Today’s Agenda

- Secure Encryption Revisited
- CCA 1 vs. CCA 2 Security
- CCA Secure Cryptosystems

1 Secure Encryption Revisited

Recall the public-key encryption setting: Let’s say Alice wants to send a message to Bob. Alice does not want to send the message in clear but sends an encryption of the message. This setting also contains the evesdropper Eve whose goal is to intercept Alice’s ciphertext to Bob and infer something about the underlying message. We let Eve be more powerful now to model the additional things happening in the system and to achieve a stronger notion of security. For example, Eve may have some independent channel of communication with Bob and be able to get some ciphertexts decrypted by Bob. So the whole scenario can be explained in the following steps:

1. Bob runs a $\text{KeyGen}(1^k)$ first to get $(pk, sk)$ and publishes the $pk$.

2. Eve generates a ciphertext $c_i$, sends it to Bob and gets its decryption $m_i$ for $1 \leq i \leq \text{poly}(k)$ (Eve is allowed to make polynomially many decryption queries).

3. Eve picks a message $m$ that she wants Alice to encrypt for Bob.

4. Alice either encrypts $m$ and outputs $c = \text{Enc}(pk, m)$ or outputs $c$ produced by the Simulator($pk$).

5. Eve gets to query Bob again to get polynomially-many ciphertexts of her choice decrypted (like in step 2).

6. Eve now has to guess REAL or SIMULATED.

Now our adversary Eve has a lot more power than in semantic security. Eve can get ciphertexts decrypted and also choose the message that Alice encrypts for her challenge. It seems like if Eve can get ciphertexts decrypted from Bob then she can perhaps just get the challenge ciphertext decrypted and then she knows the message. Well, if we allow that our notion of security would be meaningless, so what we want to allow Eve as much power as possible without letting her to be able to directly decrypt the challenge ciphertext.

Note that steps 1,4 correspond to the semantic security notion that we have seen before which is: There exists a simulator $\text{Sim}$ s.t. $\forall$ PPT Eve, there exists a negligible $\gamma(k)$ such that:

$$\Pr[(pk, sk) \leftarrow \text{KeyGen}(1^k) ; (\text{info}, m) \leftarrow \text{Eve}(pk) ; c_0 \leftarrow \text{Enc}(pk, m) ; c_1 \leftarrow \text{Sim}(pk) ; b' \leftarrow \{0, 1\} ; b' \leftarrow \text{Eve}(\text{info}, c_{b'}) : b' = b] \leq \frac{1}{2} + \gamma(k)$$
The algorithm of Eve may be invoked more than once and in that case it is reasonable to allow her to store some state information from the previous invocation. info is that state information or tape contents that Eve is allowed to maintain and refer to later.

1.1 CCA 2 Security

We will revise the notion of semantic security to what is called security against adaptive chosen ciphertext attack (CCA 2) which we described in 6 steps before:

A cryptosystem is CCA 2 secure if there exists a simulator \( \text{Sim} \) s.t. \( \forall \) PPT Eve, there exists a negligible \( \gamma(k) \) such that:

\[
\Pr[(pk, sk) \leftarrow \text{KeyGen}(1^k) \ ; \ (info, m) \leftarrow \text{Eve}^{\text{Dec}(\cdot)}(pk) \ ; \ c_0 \leftarrow \text{Enc}(pk, m) \ ; \ c_1 \leftarrow \text{Sim}(pk) \ ; \ b' \leftarrow \{0, 1\} \ ; \ b' \leftarrow \text{Eve}^{\text{Dec}(\cdot)}(sk, c_{b'}) \ ; \ b' = b] \leq \frac{1}{2} + \gamma(k)
\]

Note that \( \text{Dec}'(sk, c^*, \cdot) \) is a crippled decryption oracle such that if \( c = c^* \), reject else output \( \text{Dec}(sk, c) \). So the CCA 2 security is a way of strengthening the initial notion of semantic security by giving Eve the extra power as we discussed before. Giving Eve access to a decryption oracle models some attacks known as the *lunch-time attacks*. Basically the setting is that Bob might be out for his lunch break and then Eve may have access to his system which has the secret key embedded in it. So Eve cannot get the secret key completely, but can use Bob’s system to decrypt ciphertexts of her choice.

1.2 CCA 1 Security

CCA 1 security is same as CCA 2 security except that Eve cannot query the decryption oracle after it receives its challenge ciphertext. It can query it only in the beginning. So in our setting will now skip the step 5 that we had before.

1. Bob runs a \( \text{KeyGen}(1^k) \) first to get \( (pk, sk) \) and publishes the \( pk \)
2. Eve generates a ciphertext \( c_i \), sends it to Bob and get its decryption \( m_i \) for \( 1 \leq i \leq \text{poly}(k) \)
3. Eve picks messages \( m \) that Alice encrypts to Bob
4. Alice either encrypts \( m \) and outputs \( c = \text{Enc}(pk, m) \) or outputs \( c \) by \( \text{Simulator}(pk) \)
5. Guess REAL or SIMULATED

More formally, it states that there exists a simulator \( \text{Sim} \) s.t. \( \forall \) PPT Eve, there exists a negligible \( \gamma(k) \) such that:

\[
\Pr[(pk, sk) \leftarrow \text{KeyGen}(1^k) \ ; \ (info, m) \leftarrow \text{Eve}^{\text{Dec}(\cdot)}(pk) \ ; \ c_0 \leftarrow \text{Enc}(pk, m) \ ; \ c_1 \leftarrow \text{Sim}(pk) \ ; \ b' \leftarrow \{0, 1\} \ ; \ b' \leftarrow \text{Eve}(info, c_b) \ ; \ b' = b] \leq \frac{1}{2} + \gamma(k)
\]

One would wonder why do we need to consider CCA 1 security at all since CCA 2 is obviously a stronger notion of security. But sometimes when we design cryptosystems, it is very difficult to satisfy CCA 2 security but we can achieve CCA 1 security. Also achieving CCA 1 is sometimes the illustrative first step which helps in getting to CCA 2 security. We will now look at cryptosystems that satisfy these stronger notions of security.

19: Adaptively Secure Encryption-2
2 CCA Secure Cryptosystems

2.1 RSA-Optimal Asymmetric Encryption Padding (OAEP)

RSA-OAEP is a padded version of RSA TDP which is CCA secure cryptosystem, introduced by Bellare-Rogaway'95. The details are as follows:

- **KeyGen**(1\(^k_1\), 1\(^k_2\)): Output as \( pk \) RSA modulus \( N \) of \( n + 1 \) bits, exponent \( e \), functions \( G, H \) and \( sk \) is the RSA secret key \( d \) (For message of length \( k_0 \), pick \( n = k_0 + k_1 + k_2 \). \( k_1, k_2 \) are small enough such that RSA with \( n - k_1 - k_2 \) bits is still secure. )
- **Enc()**: On input \( m \in \{0,1\}^{n-k_1-k_2} \), let \( x \) be an encoding of \( m \), output \( y = x^e \mod N \)
- **Dec()**: On input \( y \), compute \( x = y^d \mod N \) and output decoding of \( y \).

Let us look at the encoding and decoding that we will be using:

1. **Encoding**: On input message \( m \) and randomness \( r \), output \( x_1 \odot x_2 \) where

   \[
   x_1 = G(r) \oplus (m \circ 0^{k_2})
   \]

   where \( r \in \{0,1\}^{k_1} \) and \( G \) is an expanding function (like a PRG) and

   \[
   x_2 = H(m \circ 0^{k_2}) \oplus r
   \]

2. **Decoding**: On input \((y_1 \odot y_2)\), first retrieve \( r = y_2 \oplus H(y_1) \) and then \( m \circ 0^{k_2} = y_1 \oplus G(r) \)

**Proof Intuition**: This cryptosystem is CCA secure in the random-oracle model. Why does this proof work in the random-oracle model? The high-level idea is that the reduction (trying to break RSA, given an RSA instance) is trying to provide the decryption oracle for \( A \) without knowing the RSA secret key. Since \( G, H \) are modelled as random oracles, it means that for \( A \) to create a ciphertext, it has to make the corresponding queries to \( G, H \) and the reduction can decrypt messages for \( A \) using the queries that \( A \) made. Note that this padding construction is NOT CCA secure for a general TDP but is CCA secure specifically for RSA.

---

2.2 Naor-Yung Cryptosystem

- **KeyGen**: $(pk_1, sk_1)$ and $(pk_2, sk_2)$ for a semantically secure cryptosystem and $pp$ which are parameters for a non-interactive zero-knowledge proof (NIZK) proof.

- **Enc**: Produce $c_1 \leftarrow \text{Enc}(pk_1, m)$ and $c_2 \leftarrow \text{Enc}(pk_2, m)$ and a proof $\pi$ which is a NIZK to prove that $c_1, c_2$ are encryption of the same message $m$. Think of NIZK proof as a blackbox that lets you prove a statement without revealing underlying details.

  Output $(c_1, c_2, \pi)$

- **Dec**: Verify that $\pi$ is correct. If yes, decrypt $c_1$

**Proof Intuition**: At a very high-level, the NIZK proof system allows the reduction to decrypt ciphertexts for $A$ without knowing the secret key since the reduction can set the parameters for NIZK. This gives us CCA 1 security. To achieve CCA 2 security we need stronger guarantees such as non-malleability on the NIZK proof system. Hence it is always a challenge to go from CCA 1 to CCA 2.