

# CCA-Secure and Authentication Encryption

Yu Zhang

Harbin Institute of Technology

Cryptography, Autumn, 2018

- 1 Constructing CCA-Secure Encryption Schemes**
- 2 Obtaining Privacy and Message Authentication**
- 3 Key Derivation Function (FYI)**
- 4 Deterministic Encryption (FYI)**

## **1 Constructing CCA-Secure Encryption Schemes**

## 2 Obtaining Privacy and Message Authentication

## 3 Key Derivation Function (FYI)

## 4 Deterministic Encryption (FYI)

# Recall Security Against CCA

The CCA indistinguishability experiment  $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cca}}(n)$ :

- 1  $k \leftarrow \text{Gen}(1^n)$ .
- 2  $\mathcal{A}$  is given input  $1^n$  and oracle access  $\mathcal{A}^{\text{Enc}_k(\cdot)}$  and  $\mathcal{A}^{\text{Dec}_k(\cdot)}$ , outputs  $m_0, m_1$  of the same length.
- 3 a random bit  $b \leftarrow \{0, 1\}$  is chosen. Then  $c \leftarrow \text{Enc}_k(m_b)$  is given to  $\mathcal{A}$ .
- 4  $\mathcal{A}$  continues to have oracle access **except for**  $c$ , outputs  $b'$ .
- 5 If  $b' = b$ ,  $\mathcal{A}$  succeeded  $\text{PrivK}_{\mathcal{A}, \Pi}^{\text{cca}} = 1$ , otherwise 0.

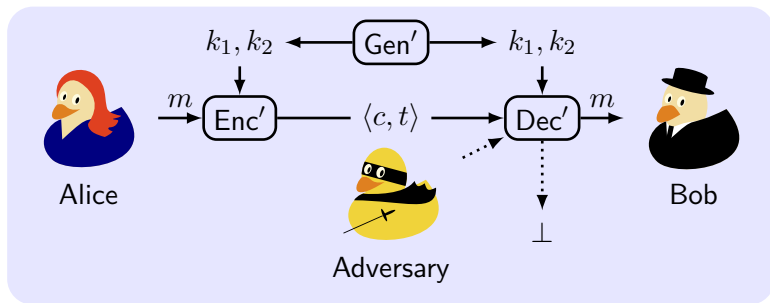
## Definition 1

$\Pi$  has **indistinguishable encryptions under a CCA (CCA-secure)** if  $\forall$  PPT  $\mathcal{A}$ ,  $\exists$   $\text{negl}$  such that

$$\Pr \left[ \text{PrivK}_{\mathcal{A}, \Pi}^{\text{cca}}(n) = 1 \right] \leq \frac{1}{2} + \text{negl}(n).$$

# Constructing CCA-Secure Encryption Schemes

**Idea:** Make decryption oracle useless. CCA = CPA-then-MAC.



## Construction 2

$\Pi_E = (\text{Gen}_E, \text{Enc}, \text{Dec})$ ,  $\Pi_M = (\text{Gen}_M, \text{Mac}, \text{Vrfy})$ .  $\Pi'$ :

- $\text{Gen}'(1^n)$ :  $k_1 \leftarrow \text{Gen}_E(1^n)$  and  $k_2 \leftarrow \text{Gen}_M(1^n)$
- $\text{Enc}'_{k_1, k_2}(m)$ :  $c \leftarrow \text{Enc}_{k_1}(m)$ ,  $t \leftarrow \text{Mac}_{k_2}(c)$  and output  $\langle c, t \rangle$
- $\text{Dec}'_{k_1, k_2}(\langle c, t \rangle) = \text{Dec}_{k_1}(c)$  if  $\text{Vrfy}_{k_2}(c, t) \stackrel{?}{=} 1$ ; otherwise  $\perp$

# Proof of CCA-Secure Encryption Schemes

## Theorem 3

*If  $\Pi_E$  is a CPA-secure private-key encryption scheme and  $\Pi_M$  is a secure MAC with unique tags, then Construction  $\Pi'$  is CCA-secure.*

## Proof.

VQ:  $\mathcal{A}$  submits a “new” query to oracle  $\text{Dec}'$  and  $\text{Vrfy} = 1$ .

$$\Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1] \leq \Pr[\text{VQ}] + \Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1 \wedge \overline{\text{VQ}}]$$

We need to prove the following claims.

- 1  $\Pr[\text{VQ}]$  is negligible.
- 2  $\Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1 \wedge \overline{\text{VQ}}] \leq \frac{1}{2} + \text{negl}(n)$ .



# Proof of “ $\Pr[\text{VQ}]$ is negligible”

**Idea:** Reduce  $\mathcal{A}_M$  (attacking  $\Pi_M$  with an oracle  $\text{Mac}_{k_2}(\cdot)$ ) to  $\mathcal{A}$ .

## Proof.

- $\mathcal{A}_M$  chooses  $i \leftarrow \{1, \dots, q(n)\}$  *u.a.r.*
- Run  $\mathcal{A}$  with the encryption/decryption oracles.
- If the  $i$ th decryption oracle query from  $\mathcal{A}$  uses a “new”  $c$ , output  $(c, t)$  and stop.
- $\text{Macforge}_{\mathcal{A}_M, \Pi_M}(n) = 1$  only if VQ occurs.
- $\mathcal{A}_M$  correctly guesses  $i$  with probability  $1/q(n)$ .

$$\Pr[\text{Macforge}_{\mathcal{A}_M, \Pi_M}(n) = 1] \geq \Pr[\text{VQ}]/q(n).$$



# Proof of “ $\Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1 \wedge \overline{\text{VQ}}] \leq \frac{1}{2} + \text{negl}(n)$ ”

**Idea:** Reduce  $\mathcal{A}_E$  (attacking  $\Pi_E$  with an oracle  $\text{Enc}_{k_1}(\cdot)$ ) to  $\mathcal{A}$ .

## Proof.

- Run  $\mathcal{A}$  with the encryption/decryption oracles.
- Run  $\text{PrivK}_{\mathcal{A}_E, \Pi_E}^{\text{cpa}}$  as  $\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}$ .
- $\mathcal{A}_E$  outputs the same  $b'$  that is output by  $\mathcal{A}$ .
- $\Pr[\text{PrivK}_{\mathcal{A}_E, \Pi_E}^{\text{cpa}}(n) = 1 \wedge \overline{\text{VQ}}] = \Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1 \wedge \overline{\text{VQ}}]$  unless VQ occurs.

$$\Pr[\text{PrivK}_{\mathcal{A}_E, \Pi_E}^{\text{cpa}}(n) = 1] \geq \Pr[\text{PrivK}_{\mathcal{A}, \Pi'}^{\text{cca}}(n) = 1 \wedge \overline{\text{VQ}}].$$





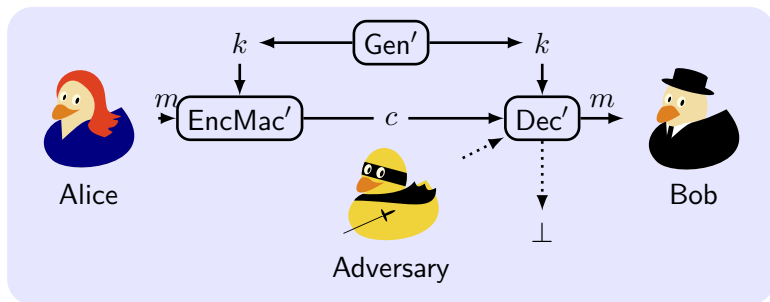
1 Constructing CCA-Secure Encryption Schemes

**2 Obtaining Privacy and Message Authentication**

3 Key Derivation Function (FYI)

4 Deterministic Encryption (FYI)

# Message Transmission Scheme



- **Key-generation** algorithm outputs  $k \leftarrow \text{Gen}'(1^n)$ .  
 $k = (k_1, k_2)$ .  $k_1 \leftarrow \text{Gen}_E(1^n)$ ,  $k_2 \leftarrow \text{Gen}_M(1^n)$ .
- **Message transmission** algorithm is derived from  $\text{Enc}_{k_1}(\cdot)$  and  $\text{Mac}_{k_2}(\cdot)$ , outputs  $c \leftarrow \text{EncMac}'_{k_1, k_2}(m)$ .
- **Decryption** algorithm is derived from  $\text{Dec}_{k_1}(\cdot)$  and  $\text{Vrfy}_{k_2}(\cdot)$ , outputs  $m \leftarrow \text{Dec}'_{k_1, k_2}(c)$  or  $\perp$ .
- **Correctness requirement**:  $\text{Dec}'_{k_1, k_2}(\text{EncMac}'_{k_1, k_2}(m)) = m$ .

# Defining Secure Message Transmission

The secure message transmission experiment  $\text{Auth}_{\mathcal{A}, \Pi'}(n)$ :

- 1  $k = (k_1, k_2) \leftarrow \text{Gen}'(1^n)$ .
- 2  $\mathcal{A}$  is given input  $1^n$  and oracle access to  $\text{EncMac}'_k$ , and outputs  $c \leftarrow \text{EncMac}'_k(m)$ .
- 3  $m := \text{Dec}'_k(c)$ .  $\text{Auth}_{\mathcal{A}, \Pi'}(n) = 1 \iff m \neq \perp \wedge m \notin \mathcal{Q}$ .

## Definition 4

$\Pi'$  achieves **authenticated communication** if  $\forall$  PPT  $\mathcal{A}$ ,  $\exists \text{negl}$  such that

$$\Pr[\text{Auth}_{\mathcal{A}, \Pi'}(n) = 1] \leq \text{negl}(n).$$

## Definition 5

$\Pi'$  is **secure Authenticated Encryption (A.E.)** if it is both CCA-secure and also achieves authenticated communication.

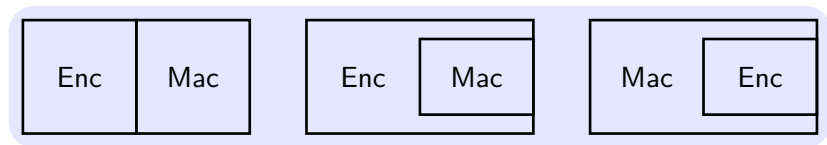
**Does CCA-security imply A.E.? (homework)**

# Questions

Suppose  $(E, D)$  provides A.E. Which of the following systems provide A.E.?

- $E'_k(m) = (E_k(m), E_k(m))$  and  $D'_k(c_1, c_2) = D_k(c_1)$
- $E'_k(m) = (E_k(m), 0)$  and  $D'_k(c, b) = \begin{cases} D_k(c) & \text{if } b = 0 \\ \perp & \text{otherwise} \end{cases}$
- $E'_k(m) = E_k(m) \oplus 1^*$  and  $D'_k(c) = D_k(c \oplus 1^*)$
- $E'_k(m) = (E_k(m), E_k(m))$  and
$$D'_k(c_1, c_2) = \begin{cases} D_k(c_1) & \text{if } D_k(c_1) = D_k(c_2) \\ \perp & \text{otherwise} \end{cases}$$
- $E'_k(m) = (E_k(m), H(m))$  ( $H$  is a CRHF) and
$$D'_k(c, h) = \begin{cases} D_k(c) & \text{if } H(D_k(c)) = h \\ \perp & \text{otherwise} \end{cases}$$

# Combining Encryption and Authentication



- **Encrypt-and-authenticate** (e.g., SSH):

$$c \leftarrow \text{Enc}_{k_1}(m), t \leftarrow \text{Mac}_{k_2}(m).$$

- **Authenticate-then-encrypt** (e.g., SSL):

$$t \leftarrow \text{Mac}_{k_2}(m), c \leftarrow \text{Enc}_{k_1}(m||t).$$

- **Encrypt-then-authenticate** (e.g., IPsec):

$$c \leftarrow \text{Enc}_{k_1}(m), t \leftarrow \text{Mac}_{k_2}(c).$$

# Analyzing Security of Combinations

**All-or-nothing:** Reject any combination for which there exists even a single counterexample is insecure.

- **Encrypt-and-authenticate:**  $\text{Mac}'_k(m) = (m, \text{Mac}_k(m))$ .

- **Authenticate-then-encrypt:**

- $\text{Trans} : 0 \rightarrow 00; 1 \rightarrow 10/01$ ;  $\text{Enc}'$  uses CTR mode;  
 $c = \text{Enc}'(\text{Trans}(m \parallel \text{Mac}(m)))$ .
- Flip the first two bits of  $c$  and verify whether the ciphertext is valid.  $10/01 \rightarrow 01/10 \rightarrow 1$ ,  $00 \rightarrow 11 \rightarrow \perp$ .
- If valid, the first bit of message is 1; otherwise 0.
- For any MAC, this is not CCA-secure.

- **Encrypt-then-authenticate:**

Decryption: If  $\text{Vrfy}(\cdot) = 1$ , then  $\text{Dec}(\cdot)$ ; otherwise output  $\perp$ .

# Authenticated Encryption Theory and Practice

## Theorem 6

$\Pi_E$  is CPA-secure and  $\Pi_E$  is a secure MAC with unique tags,  $\Pi'$  deriving from encrypt-then-authenticate approach is secure.

**GCM(Galois/Counter Mode):** CTR encryption then Galois MAC. (RFC4106/4543/5647/5288 on IPsec/SSH/TLS)

**EAX:** CTR encryption then CMAC.

## Proposition 7

*Authenticate-then-encrypt approach is secure if  $\Pi_E$  is rand-CTR mode or rand-CBC mode.*

**CCM (Counter with CBC-MAC):** CBC-MAC then CTR encryption. (802.11i, RFC3610)

**OCB (Offset Codebook Mode):** integrating MAC into ENC. (two times fast as CCM, EAX)

**All support AEAD (A.E. with associated data):** part of message is in clear, and all is authenticated

# Remarks on Secure Message Transmission

- Authentication may leak the message.
- Secure message transmission implies CCA-security. The opposite direction is not necessarily true.
- Different security goals should always use different keys.
  - otherwise, the message may be leaked if  $\text{Mac}_k(c) = \text{Dec}_k(c)$ .
- Implementation may destroy the security proved by theory.
  - **Attack with padding oracle** (in TLS 1.0):  
**Dec** return two types of error: padding error, MAC error.  
**Adv.** learns last bytes if no padding error with guessed bytes.
  - **Attack non-atomic dec.** (in SSH Binary Packet Protocol):  
**Dec** (1)decrypt length field; (2)read packets as specified by the length; (3)check MAC.  
**Adv.** (1)send  $c$ ; (2)send  $l$  packets until “MAC error” occurs; (3)learn  $l = \text{Dec}(c)$ .



- 1 Constructing CCA-Secure Encryption Schemes
- 2 Obtaining Privacy and Message Authentication
- 3 Key Derivation Function (FYI)**
- 4 Deterministic Encryption (FYI)

# Key Derivation Function (KDF)

**Key Derivation Function (KDF)** generates many keys from a secret source key  $sk$ .

**For uniformly random  $sk$ :**  $F$  is PRF,  $ctx$  is a unique string identifying application,

$$\text{KDF}(sk, ctx, l) = \langle F_{sk}(ctx\|0), F_{sk}(ctx\|1) \cdots, F_{sk}(ctx\|l) \rangle.$$

**For not-uniform  $sk$ :** extract-then-expand paradigm.

**extract:** HKDF  $k \leftarrow \text{HMAC}(salt, sk)$ .  $salt$  is a random number.

**expand:** as the above.

# Password-Based KDF (PBKDF)

**Key stretching** increases the time of testing key (with slow hash function).

**Key strengthening** increases the length/randomness of key (with salt).

**PKCS#5 (PBKDF1):**  $H^{(c)}(pwd || salt)$ , iterate hash function  $c$  times.

**Attack:** either try the enhanced key (larger key space), or else try the initial key (longer time per key).

## IV, Nonce, Counter and Salt

**IV** an input to a cryptographic primitive, providing randomness.

**nonce** a number used only once to sign a communication.

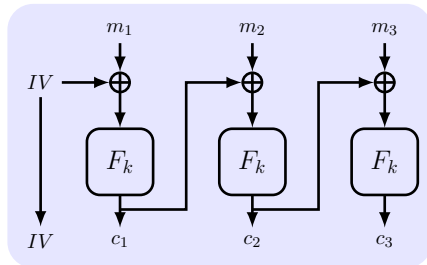
**counter** a sequence number used as nonce.

**salt** consists of random bits, creating the input to a function.

- 1 Constructing CCA-Secure Encryption Schemes
- 2 Obtaining Privacy and Message Authentication
- 3 Key Derivation Function (FYI)
- 4 Deterministic Encryption (FYI)**

# Deterministic CPA Security

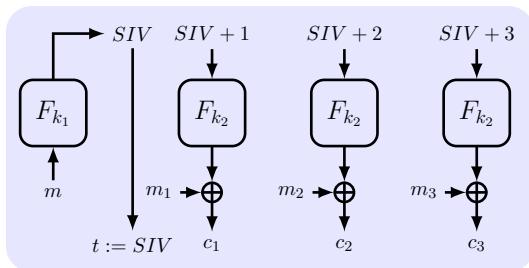
- **Application:** encrypted database index, disk encryption
- **Deterministic encryption:** the same message is encrypted to the same ciphertext under the same key.
- **Q: would it be CPA-secure?**
- **Deterministic CPA Security:** CPA-secure if *never encrypt same message twice* using same key. The pair  $\langle k, m \rangle$  is unique.
- **Deterministic Authenticated Encryption (DAE)**
- **Common Mistake:** CBC/CTR with **fixed**  $IV$ .



Adversary can query  $(m_{q1}, m_{q2})$  and get  $(c_{q1}, c_{q2})$ ; then output PT:  $IV \oplus c_{q1} \oplus m_{q2}$  and expect CT:  $c_{q2}$ .

# Synthetic $IV$ (SIV) for Det. Encryption

- **SIV** (fixed  $IV$  for the same  $\langle k, m \rangle$ ):  
PRF  $F$ , CPA-secure  $\Pi : (\text{Enc}_k(r, m), \text{Dec}_k(r, s))$   
 $(k_1, k_2) \leftarrow \text{Gen}$ ;  $SIV \leftarrow F_{k_1}(m)$   
 $c = \langle SIV, \text{Enc}_{k_2}(SIV, m) \rangle$ .
- **DAE for free with SIV-CTR**:



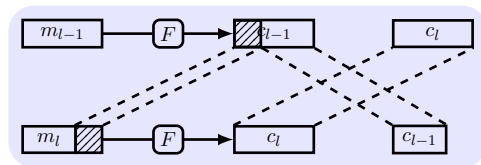
# Wide Block PRP for Det. Encryption

- **Just PRP:** If  $F$  is PRP, then  $F$  is also det. CPA-secure.
- **PRP-based DAE:**  $\text{Enc}_k(m\|0^\ell)$ . In Dec, if  $\neq 0^\ell$ , output  $\perp$ .
- **Narrow block** may leak info. as some blocks are the same.
- **Wide block PRP:** PRP with longer block length (e.g. a sector on disk) from PRP with short block length (e.g. AES).
- **Standards:** CBC-mask-CBC (CMC) and ECB-mask-ECB (EME) in IEEE P1619.2.
- **Cost:** 2x slower than SIV due to two-pass encryption.



# Tweakable Encryption

- **Encryption without expansion:**  $\mathcal{M} = \mathcal{C}$  implies det. encryption without integrity (e.g., disk encryption).
- **Trivial solution:**  $k_t = F_k(t), t = 1, \dots, \ell$ .
- **Tweakable block ciphers:** many PRPs from one key  $\mathcal{K} \times \mathcal{T} \times \mathcal{X} \rightarrow \mathcal{X}$ ,  $\mathcal{T}$  is the set of tweaks.
- **XTS:** XEX(Xor-Encrypt-Xor)-based tweaked-codebook mode with ciphertext stealing. (XTS-AES, NIST SP 800-38E)
- **XEX:** To encrypt block  $j$  in sector  $I$ ,  $c = F_k(m \oplus x) \oplus x$ , where  $x = F_k(I) \otimes 2^j$  in Galois field,  $(I, j)$  is tweak.
- **Ciphertext stealing (CTS):** no padding, **no expansion**.



# Summary

- CCA-secure, AE, det. enc., det. CPA-secure, DAE.
- Enc-then-auth, KDF, SIV, wide block cipher, tweakable encryption.
- SIV-CTR, PBKDF, salt, enc. w/o expansion, CTS.