## 1. Security Scheme Definitions

Security schemes are normally presented as a game, and cryptographic systems are tested in these games to see if they are secure if an adversary cannot win a disporportionate amount of the time it.

### 1.1. **IND-CCA (Indistinguishability under Chosen Ciphertext Attack).** Phase I:

- Examiner produces $(PK, SK) \leftarrow Keygen(1^\lambda)$.
- Adversary is given $PK$.
- Adversary computes in time $poly(\lambda)$ with access to decryption oracle $Dec(SK, \cdot)$ and encryption oracle $Enc(PK, \cdot)$. Adversary outputs $m_0, m_1$ where $|m_0| = |m_1|$. The adversary can also store state information $s$ and obtain this information in the next phase.

Phase II:

- Examiner chooses $b \leftarrow \{0, 1\}$ and computes $y = Enc(SK, m_b)$.
- Adversary is given access to state information $s$ and allowed to compute in time $poly(\lambda)$. Then, he produces a guess $\hat{b}$.

If adversary's advantage, defined as $|P(\hat{b} = b) - \frac{1}{2}|$, is negligible then the encryption scheme is deemed secure.

Note: Encryption must be randomized, and random values cannot be easily observable for IND-CCA security.

### 1.2. **IND-CCA2 (Indistinguishability under Adaptive Chosen Ciphertext Attack).**

Adapativity is a stronger security claim than IND-CCA. Everything in IND-CCA2 is the same, except that in phase II, the adversary is given access to the decryption block $Dec(SK, \cdot)$ on all inputs except for $y$.

### 1.3. **IND-CPA (Indistinguishability under Chosen Plaintext Attack).**

Almost the same as IND-CCA, but it is a little weaker. Under IND-CPA, the adversary is given access only to the encryption block, and never to the decryption block. Thus, the adversary can compute any encryptions in $poly(\lambda)$, but cannot use $Dec(SK, \cdot)$ for either Phase I or II.

Again, this security scheme requires that encryption is randomized.

### 1.4. **Semantic Security.**

Semantic Security is equivalent to IND-CPA.

This means that if you want to show IND-CPA, then it is sufficient to show semantic security, and if you want to show semantic security, it is sufficient to show IND-CPA.

*Definition:* A cryptosystem is *semantically secure* if any probabilistic polynomial time algorithm that is given the cipher $c$ of message $m$ and given $|m|$, cannot determine any other information about the message with non-neglible probability. In other words, it must be infeasible for a computationally bounded adversary to obtain significant information about a plaintext from a ciphertext and the public encryption key.

Note that semantic security does not consider CCA where the attacker is able to request the decryption of chosen ciphertexts.

Examples of semantically secure algorithms:

- Goldwasser Micali
- El Gamal
- Paillier

## 2. Cryptographic Systems

### 2.1. **ElGamal Encryption.**

### 2.2. **RSA.**

RSA is a public key encryption system.

Key Generation: We define $Keygen(1^\lambda)$ as the following process:

- Pick two large primes $p, q$ and set $n = pq$.
- Pick some number $e \leftarrow Z^*_{\phi(n)}$ and compute $d = e^{-1} \pmod{\phi(n)}$. Note that picking $e$ is equivalent to picking some $e$ from $Z^*_n$ where $gcd(e, \phi(n)) = 1$.
- Set the keys as $PK = (n, e)$ and $SK = (d, p, q)$.

Encryption: We have some message $m \in Z_n$ and we define the encryption operation as $Enc(PK, m) = m^e \pmod{n}$.

Decryption: We have some ciphertext $c$ and define decryption as $Dec(SK, c) = c^d \pmod{n}$.

Proof of correctness: Note that by CRT, proving correctness for $(m^e)^d \pmod{n}$ is equivalent to proving correctness for $(m^e)^d \pmod{p}$ and $(m^e)^d \pmod{q}$. WLOG we will prove it correct for $p$. In this case, if $m = 0$

$\pmod p$, then $(m^e)^d \equiv 0 \pmod p$ so we have the relation $m = (m^e)^d \pmod p$. If $m \neq 0 \pmod p$, then we have $(m^e)^d = m^{ed} \pmod p$. But we know that $d = e^{-1} \pmod{\phi(n)}$ so that $ed = 1 \pmod{(p-1)(q-1)}$. This implies that $ed = 1 + t(q-1)(p-1) = 1 + u(p-1)$. Therefore, we have $m^{ed} = m^{1+u(p-1)} \pmod p$ and by FLT, we have $m^{ed} = m \pmod p$. Implies that $(m^e)^d = m \pmod p$, which completes the proof.

The basic assumptions underlying RSA is that the RSA problem is hard, i.e that given $c$ and $e$ the computation of $c = m^e \pmod n$ is difficult for $n$ large. The best known current way to solve this problem is by factoring $n$ and solving the congruence modulo smaller primes, via the Chinese Remainder Theorem.

2.2.1. *RSA-OAEP.* Regular RSA is not IND-CCA2 because it isn't randomized. In order to get it to IND-CCA2, people have developed a new method for encryption which builds off of RSA.