CS 388H: Cryptography

Cryptographic Definitions

Instructor: David Wu

In this note, we will recall the main definitions of the cryptographic notions encountered in this course.

1 Cryptographic Building Blocks

Pseudorandom generators (PRGs). Let $G: \{0,1\}^\lambda \to \{0,1\}^n$ be an efficiently-computable function where $n > \lambda$. We define the following PRG security experiments:

Experiment $b = 0$:	**Experiment** $b = 1$:
1. The challenger samples $s \xleftarrow{R} \{0,1\}^\lambda$ and sends $t \leftarrow G(s)$ to $\mathcal{A}$. 2. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.	1. The challenger samples $t \xleftarrow{R} \{0,1\}^n$ and gives t to $\mathcal{A}$. 2. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

We say G is a secure PRG if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{PRGAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \text{negl}(\lambda).$$

Pseudorandom functions (PRFs). Let $F: \mathcal{K} \times \mathcal{X} \to \mathcal{Y}$ be an efficiently-computable function with a key space $\mathcal{K}$, domain $\mathcal{X}$, and range $\mathcal{Y}$ (technically, each of these sets is a function of the security parameter λ). We now define the following PRF security experiments:

Experiment $b = 0$:	**Experiment** $b = 1$:
1. The challenger samples $k \xleftarrow{R} \mathcal{K}$. 2. The adversary can now adaptively make queries to the challenger. In each query, the adversary chooses an input $x \in \mathcal{X}$, and the challenger replies with $F(k, x)$. 3. The adversary outputs a bit $b' \in \{0,1\}$.	1. The challenger samples a function $f \xleftarrow{R} \mathsf{Funs}[\mathcal{X}, \mathcal{Y}]$. 2. The adversary can now adaptively make queries to the challenger. In each query, the adversary chooses an input $x \in \mathcal{X}$, and the challenger replies with $f(x)$. 3. The adversary outputs a bit $b' \in \{0,1\}$.

We say that F is a secure PRF if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{PRFAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \text{negl}(\lambda).$$

In the above definition, $\mathsf{Funs}[\mathcal{X}, \mathcal{Y}]$ denotes the set of *all* functions $f: \mathcal{X} \to \mathcal{Y}$.

Pseudorandom permutations (PRPs). Let $F: \mathcal{K} \times \mathcal{X} \to \mathcal{X}$ be an efficiently-computable function with a key space $\mathcal{K}$ and domain $\mathcal{X}$ (technically, each of these sets is a function of the security parameter λ). We say that F is a pseudorandom permutation (PRP) if the following properties hold:

- For every key $k \in \mathcal{K}$, the function $F(k, \cdot)$ is a permutation on $\mathcal{X}$.
- There exists an efficiently-computable function $F^{-1}: \mathcal{K} \times \mathcal{X} \to \mathcal{X}$ such that for all $k \in \mathcal{K}$ and all $x \in \mathcal{X}$,

$$F^{-1}(k, F(k, x)) = x.$$

For security, we define the following PRP security experiments:

Experiment $b = 0$:	**Experiment** $b = 1$:
1. The challenger samples $k \xleftarrow{R} \mathcal{K}$. 2. The adversary can now adaptively make queries to the challenger. In each query, the adversary chooses an input $x \in \mathcal{X}$, and the challenger replies with $F(k, x)$. 3. The adversary outputs a bit $b' \in \{0,1\}$.	1. The challenger samples a function $f \xleftarrow{R} \mathsf{Perm}[\mathcal{X}]$. 2. The adversary can now adaptively make queries to the challenger. In each query, the adversary chooses an input $x \in \mathcal{X}$, and the challenger replies with $f(x)$. 3. The adversary outputs a bit $b' \in \{0,1\}$.

We say that F is a secure PRP if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{PRPAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathrm{negl}(\lambda).$$

In the above definition, $\mathsf{Perm}[\mathcal{X}]$ denotes the set of *all* permutations $f \colon \mathcal{X} \to \mathcal{X}$.

Collision-resistant hash functions (CRHFs). Let $H \colon \{0,1\}^n \to \{0,1\}^m$ where $m < n$ (for full formality, the hash function would be indexed by a security parameter λ and n, m are polynomials in λ). We say that H is a collision-resistant hash function if for all efficient (uniform) adversaries $\mathcal{A}$ (that takes the security parameter λ as input),

$$\mathsf{CRHFAdv}[\mathcal{A}] = \Pr[(x, y) \leftarrow \mathcal{A} : H(x) = H(y) \text{ and } x \neq y] = \mathrm{negl}(\lambda).$$

2 Symmetric Encryption

A symmetric encryption scheme (also called a cipher) is defined over a key space $\mathcal{K}$, a message space $\mathcal{M}$, and a ciphertext space $\mathcal{C}$ (technically, each of these sets is a function of the security parameter λ) and consists of two efficient algorithms:

- $\mathsf{Encrypt}(k, m) \to \mathsf{ct}$: On input a key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$, the encryption algorithm outputs a ciphertext ct.
- $\mathsf{Decrypt}(k, \mathsf{ct}) \to m/\perp$: On input a key $k \in \mathcal{K}$ and a ciphertext $\mathsf{ct} \in \mathcal{C}$, the decryption algorithm either outputs a message $m \in \mathcal{M}$ or a special symbol $\perp$ (to indicate a decryption failure).

Correctness. The encryption scheme is correct if for all keys $k \in \mathcal{K}$ and all messages $m \in \mathcal{M}$,

$$\Pr[\mathsf{Decrypt}(k, \mathsf{Encrypt}(k, m)) = m] = 1.$$

Perfect secrecy. The encryption scheme satisfies perfect secrecy if for all pairs of messages $m_0, m_1 \in \mathcal{M}$ and all ciphertext $\mathsf{ct} \in \mathcal{C}$,

$$\Pr[k \xleftarrow{\mathrm{R}} \mathcal{K} : \mathsf{Encrypt}(k, m_0) = c] = \Pr[k \xleftarrow{\mathrm{R}} \mathcal{K} : \mathsf{Encrypt}(k, m_1) = c].$$

Semantic security. We start by defining the semantic security experiment:

Experiment $b = 0$:	**Experiment** $b = 1$:
1. The challenger samples a key $k \xleftarrow{\mathrm{R}} \mathcal{K}$. 2. The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger. 3. The challenger replies with $\mathsf{Encrypt}(k, m_0)$. 4. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.	1. The challenger samples a key $k \xleftarrow{\mathrm{R}} \mathcal{K}$. 2. The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger. 3. The challenger replies with $\mathsf{Encrypt}(k, m_1)$. 4. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

We say the encryption scheme satisfies semantic security if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{SSAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathrm{negl}(\lambda).$$

Note that when the message space $\mathcal{M}$ contains *variable-length* messages, then each of the adversary's encryption queries (m_0, m_1) in the semantic security experiment must additionally satisfy $|m_0| = |m_1|$.

Security against chosen-plaintext attacks (CPA-security). We start by defining the CPA-security experiment:

Experiment $b = 0$:	**Experiment** $b = 1$:

• The challenger samples a key $k \xleftarrow{\mathrm{R}} \mathcal{K}$.
• The adversary can now make queries to the challenger:
  – **Encryption query:** The adversary sends $m_0, m_1 \in \mathcal{M}$ to the challenger. The challenger replies with $\mathsf{Encrypt}(k, m_0)$.
• The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.
	• The challenger samples a key $k \xleftarrow{\mathrm{R}} \mathcal{K}$.
	• The adversary can now make queries to the challenger:
	  – **Encryption query:** The adversary sends $m_0, m_1 \in \mathcal{M}$ to the challenger. The challenger replies with $\mathsf{Encrypt}(k, m_1)$.
	• The adversary $\mathcal{A}$ outputs a bit $b' \in \{0,1\}$.

We say the encryption scheme satisfies security against chosen-plaintext attacks (CPA-security) if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{CPAAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathsf{negl}(\lambda).$$

Note that when the message space $\mathcal{M}$ contains *variable-length* messages, then each of the adversary's encryption queries (m_0, m_1) in the CPA-security experiment must additionally satisfy $|m_0| = |m_1|$.

Security against chosen-ciphertext attacks (CCA-security). We start by defining the CCA-security experiment:

Experiment $b = 0$:

- The challenger samples a key $k \xleftarrow{R} \mathcal{K}$.
- The adversary can now make queries to the challenger:
  - **Encryption query:** The adversary sends $m_0, m_1 \in \mathcal{M}$ to the challenger. The challenger replies with $\mathsf{Encrypt}(k, m_0)$.
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \in \mathcal{C}$ to the challenger. The challenger replies with $\mathsf{Decrypt}(k, \mathsf{ct})$.
- The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

Experiment $b = 1$:

- The challenger samples a key $k \xleftarrow{R} \mathcal{K}$.
- The adversary can now make queries to the challenger:
  - **Encryption query:** The adversary sends $m_0, m_1 \in \mathcal{M}$ to the challenger. The challenger replies with $\mathsf{Encrypt}(k, m_1)$.
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \in \mathcal{C}$ to the challenger. The challenger replies with $\mathsf{Decrypt}(k, \mathsf{ct})$.
- The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

We say an adversary $\mathcal{A}$ is admissible for the CCA-security game if it does not issue a decryption query on a ciphertext ct it *previously* received from the challenger (in response to an encryption query). We say the encryption scheme satisfies security against chosen-ciphertext attacks (CCA-security) if for all efficient and admissible adversaries $\mathcal{A}$,

$$\mathsf{CCAAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathsf{negl}(\lambda).$$

Note that when the message space $\mathcal{M}$ contains *variable-length* messages, then each of the adversary's encryption queries (m_0, m_1) in the CCA-security experiment must additionally satisfy $|m_0| = |m_1|$.

Ciphertext integrity. We start by defining the ciphertext integrity experiment:

Ciphertext integrity experiment:

- The challenger samples a key $k \xleftarrow{R} \mathcal{K}$.
- The adversary can now make encryption queries to the challenger:
  - **Encryption query:** The adversary sends $m \in \mathcal{M}$ to the challenger. The challenger replies with $\mathsf{ct} \leftarrow \mathsf{Encrypt}(k, m)$.
- The adversary $\mathcal{A}$ outputs a ciphertext $\mathsf{ct}^* \in \mathcal{C}$.

Let $\mathsf{ct}_1, \ldots, \mathsf{ct}_Q \in \mathcal{C}$ be the ciphertexts that the challenger gives the adversary in the security game (when responding to encryption queries). We say an adversary $\mathcal{A}$ is admissible for the existential unforgeability game if $\mathsf{ct}^* \notin \{\mathsf{ct}_1, \ldots, \mathsf{ct}_Q\}$. We say that the encryption scheme satisfies ciphertext integrity if for all efficient and admissible adversaries $\mathcal{A}$,

$$\Pr[\mathsf{Decrypt}(k, \mathsf{ct}^*) \neq \bot] = \mathsf{negl}(\lambda).$$

Authenticated encryption. We say the encryption scheme is an authenticated encryption if it satisfies CPA-security *and* ciphertext integrity.

3 Message Authentication Codes

A message authentication code (MAC) is defined over a key space $\mathcal{K}$, a message space $\mathcal{M}$, and a tag space $\mathcal{T}$ (technically, each of these sets is a function of the security parameter λ) and consists of two efficient algorithms:

- $\mathsf{Sign}(k, m) \to t$: On input a key $k \in \mathcal{K}$ and a message $m \in \mathcal{M}$, the signing algorithm outputs a tag t.
- $\mathsf{Verify}(k, m, t) \to 0/1$: On input a key $k \in \mathcal{K}$, a message $m \in \mathcal{M}$, and a tag $t \in \mathcal{T}$, the verification algorithm outputs a bit $b \in \{0, 1\}$ (indicating whether the tag is valid or not).

Correctness. The MAC is correct if for all keys $k \in \mathcal{K}$ and all messages $m \in \mathcal{M}$,

$$\Pr[\mathsf{Verify}(k, m, \mathsf{Sign}(k, m)) = 1] = 1.$$

Existential unforgeability. We start by defining the existential unforgeability experiment:

Existential unforgeability experiment:
- The challenger samples a key $k \xleftarrow{R} \mathcal{K}$.
- The adversary can now make signing queries to the challenger:
  - **Signing query:** The adversary sends $m \in \mathcal{M}$ to the challenger. The challenger replies with $t \leftarrow \mathsf{Sign}(k, m)$.
- The adversary $\mathcal{A}$ outputs a message $m^* \in \mathcal{M}$ and tag $t^* \in \mathcal{T}$.

Let $m_1, \ldots, m_Q \in \mathcal{M}$ be the signing queries the adversary makes and let $t_1, \ldots, t_Q \in \mathcal{T}$ be the respective tags that the challenger responds with. We say an adversary $\mathcal{A}$ is admissible for the existential unforgeability game if $(m^*, t^*) \notin \{(m_1, t_1), \ldots, (m_Q, t_Q)\}$. We say the MAC satisfies existential unforgeability against chosen-message attacks if for all efficient and admissible adversaries $\mathcal{A}$,

$$\Pr[\mathsf{Verify}(k, m^*, t^*) = 1] = \mathrm{negl}(\lambda).$$

4 Block Cipher Modes of Operation

We now recall two common ways to use block ciphers to construct CPA-secure encryption schemes.

Counter mode. Let $F \colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a secure PRF. In the following, k is the PRF key and $m = (m_1, \ldots, m_n)$ are the blocks of the message (i.e., $m_i \in \{0,1\}^n$). In randomized counter-mode encryption, sample $\mathsf{IV} \xleftarrow{R} \{0,1\}^n$, and the ciphertext is $(\mathsf{IV}, c_1, \ldots, c_n)$. We view IV as an integer between 0 and $2^n - 1$, and perform arithmetic operations modulo 2^n.

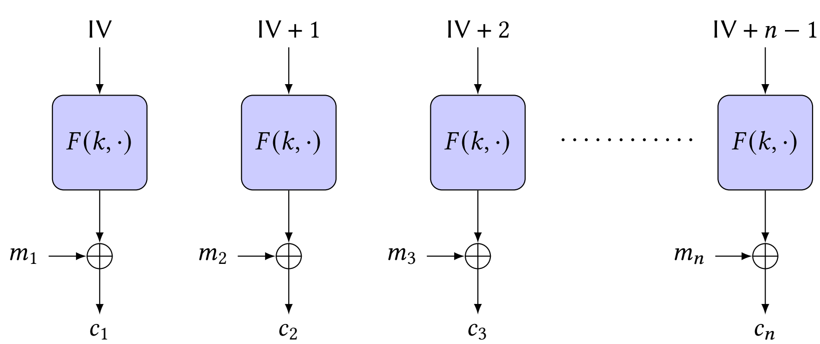

Figure 1: Counter-mode encryption

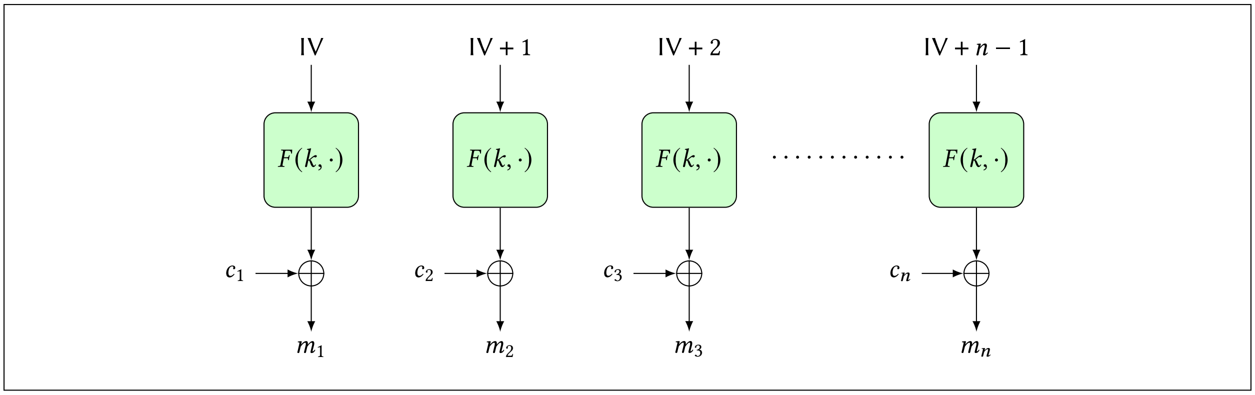

Figure 2: Counter-mode decryption

Cipherblock chaining (CBC). Let $F\colon \mathcal{K} \times \{0,1\}^n \to \{0,1\}^n$ be a block cipher (i.e., a secure PRP). In the following, k is the PRP key and $m = (m_1, \ldots, m_n)$ are the blocks of the message (i.e., $m_i \in \{0,1\}^n$). In CBC encryption, sample $\mathsf{IV} \xleftarrow{\text{R}} \{0,1\}^n$, and the ciphertext is $(\mathsf{IV}, c_1, \ldots, c_n)$.

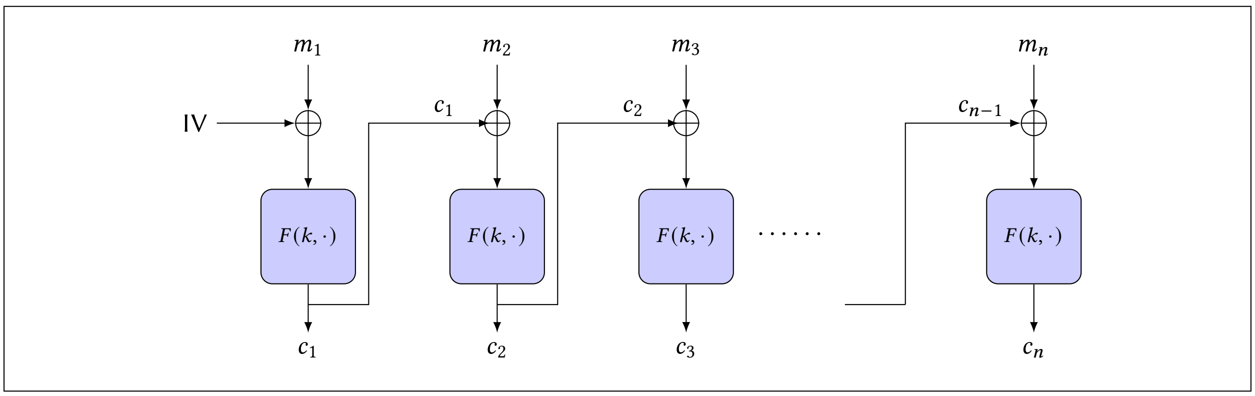

Figure 3: CBC encryption

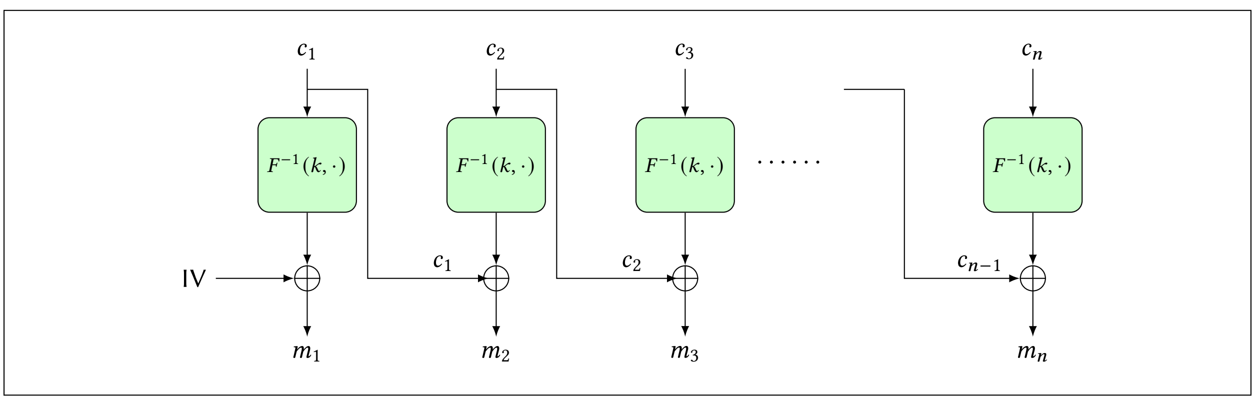

Figure 4: CBC decryption

5 Public-Key Encryption

A public-key encryption scheme is define with respect to a message space $\mathcal{M}$ and a ciphertext space $\mathcal{C}$ (technically, each of these sets can be a function of the security parameter λ) and consists of three algorithms:

- $\mathsf{Setup} \rightarrow (\mathsf{pk}, \mathsf{sk})$: The setup algorithm outputs a public key pk and a secret key sk. (Technically, this algorithm takes the security parameter λ as input).
- $\mathsf{Encrypt}(\mathsf{pk}, m) \rightarrow \mathsf{ct}$: On input the public key pk and a message $m \in \mathcal{M}$, the encryption algorithm outputs a ciphertext ct.
- $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct}) \rightarrow m$: On input a secret key sk and a ciphertext ct, the decryption algorithm either outputs a message $m \in \mathcal{M}$ or a special symbol $\perp$ (to indicate a decryption failure).

Correctness. A public-key encryption scheme is correct if for all $(\mathsf{pk}, \mathsf{sk})$ output by Setup and all messages $m \in \mathcal{M}$,

$$\Pr[\mathsf{Decrypt}(\mathsf{sk}, \mathsf{Encrypt}(\mathsf{pk}, m)) = m] = 1.$$

Semantic security. The semantic security experiment is defined analogously to the corresponding notion in the secret-key setting:

Experiment $b = 0$:

1. The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}$ and gives pk to $\mathcal{A}$.
2. The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger.
3. The challenger replies with $\mathsf{Encrypt}(\mathsf{pk}, m_0)$.
4. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

Experiment $b = 1$:

1. The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}$ and gives pk to $\mathcal{A}$.
2. The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger.
3. The challenger replies with $\mathsf{Encrypt}(\mathsf{pk}, m_1)$.
4. The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

We say the encryption scheme satisfies semantic security if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{SSAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathrm{negl}(\lambda).$$

CCA security. We start by defining the CCA-security experiment for public-key encryption. This is the analog of the corresponding secret-key notion.

Experiment $b = 0$:

- The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}$ and gives pk to $\mathcal{A}$.
- The adversary can now issue decryption queries to the challenger:
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \in \mathcal{C}$ to the challenger. The challenger replies with $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$.
- The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger.
- The challenger replies with $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_0)$.
- The adversary can make more decryption queries to the challenger, with the restriction that it is not allowed to query on ct^*.
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \neq \mathsf{ct}^*$ to the challenger. The challenger replies with $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$.
- The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

Experiment $b = 1$:

- The challenger samples $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{Setup}$ and gives pk to $\mathcal{A}$.
- The adversary can now issue decryption queries to the challenger:
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \in \mathcal{C}$ to the challenger. The challenger replies with $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$.
- The adversary $\mathcal{A}$ sends messages $m_0, m_1 \in \mathcal{M}$ to the challenger.
- The challenger replies with $\mathsf{ct}^* \leftarrow \mathsf{Encrypt}(\mathsf{pk}, m_1)$.
- The adversary can make more decryption queries to the challenger, with the restriction that it is not allowed to query on ct^*.
  - **Decryption query:** The adversary sends a ciphertext $\mathsf{ct} \neq \mathsf{ct}^*$ to the challenger. The challenger replies with $\mathsf{Decrypt}(\mathsf{sk}, \mathsf{ct})$.
- The adversary $\mathcal{A}$ outputs a bit $b' \in \{0, 1\}$.

We say the encryption scheme satisfies security against chosen-ciphertext attacks (CCA-security) if for all efficient adversaries $\mathcal{A}$,

$$\mathsf{CCAAdv}[\mathcal{A}] = |\Pr[b' = 1 \mid b = 0] - \Pr[b' = 1 \mid b = 1]| = \mathrm{negl}(\lambda).$$

6 Digital Signatures

A digital signature scheme is defined over a message space $\mathcal{M}$ and a signature space $\mathcal{S}$ (technically, each of these sets can be a function of the security parameter λ) and consists of three main algorithms:

- $\mathsf{Setup} \rightarrow (\mathsf{vk}, \mathsf{sk})$: The setup algorithm outputs a public verification key vk and a secret signing key sk. (Technically, this algorithm takes the security parameter λ as input).
- $\mathsf{Sign}(\mathsf{sk}, m) \rightarrow \sigma$: On input the signing key sk and a message $m \in \mathcal{M}$, the signing algorithm outputs a signature $\sigma \in \mathcal{S}$.

- $\mathsf{Verify}(\mathsf{vk}, m, \mathsf{ct}) \to \{0,1\}$: On input the verification key vk, a message $m \in \mathcal{M}$, and a signature $\sigma \in \mathcal{S}$, the verification algorithm outputs a bit $b \in \{0,1\}$ (indicating whether the signature is valid or not).

Correctness. The signature scheme is correct if for all (vk, sk) output by Setup and all messages $m \in \mathcal{M}$,

$$\Pr[\mathsf{Verify}(\mathsf{vk}, m, \mathsf{Sign}(\mathsf{sk}, m)) = 1] = 1.$$

Unforgeability. We start by defining the unforgeability experiment:

Existential unforgeability experiment:

- The challenger samples $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{Setup}$ and gives vk to the adversary.
- The adversary can now make signing queries to the challenger:
  - **Signing query:** The adversary sends $m \in \mathcal{M}$ to the challenger. The challenger replies with $\sigma \leftarrow \mathsf{Sign}(\mathsf{sk}, m)$.
- The adversary $\mathcal{A}$ outputs a message $m^* \in \mathcal{M}$ and signature $\sigma^* \in \mathcal{S}$.

We say an adversary $\mathcal{A}$ is admissible for the signature unforgeability game if the adversary does not make a signing query on the message m^*. We say the signature scheme satisfies unforgeability if for all efficient and admissible adversaries $\mathcal{A}$,

$$\Pr[\mathsf{Verify}(\mathsf{sk}, m^*, \sigma^*) = 1] = \mathrm{negl}(\lambda).$$