CS 600.442 – Modern Cryptography Nov 7, 2016

Lecture 1: Chosen-Ciphertext Security (I)

Instructor: Abhishek Jain *Scribe: Ke Wu*

1 Recall

Public-Key Encryption

- Syntax
  - $\mathsf{Gen}(1^n) \to (pk, sk)$
  - $\mathsf{Enc}(1^n) \to (pk, sk)$
  - $\mathsf{Dec}(1^n) \to (pk, sk)$
- **Correctness**: For every m, $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$, where $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$.
- **IND-CPA Security**: For all n.u. PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(n)$ s.t.

$$\begin{bmatrix} (pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^n) & \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n, pk), & : \mathcal{A}(pk, \mathsf{Enc}(m_b)) = b \\ b \xleftarrow{\$} \{0,1\} & \end{bmatrix} \leq \frac{1}{2} + \mu(n)$$

Notice: for public-key encryption scheme, IND-CPA security for one-message implies IND-CPA security for multiple messages.

2 Security against Chosen-Ciphertext Attack (CCA)

2.1 Definition

Motivation: IND-CPA is not secure enough if an adversary is able to find an oracle that decrypts ciphertexts, which could be real-world possible attack. Hence we need to augment IND-CPA security to allow the adversary to make decryption queries of its choices. We then get two kinds of CCA security definitions.

Definition 1 *(IND-CCA-1 Security) A public key encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is IND-CCA-1 secure if for all n.u. PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\mu(n)$ *s.t. for all auxiliary inputs* $z \in \{0,1\}^*$*:*

$$|Pr[\boldsymbol{Expt}_{\mathcal{A}}^{CCA1}(1, z) = 1] - Pr[\boldsymbol{Expt}_{\mathcal{A}}^{CCA1}(0, z) = 1]| \leq \mu(n)$$

where $\boldsymbol{Expt}_{\mathcal{A}}^{CCA1}(b, z)$ *is defined as:*

$\boldsymbol{Expt}_{\mathcal{A}}^{CCA1}(0, z)$

- $st = z$
- $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$

- *Decryption query phase (repeated polynomial times)*
  - $c \leftarrow \mathcal{A}(pk, st)$
  - $m \leftarrow \mathsf{Dec}(sk, c)$
  - $st = (st, m)$
- $(m_0, m_1) \leftarrow \mathcal{A}(pk, st)$
- $c^* \leftarrow \mathsf{Enc}(pk, m_b)$
- *Output* $b' \leftarrow \mathcal{A}(pk, c^*, st)$

Definition 2 *(IND-CCA-2 Security) A public key encryption scheme* $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ *is IND-CCA-2 secure if for all n.u. PPT adversaries* $\mathcal{A}$*, there exists a negligible function* $\nu(n)$ *s.t. for all auxiliary inputs* $z \in \{0,1\}^*$*:*

$$|Pr[\boldsymbol{Expt}_{\mathcal{A}}^{CCA2}(1, z) = 1] - Pr[\boldsymbol{Expt}_{\mathcal{A}}^{CCA2}(0, z) = 1]| \leq \nu(n)$$

where $\boldsymbol{Expt}_{\mathcal{A}}^{CCA2}(b, z)$ *is defined as:*
$\boldsymbol{Expt}_{\mathcal{A}}^{CCA2}(0, z)$

- $st = z$
- $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$
- *Decryption query phase 1 (repeated polynomial times)*
  - $c \leftarrow \mathcal{A}(pk, st)$
  - $m \leftarrow \mathsf{Dec}(sk, c)$
  - $st = (st, m)$
- $(m_0, m_1) \leftarrow \mathcal{A}(pk, st)$
- $c^* \leftarrow \mathsf{Enc}(pk, m_b)$
- *Decryption query phase 2 (repeated polynomial times)*
  - $c \leftarrow \mathcal{A}(pk, c^*, st)$
  - *If* $c = c^*$*, output reject*
  - $m \leftarrow \mathsf{Dec}(sk, c)$
  - $st = (st, m)$
- *Output* $b' \leftarrow \mathcal{A}(pk, c^*, st)$

Note: CCA-2 is stronger than CCA-1 as it can make queries not only before challenge (as CCA-1) and also after challenge. And to prevent trivial attacks, decryption queries c should be different from the challenge ciphertext c^*.

2.2 Construction

Main Challenge When building IND-CCA-1 secure PKE starting from IND-CPA secure PKE, we should not use the secret key in the secure experiment. However, we need the secret key to answer the decryption queries of the adversary. Thus the main idea is to use *two copies of the encryption scheme.*

Main Idea We could encrypt a message twice, using each of the two copies of the encryption scheme. To answer a decryption query (c_1, c_2), we only need to decrypt one of the two ciphertext. That means, we only need to know one of the secret key to answer the decryption queries. We can then use the IND-CPA security of another encryption scheme whose secret key is not used to answer decryption queries. Then switch the secret key and use IND-CPA security of the other one.

But there's a problem. What if the adversary sends (c_1, c_2) such that c_1 and c_2 are ciphertext of different messages? To solve this, we modify the scheme such that the encryption of messages m contains a NIZK proof that proves that c_1 and c_2 encrypts same message m.

Theorem 1 *(Naor-Yung) Assuming that NIZKs in the CRS model and IND-CPA secure public-key encryption, the encryption scheme* $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ *below is IND-CCA-1 secure public-key encryption.*

Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA encryption scheme.

Let $(\mathsf{K}, \mathsf{P}, \mathsf{V})$ be an adaptive NIZK with Simulator $\mathcal{S} = (\mathcal{S}_0, \mathcal{S}_1)$.

$\mathsf{Gen}'(1^n)$:

- Compute (pk_1, sk_1) and (pk_2, sk_2) using $\mathsf{Gen}(1^n)$
- Compute $\sigma \leftarrow \mathsf{K}(1^n)$
- Output $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$

$\mathsf{Enc}'(pk', m)$:

- Compute $c_i \leftarrow \mathsf{Enc}(pk_i, m; r_i)$ for $i \in [2]$
- Compute $\pi \leftarrow \mathsf{P}(\sigma, x, w)$ where $x = (pk_1, pk_2, c_1, c_2)$, $w = (m, r_1, r_2)$ and $R(x, w) = 1$ iff c_1 and c_2 encrypts the same message m.
- Output $C =)c_1, c_2, \pi$

$\mathsf{Dec}'(sk', c')$: If $\mathsf{V}(\sigma, \pi) = 0$, output $\perp$. Else, output $\mathsf{Dec}(sk_1, c_1)$.

Proof. We use Hybrid Lemma to prove the theorem. We construct hybrids as follows:

Hybrids H_0: $= \mathbf{Expt}_{\mathcal{A}}^{CCA1}(0, z)$

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $\sigma \leftarrow \mathsf{K}(1^n)$
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_0; r_2^*)$
- $\pi^* \leftarrow \mathsf{P}(\sigma, x^* = (c_1^*, c_2^*), w^* = (m_0, r_1, r_2))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_1:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_0; r_2^*)$
- $\pi^* \leftarrow \mathcal{S}_1(\sigma, \tau, x^* = (c_1^*, c_2^*))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_2:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_1; r_2^*)$
- $\pi^* \leftarrow \mathcal{S}_1(\sigma, \tau, x^* = (c_1^*, c_2^*))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_3:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma)$, $sk' = sk_2$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_2, c_2)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_1; r_2^*)$
- $\pi^* \leftarrow \mathcal{S}_1(\sigma, \tau, x^* = (c_1^*, c_2^*))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_4:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma)$, $sk' = sk_2$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_2, c_2)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_1; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_1; r_2^*)$
- $\pi^* \leftarrow \mathcal{S}_1(\sigma, \tau, x^* = (c_1^*, c_2^*))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_5:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma)$, $sk' = sk_1$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_1; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_1; r_2^*)$
- $\pi^* \leftarrow \mathcal{S}_1(\sigma, \tau, x^* = (c_1^*, c_2^*))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

Hybrids H_6: $= \mathbf{Expt}_{\mathcal{A}}^{CCA1}(0, z)$

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $\sigma \leftarrow \mathsf{K}(1^n)$
- $pk' = (pk_1, pk_2, \sigma)$, $sk' = sk_1$
- On receiving a decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_1; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_1; r_2^*)$
- $\pi^* \leftarrow \mathsf{P}(\sigma, x^* = (c_1^*, c_2^*), w^* = (m_1, r_1, r_2))$
- Output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi^*))$

In short, the changes in hybrids are:

- $H_0 : \mathrm{Expt}_{\mathcal{A}}^{CCA1}(1, z)$.
- H_1: Simulate the CRS in public-key and simulate the proof in challenge ciphertext.
- H_2: Modify c_2^* in challenge ciphertext to be an encryption of m_1.
- H_3: Change the decryption key to sk_2.

- H_4: Modify c_2^* in challenge ciphertext to be an encryption of m_1.
- H_5: Change the decryption key back to sk_1.
- H_6 : $\text{Expt}_{\mathcal{A}}^{CCA1}(0, z)$:

Now we argue the indistinguishability of these hybrids.

$H_0 \approx H_1$: This follows from the zero knowledge property of NIZK. Suppose that $\mathcal{A}'$ can distinguish H_0 and H_1 with at least a noticeable probability $\frac{1}{p(n)}$ where $p(n)$ is a polynomial function. Then we can build a distinguisher $\mathcal{D}$ against the zero-knowledge property of NIZK: on input (σ, π), $\mathcal{D}$ runs the experiment with (σ) and π^* replaced by input. $\mathcal{D}$ runs as follows:

$\mathcal{D}(\sigma, \pi)$

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- Receive decryption queries from $\mathcal{A}$: $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$
- $(m_0, m_1) \leftarrow \mathcal{A}(z, pk')$
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- $c_2^* \leftarrow \mathsf{Enc}(pk_2, m_0; r_2^*)$
- Pass the output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi))$ to $\mathcal{A}'$. If $\mathcal{A}'$ says the output is sampled from H_0, output "real proof". Else if $\mathcal{A}'$ says the output is from H_1, output "simulated proof".

Notice that $Pr[\mathcal{D}$ outputs real proof$] = Pr[\mathcal{A}'$ output $H_0]$ and that $Pr[\mathcal{D}$ outputs simulated proof$] = Pr[\mathcal{A}'$ output $H_1]$. It follows that $\mathcal{D}$ can distinguishes the real and simulated proof with noticeable probability $\frac{1}{p(n)}$. This contradicts the zero-knowledge property of NIZK.

Actually, notice that even though $x \notin L$, simulator $(\mathcal{S}_0, \mathcal{S}_1)$ can still come up with a simulated proof. Otherwise, simulator can actually decide L in polynomial time!

$H_1 \approx H_2$: This follows from the IND-CPA security of $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ with sk_2. Suppose that $\mathcal{A}'$ can distinguish H_1 and H_2 with at least a noticeable probability $\frac{1}{p(n)}$ where $p(n)$ is a polynomial function. Then we can build an adversary $\mathcal{B}$ against the IND-CPA security. $\mathcal{B}$ runs as follows:

- $(pk_1, sk_1) \leftarrow \mathsf{Gen}(1^n)$.
- Let pk_2 be the public key $\mathcal{B}$ got from challenger.
- $(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n)$
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- Receive decryption queries from $\mathcal{A}$: $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return $\mathsf{Dec}(sk' = sk_1, c_1)$

- Run $\mathcal{A}$ to get message query (m_0, m_1) and pass (m_0, m_1) to challenger.
- $c_1^* \leftarrow \mathsf{Enc}(pk_1, m_0; r_1^*)$
- Let c_2^* be the cipher text $\mathcal{B}$ got from challenger.
- Pass the output $\mathcal{A}(z, pk', C = (c_1^*, c_2^*, \pi))$ to $\mathcal{A}'$. If $\mathcal{A}'$ says the output is sampled from H_1, output $b = 0$. Else if $\mathcal{A}'$ says the output is from H_2, output $b = 1$.

When challenger choose to encrypt m_0, the output passed to $\mathcal{A}'$ is identical to that in H_1; if it is m_1 that is encrypted, the output is identical to H_2. Note that $\mathcal{B}$ can handle the decryption queries from $\mathcal{A}$ because $\mathcal{B}$ generates (pk_1, sk_1) itself. Also, it doesn't matter that $\mathcal{B}$ has no access to the randomness used to encrypt m_0, as the simulator $\mathcal{S}_1$ doesn't need r_2 to simulate the proof (unlike the real prover). Thus

$$Pr[\mathcal{B} \text{ distinguishes encryption of } m_0 \text{ and } m_1] = Pr[\mathcal{A} \text{ distinguishes } H_1 \text{ and } H_2] \geq \frac{1}{p(n)}$$

This contradicts the IND-CPA security of the PKE.

$H_2 \approx H_3$: This follows from the soundness of NIZK. Notice that the adversary can only makes successful decryption queries (c_1, c_2) if c_1 and c_2 encrypts the same message. Suppose $\mathcal{A}'$ can distinguishes H_2 and H_3 with noticeable probability. Then $Pr[\mathcal{A} \text{ distinguishes } H_2 \text{ and } H_3] = Pr[E]$ where E denotes the event that c_1 and c_2 encrypts different messages but $V(\sigma, (c_1, c_2), \pi) = 1$. Let $L = \{(c_1, c_2) | c_1 \text{ and } c_2 \text{ encrypts same message}\}$. According to the soundness property of NIZK, there exists $\nu(n)$ such that

$$Pr[\sigma \leftarrow \mathsf{K}(1^n), \exists (x, \pi) s.t. x \notin L \wedge V(\sigma, x, \pi) = 1] \leq \nu(n)$$

Now we argue that

$$Pr[(\sigma, \tau) \leftarrow \mathcal{S}_0(1^n), \exists (x, \pi) s.t. x \notin L \wedge V(\sigma, x, \pi) = 1] \leq \nu(n)$$

If not, suppose the probability above is at least $\frac{1}{p(n)}$ where $p(\cdot)$ is a polynomial function. We can then build distinguisher $\mathcal{B}$ that can tell the random string and the simulated string apart. On input σ, $\mathcal{B}$ runs as follows:

- $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$.
- $pk' = (pk_1, pk_2, \sigma), sk' = sk_1$
- On each decryption query $c = (c_1, c_2, \pi)$ from $\mathcal{A}(z, pk')$, if $\mathsf{Dec}(sk' = sk_1, c_1) \neq \mathsf{Dec}(sk' = sk_2, c_2)$ but $\mathsf{V}(\sigma, x = (c_1, c_2), \pi) = 1$, return 1. Otherwise, repeat dealing with next query.

It's obvious that if σ is real random string, then the probability $\mathcal{B}$ outputs 1 is negligible. If σ is generated by simulator, then the probability $\mathcal{B}$ outputs 1 is at least $1 - (1 - \frac{1}{p(n)})^N$ where N is the number of queries made by $\mathcal{A}$. Hence $\mathcal{B}$ could distinguish the real random string with the one simulated by the simulator, which is a contradiction that NIZK is zero-knowledge. Hence $Pr[E]$ is negligible, which implies that $H_2 \approx H_3$.

$H_3 \approx H_4$: follows in the same manner as $H_1 \approx H_2$.

$H_4 \approx H_5$: follows in the same manner as $H_2 \approx H_3$.

$H_3 \approx H_4$: follows in the same manner as $H_0 \approx H_1$. Notice that now c_1^* and c_2^* are encrypting same message, hence P can come up with a valid proof.

Above all, $H_0 \approx H_6$, which implies the IND-CCA-1 security of the scheme $\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}'$.