Identity-based Encryption with Post-Challenge Auxiliary Inputs for Secure Cloud Applications and Sensor Networks

Tsz Hon Yuen - *Huawei, Singapore*
Ye Zhang - *Pennsylvania State University, USA*
Siu Ming Yiu - *University of Hong Kong, Hong Kong*
Joseph K. Liu - *Institute for Infocomm Research, Singapore*

Table of Content

Introduction - IBE

- Identity-based encryption (IBE)
  - Use identity (e.g. name, email etc.) to encrypt
  - Private key issued by a trusted party called Private Key Generator (PKG)
  - No certificate required
- IBE can be used to protect data confidentiality in cloud computing era; or wireless sensor network
- More convenience

Introduction – Practical Threats of Using IBE

- Side Channel Attacks to the Decryptor
  - Real world attackers can obtain partial information about the secret key of the decryptor
  - Side-channel attacks explore the physical weakness of the implementation of cryptosystems
  - Some bits of the secret key can be leaked by observing the running time of the decryption process, or the power consumption used

Introduction – Practical Threats of Using IBE

- Weak Randomness Used by the Encryptor
  - The randomness used in the encryption process may be leaked by poor implementation of pseudorandom number generator (PRNG)
  - In big data applications, data are usually generated by some devices with limited computational power
  - It is possible that the data are encrypted using such weak randomness from java runtime libraries
  - wireless sensors as they are usually exposed in the open air but contain only very limited computation power
  - Attackers may easily guess the randomness they are using for generating the ciphertext

Motivation for Post-Challenge Auxiliary Inputs

- We need to provide leakage-resilient protection for users of the cloud applications and wireless sensor network
- It includes the encryptor and the decryptor
- Protecting the Decryptor: Leakage-Resilient Cryptography

Leakage-Resilient Cryptography

- In modern cryptography, we use a security model to capture the abilities of a potential attacker (the adversary)
- For example, in the chosen-ciphertext attack (CCA) model the adversary is allowed to ask for the decryption of arbitrary ciphertexts, except for the one that he intends to attack
- But if the adversary has some extra abilities, the security of the scheme is no longer guaranteed
- In most traditional security models, it is assumed that the adversary does not have the ability to obtain any information (even one single bit)

Leakage-Resilient Cryptography

- However, due to the advancement of a large class of side-channel attacks, obtaining partial information of the secret key becomes easier
- the assumption for absolute secrecy of the secret key may not hold
- leakage-resilient cryptography to formalize these attacks in the security model
- models various side-channel attacks by allowing the adversary to specify a function f and to obtain the output of f applied to the secret key sk (auxiliary input)

Restriction of the Auxiliary Input Model

- CCA security model for PKE and IBE, the adversary A is allowed to ask for the decryption of arbitrary ciphertexts before and after receiving the challenge ciphertext C*
- But for most leakage-resilient PKE or IBE, the adversary A can only specify and query the leakage function f(sk) before getting C*
  - Reason: If we allow A to specify the leakage function after getting C*, he can easily embed the decryption of C* as the leakage function, which will lead to a trivial break to the security game
- Cannot exactly reflect the real situation!
- Need a model with minimal restriction needed to allow post-challenge leakage query after getting the challenge ciphertext, while avoiding the above trivial attack

Protecting the Encryptor

- Leakage-Resilient from the Encryptor‘s Randomness
- If the adversary A can obtain the entire r (randomness), it can encrypt the two challenge messages m0 and m1 by itself using r and compare if they are equal to the challenge ciphertext
- It wins the game easily!
- Consider the following example:

  – $\mathtt{Enc}'$: On input a message M and a public key pk, pick a random one-time pad P for M and calculate $C_1 = \mathtt{Enc}(\mathsf{pk}, P), C_2 = P \oplus M$, where $\oplus$ is the bit-wise XOR. Return the ciphertext $C = (C_1, C_2)$.
  – $\mathtt{Dec}'$: On input a secret key sk and a ciphertext $C = (C_1, C_2)$, calculate $P' = \mathtt{Dec}(\mathsf{sk}, C_1)$ and output $M = C_2 \oplus P'$.

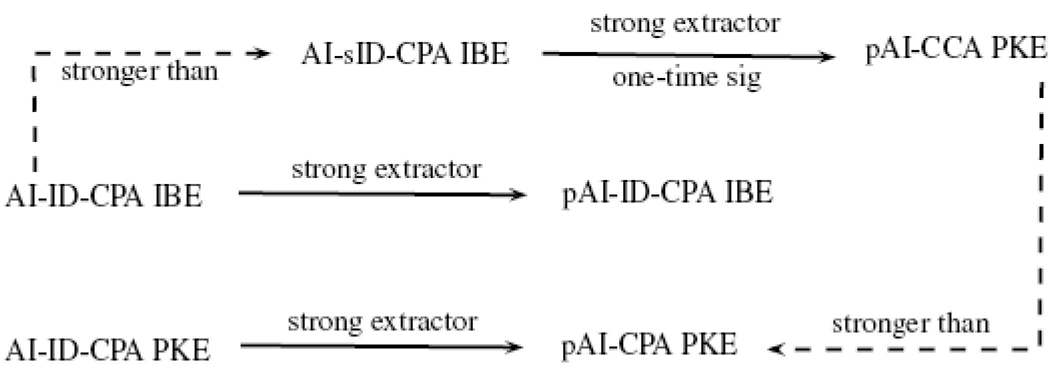
- The randomness used in Enc’ by the encryptor is P and the randomness in Enc
- Leaking the n-th bit of P leads to the leakage of the n-th bit in M

Contribution

- We propose the post-challenge auxiliary input model for public key and identity-based encryption
  - it allows the leakage after seeing the challenge ciphertext
  - it considers the leakage of two different parties: the secret key owner and the encryptor
- To the best of the authors' knowledge, no existing leakage-resilient PKE or IBE schemes consider the leakage of secret key and randomness at the same time
- We propose a generic construction of CPA-secure PKE in our new post-challenge auxiliary input model
- It is a generic transformation from the CPA-secure PKE in the auxiliary input model (AI-CPA PKE) and a new primitive called the strong extractor with hard-to-invert auxiliary inputs

Contribution

- Similar transformation can also be applied to identity-based encryption (IBE). Therefore we are able to construct pAI-ID-CPA IBE from AI-ID-CPA IBE
- We extend the generic transformation for CPA-secure IBE to CCA-secure PKE (by Canetti et al.) into the leakage-resilient setting
- Our contributions on encryption can be summarized in the following figure:



Security Model

- The basic setting of our new security model is similar to the classic IND-CCA model and the auxiliary input model for public key encryption
- Our improvement is to require the adversary *A* to submit a set of possible leakages F_0 that may be asked later in the security game
- *A* is only allowed to ask for at most q queries $f_1', \dots, f_q' \in F_0$ to the post-challenge leakage oracle and obtains $f_1'(r'), \dots, f_q'(r')$, where r' is the encryption randomness of the challenge ciphertext
- But A cannot recover r' with probability better than ϵ_r
- The security against post-challenge auxiliary inputs and adaptive chosen-ciphertext attacks is defined as the following game pAI-CCA

Security Model

1. The adversary $\mathcal{A}$ submits a set of leakage functions $\mathcal{F}_0$ to the challenger $\mathcal{C}$ with $m := |\mathcal{F}_0|$ is polynomial in λ.
2. $\mathcal{C}$ runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathtt{Gen}(1^\lambda)$ and outputs pk to $\mathcal{A}$.
3. $\mathcal{A}$ may adaptively query the (pre-challenge) leakage oracle:
   – $\mathcal{LO}_s(f_i)$ with f_i. $\mathcal{LO}_s(f_i)$ returns $f_i(\mathsf{sk}, \mathsf{pk})$ to $\mathcal{A}$.
4. $\mathcal{A}$ submits two messages $m_0, m_1 \in \mathcal{M}$ of the same length to $\mathcal{C}$. $\mathcal{C}$ samples $b \leftarrow \{0,1\}$ and the randomness of encryption $r' \leftarrow \{0,1\}^*$. It returns $C^* \leftarrow \mathtt{Enc}(\mathsf{pk}, m_b; r')$ to $\mathcal{A}$.
5. $\mathcal{A}$ may adaptively query the (post-challenge) leakage oracle and the decryption oracle:
   – $\mathcal{LO}_r(f_i')$ with $f_i' \in \mathcal{F}_0$. It returns $f_i'(r')$ to $\mathcal{A}$.
   – $\mathcal{DEC}(C)$ with $C \neq C^*$. It returns $\mathtt{Dec}(\mathsf{sk}, C)$ to $\mathcal{A}$.
6. $\mathcal{A}$ outputs its guess $b' \in \{0,1\}$. The advantage of $\mathcal{A}$ is $Adv_{\mathcal{A}}^{\mathrm{pAI-CCA}}(\Pi) = |\Pr[b = b'] - \frac{1}{2}|$.

Scheme Description

- Strong Extractor with Hard-to-invert Auxiliary Inputs

Definition : $((\epsilon, \delta)$-Strong extractor with auxiliary inputs). *Let* Ext : $\{0,1\}^{l_1} \times \{0,1\}^{l_2} \to \{0,1\}^{m'}$, *where* l_1, l_2 *and* m' *are polynomial in* λ. Ext *is said to be a* (ϵ, δ)*-strong extractor with auxiliary inputs, if for every PPT adversary* $\mathcal{A}$*, and for all pairs* (x, f) *such that* $x \in \{0,1\}^{l_2}$ *and* $f \in \mathcal{H}_{\mathsf{ow}}(\epsilon)$*, we have:*

$$|\Pr[\mathcal{A}(r, f(x), \mathsf{Ext}(r, x)) = 1] - \Pr[\mathcal{A}(r, f(x), u) = 1]| < \delta.$$

where $r \in \{0,1\}^{l_1}$, $u \in \{0,1\}^{m'}$ *are chosen uniformly random.*

- Interestingly, we found out that a (ϵ, δ)-strong extractor with auxiliary inputs can be constructed from

$\langle r, x \rangle = \sum_{i=1}^{l} r_i x_i$ the inner product of $x = (x_1, \ldots x_l)$ and $r = (r_1, \ldots, r_l)$

(Proof is in the paper)

Construction of pAI-CPA Secure PKE

Let $\mathcal{H}_{\mathsf{ow}}(\epsilon_r)$ be the class of all polynomial-time computable functions $h : \{0,1\}^{|r'|} \to \{0,1\}^*$, such that given $h(r')$ (for a randomly generated r'), no PPT algorithm can find r' with probability greater than ϵ_r . The function $h(r')$ can be viewed as a composition of $q \in \mathbb{N}^+$ functions: $h(r') = (h_1(r'), \ldots, h_q(r'))$. Therefore $\{h_1, \ldots, h_q\} \in \mathcal{H}_{\mathsf{ow}}(\epsilon_r)$.

Let $\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$ be the class of all polynomial-time computable functions $h : \{0,1\}^{|\mathsf{sk}|+|\mathsf{pk}|} \to \{0,1\}^*$, such that given $(\mathsf{pk}, h(\mathsf{sk},\mathsf{pk}))$ (for a randomly generated $(\mathsf{sk},\mathsf{pk})$), no PPT algorithm can find sk with probability greater than ϵ_s . The function $h(\mathsf{sk},\mathsf{pk})$ can be viewed as a composition of q' functions: $h(\mathsf{sk},\mathsf{pk}) = (h_1(\mathsf{sk},\mathsf{pk}), \ldots, h_{q'}(\mathsf{sk},\mathsf{pk}))$. Therefore $\{h_1, \ldots, h_{q'}\} \in \mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$.

Construction of pAI-CPA Secure PKE

Let $\Pi' = (\mathtt{Gen}', \mathtt{Enc}', \mathtt{Dec}')$ be an AI-CPA secure encryption (with respect to family $\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$) where the encryption randomness is in $\{0,1\}^{m'}$, $\mathsf{Ext} : \{0,1\}^{l_1} \times \{0,1\}^{l_2} \rightarrow \{0,1\}^{m'}$ is a $(\epsilon_r, \mathsf{neg}(\lambda))$-strong extractor with auxiliary inputs, then a pAI-CPA secure (with respect to families $(\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s), \mathcal{H}_{\mathsf{ow}}(\epsilon_r))$) encryption scheme Π can be constructed as follows.

1. $\mathtt{Gen}(1^\lambda)$: It runs $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathtt{Gen}'(1^\lambda)$ and chooses r uniformly random from $\{0,1\}^{l_1}$. Then, we set the public key $\mathsf{PK} = (\mathsf{pk}, r)$ and the secret key $\mathsf{SK} = \mathsf{sk}$.
2. $\mathtt{Enc}(\mathsf{PK}, M)$: It picks x uniformly random from $\{0,1\}^{l_2}$. Then, it computes $y = \mathsf{Ext}(r, x)$. The ciphertext is $c = \mathtt{Enc}'(\mathsf{pk}, M; y)$.
3. $\mathtt{Dec}(\mathsf{SK}, c)$: It returns $\mathtt{Dec}'(\mathsf{sk}, c)$.

Theorem 3. *If Π' is an AI-CPA secure encryption with respect to family $\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$ and Ext is a $(\epsilon_r, \mathsf{neg}(\lambda))$-strong extractor with auxiliary inputs, then Π is pAI-CPA secure with respect to families $(\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s), \mathcal{H}_{\mathsf{ow}}(\epsilon_r))$.*

Extension to IBE setting

Extension to IBE. We can use the same technique to construct pAI-ID-CPA secure IBE. Let $\Sigma' = (\mathsf{Setup}', \mathsf{Extract}', \mathtt{Enc}', \mathtt{Dec}')$ be an AI-ID-CPA secure IBE (e.g. [19]) where the encryption randomness is in $\{0,1\}^{m'}$, $\mathsf{Ext} : \{0,1\}^{l_1} \times \{0,1\}^{l_2} \to \{0,1\}^{m'}$ is a $(\epsilon_r, \mathsf{neg}(\lambda))$-strong extractor with auxiliary inputs, then construct a pAI-ID-CPA secure IBE scheme Σ as follows.

1. $\mathsf{Setup}(1^\lambda)$: It runs $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'(1^\lambda)$ and chooses r uniformly random from $\{0,1\}^{l_1}$. Then, we set the master public key $\mathsf{MPK} = (\mathsf{mpk}, r)$ and the master secret key $\mathsf{MSK} = \mathsf{msk}$.
2. $\mathsf{Extract}(\mathsf{MSK}, \mathsf{ID})$: It returns $\mathsf{sk}_{\mathsf{ID}} \leftarrow \mathsf{Extract}(\mathsf{MSK}, \mathsf{ID})$.
3. $\mathtt{Enc}(\mathsf{MPK}, \mathsf{ID}, M)$: It chooses x uniformly random from $\{0,1\}^{l_2}$. Then, it computes $y = \mathsf{Ext}(r, x)$. The ciphertext is $c = \mathtt{Enc}'(\mathsf{mpk}, \mathsf{ID}, M; y)$.
4. $\mathtt{Dec}(\mathsf{sk}_{\mathsf{ID}}, c)$: It returns $\mathtt{Dec}'(\mathsf{sk}_{\mathsf{ID}}, c)$.

Theorem 4. *If Σ' is an AI-ID-CPA secure IBE with respect to family $\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$ and* Ext *is a $(\epsilon_r, \mathsf{neg}(\lambda))$-strong extractor with auxiliary inputs, then Σ is pAI-ID-CPA secure with respect to families $(\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s), \mathcal{H}_{\mathsf{ow}}(\epsilon_r))$.*

CCA Public Key Encryption from CPA IBE

- We give a first attempt, using the transformation given by Canetti (simply change the underlying IBE to be secure in the corresponding post-challenge auxiliary input model)

Let $(\mathtt{Gen}_s, \mathtt{Sign}, \mathtt{Verify})$ be a strong one-time signature scheme. Let $(\mathsf{Setup}', \mathsf{Extract}', \mathtt{Enc}', \mathtt{Dec}')$ be an auxiliary-inputs CPA secure IBE scheme

1. $\mathtt{Gen}(1^\lambda)$: Run $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'(1^\lambda)$. Set the public key $\mathsf{pk} = \mathsf{mpk}$ and the secret key $\mathsf{sk} = \mathsf{msk}$.
2. $\mathtt{Enc}(\mathsf{pk}, M)$: Run $(\mathsf{vk}, \mathsf{sk}_s) \leftarrow \mathtt{Gen}_s(1^\lambda)$. Calculate $c \leftarrow \mathtt{Enc}'(\mathsf{pk}, \mathsf{vk}, M)$ and $\sigma \leftarrow \mathtt{Sign}(\mathsf{sk}_s, c)$. Then, the ciphertext is $C = (c, \sigma, \mathsf{vk})$.
3. $\mathtt{Dec}(\mathsf{sk}, C)$: First, test $\mathtt{Verify}(\mathsf{vk}, c, \sigma) \stackrel{?}{=} 1$. If it is "1", compute $\mathsf{sk}_{\mathsf{vk}} = \mathsf{Extract}'(\mathsf{sk}, \mathsf{vk})$ and return $\mathtt{Dec}'(\mathsf{sk}_{\mathsf{vk}}, c)$. Otherwise, return $\perp$.

- The main challenge of pAI-CCA secure PKE is how to handle the leakage of the randomness used in the challenge ciphertext
- It includes the randomness used in $\mathtt{Gen}_s$, $\mathtt{Sign}$ and $\mathtt{Enc}'$, denoted as r_{sig_1}, r_{sig_2} and r_{enc}

CCA Public Key Encryption from CPA IBE

- We can re-write as $(\mathsf{vk}, \mathsf{sk}_s) \leftarrow \mathtt{Gen}_s(1^\lambda; r_{\mathsf{sig}_1}),\ \sigma \leftarrow \mathtt{Sign}(\mathsf{sk}_s, c; r_{\mathsf{sig}_2})$

  and $c \leftarrow \mathtt{Enc}'(\mathsf{mpk}, \mathsf{vk}, m_b; r_{\mathsf{enc}})$
- The adversary may ask:

  – $f_1(r') = r_{\mathsf{enc}}$, such that f_1 is still hard-to-invert upon r'. In this case, $\mathcal{A}$ can test $c^* \stackrel{?}{=} \mathtt{Enc}'(\mathsf{mpk}, \mathsf{vk},\ m_0; r_{\mathsf{enc}})$ to win the pAI-CCA game; or

  – $f_2(r') = (r_{\mathsf{sig}_1}, r_{\mathsf{sig}_2})$, such that f_2 is still hard-to-invert upon r'. In this case, given r_{sig_1}, $\mathcal{A}$ can generate $(\mathsf{vk}, \mathsf{sk}_s) = \mathtt{Gen}_s(1^\lambda; r_{\mathsf{sig}_1})$ which causes $\Pr[\mathsf{Forge}]$ defined in [5] to be non-negligible ("Forge" is the event that $\mathcal{A}$ wins the game by outputting a forged strong one-time signature).

CCA Public Key Encryption from CPA IBE

- Our Solution: set $r_{sig_1}, r_{sig_2}, r_{enc}$ are generated by the same source
- The randomness used in the IBE and the one-time signature can be calculated by $r_{\mathsf{enc}} = \mathsf{Ext}(r_1, x)$ and $(r_{\mathsf{sig}_1} \| r_{\mathsf{sig}_2}) = \mathsf{Ext}(r_2, x)$ for some random *x*
- The pAI-CCA adversary *A* can ask for the leakage of f(x), where f is any hard-to-invert function

1. $\mathtt{Gen}(1^\lambda)$: Run $(\mathsf{mpk}, \mathsf{msk}) \leftarrow \mathsf{Setup}'(1^\lambda)$. Choose r_1, r_2 uniformly random from $\{0,1\}^{l_1}$. Set the public key $\mathsf{pk} = (\mathsf{mpk}, r_1, r_2)$ and the secret key $\mathsf{sk} = \mathsf{msk}$.
2. $\mathtt{Enc}(\mathsf{pk}, m)$: Randomly sample $x \in \{0,1\}^{l_2}$, calculate $r_{\mathsf{enc}} = \mathsf{Ext}_1(r_1, x)$ and $r_{\mathsf{sig}_1} \| r_{\mathsf{sig}_2} = \mathsf{Ext}_2(r_2, x)$. Run $(\mathsf{vk}, \mathsf{sk}_s) = \mathtt{Gen}_s(1^\lambda; r_{\mathsf{sig}_1})$. Let $c = \mathtt{Enc}'(\mathsf{pk}, \mathsf{vk}, m; r_{\mathsf{enc}})$; $\sigma = \mathtt{Sign}(\mathsf{sk}_s, c; r_{\mathsf{sig}_2})$. Then, the ciphertext is $C = (c, \sigma, \mathsf{vk})$.
3. $\mathtt{Dec}(\mathsf{sk}, C)$: First, test $\mathtt{Verify}(\mathsf{vk}, c, \sigma) \stackrel{?}{=} 1$. If it is "1", compute $\mathsf{sk}_{\mathsf{vk}} = \mathsf{Extract}(\mathsf{sk}, \mathsf{vk})$ and return $\mathtt{Dec}'(\mathsf{sk}_{\mathsf{vk}}, c)$. Otherwise, return $\perp$.

Theorem 5. *Assuming that Π' is a AI-sID-CPA secure IBE scheme with respect to family $\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s)$, Π_s is a strong one-time signature, and Ext_1 is $(\epsilon_r, \mathsf{neg}_1)$-strong extractor with auxiliary inputs and Ext_2 is $(2\mathsf{neg}_1, \mathsf{neg}_2)$-strong extractor with auxiliary inputs, then there exists a PKE scheme Π which is pAI-CCA secure with respect to families $(\mathcal{H}_{\mathsf{pk-ow}}(\epsilon_s), \mathcal{H}_{\mathsf{ow}}(\epsilon_r))$.*

Conclusion

- We propose a new model to capture:
  - the leakage after the adversary seeing the challenge ciphertext
  - the leakage of two different parties: the secret key owner and the encryptor
- We give a generic construction of PKE + IBE in this new model (CPA secure)
- We also give a generic construction of CCA-PKE from CPA-IBE under this new model

~~~ END ~~~
~~~