).
> Depending on the scheme at hand, both approaches are necessary to assess cryptographic security.

Suggested reading. Reading the following sections in the Boneh-Shoup book [1] might help with the problems on this exercise sheet: Sections 2.1, 2.2.1-2.2.3, 3.1, 3.2, and 3.3 (One-time pad, introduction to security games and reductions). Sections 4.1.1-4.1.2 (Block ciphers).

Problem 1 (Malleability of the One-time Pad). You just sat through the worst exam of your life and are heading home to start cramming for the next one, which is already tomorrow. Life feels hopeless and you start taking your frustration out on the world. In a moment of overconfidence, you decide to send your professor an angry email. You start typing "`Dear Professor, I hated your class so much. It sucked! Sincerely...`", but just before you hit send you get cold feet and decide to encrypt the message. You quickly choose a random key, *s bxbpy pbtz scomu dn qmhy af hxvysz*, encrypt the controversial parts of your message and hit send. The email reads: "`Dear Professor, a ixutb npnq unoem vb cgjf. iy zrxiwc! Hint: The message is encrypted with a one-time pad. Sincerely...`".

Next morning when you wake up, full of regrets, you have a message in your inbox from the professor asking you to come to their office and explain yourself. What will you do?! Devise a strategy to prove to the professor that your message was completely innocent.

> **Solution.** The perfect security of the one-time pad saves you. Any message of the same length as the original could have produced the given ciphertext, so all you have to do is come up with a harmless message of the correct length and give your professor the one-time pad key that decrypts the ciphertext to the innocuous plaintext.
>
> As per the problem statement, we have
>
> $$m_0 = \texttt{I hated your class so much. It sucked!}$$

and

$$k_0 = \texttt{s bxbpy pbtz scomu dn qmhy af hxvysz}.$$

If we ignore case and punctuation (and leave spaces unchanged for readability), we get

$$c = (m_0 + k_0) \bmod 26 = \texttt{a ixutb npnq unoem vb cgjf iy zrxiwc}^{a}.$$

This was the ciphertext you sent.

Now, to clear your name, pick a new message, for example

$$m_1 = \texttt{I loved your class so much. It rocked!}$$

and compute the key k_1 such that $c = (m_1 + k_1) \mod 26$ as follows:

$$k_1 = (c - m_1) \bmod 26 = \texttt{s xjzpy pbtz scomu dn qmhy af idvysz}.$$

Now you just have to explain why you decided to encrypt your praise of the class in the first place...

[a] To perform addition modulo 26 on the letters, associate to each letter its index in the alphabet, starting with a = 0.

Problem 2 (One-time Pad Cryptanalysis). You have intercepted two ciphertexts c and c'. You know that both are OTP ciphertexts encrypted with the *same* key k. For some fixed bit-strings m_0, m_0', m_1, m_1' you know that **either** c is an encryption of m_0 and c' is an encryption of m_0' **or** c is an encryption of m_1 and c' is an encryption of m_1'.

(a) Explain how to determine which of the two possibilities is true. Does it work in all cases?

Solution. According to the problem statement, we have $c = k \oplus m_b$ and $c' = k \oplus m_b'$ for some $b \in \{0, 1\}$. We know the values of c, c', m_0, m_0', m_1, and m_1'. Let us calculate $d = c \oplus c'$. Because XOR is a commutative operation, and because XORing any string with itself yields the identity (i.e., $x \oplus x = 0^{|x|}$ for any string x with length $|x|$), this "cancels out" the key k:

$$d = c \oplus c' = (k \oplus m_b) \oplus (k \oplus m_b') = (k \oplus k) \oplus (m_b \oplus m_b') = m_b \oplus m_b'.$$

It now remains to check whether $d = m_0 \oplus m_0'$ or $d = m_1 \oplus m_1'$. This allows to determine the value of b whenever $m_0 \oplus m_0' \neq m_1 \oplus m_1'$. If $m_0 \oplus m_0' = m_1 \oplus m_1'$, it is impossible to determine the correct value of b.

(b) Now let m_0 = "a", m_0' = "b", m_1 = "c", and m_1' = "d" (all converted to binary from ASCII in the standard way). Let $c = 11111001$ and $c' = 11111010$.

Which of the two possibilities is correct? What was the key k?

Solution. We have $d = c \oplus c' = 00000011$. The candidate ASCII messages correspond to the following bitstrings:

$$m_0 = 01100001 \qquad m_1 = 01100011$$
$$m_0' = 01100010 \qquad m_1' = 01100100$$

Observe that $d = m_0 \oplus m_0'$, but $d \neq m_1 \oplus m_1'$. So, c and c' encrypt m_0 and m_0', respectively. We can compute the key as $k = c \oplus m_0 = 10011000$.

Problem 3 (One-time Pad with Inconvenient Message and Key Spaces). Suppose you want to encrypt a single message $m \in \{0, 1, 2\}$ using a uniformly random shared key $k \in \{0, 1, 2\}$. Suppose you do this by representing each of k and m using two bits (00, 01, or 10), and then XOR-ing the two representations. Does this seem like a good algorithm to you? Explain. If not, then explain a better way to do this.

Solution. Ciphertexts of the proposed encryption scheme leak some information about the encrypted message, so it is not a good algorithm. For example, observe that—depending on the choice of the shared key—message 00 maps to one of the ciphertexts in $\{00, 01, 10\}$. So if an eavesdropper Eve intercepts a ciphertext 11 sent from Alice to Bob, then Eve knows that Alice encrypted either 01 or 10 (i.e., Alice *did not* encrypt message 00). More generally, Eve knows that a ciphertext $c \in \{01, 10, 11\}$ cannot be the encryption of the message $m = c \oplus 11$, since there is no key 11.

To build a better scheme, consider the following. Suppose you want to encrypt a single message $m \in \{0, 1, 2\}$ using a random shared key $k \in \{0, 1, 2\}$. For any values of m and k let the ciphertext be $c = (m + k) \bmod 3$. Then for any ciphertext c and key k we decrypt by calculating the message as $m = (c + k) \bmod 3$. This scheme ensures that any message $m \in \{0, 1, 2\}$ – depending on the choice of the shared key – can get encrypted into *any* ciphertext $c \in \{0, 1, 2\}$. Intuitively, this shows that the ciphertext leaks no information about the message. You can think of our scheme as the one-time pad scheme in $\mathbb{Z}_3$ (using addition modulo 3 instead of the XOR operation $\oplus$).

Problem 4 (Block Ciphers). Recall that a function $\mathsf{E}\colon \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ is called a *block cipher* if for all $\mathsf{K} \in \{0,1\}^k$, E_K is an efficiently computable permutation on $\{0,1\}^n$. Here $\mathsf{E}_\mathsf{K}(x) := \mathsf{E}(\mathsf{K}, x)$.

Let $\mathsf{E} : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher.

(a) Let the function $\mathsf{F}_1 : \{0,1\}^k \times \{0,1\}^n \to \{0,1\}^n$ be defined by

$$\mathsf{F}_1(\mathsf{K}, x) = \mathsf{E}(\mathsf{K}, x) \oplus x.$$

Give an example illustrating that F_1 is not in general a block cipher.

Solution. For all $\mathsf{K} \in \{0,1\}^k$, $\mathsf{F}_1(\mathsf{K}, \cdot)$ should be a permutation on the set $\{0,1\}^n$. This is not always the case. For example, if the underlying block cipher is defined such that for a specific key, say $\mathsf{K} = 0^k$, $\mathsf{E}_{0^k}(x) = x$ for all $x \in \{0,1\}^n$, (i.e., E_{0^k} is the identity function), then

$$\mathsf{F}_1(0^k, x) = \mathsf{E}(0^k, x) \oplus x = x \oplus x = 0^n,$$

for all $x \in \{0,1\}^n$, so $\mathsf{F}_1(0^k, \cdot)$ is not a permutation. Note that this does not violate any of the assumptions. We only assumed that E was a block cipher, not necessarily a *secure* block cipher, so it might very well be that under certain keys (or all!) it is the identity function.

Recall the definition of PRP security of a block cipher E, in which an adversary $\mathcal{A}$ with access to an oracle $\textsc{Fn}$ is asked to distinguish a "real world" ($b = 0$) where $\textsc{Fn}(x)$ returns the evaluation $\mathsf{E}(\mathsf{K}, x)$ under a random key K from a "random world" ($b = 1$) where $\textsc{Fn}(x)$ returns the evaluation $\Pi(x)$ under a randomly sampled permutation $\Pi\colon \{0,1\}^n \to \{0,1\}^n$. The advantage of $\mathcal{A}$ in this game is defined as $\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{E}}(\mathcal{A}) = 2 \cdot \left|\Pr[\,\text{Game } \mathbf{PRP}(\mathcal{A}, \mathsf{E}) \Rightarrow \mathsf{true}\,] - \frac{1}{2}\right|$, where $\Pr[\,\text{Game } \mathbf{PRP}(\mathcal{A}, \mathsf{E}) \Rightarrow \mathsf{true}\,]$ denotes the probability that the output of Game **PRP** is true.

Game $\mathbf{PRP}(\mathcal{A}, \mathsf{E})$:	Oracle $\textsc{Fn}(x)$:
$b \leftarrow\!\!\$\ \{0,1\}$	If $b = 0$ then:
$\mathsf{K} \leftarrow\!\!\$\ \{0,1\}^k$	$\quad y \leftarrow \mathsf{E}_\mathsf{K}(x)$
$\Pi \leftarrow\!\!\$\ \mathsf{Perms}[\{0,1\}^n]$	Else if $b = 1$ then:
$b' \leftarrow\!\!\$\ \mathcal{A}^{\textsc{Fn}}()$	$\quad y \leftarrow \Pi(x)$
Return $b' = b$	Return y

(b) Let the function $\mathsf{F}_2 : (\{0,1\}^k \times \{0,1\}^n) \times \{0,1\}^n \to \{0,1\}^n$ be defined by

$$\mathsf{F}_2((\mathsf{K}_1, \mathsf{K}_2), x) = \mathsf{E}(\mathsf{K}_1, \mathsf{K}_2 \oplus x).$$

The keyspace of F_2 is $\{0,1\}^k \times \{0,1\}^n$.

1) Show that F_2 is a block cipher.

Solution. We have to show that for all $\mathsf{K} \in \{0,1\}^k \times \{0,1\}^n$, $\mathsf{F}_2(\mathsf{K},\cdot)\colon \{0,1\}^n \to \{0,1\}^n$ is a permutation on the set $\{0,1\}^n$. To do this, it suffices to give the inverse function $\mathsf{F}_2^{-1}(\mathsf{K},\cdot)$ for $\mathsf{K} = (\mathsf{K}_1, \mathsf{K}_2)$ (because an injective function is trivially a bijection if the domain is equal to the range).
The inverse is given by

$$\mathsf{F}_2^{-1}((\mathsf{K}_1, \mathsf{K}_2), y) = \mathsf{E}_{\mathsf{K}_1}^{-1}(y) \oplus \mathsf{K}_2.$$

Here we use the fact that E is a block cipher, and hence for each key in the key space has an efficiently computable inverse.

2) Argue informally that F_2 is a pseudorandom permutation (i.e., is PRP-secure) assuming that E is a pseudorandom permutation.

Solution. Observe that $\mathsf{K}_2 \oplus x$ essentially is just a permutation of the input x via the random key K_2. So, even if an attacker would know K_2, or K_2 was a fixed value, the outer block cipher operation $\mathsf{E}(\mathsf{K}_1,\cdot)$ still serves as a pseudorandom permutation.

We now want to formally prove that F_2 is PRP-secure assuming that E is PRP-secure.

For proving relations like this, we use a *reductionist* proof approach (conceptually following complexity-theoretic reductions). We start by taking the contra-position of the claim

$$\mathsf{E} \text{ PRP-secure} \implies \mathsf{F}_2 \text{ PRP-secure},$$

to arrive at the equivalent

$$\mathsf{F}_2 \underline{\text{ not}} \text{ PRP-secure} \implies \mathsf{E} \underline{\text{ not}} \text{ PRP-secure}.$$

The result is the reduction we want to construct: Assume F_2 is not PRP-secure, i.e., there exists an adversary $\mathcal{A}$ with high advantage against the PRP security of F_2. Then we construct a reduction algorithm $\mathcal{B}$ with high advantage against the PRP security of E (i.e., show that E is not PRP-secure).

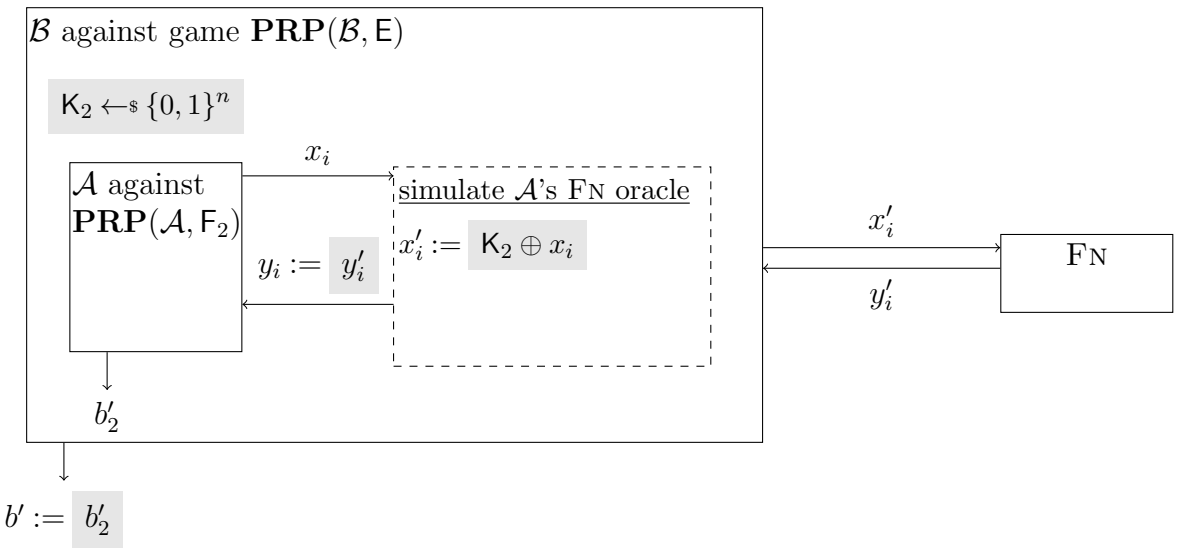
The idea for the reduction $\mathcal{B}$ is *simulate* the game $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$ for $\mathcal{A}$ by using the oracle $\mathcal{B}$ has available in its game $\mathbf{PRP}(\mathcal{B}, \mathsf{E})$. This should be done in such a way that, when $\mathcal{B}$'s oracle gives real (resp. random) responses, the simulated response to $\mathcal{A}$ should likewise be real (resp. random). We call such a simulation *sound*, if for $\mathcal{A}$ there's no observable difference between interacting with the game $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$ it expects and $\mathcal{B}$'s simulation. As a result, $\mathcal{B}$ can use $\mathcal{A}$'s bit guess to solve its own PRP security challenge.

Building the reduction $\mathcal{B}$ involves four steps:

1) Identify what $\mathcal{B}$ needs to do. We begin by comparing the games $\mathbf{PRP}(\mathcal{B}, \mathsf{E})$ and $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$ to determine what $\mathcal{B}$ needs to do/simulate "on top of" what $\mathbf{PRP}(\mathcal{B}, \mathsf{E})$ does already. To determine this, compare $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$ below with $\mathbf{PRP}(\mathcal{B}, \mathsf{E})$ and highlight those steps which are different/extra:

Game $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$:	Oracle $\textsc{Fn}(x)$:
1 $b \leftarrow\$ \{0,1\}$	6 If $b = 0$ then:
2 $\mathsf{K}_1 \leftarrow\$ \{0,1\}^k$; $\mathsf{K}_2 \leftarrow\$ \{0,1\}^n$	7 $y \leftarrow \mathsf{F}_2((\mathsf{K}_1, \mathsf{K}_2), x) = \mathsf{E}(\mathsf{K}_1, \mathsf{K}_2 \oplus x)$
3 $\Pi \leftarrow\$ \mathsf{Perms}[\{0,1\}^n]$	8 Else if $b = 1$ then:
4 $b' \leftarrow\$ \mathcal{A}^{\textsc{Fn}}()$	9 $y \leftarrow \Pi(x)$
5 Return $b' = b$	10 Return y

2) Define reduction $\mathcal{B}$. This can be done in words, pictorially, or in pseudocode. You may start with the pictorial representation below, similar to those used in the lecture, and fill in the "..." gaps. Then complete the pseudocode for $\mathcal{B}$. Make sure you can explain all this in words to a peer student.



Reduction $\mathcal{B}^{\textsc{Fn}}$:	Oracle $\mathsf{FnSim}(x_i)$:
1 $\mathsf{K}_2 \leftarrow\$ \{0,1\}^n$	4 $x'_i := \mathsf{K}_2 \oplus x_i$
2 $b'_2 \leftarrow\$ \mathcal{A}^{\mathsf{FnSim}}()$	5 $y'_i \leftarrow \textsc{Fn}(x'_i)$
3 Return b'_2	6 $y_i := y'_i$
	7 Return y_i

Solution: The reduction in words. To prove that F_2 is PRP-secure, we give a reduction to the assumed security of the underlying block cipher E. For any adversary $\mathcal{A}$ against the PRP security of F_2, we construct an adversary $\mathcal{B}$ against the PRP security of E which runs $\mathcal{A}$, simulating Game $\mathbf{PRP}(\mathcal{A}, \mathsf{F}_2)$. Let b_2 and b denote the challenge bits in the PRP games for F_2 and E respectively, and let b'_2 and b' denote the adversary responses (of $\mathcal{A}$, resp. $\mathcal{B}$) in the PRP games for F_2 and E respectively.

- Adversary $\mathcal{B}$ begins by picking a value for K_2 uniformly at random from $\{0,1\}^n$.
  Intuition: The Game $\mathbf{PRP}(\mathcal{B},\mathsf{E})$ that $\mathcal{B}$ is in will pick K_1 (or a random permutation). $\mathcal{B}$ has to take care of K_2 to complete the key needed to simulate $\mathbf{PRP}(\mathcal{A},\mathsf{F}_2)$. $\mathcal{B}$ can do this on its own, since K_2 goes into the message input of E. I.e., it is not a part of the key in $\mathcal{B}$'s game (which is unknown to $\mathcal{B}$).
- Whenever adversary $\mathcal{A}$ makes a query x_i to its oracle Fn (which $\mathcal{B}$ has to simulate), $\mathcal{B}$ queries its own oracle Fn on $x_i' = \mathsf{K}_2 \oplus x_i$.
  Intuition: $\mathcal{A}$ expects its input to be XOR-ed with K_2 before being input to the block cipher; $\mathcal{B}$ takes care of that.
  In return, $\mathcal{B}$ obtains y_i' which it passes on to $\mathcal{A}$.
  Intuition: $\mathcal{A}$ expects $\mathsf{E}(\mathsf{K}_1, \mathsf{K}_2 \oplus x_i)$ or $\Pi(x_i)$ as a response. If $\mathcal{B}$'s oracle is real, this is exactly what y_i' is, so we just forward. (See below for why the random part is also fine.)
- When adversary $\mathcal{A}$ halts and returns bit b_2', adversary $\mathcal{B}$ also halts and returns $b' := b_2'$.
  Intuition: $\mathcal{A}$ tries to guess if its oracle was real or random. As discussed next, $\mathcal{B}$ simulates real resp. random if and only if its own oracle was real resp. random. So $\mathcal{B}$ can use b_2' as its own guess.

3) Argue why $\mathcal{B}$'s simulation for $\mathcal{A}$ is sound. That is, argue that when $\mathcal{B}$ gets real responses ($b = 0$ in $\mathbf{PRP}(\mathcal{B},\mathsf{E})$), it correctly simulates real responses for $\mathcal{A}$, and vice versa for the random world ($b = 1$). Also say why $\mathcal{B}$ is efficient.

Solution. The simulation ensures that $\mathcal{B}$ perfectly answers $\mathcal{A}$'s queries: When the bit b in $\mathcal{B}$'s PRP-game is 0 (i.e., it is in the "real world"), the response y_i' that $\mathcal{B}$ gets from oracle Fn is equal to $\mathsf{F}_2((\mathsf{K}_1,\mathsf{K}_2),x_i)$ for uniformly random K_1 and K_2. When the bit b in $\mathcal{B}$'s PRP-game is 1, the response y_i' of $\mathcal{B}$ is equal to $\Pi(x_i)$ for a random permutation $\Pi : \{0,1\}^n \to \{0,1\}^n$. (For the latter, observe that $\mathcal{B}$ actually obtains $\Pi(\mathsf{K}_2 \oplus x_i)$, but the XOR-ing with K_2 is just another permutation, so $\Pi(\mathsf{K}_2 \oplus x_i)$ still behaves like a random permutation.)

Also note that $\mathcal{B}$ is efficient: All it does is sampling a random key (once) and performing one XOR operation per query of $\mathcal{A}$. In particular, it makes as many Fn queries as $\mathcal{A}$ makes Fn queries.

4) Translate the reduction into an advantage bound. Finally, argue that due to the above, $\mathcal{A}$'s advantage in breaking PRP security of F_2 is bounded by $\mathcal{B}$'s advantage in breaking PRP security of E.

Solution. As a result of the sounds simulation, $\mathcal{B}$ can make direct use of $\mathcal{A}$'s bit guess: we have that

$$\Pr[\,\text{Game } \mathbf{PRP}(\mathcal{A},\mathsf{F}_2) \Rightarrow \mathsf{true}\,] = \Pr[\,\text{Game } \mathbf{PRP}(\mathcal{B},\mathsf{E}) \Rightarrow \mathsf{true}\,],$$

and so

$$\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{E}}(\mathcal{B}) = 2\left|\Pr[\,b' = b\,] - \frac{1}{2}\right| = 2\left|\Pr[\,b_2' = b_2\,] - \frac{1}{2}\right| = \mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{F}_2}(\mathcal{A}).$$

Hence by the assumption that E is PRP-secure, so is F_2.

(c) Let the function $\mathsf{F}_3 : (\{0,1\}^k \times \{0,1\}^n) \times \{0,1\}^n \to \{0,1\}^n$ be defined by

$$\mathsf{F}_3((\mathsf{K}_1,\mathsf{K}_2),x) = \mathsf{E}(\mathsf{K}_1,\mathsf{K}_2) \oplus x.$$

1) Show that F_3 is a block cipher.

2) Argue informally why F_3 is *not* a secure PRP.

3) To make the latter argument formal, define an adversary $\mathcal{A}$ attacking the PRP security of F_3.

   Hint: There is an adversary $\mathcal{A}$ that makes exactly two *distinct* queries to its challenger (we call this a 2-query adversary) and achieves $\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{F}_3}(\mathcal{A}) = 1 - \frac{1}{2^n - 1}$. (This almost-perfect distinguishing advantage is essentially the highest possible achievable advantage.)

Note that despite the similarities between F_2 and F_3, one is secure while the other is not. Think about how misplaced parentheses can have important consequences for security!

Solution.

1) As before, we show that F_3 is a block cipher by giving the inverse function $\mathsf{F}_3^{-1}(\mathsf{K}, \cdot)$ for $\mathsf{K} = (\mathsf{K}_1, \mathsf{K}_2)$. For all $y \in \{0,1\}^n$, we have
$$\mathsf{F}_3^{-1}((\mathsf{K}_1, \mathsf{K}_2), y) = \mathsf{E}(\mathsf{K}_1, \mathsf{K}_2) \oplus y,$$
so F_3 is its own inverse.

2) Observe that the $\mathsf{E}(\mathsf{K}_1, \mathsf{K}_2)$ part in F_3 is *constant* for a fixed key $(\mathsf{K}_1, \mathsf{K}_2)$. In the PRP security game, an adversary querying two values $x_1 \neq x_2$ and receiving responses y_1, y_2, can hence expect to see that the XOR between x_1 and x_2 and the XOR between y_1 and y_2 is the *same* if the FN oracle computes F_3. If instead the FN oracle is a random permutation, this will only happen with very small probability. This allows the adversary to distinguish the two cases and determine the challenge bit b.

3) To make this formal, we are asked to construct a 2-query adversary $\mathcal{A}$ attacking the PRP security of F_3 and achieving $\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{F}_3}(\mathcal{A}) = 1 - \frac{1}{2^n - 1}$. The adversary works as follows:

   Adversary $\mathcal{A}^{\mathrm{FN}}()$:

   1 $x_1 \leftarrow 0^n \,;\, x_2 \leftarrow 1^n$
   2 $y_1 \leftarrow \mathrm{FN}(x_1) \,;\, y_2 \leftarrow \mathrm{FN}(x_2)$
   3 If $y_1 \oplus y_2 = x_1 \oplus x_2$ then return 0
   4 Else return 1

   Let $b \in \{0,1\}$ be the bit and $(\mathsf{K}_1, \mathsf{K}_2) \in \{0,1\}^k \times \{0,1\}^n$ the key used in the game (both drawn uniformly at random during the setup phase), and let b' be the bit

returned by adversary $\mathcal{A}$.

$$\begin{aligned}
\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{F}_3}(\mathcal{A}) &= 2 \cdot \left| \Pr\left[\, b' = b \,\right] - \frac{1}{2} \right| \\
&= 2 \cdot \left| \Pr\left[\, b' = b \,\middle|\, b = 0 \,\right] \Pr\left[\, b = 0 \,\right] + \Pr\left[\, b' = b \,\middle|\, b = 1 \,\right] \Pr\left[\, b = 1 \,\right] - \frac{1}{2} \right| \\
&= \left| \Pr\left[\, b' = b \,\middle|\, b = 0 \,\right] + \Pr\left[\, b' = b \,\middle|\, b = 1 \,\right] - 1 \right| \\
&= \left| \Pr\left[\, b' = 0 \,\middle|\, b = 0 \,\right] + (1 - \Pr\left[\, b' = 0 \,\middle|\, b = 1 \,\right]) - 1 \right| \\
&= \left| \Pr\left[\, b' = 0 \,\middle|\, b = 0 \,\right] - \Pr\left[\, b' = 0 \,\middle|\, b = 1 \,\right] \right| .
\end{aligned}$$

Now, $\Pr\left[\, b' = 0 \,\middle|\, b = 0 \,\right] = 1$, because if $b = 0$ then $y_i = \mathsf{F}_3((\mathsf{K}_1, \mathsf{K}_2), x_i) = \mathsf{E}(\mathsf{K}_1, \mathsf{K}_2) \oplus x_i$ for $i \in \{1, 2\}$, so $y_1 \oplus y_2 = (\mathsf{E}(\mathsf{K}_1, \mathsf{K}_2) \oplus x_1) \oplus (\mathsf{E}(\mathsf{K}_1, \mathsf{K}_2) \oplus x_2) = x_1 \oplus x_2$, and hence adversary $\mathcal{A}$ always returns $b' = 0$ in this case.

For $\Pr\left[\, b' = 0 \,\middle|\, b = 1 \,\right]$, recall that when $b = 1$, $\textsc{Fn}$ is implemented by a uniformly random permutation Π. Hence $\Pr\left[\, b' = 0 \,\middle|\, b = 1 \,\right] = \Pr\left[\, \Pi(x_1) \oplus \Pi(x_2) = x_1 \oplus x_2 \,\right]$. Here $\Pi(x_1)$ and $\Pi(x_2)$ are essentially two random, but distinct, elements in $\{0,1\}^n$, and $x_1 \oplus x_2 = 1^n$ is a fixed string. Hence $\Pr\left[\, \Pi(x_1) \oplus \Pi(x_2) = x_1 \oplus x_2 \,\right] = \frac{1}{2^n - 1}$: fixing $\Pi(x_1)$, there is only one out of $2^n - 1$ possible values to assign to $\Pi(x_2)$ that make $\Pi(x_1) \oplus \Pi(x_2) = 1^n$, and that assignment is uniformly random.

Plugging this in we get the required $\mathbf{Adv}^{\mathrm{PRP}}_{\mathsf{F}_3}(\mathcal{A}) = 1 - \frac{1}{2^n - 1}$.

Problem 5 (Bonus: Historic Ciphers). Kerckhoff's principle essentially states that a cryptosystem should be secure even if everything about the system, except the key, is public knowledge.

(a) Recover the English phrase that is encoded below using the Caesar cipher:

`BPM MVMUG SVWEA BPM AGABMU`

Solution. The encoded phrase is: `THE ENEMY KNOWS THE SYSTEM`. The key is $k = 8$, meaning that the encryption algorithm shifts everything forward by 8 letters, and the decryption algorithm shifts everything back by 8 letters.

(b) The following explains the origin of the above phrase:

`OXYGDOSYMW IYMGJMISI TZ JHYQNSEMLH UF GEHYWL WAHRGVR`

Decrypt it using the Vigenère cipher with the key `ETH`.

Solution. The encoded sentence is meant to point out that the above phrase is an alternative description of `KERCKHOFFS PRINCIPLE(,) AS FORMULATED BY CLAUDE SHANNON`.

You can use websites like cryptii.com to further play around with the Caesar, Vigenère, or other ciphers.

Problem 6 (Challenge: Encryption with a Deck of Cards). Alice, Bob, and Eve are playing a card game. Alice shuffles a deck of cards and deals it all out to herself and Bob (each gets half of the 52 distinct cards). Alice now wishes to send a secret message m to Bob by saying something aloud. Everybody is in the same room, and eavesdropper Eve is listening in: she hears everything Alice says (but Eve cannot see the face of Alice's and Bob's cards).

Let us count the number of possible 26-card combinations Alice can get, out of the 52-card collection:

$$N = C(52, 26) = \binom{52}{26} = \frac{52!}{26! \cdot (52-26)!} = 495918532948104.$$

Note that $2^{48} < N < 2^{49}$.

Both Alice and Bob know the cards that were dealt to Alice (Bob can deduce it by checking which cards he is missing). So they can agree to assign a unique number between 1 to 52 to each of the cards, and then enumerate all 26-card combinations in the lexicographic order from 0 to $N-1$:

Number	Combination
0	$(1, 2, \ldots, 25, 26)$
1	$(1, 2, \ldots, 25, 27)$
2	$(1, 2, \ldots, 26, 27)$
$\ldots$	$\ldots$
$N-1$	$(27, 28, \ldots, 51, 52)$

(a) Suppose Alice's message m is a string of 48 bits. Describe how Alice can communicate m to Bob in such a way that Eve will have *no* information about the value of m.

Hint: Alice and Bob are allowed to devise a public strategy together *before* the cards are dealt.

Solution. Let $k \in \{0, 1, \ldots, N-1\}$ be the number of the 26-card combination dealt to Alice. In order to encrypt a message $m \in \{0,1\}^{48}$, first Alice converts it from the (binary) bit-string into a number x in the decimal system. She then calculates the ciphertext as $c = (x + k) \bmod N$. To decrypt a ciphertext c, Bob recovers $x = (c - k) \bmod N$ and then converts x from the decimal system to the corresponding 48-bit string message $m \in \{0,1\}^{48}$.

This provides perfect security, as, for a given ciphertext, every message has the same probability of being encrypted to that ciphertext: For every message-ciphertext pair, there exists a single key $k \in \{0, 1, \ldots, N-1\}$ such that $c = (x + k) \bmod N$. And (crucially!) each value from $\{0, 1, \ldots, N-1\}$ is equally likely to be chosen as the key if the deck is shuffled perfectly.

(b) Now suppose that Alice's message m is 49 bits. Show that there exists no protocol that allows Alice to communicate m to Bob in such a way that Eve will have no information about m.

Solution. This question asks to come up with a perfectly secure encryption scheme that has message space $\mathcal{M} = \{0,1\}^{49}$ and key space $\mathcal{K}$ of size $|\mathcal{K}| = \binom{52}{26}$, and hence $|\mathcal{M}| > |\mathcal{K}|$. But from the lecture we know that such perfectly secure encryption scheme needs to satisfy $|\mathcal{M}| \leq |\mathcal{K}|$. This leads to a contradiction, i.e., it is impossible to encrypt a 49-bit message without leaking any information to Eve. You should also try to convince yourself of the fact that there is no way to "squeeze" more information out of Alice's hand than we do here, i.e., we could not possibly construct a larger key space and still have perfect security. (For example, even using knowledge of Bob's hand will not help—why?).

Acknowledgements. This exercise sheet is in part inspired by (and adapted from) problems by Rafael Pass, Tom Ristenpart, Mihir Bellare, Phillip Rogaway and Mark Zhandry, as well as from the book "A Graduate Course in Applied Cryptography" by Dan Boneh and Victor Shoup.

References

[1] D. Boneh and V. Shoup. *A Graduate Course in Applied Cryptography.* Online, version 0.6 edition, Jan. 2023.