Midterm II

Name:

SID:

- **You may consult at most *1 double-sided sheet of handwritten notes*. Apart from that, you may not look at books, notes, etc. Calculators, phones, computers, and other electronic devices are NOT permitted for looking up content. However, you may use an electronic device such as a tablet for writing your answers.**
- **DSP Students: If you are allowed $1.5\times$ (resp. $2\times$) the regular exam duration, then you must submit your exam within $120 = 80 * 1.5$ (resp. $160 = 80 * 2$) mins.**
- **The instructors will not be answering questions during the exam. If you feel that something is unclear, please write a note in your answer.**

1 Multiple Choice (20 points)

In the multiple choice section, no explanations are needed for your answers. No points are deducted for wrong answers. Please mark your answers clearly.

1. **Public-Key Encryption:** For each of the following statements, indicate whether it is true or false.

   (a) Encrypting a message using PKE (public-key encryption) is usually slower in practice than encrypting the message using SKE (secret-key encryption).
   ◯ True
   ◯ False
   Solution: True. This is why hybrid encryption is desirable. See Katz & Lindell, 3rd Edition, Section 12.3 for more information.

   (b) EAV security is equivalent to CPA security for PKE schemes.
   ◯ True
   ◯ False
   Solution: True. See lecture 14, slide 14.

   (c) CPA-secure PKE can be constructed from key-exchange protocols and vice versa: key-exchange protocols can be constructed from CPA-secure PKE.
   ◯ True
   ◯ False
   Solution: True. Discussion 8, question 1 shows that key-exchange implies PKE. The midterm review session, slides 48-50, shows that PKE implies key-exchange.

   (d) In hybrid encryption, SKE is used to encrypt a shared public key pk for a PKE scheme.
   ◯ True
   ◯ False
   Solution: False. In hybrid encryption, PKE is used to encrypt a shared secret key k for an SKE scheme.

2. **Hard-Concentrate Predicates:** Let $f : \{0,1\}^n \to \{0,1\}^n$ be a function that has a hard-concentrate predicate $h : \{0,1\}^n \to \{0,1\}$. Also, let $f(x)_{[1,n-1]}$ be $f(x)$ without the nth bit, and let $f(x)_n$ be the nth bit of $f(x)$.

   Select all of the functions below for which h is (necessarily) a hard-concentrate predicate.

   ◯ $g_1(x) = f(x)_{[1,n-1]}$
   ◯ $g_2(x) = f(x)||\big(h(x) \oplus 1\big)$
   ◯ $g_3(x) = f(x)_n \oplus h(x)$
   ◯ $g_4(x) = f(x)_{[1,n-1]}||\big(f(x)_n \oplus h(x)\big)$

   Solution: h is necessarily a hard-concentrate predicate for g_1, but not for g_2, g_3, or g_4.

   For g_1: We know that given $f(x)$, no PPT adversary can guess $h(x)$ with probability $\frac{1}{2} + \mathsf{non\text{-}negl}(n)$. Therefore if the adversary is given $g_1(x) = f(x)_{[1,n-1]}$, which is less information than $f(x)$, it is also hard for them to guess $h(x)$ with probability $\frac{1}{2} + \mathsf{non\text{-}negl}(n)$.

For g_2: Given $g_2(x)$, it is easy for an adversary to learn $h(x)$. They can take the last bit $g_2(x)_{n+1} = h(x) \oplus 1$ and compute $g_2(x)_{n+1} \oplus 1 = h(x)$.

For g_3 and g_4: It's possible that $f(x)_n = 0$ for all inputs x. In this case $g_3(x) = h(x)$, and $g_4(x)_n = h(x)$. So given $g_3(x)$ or $g_4(x)$, it's easy for an adversary to learn $h(x)$.

3. **Constructing A from B:** For each of the following statements, indicate whether it is true or false.

   (a) PRGs can be used to construct PRFs, but PRFs are not sufficient to construct PRGs.
   ◯ True
   ◯ False
   Solution: False. PRGs can be used to construct PRFs, and PRFs can be used to construct PRGs.
   Lecture 12, slides 22-26, shows how to use a PRG to construct a PRF. Discussion 8, question 2.3 shows how to use a PRP to construct a PRG, and by the same argument, it's possible to use a (generic) PRF to construct a PRG.

   (b) Any OWF $f : \{0,1\}^n \to \{0,1\}^{2n}$ is also a PRG.
   ◯ True
   ◯ False
   Solution: False. Consider a OWF whose first bit is always 0. Such a OWF f can be constructed from any OWF $g : \{0,1\}^n \to \{0,1\}^{n-1}$. But f's output can be distinguished from random with advantage $\frac{1}{2}$, so f is not a PRG.

   (c) Any PRG $g : \{0,1\}^n \to \{0,1\}^{2n}$ is also a OWF.
   ◯ True
   ◯ False
   Solution: True. Essentially the same claim was proven in Homework 7, question 1.

   (d) Any length-preserving PRP (pseudorandom permutation) is also a PRF.
   ◯ True
   ◯ False
   Solution: True. This is stated in Katz & Lindell, 3rd Edition, Proposition 3.26.

4. **El Gamal Encryption:** In the El Gamal encryption scheme, let the public key be $\mathsf{pk} = (\mathbb{G}, p, g, g^a)$, where $\mathbb{G}$ is a cyclic group, p is the size of the group, g is a generator of the group, and $a \in \mathbb{Z}_p$ is part of the secret key.

   Which *one* of the following algorithms correctly describes the process to encrypt a message $m \in \mathbb{G}$?

   ◯ Sample $k \leftarrow \mathbb{Z}_p$. Compute $c_1 = g^k$ and $c_2 = g^a \cdot g^k \cdot m$. Output (c_1, c_2).
   ◯ Sample $k \leftarrow \mathbb{Z}_p$. Compute $c_1 = (g^a)^k$ and $c_2 = g^k + m$. Output (c_1, c_2).
   ◯ Sample $k \leftarrow \mathbb{Z}_p$. Compute $c_1 = g^k$ and $c_2 = (g^a)^k \cdot m$. Output (c_1, c_2).
   ◯ Sample $k \leftarrow \mathbb{Z}_p$. Compute $c_1 = (g^a)^k$ and $c_2 = g^k \cdot m$. Output (c_1, c_2).

   Solution: Option C

2 One-Way Functions (15 points)

Question: Let $f : \{0,1\}^n \to \{0,1\}^n$ be a OWF. Use f to construct another OWF g such that $g : \{0,1\}^n \to \{0,1\}^n$ and $g(0^n) = 0^n$. Your answer should describe a construction of g and prove that g is a OWF.

Give a construction of g.

Solution:

$$\text{Let } g(x) = \begin{cases} 0^n & \text{if } x = 0^n, \\ f(x) & \text{otherwise} \end{cases}$$

Note that $g(x)$ satisfies $g(0^n) = 0^n$ for every n as required. Also note that g is efficiently computable: it runs in polynomial time in n.

Name: _____

Prove that the function g constructed above is a secure OWF.

Solution:

1. Overview: Assume for sake of contradiction that g is not one-way. This implies the existence of the following PPT algorithm $\mathcal{A}$ that wins the OWF game for g – $\mathsf{Invert}_{\mathcal{A},g}(n)$ – with non-negligible probability. In math notation, that is:

   $$\Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1] \geq \mathsf{nonnegl}_1(n).$$

   Then we will show that $\mathcal{A}$ can actually break the OWF security of f as well:

   $$\Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1] \geq \mathsf{nonnegl}_2(n).$$

   This is a contradiction because f is a secure OWF. Therefore, our original assumption was wrong, and in fact, g is a secure OWF.

2. The first key insight is that if we play $\mathsf{Invert}_{\mathcal{A},g}(n)$ and condition on the event that the challenger does not sample $x = 0^n$, then $\mathcal{A}$'s success probability changes by a negligible amount. This is because $\Pr[x = 0^n] = \frac{1}{2^n}$.

   Here's more detail:

   $$\begin{aligned}
   \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1] &= \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n] \cdot \Pr[x \neq 0^n] + \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x = 0^n] \cdot \Pr[x = 0^n] \\
   &\leq \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n] + \Pr[x = 0^n] \\
   &= \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n] + \frac{1}{2^n} \\
   \mathsf{non\text{-}negl}_1(n) - \frac{1}{2^n} &\leq \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n]
   \end{aligned}$$

3. The second key insight is that if we condition on the event that $x \neq 0^n$, then the OWF games for f and g are the same. This implies:

   $$\Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1 | x \neq 0^n] = \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n]$$

4. Now we will put these ideas together to show that $\Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1] \geq \mathsf{non\text{-}negl}_2(n)$.

   $$\begin{aligned}
   \Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1] &= \Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1 | x \neq 0^n] \cdot \Pr[x \neq 0^n] + \Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1 | x = 0^n] \cdot \Pr[x = 0^n] \\
   &\geq \Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1 | x \neq 0^n] \cdot \Pr[x \neq 0^n] \\
   &= \Pr[\mathsf{Invert}_{\mathcal{A},f}(n) = 1 | x \neq 0^n] \cdot \left(1 - \frac{1}{2^n}\right) \\
   &= \Pr[\mathsf{Invert}_{\mathcal{A},g}(n) = 1 | x \neq 0^n] \cdot \left(1 - \frac{1}{2^n}\right) \\
   &\geq \left(\mathsf{nonnegl}_1(n) - \frac{1}{2^n}\right) \cdot \left(1 - \frac{1}{2^n}\right) \\
   &= \mathsf{nonnegl}_2(n).
   \end{aligned}$$

5. Thus, we have shown that if g is not one-way, then f is not one-way either. The contrapositive is also true: if f is one-way, then g is one-way.

3 Domain Extension with CRHFs (25 Points)

We will examine a simple way to extend the domain of a MAC by first hashing the message with a CRHF.

Let $F : \{0,1\}^n \times \{0,1\}^n \to \{0,1\}^n$ be a pseudorandom function.

Let $\mathcal{H} = (\mathsf{Gen}, H)$ be a collision-resistant hash function with key space $\{0,1\}^n$ and input space $\mathcal{X}$, which may be very large. For every key $s \leftarrow \mathsf{Gen}(1^n)$, $s \in \{0,1\}^n$ and $H^s : \mathcal{X} \to \{0,1\}^n$.

Let $G : \{0,1\}^{2n} \times \mathcal{X} \to \{0,1\}^n$ be defined as follows:

$$G((k,s),x) = F\big(k, H^s(x)\big)$$

3.1 Pseudorandom Function (15 Points)

Question: Prove that G is a pseudorandom function.

You may wish to follow the template provided below.

Let's define several hybrids. For a given adversary $\mathcal{A}$:

1. Let $\mathsf{Hyb}_0(\mathcal{A}, n)$ be the PRF security game in which the adversary $\mathcal{A}$ gets query access to G. In particular:
   (a) The PRF challenger samples $k \leftarrow \{0,1\}^n$ and $s \leftarrow \mathsf{Gen}(1^n)$.
   (b) The adversary $\mathcal{A}$ gets query access to the following function:
   $$G(\cdot) = F(k, H^s(\cdot))$$
   (c) The adversary outputs a bit b, which is the output of the hybrid.
2. Let $\mathsf{Hyb}_1(\mathcal{A}, n)$ be the same as $\mathsf{Hyb}_0(\mathcal{A}, n)$, except $F(k,\cdot)$ is replaced with a uniformly random function $R_1 : \{0,1\}^n \to \{0,1\}^n$:
   (a) The PRF challenger samples a function R_1 uniformly at random from the set of all functions mapping $\{0,1\}^n \to \{0,1\}^n$. They also sample $s \leftarrow \mathsf{Gen}(1^n)$.
   (b) The adversary $\mathcal{A}$ gets query access to the following function:
   $$R_1(H^s(\cdot))$$
   (c) The adversary outputs a bit b, which is the output of the hybrid.

3. Let $\mathsf{Hyb}_2(\mathcal{A}, n)$ be the same as $\mathsf{Hyb}_0(\mathcal{A}, n)$ except $F(k, H^s(\cdot))$ is replaced with a uniformly random function $R_2 : \mathcal{X} \to \{0,1\}^n$:

   (a) The PRF challenger samples a function R_2 uniformly at random from the set of all functions mapping $\mathcal{X} \to \{0,1\}^n$.

   (b) The adversary $\mathcal{A}$ gets query access to:

   $$R_2(\cdot)$$

   (c) The adversary outputs a bit b, which is the output of the hybrid.

Lemma 3.1 *For any PPT adversary $\mathcal{A}$,* $\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right| \leq \mathsf{negl}(n)$.

Proof:

Solution: This follows from the PRF security of F.

1. Overview: Assume toward contradiction that there exists a PPT adversary $\mathcal{A}$ such that $\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right|$ is $\mathsf{non\text{-}negl}(n)$. Then we will use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that breaks the PRF security of F.

2. $\mathcal{B}$ will play in the PRF security game. $\mathcal{B}^{F(k,\cdot)}$ will end up simulating $\mathsf{Hyb}_0(\mathcal{A}, n)$, and $\mathcal{B}^{R_1(\cdot)}$ will end up simulating $\mathsf{Hyb}_1(\mathcal{A}, n)$.

   Construction of $\mathcal{B}$:

   (a) The PRF challenger either samples $k \leftarrow \{0,1\}^n$ and gives $\mathcal{B}$ query access to $F(k, \cdot)$, or they sample $R_1 : \{0,1\}^n \to \{0,1\}^n$ uniformly at random and give $\mathcal{B}$ query access to $R_1(\cdot)$.

   (b) $\mathcal{B}$ samples $s \leftarrow \mathsf{Gen}(1^n)$.

   (c) $\mathcal{B}$ runs $\mathcal{A}$ internally. Whenever $\mathcal{A}$ produces a query x, $\mathcal{B}$ computes $y = H^s(x)$ and forwards y as a query to its PRF challenger. $\mathcal{B}$ forwards whatever response it gets from its challenger to $\mathcal{A}$.

   (d) Finally, $\mathcal{A}$ outputs a bit b, which $\mathcal{B}$ outputs as well.

3. Analysis: If $\mathcal{B}$ was given query access to $F(k, \cdot)$, then $\mathcal{B}$ ended up simulating $\mathsf{Hyb}_0(\mathcal{A}, n)$. Therefore:

   $$\Pr[\mathcal{B}^{F(k,\cdot)} \to 1] = \Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1]$$

   If $\mathcal{B}$ was given query access to $R_1(\cdot)$, then $\mathcal{B}$ ended up simulating $\mathsf{Hyb}_1(\mathcal{A}, n)$. Therefore:

   $$\Pr[\mathcal{B}^{R_1(\cdot)} \to 1] = \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]$$

   In summary:

   $$\left|\Pr[\mathcal{B}^{F(k,\cdot)} \to 1] - \Pr[\mathcal{B}^{R_1(\cdot)} \to 1]\right| = \left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right|$$

4. If $\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right| = \mathsf{non\text{-}negl}(n)$, then $\mathcal{B}$ breaks the PRF security of F. This is a contradiction because F is a secure PRF, so our initial assumption was wrong. In fact, for any PPT adversary $\mathcal{A}$,

$$\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right| \leq \mathsf{negl}(n)$$

Lemma 3.2 *For any PPT adversary $\mathcal{A}$, $\left|\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1]\right| \leq \mathsf{negl}(n)$.*

Proof:

Solution: Note: The text in gray is included to clarify our argument, but it goes beyond the level of detail expected from students on an exam.

1. Intuition: This follows from the CRHF security of H. In Hyb_1, if $\mathcal{A}$ never queries its oracle on two inputs that collide in H^s, then each distinct query submitted by $\mathcal{A}$ will receive in response a random string sampled independently of the other responses. In this case, $\mathcal{A}$ cannot distinguish Hyb_1 and Hyb_2 because because the distribution of responses will be identical in both hybrids.

   The flip side is that if $\mathcal{A}$ can distinguish Hyb_1 and Hyb_2, then we can use $\mathcal{A}$ to find collisions in H^s by recording the queries that $\mathcal{A}$ makes and checking if any of them collide.

2. Let $\mathcal{A}$ be a PPT adversary for Hyb_1 and Hyb_2. Next, we will use $\mathcal{A}$ to construct an adversary $\mathcal{B}$ that tries to find collisions in H^s.

   Construction of $\mathcal{B}$:

   (a) The CRHF challenger samples $s \leftarrow \mathsf{Gen}(1^n)$ and gives s to $\mathcal{B}$.

   (b) $\mathcal{B}$ will simulate Hyb_1 with $\mathcal{A}$ and keep a database of the queries from $\mathcal{A}$ and responses from $\mathcal{B}$. The database contains entries of the form (x, y, r_y), where $x \in \mathcal{X}$, $y = H^s(x)$, and $r_y \in \{0,1\}^n$ serves as the response $R_1(y)$.

   (c) $\mathcal{B}$ runs $\mathcal{A}$ internally. Whenever $\mathcal{A}$ outputs a query x to the $R_1(H^s(\cdot))$ oracle:

      i. $\mathcal{B}$ computes $y = H^s(x)$ and searches for y in the database.

      ii. A. Case 1: This y-value does not appear in any entries from the database. Then $\mathcal{B}$ samples a random string $r_y \leftarrow \{0,1\}^n$.

      B. Case 2: This y-value does appear in a database entry. The entry has the form (x', y, r_y), where $H^s(x') = y$. Then $\mathcal{B}$ uses the value of r_y given in this entry.

      iii. $\mathcal{B}$ sends r_y to $\mathcal{A}$.

      iv. $\mathcal{B}$ adds (x, y, r_y) to the database if it's not there already.

   (d) When $\mathcal{A}$ finishes running, $\mathcal{B}$ searches the database for a collision in H^s. It looks for two values x and x' such that $x \neq x'$, and $H^s(x) = H^s(x')$. If $\mathcal{B}$ finds such a pair, it ouputs (x, x'). Otherwise $\mathcal{B}$ outputs $\perp$.

3. Let C be the event that $\mathcal{A}$'s queries include a collision in H^s.[1] Since $\mathcal{H}$ is collision resistant,

$$\Pr[\mathsf{C}] = \Pr[\mathcal{B} \text{ finds a collision in } H^s] = \mathsf{negl}(n)$$

4. Next, we'll show that

$$\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 | \neg\mathsf{C}] = \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1 | \neg\mathsf{C}]$$

In Hyb_1, if $\neg\mathsf{C}$ occurs, then every distinct query made by $\mathcal{A}$ will receive in response a uniformly random string that is independent of all the other inputs that $\mathcal{A}$ queried on. This is the same distribution of responses that $\mathcal{A}$ receives in Hyb_2 if $\neg\mathsf{C}$ occurs.

5. We will finish with some algebra:

$$\begin{aligned}\text{Let } \alpha &= \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1] \\ \beta &= \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1] \\ \gamma &= \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 | \neg\mathsf{C}] = \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1 | \neg\mathsf{C}]\end{aligned}$$

Claim 3.1 $|\alpha - \gamma| \leq \mathsf{negl}(n)$.

Proof:

$$\begin{aligned}\alpha = \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1] &= \underbrace{\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 \text{ AND } \neg\mathsf{C}]}_{=\Pr[\neg\mathsf{C}]\cdot\Pr[\mathsf{Hyb}_1(\mathcal{A},n)\to 1|\neg\mathsf{C}]} + \underbrace{\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 \text{ AND } \mathsf{C}]}_{\leq \Pr[\mathsf{C}]} \\ &\leq \Pr[\neg\mathsf{C}] \cdot \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 | \neg\mathsf{C}] + \Pr[\mathsf{C}] \\ &\leq \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1 | \neg\mathsf{C}] + \Pr[\mathsf{C}] \\ &= \gamma + \mathsf{negl}(n) \\ |\alpha - \gamma| &\leq \mathsf{negl}(n)\end{aligned}$$

■

6. By a similar argument, we can show that $|\beta - \gamma| \leq \mathsf{negl}(n)$.

7. Next, we use the triangle inequality to conclude that $\left|\Pr\left[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1\right] - \Pr\left[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1\right]\right|$ is negligible:

$$\begin{aligned}\left|\Pr\left[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1\right] - \Pr\left[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1\right]\right| &= |\alpha - \beta| \\ &= |(\alpha - \gamma) - (\beta - \gamma)| \\ &\leq |\alpha - \gamma| + |\beta - \gamma| \\ &\leq 2 \cdot \mathsf{negl}(n)\end{aligned}$$

[1] We can define C for Hyb_2 as well as for Hyb_1. In Hyb_2, we can imagine that the challenger samples $s \leftarrow \mathsf{Gen}(1^n)$ at the beginning but doesn't use it. Then at the end of the hybrid, we can check whether any of $\mathcal{A}$'s queries include a collision in H^s to decide whether C occurred.

Finish the proof.

Solution: In summary, for any PPT adversary $\mathcal{A}$, there exist negligible functions $(\mathsf{negl}_1, \mathsf{negl}_2, \mathsf{negl}_3)$ such that:

$$\begin{aligned}\left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1]\right| &\leq \left|\Pr[\mathsf{Hyb}_0(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1]\right| \\ &\quad + \left|\Pr[\mathsf{Hyb}_1(\mathcal{A}, n) \to 1] - \Pr[\mathsf{Hyb}_2(\mathcal{A}, n) \to 1]\right| \\ &= \mathsf{negl}_1(n) + \mathsf{negl}_2(n) = \mathsf{negl}_3(n)\end{aligned}$$

Since $\mathsf{Hyb}_0(\mathcal{A}, n)$ and $\mathsf{Hyb}_2(\mathcal{A}, n)$ are the two hybrids that the PRF adversary for G is asked to distinguish, this implies that G is a secure PRF.

Name:

3.2 Message Authentication Code (10 points)

Question: Use G (defined above) to construct a secure MAC $\Pi = (\mathsf{Gen}_\Pi, \mathsf{Mac}_\Pi, \mathsf{Verify}_\Pi)$ that takes messages $m \in \mathcal{X}$.

You may use the template provided below. You do not need to prove that your construction is secure.

1. $\mathsf{Gen}_\Pi(1^n)$: Sample $k \leftarrow \{0,1\}^n$ and $s \leftarrow \mathsf{Gen}(1^n)$, and output $k_\Pi = (k, s)$.
2. $\mathsf{Mac}_\Pi(k_\Pi, m)$: Output
$$t = G(k_\Pi, m) = F(k, H^s(m))$$
3. $\mathsf{Verify}_\Pi(k_\Pi, m, t)$: If $t = \mathsf{Mac}_\Pi(k_\Pi, m)$, then return 1. Otherwise, return 0.

The proof that Π is a secure MAC is essentially the same as the proof of theorem 4.6 in Katz & Lindell, 3rd Edition or the proof in lecture 9, slides 8-9.

4 Public-Key Encryption (20 points)

The composition of two PKE schemes with independent keys is CPA-secure as long as at least one of the schemes is CPA-secure. We will show most of the proof of this claim.

Question: Follow the outline given below and fill in any blanks.

Let us be given two public-key encryption schemes $\Pi_1 = (\mathsf{Gen}_1, \mathsf{Enc}_1, \mathsf{Dec}_1)$ and $\Pi_2 = (\mathsf{Gen}_2, \mathsf{Enc}_2, \mathsf{Dec}_2)$. Let the ciphertext space of Enc_2 be the same as the message space of Enc_1. Also, one of Π_1 or Π_2 is CPA secure, and the other one is not, but we don't know which one is secure.

Define the composed scheme $\Pi = (\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ as follows. **Fill in the algorithm for** Dec **so that** Π **satisfies correctness.**

- $\mathsf{Gen}(1^n)$: Run $\mathsf{Gen}_1(1^n) \to (\mathsf{pk}_1, \mathsf{sk}_1)$ and $\mathsf{Gen}_2(1^n) \to (\mathsf{pk}_2, \mathsf{sk}_2)$. Return $((\mathsf{pk}_1, \mathsf{pk}_2), (\mathsf{sk}_1, \mathsf{sk}_2))$.
- $\mathsf{Enc}((\mathsf{pk}_1, \mathsf{pk}_2), m)$: Return $c = \mathsf{Enc}_1(\mathsf{pk}_1, \mathsf{Enc}_2(\mathsf{pk}_2, m))$.
- $\mathsf{Dec}((\mathsf{sk}_1, \mathsf{sk}_2), c)$: Return $m' = \mathsf{Dec}_2(\mathsf{sk}_2, \mathsf{Dec}_1(\mathsf{sk}_1, c))$

Theorem 4.1 *If* Π_1 *is CPA-secure or* Π_2 *is CPA-secure, then* Π *is CPA-secure.*

Proof:

1. Overview: To show that Π is CPA-secure, we will give a proof by contradiction. Suppose that there is a PPT adversary $\mathcal{A}$ that wins the CPA security game for Π with non-negligible probability. Then we will construct an adversary $\mathcal{B}_1$ for the CPA game for Π_1 and an adversary $\mathcal{B}_2$ for the CPA game for Π_2. Both $\mathcal{B}_1$ and $\mathcal{B}_2$ will succeed with non-negligible probability, which breaks CPA security for both Π_1 and Π_2. This contradicts the fact that at least one of them was CPA-secure.

Name:

2. **Use $\mathcal{A}$ to construct an adversary $\mathcal{B}_1$ for the CPA game for Π_1. $\mathcal{B}_1$ should win the CPA game for Π_1 with the same probability that $\mathcal{A}$ wins the CPA game for Π. Do not include the proof that your adversary works, just construct the adversary.**

Solution: Consider the CPA game for Π_1 with a challenger and an adversary $\mathcal{B}_1$ that we will construct:

(a) The challenger samples $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{Gen}_1(1^n)$ and sends pk_1 to $\mathcal{B}_1$.

(b) $\mathcal{B}_1$ samples $(\mathsf{sk}_2, \mathsf{pk}_2) \leftarrow \mathsf{Gen}_2(1^n)$ and sends $(\mathsf{pk}_1, \mathsf{pk}_2)$ to $\mathcal{A}$.

(c) $\mathcal{B}_1$ receives the challenge messages (m_0, m_1) from $\mathcal{A}$, and then computes

$$(m'_0, m'_1) = (\mathsf{Enc}_2(\mathsf{pk}_2, m_0), \mathsf{Enc}_2(\mathsf{pk}_2, m_1))$$

Then $\mathcal{B}_1$ sends (m'_0, m'_1) as its challenge messages to the challenger.

(d) The challenger samples a bit b and returns $\mathsf{Enc}_1(\mathsf{pk}_1, m'_b)$ to $\mathcal{B}_1$ who sends the same challenge ciphertext unchanged to $\mathcal{A}$.

(e) $\mathcal{A}$ returns a guess b' and $\mathcal{B}_1$ outputs the same bit as its guess.

Note that by construction, $\mathcal{A}$ gets the correct input distribution of the public key as well as the challenge ciphertext, since $\mathsf{Enc}_1(\mathsf{pk}_1, m'_b) = \mathsf{Enc}_1(\mathsf{pk}_1, \mathsf{Enc}_2(\mathsf{pk}_2, m_b))$. Hence, the success probability of $\mathcal{A}$ in the CPA game for Π is the same as the success probability of $\mathcal{B}_1$ in the CPA game for Π_1.

Name:

3. **Use $\mathcal{A}$ to construct an adversary $\mathcal{B}_2$ for the CPA game for Π_2. $\mathcal{B}_2$ should win the CPA game for Π_2 with the same probability that $\mathcal{A}$ wins the CPA game for Π. Do not include the proof that your adversary works, just construct the adversary.**

Solution: Consider the CPA game for Π_2 with a challenger and an adversary $\mathcal{B}_2$ that we will construct:

(a) The challenger samples $(\mathsf{sk}_2, \mathsf{pk}_2) \leftarrow \mathsf{Gen}_2(1^n)$ and sends pk_2 to $\mathcal{B}_2$.

(b) $\mathcal{B}_2$ samples $(\mathsf{sk}_1, \mathsf{pk}_1) \leftarrow \mathsf{Gen}_1(1^n)$ and sends $(\mathsf{pk}_1, \mathsf{pk}_2)$ to $\mathcal{A}$.

(c) $\mathcal{B}_2$ gets the challenge messages (m_0, m_1) from $\mathcal{A}$, and sends them as its own challenge messages to the challenger.

(d) The challenger samples a bit b and returns $c_b = \mathsf{Enc}_2(\mathsf{pk}_2, m_b)$ to $\mathcal{B}_2$ who computes $\mathsf{Enc}_1(\mathsf{pk}_1, c_b)$ and sends the modified ciphertext to $\mathcal{A}$.

(e) $\mathcal{A}$ returns a guess b' and $\mathcal{B}_2$ outputs the same bit as its guess.

Note that by construction, $\mathcal{A}$ gets the correct input distribution of the public key as well as the challenge ciphertext, since $\mathsf{Enc}_1(\mathsf{pk}_1, c_b) = \mathsf{Enc}_1(\mathsf{pk}_1, \mathsf{Enc}_2(\mathsf{pk}_2, m_b))$. Hence, the success probability of $\mathcal{A}$ in the CPA game for Π is the same as the success probability of $\mathcal{B}_2$ in the CPA game for Π_2.

■