This is a non-collaborative assignment. You may not discuss the problems with any other students. The only resources you may use are the useful links given on the course website, the lecture notes, and the course staff.

1 Midterm (MI) Security

In this problem, you will consider two notions of security for symmetric-key encryption schemes. Recall that in a symmetric-key encryption scheme, Alice and Bob have the same secret key $sk$. There is no public key.

**Definition 1 (MI6 security)** A symmetric-key encryption scheme $(KeyGen, Enc, Dec)$ is MI6-secure if for all p.p.t. algorithms $A$, there is a negligible function $\nu$ such that:

$$
\Pr[\text{sk} \leftarrow \text{KeyGen}(1^k); (m_0, m_1, s) \leftarrow A(1^k); b \leftarrow \{0, 1\}; c \leftarrow \text{Enc}(sk, m_b); b' \leftarrow A(1^k, c, s) : b' = b] \leq \frac{1}{2} + \nu(k)
$$

The $s$ produced by $A$ can be thought of as a form of state or memory between $A$’s two runs.

In other words, an encryption scheme is MI6-secure if no p.p.t. adversary can distinguish between encryptions of a chosen $m_0$ and $m_1$ with nonnegligible advantage.

**Definition 2 (MI7 security)** A symmetric-key encryption scheme $(KeyGen, Enc, Dec)$ is MI7-secure if for all p.p.t. algorithms $A$, there is a negligible function $\nu$ such that:

$$
\Pr[sk \leftarrow \text{KeyGen}(1^k); (m_0, m_1, s) \leftarrow A(1^k); b \leftarrow \{0, 1\}; c_0 \leftarrow \text{Enc}(sk, m_0); c_1 \leftarrow \text{Enc}(sk, m_1); z \leftarrow \text{if } b = 0 \text{ then } (c_0, c_1) \text{ else } (c_1, c_0); b' \leftarrow A(1^k, z, s) : b' = b] \leq \frac{1}{2} + \nu(k)
$$

In contrast to MI6 security, an encryption scheme is MI7-secure if no p.p.t. adversary can distinguish between $(c_0, c_1)$ and $(c_1, c_0)$ for chosen messages $m_0$ and $m_1$ with nonnegligible advantage.

Consider the relationship between MI6 and MI7 security. Does one necessarily imply the other? Are they equivalent? Prove your answer using reductions and/or counterexamples.
2 More Fun with PRGs

Let $G : \{0,1\}^k \rightarrow \{0,1\}^{2k}$ be a pseudorandom generator.

a. Define $G_a : \{0,1\}^k \rightarrow \{0,1\}^{4k}$ as follows:

$$G_a(x) = G(G(x))$$

Prove that $G_a$ is a PRG or provide a counterexample to the contrary.

b. Define $G_b : \{0,1\}^k \rightarrow \{0,1\}^{\text{poly}(k)}$ as follows:

$$G_b(x) = \text{the number of 0's in } G(x), \text{ written as a poly}(k)\text{-bit string}$$

Prove that $G_b$ is a PRG or provide a counterexample to the contrary.

3 Even More Fun with OWFs

a. Let $g : \{0,1\}^k \rightarrow \{0,1\}^k$ be a one-way permutation. Define $f_a$ as:

$$f_a(x) = g(x) \mod 2^{k-1}$$

Assuming that OWPs exist, does it follow that $f_a$ is a OWF? If so, prove it. If not, give a counterexample.

b. Let $f : \{0,1\}^k \rightarrow \{0,1\}^k$ be a length-preserving one-way function. Define $f_b$ as:

$$f_b(x) = f(x) \mod 2^{k-1}$$

Assuming that length-preserving OWFs exist, does it follow that $f_b$ is a OWF? If so, prove it. If not, give a counterexample.

c. Consider the following proof that $f_b$ is not necessarily a OWF. Does it hold? If not, explain what is wrong with the reasoning.

**Proof:** As we saw in lecture, $f(f(x))$ is not necessarily a OWF because we can construct a OWF $f$ such that $f(f(x))$ is an all-to-one mapping, meaning that every $x$ maps to the all-zero string.

Let $f_1 : \{0,1\}^k \rightarrow \{0,1\}^k$ be a OWF, and for every $i \in \{1,2,\ldots,k\}$, let:

$$f_{i+1}(x) = f_i(x) \mod 2^{k-i}$$

Note that leading zeros are kept when reducing mod $2^{k-i}$. Then since $f_{k+1}(x)$ is always the all-zero string, it is easy to invert. By the hybrid argument, there must exist some $i \in \{1,2,\ldots,k\}$ such that $f_i$ is easy to invert but $f_{i-1}$ is a OWF. This $f_i$ is a counterexample to $f_b$ necessarily being a OWF. □
4 Breaking our Assumptions

In both RSA and QR-based cryptosystems, we have implicitly assumed that factoring the public value $N = pq$ is hard. Specifically, it has been crucial that an adversary cannot efficiently learn $\varphi(N) = (p-1)(q-1)$. However, it is not even necessary for the adversary to learn $\varphi(N)$ exactly to break the scheme. There are in fact whole classes of values related to $\varphi(N)$ that are sufficient to mount an equally powerful attack.

a. **RSA warmup**: Given an RSA public modulus $N = pq$, public exponent $e$, and the prime factors $p$ and $q$ of $N$, show how to efficiently compute the private exponent $d$.

b. Given an RSA public modulus $N = pq$, public exponent $e$, and $\lambda(N) = \alpha\varphi(N)$ for some unknown positive integer $\alpha$, show how to efficiently compute an exponent $d'$ which allows you to invert the RSA trapdoor permutation.

c. **QR warmup**: Recall that $\text{QR}_N$ denotes the set of quadratic residues mod $N$. We use $\text{QNR}_N$ to denote the set of elements mod $N$ that are quadratic nonresidues both mod $p$ and mod $q$. Given a Blum integer $N = pq$ with $p \equiv q \equiv 3 \pmod{4}$ as well as its prime factors $p$ and $q$, show how to efficiently distinguish between elements of $\text{QR}_N$ and $\text{QNR}_N$.

d. Given a Blum integer $N = pq$ with $p \equiv q \equiv 3 \pmod{4}$ as well as $\lambda(N) = \alpha\varphi(N)$ for some unknown positive odd integer $\alpha$, show how to efficiently distinguish between elements of $\text{QR}_N$ and $\text{QNR}_N$.

e. Can you extend your scheme from part (d) to efficiently handle an even $\alpha$? You may assume that the bit length of $\alpha$ is polynomial in the security parameter $k$. If so, explain why your scheme works and how to proceed when you don’t know whether $\alpha$ is even or odd. If not, argue why such a scheme is impossible.
5 Hardcore Bits and OWFs

Consider the following definition of a hardcore bit:

**Definition 3 (Hardcore bit)** Let $\text{KeyGen}$ be a key generation algorithm. Let $B_{pk}$ be an efficiently computable Boolean function $B_{pk} : D_{pk} \rightarrow \{0, 1\}$. Note that potentially both $B_{pk}$ and the domain $D_{pk}$ depend on the public key $pk \leftarrow \text{KeyGen}(1^k)$. Let $f_{pk} : D_{pk} \rightarrow R_{pk}$ be any efficiently computable function from the domain $D_{pk}$ to the range $R_{pk}$. $B_{pk}$ is a hardcore bit of $f_{pk}$ if the following two distributions are indistinguishable:

$$D_0(1^k) = \{ pk \leftarrow \text{KeyGen}(1^k); x \leftarrow D_{pk} : (f_{pk}(x), B_{pk}(x)) \}$$

$$D_1(1^k) = \{ pk \leftarrow \text{KeyGen}(1^k); x \leftarrow D_{pk} : (f_{pk}(x), B_{pk}(x) \oplus 1) \}$$

Show that any injective (one-to-one) function that has a hardcore bit must be one-way.

**Note:** To convince yourself that this is true, it may be helpful to try to think of a many-to-one function with a hardcore bit that is not one-way.