Chosen-Ciphertext Security (I)

CS 601.442/642 Modern Cryptography

Fall 2019

Recall: Public-Key Encryption

- **Syntax:**
  - $\mathsf{Gen}(1^n) \rightarrow (pk, sk)$
  - $\mathsf{Enc}(pk, m) \rightarrow c$
  - $\mathsf{Dec}(sk, c) \rightarrow m'$ or $\perp$

  All algorithms are polynomial time

- **Correctness:** For every m, $\mathsf{Dec}(sk, \mathsf{Enc}(pk, m)) = m$, where $(pk, sk) \leftarrow \mathsf{Gen}(1^n)$

Recall: IND-CPA Security

Definition (IND-CPA Security)

A public-key encryption scheme $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ is indistinguishably secure under chosen plaintext attack (IND-CPA) if for all n.u. PPT adversaries $\mathcal{A}$, there exists a negligible function $\mu(\cdot)$ s.t.:

$$\Pr\left[\begin{array}{r} (pk, sk) \xleftarrow{\$} \mathsf{Gen}(1^n), \\ (m_0, m_1) \leftarrow \mathcal{A}(1^n, pk), \\ b \xleftarrow{\$} \{0,1\} \end{array} : \mathcal{A}\left(pk, \mathsf{Enc}\left(m_b\right)\right) = b \right] \leqslant \frac{1}{2} + \mu(n)$$

1. IND-CPA for one-message implies IND-CPA for multiple messages

Need for Stronger Security

Motivation:

- An adversary may be able to find an oracle that decrypts ciphertexts. In this case, IND-CPA security may break down!
- Real-world attacks possible, e.g., chosen ciphertext attacks on Apple imessage [Garman-Green-Kaptchuk-Miers-Rushanan'16]

Security against Chosen-Ciphertext Attacks (CCA)

- Augment the IND-CPA security experiment
- Adversary can make decryption queries over ciphertexts of its choice
- Two security definitions:
  - CCA-1: Decryption queries only before challenge ciphertext query
  - CCA-2: Decryption queries before and after challenge ciphertext query

Note: To prevent trivial attacks, decryption queries c made by the adversary should be different from the challenge ciphertext c^*!

CCA-1 Security

$\mathbf{Expt}^{\mathsf{CCA1}}_{\mathcal{A}}(b, z)$:

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

CCA-1 Security (contd.)

Definition (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 $\mu(\cdot)$ s.t. for all auxiliary inputs $z \in \{0,1\}^*$:

$$\left|\Pr\left[\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA1}}(1,z)=1\right] - \Pr\left[\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA1}}(0,z)=1\right]\right| \leqslant \mu(n)$$

CCA-2 Security

$\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA2}}(b, z)$:

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

CCA-2 Security (contd.)

Definition (IND-CCA-2 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 $\nu(\cdot)$ s.t. for all auxiliary inputs $z \in \{0,1\}^*$:

$$\left|\Pr\left[\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA2}}(1,z)=1\right] - \Pr\left[\mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA2}}(0,z)=1\right]\right| \leqslant \nu(n)$$

How to Construct CCA-1 Secure PKE?

Main Challenge: How should the reduction answer decryption queries?

- Suppose we want to build IND-CCA-1 secure PKE starting from IND-CPA secure PKE
- In order to rely on IND-CPA security of underlying PKE, we should not use secret key in the reduction
- However, in order to answer decryption queries of the adversary, we need the secret key!
- How to resolve this seeming paradox?

How to Construct CCA-1 Secure PKE?

Main Idea: Use *two copies* of the encryption scheme

- Encrypt a message twice, using each of the two copies of the encryption scheme
- To answer a decryption query (c_1, c_2), the reduction only needs to decrypt one of the two ciphertexts. Therefore, the reduction only need to know one of the secret keys
- Use IND-CPA security of the second encryption scheme when the reduction uses sk_1 in the experiment (but not sk_2)
- Then, switch the secret key to sk_2 and use IND-CPA security of the first encryption scheme
- **Problem:** What if adversary sends decryption queries (c_1, c_2) such that c_1 and c_2 decrypt different messages?
- **Solution:** Modify the scheme so that encryption of message m also contains a NIZK proof that proves that c_1 and c_2 encrypt the same message m

CCA-1 Secure Public-Key Encryption

Theorem (Naor-Yung)

Assuming NIZKs and IND-CPA secure public-key encryption, there exists IND-CCA-1 secure public-key encryption

- *Standard model*: If we use NIZKs in the common random string (CRS) model, we can obtain an IND-CCA-1 encryption scheme in the standard model. The CRS of the NIZK is generated by the key generation algorithm of the encryption scheme.
- *Random Oracle model*: If we use NIZKs in the random oracle (RO) model, the resulting encryption scheme is also in the RO model.

For simplicity of notation, we will use NIZKs in RO model, but the same transformation also works if we use NIZKs in CRS model.

Construction

Let $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ be an IND-CPA encryption scheme.
Let (P, V) be a NIZK for NP with Simulator $\mathcal{S}$.

Construction of $(\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$:

$\mathsf{Gen}'(1^n)$: Execute the following steps:

- Compute $(pk_i, sk_i) \leftarrow \mathsf{Gen}(1^n)$ for $i \in [2]$
- Output $pk' = (pk_1, pk_2)$, $sk' = sk_1$

$\mathsf{Enc}'(pk', m)$: Execute the following steps:

- Compute $c_i \leftarrow \mathsf{Enc}(pk_i, m; r_i)$ for $i \in [2]$
- Compute $\pi \leftarrow \mathsf{P}(x, w)$ 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 encrypt the same message m
- Output $C = (c_1, c_2, \pi)$

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

Hybrids

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

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

Hybrids (contd.)

Hybrid H_1: Simulate the proof in challenge ciphertext

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

$H_0 \approx_c H_1$**:** Follows from ZK property of NIZK

Hybrids (contd.)

Hybrid H_2: Modify c_2^* in challenge ciphertext to be an encryption of m_1

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

$H_1 \approx_c H_2$**:** Follows from IND-CPA security of the PKE

Hybrids (contd.)

Hybrid H_3: Change decryption key to sk_2

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

$H_2 \approx H_3$**:** From soundness of NIZK, it follows that in every decryption query (c_1, c_2, π) made by $\mathcal{A}$, c_1 and c_2 encrypt the *same* message. Therefore, decrypting c_1 or c_2 yields the same answer

Hybrids (contd.)

Hybrid H_4: Modify c_1^* in challenge ciphertext to be an encryption of m_1

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

$H_3 \approx_c H_4$**:** Follows from IND-CPA security of the PKE

Hybrids (contd.)

Hybrid H_5: Change decryption key back to sk_1

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

$H_4 \approx H_5$**:** Follows in the same manner as $H_2 \approx H_3$

Hybrids (contd.)

Hybrid $H_6 := \mathbf{Expt}_{\mathcal{A}}^{\mathsf{CCA1}}(1, z)$**:** Honestly compute the proof in challenge ciphertext

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

$H_5 \approx_c H_6$**:** Follows from the ZK property of NIZK