Katz, Lindell
Introduction to Modern Cryptrography
Slides Chapter 11

Markus Bläser, Saarland University

Public-key encryption

Definition (11.1)

A *public-key encryption scheme* is a triple $(\mathsf{Gen}, \mathsf{Enc}, \mathsf{Dec})$ of ppt algorithms such that

1. $\mathsf{Gen}(1^n)$ outputs a pair of keys (pk, sk), the *public key* and the *private key*. (Both have length at least n and n can be deduced from them.)
2. Enc takes as input a public key pk and a message m from some message space (that may depend on pk) and outputs a ciphertext $c \leftarrow \mathsf{Enc}_{pk}(m)$.
3. Dec takes as input a private key sk and a ciphertext c and outputs a message m or $\perp$. We write $m := \mathsf{Dec}_{sk}(c)$.

We require that $\mathsf{Dec}_{sk}(\mathsf{Enc}_{pk}(m)) = m$ for any legal message m except with negligible probability over (pk, sk) output by $\mathsf{Gen}(1^n)$

EAV-security

The eavesdropping indistinguishability experiment
$\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n)$:

1. $\mathsf{Gen}(1^n)$ is run to generate keys (pk, sk).
2. $\mathcal{A}$ is given pk and outputs valid messages m_0, m_1 of the same length.
3. $b \in \{0,1\}$ is chosen uniformly at random and $c \leftarrow \mathsf{Enc}_{pk}(m_b)$ is given to $\mathcal{A}$.
4. $\mathcal{A}$ outputs a bit b'. The outcome of the experiment is 1 if $b = b'$ and 0 otherwise.

Definition (11.2)

A public-key encryption scheme Π is *EAV-secure* if for every ppt $\mathcal{A}$,

$$\Pr[\mathsf{PubK}^{\mathsf{eav}}_{\mathcal{A},\Pi}(n) = 1] \leq \tfrac{1}{2} + \mathsf{negl}(n).$$

CPA-security

Proposition (11.3)

If a public key encryption scheme is EAV-secure, then it is CPA-secure.

Theorem (11.4)

No deterministic public key encryption scheme is CPA-secure.

Multiple encryptions

Recall: $\mathsf{LR}_{pk,\mathrm{b}}(\mathrm{m}_0, \mathrm{m}_1)$ computes $\mathrm{c} \leftarrow \mathsf{Enc}_{pk}(\mathrm{m}_\mathrm{b})$

The LR-oracle experiment $\mathsf{PubK}^{\mathsf{LR\text{-}cpa}}_{\mathcal{A},\Pi}(\mathrm{n})$**:**

1. $\mathsf{Gen}(1^\mathrm{n})$ is run to obtain (pk, sk).
2. $\mathrm{b} \in \{0,1\}$ is chosen uniformly at random.
3. $\mathcal{A}$ is given pk and oracle access to $\mathsf{LR}_{pk,\mathrm{b}}(.,.)$.
   ($\mathcal{A}$ does not see b!)
4. $\mathcal{A}$ outputs a bit b'.
5. The outcome of the experiment is 1 if $\mathrm{b} = \mathrm{b}'$ and 0 otherwise.

Multiple encryptions (2)

Definition (11.5)

A public-key encryption scheme π has *indistinguishable multiple encryptions* if for all ppt $\mathcal{A}$,

$$\Pr[\mathsf{PubK}^{\mathsf{LR\text{-}cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n)$$

Theorem (11.6)

If a public-key encryption scheme Π *is CPA-secure, then it has indistinguishable multiple encryptions.*

Consequence: fixed length encryption extends to arbitrary length encryption.

Proof of Theorem 11.6

Adversary $\mathcal{A}'(pk)$:

1. $\mathcal{A}'$ chooses $\mathrm{i} \leftarrow \{1, \ldots, \mathrm{t}\}$.
2. $\mathcal{A}'$ runs $\mathcal{A}(pk)$ answering the jth query $(\mathrm{m}_{\mathrm{j},0}, \mathrm{m}_{\mathrm{j},1})$ as follows:
   - 2.1 if $\mathrm{j} < \mathrm{i}$, $\mathcal{A}'$ computes $\mathrm{c}_\mathrm{j} \leftarrow \mathrm{Enc}_{pk}(\mathrm{m}_{\mathrm{j},0})$ and returns c_j to $\mathcal{A}$.
   - 2.2 if $\mathrm{j} = \mathrm{i}$, then $\mathcal{A}'$ outputs $(\mathrm{m}_{\mathrm{j},0}, \mathrm{m}_{\mathrm{j},1})$ and get a challenge c_j. $\mathcal{A}'$ returns the challenge c_j to $\mathcal{A}$.
   - 2.3 if $\mathrm{j} > \mathrm{i}$, $\mathcal{A}'$ computes $\mathrm{c}_\mathrm{j} \leftarrow \mathrm{Enc}_{pk}(\mathrm{m}_{\mathrm{j},1})$ and returns c_j to $\mathcal{A}$.
3. $\mathcal{A}'$ outputs the bit b' that is output by $\mathcal{A}$.

CCA-security

The CCA indistinguishability experiment $\mathsf{PubK}^{\mathsf{cca}}_{\mathcal{A},\pi}(n)$**:**

1. $\mathsf{Gen}(1^n)$ is run to obtain (pk, sk).
2. $\mathcal{A}$ gets pk and oracle access to $\mathsf{Dec}_{sk}(.)$. $\mathcal{A}$ outputs two valid messages m_0, m_1 of the same length.
3. $b \in \{0,1\}$ is chosen uniformly at random and $c \leftarrow \mathsf{Enc}_{pk}(m_b)$ is given to $\mathcal{A}$.
4. $\mathcal{A}$ has still access to $\mathsf{Dec}_{sk}(.)$ but may not query c. $\mathcal{A}$ outputs a bit b'.
5. The outcome of the experiment is 1 if $b = b'$ and 0 otherwise.

Definition (11.8)

Π is *CCA-secure* if for all ppt $\mathcal{A}$,

$$\Pr[\mathsf{PubK}^{\mathsf{cca}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n),$$

Hybrid encryption

- Private key schemes are more efficient than public key ones.
- Hyprid approach:
  - Choose a key k uniformly at random.
  - Encrypt k using pk and send it to Bob.
  - Encrypt the message using k and send it to Bob
- This is a full-fledged public key scheme, since k is not shared in advance.
- Uses a two-stage approach.

Key encapsulation

Definition (11.9)

A *key-encapsulation mechanism (KEM)* is a triple of ppt algorithms $(\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ such that

1. Gen on input 1^n outputs a pair of keys (pk, sk). Both have length at least n and n can be determined from pk.
2. Encaps on input pk and 1^n outputs a ciphertext c and a key $k \in \{0,1\}^{\ell(n)}$, where ℓ is the key length. We write $(c, k) \leftarrow \mathsf{Encaps}_{pk}(1^n)$.
3. Decaps is deterministic and takes as input sk and c and outputs a key k or $\perp$. We write $k := \mathsf{Decaps}_{sk}(c)$.

Key encapsulation (2)

- Sender runs $\text{Encaps}_{pk}(1^n)$ to obtain c and k.
- Then uses a private key encryption scheme ("data encapsulation mechanism") to encrypt m using k yielding c'
- He then sends c and c'.

What is the message length? Let

- α denote the cost of encapsulating an n-bit string and
- β be the cost per bit to encrypt m

Total time per bit:

$$\frac{\alpha + \beta|m|}{|m|} = \frac{\alpha}{|m|} + \beta.$$

Key encapsulation (3)

Construction 11.10:
Let $\Pi = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a KEM and
$\Pi' = (\mathsf{Gen}', \mathsf{Enc}', \mathsf{Dec}')$ be a private-key encryption scheme.
Construct a public-key encryption scheme
$\Pi^{\mathsf{hy}} = (\mathsf{Gen}^{\mathsf{hy}}, \mathsf{Enc}^{\mathsf{hy}}, \mathsf{Dec}^{\mathsf{hy}})$ as follows:

- $\mathsf{Gen}^{\mathsf{hy}}$: on input 1^n run $\mathsf{Gen}(1^n)$ to obtain (pk, sk).
- $\mathsf{Enc}^{\mathsf{hy}}$: on input pk and message $m \in \{0,1\}^*$ do
  1. Compute $(c, k) \leftarrow \mathsf{Encaps}_{pk}(1^n)$
  2. Compute $c' \leftarrow \mathsf{Enc}'_k(m)$
  3. Output (c, c')
- $\mathsf{Dec}^{\mathsf{hy}}$: on input sk and (c, c') do
  1. Compute $k := \mathsf{Decaps}_{sk}(c)$
  2. Output $m := \mathsf{Dec}'_k(c')$

CPA-security

Let $\Pi = (\mathsf{Gen}, \mathsf{Encaps}, \mathsf{Decaps})$ be a KEM:

The CPA indistinguishability experiment $\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n)$**:**

1. $\mathsf{Gen}(1^n)$ is run to obtain keys (pk, sk). Then $\mathsf{Encaps}_{pk}(1^n)$ is run to generate (c, k) with $k \in \{0,1\}^n$
2. Choose $b \in \{0,1\}$ uniformly at random. If $b = 0$ set $\hat{k} := k$ Else choose $\hat{k} \in \{0,1\}^n$ uniformly at random.
3. Give $(pk, c, \hat{k})$ to $\mathcal{A}$. $\mathcal{A}$ outputs a bit b'. The outcome of the experiment is 1 if $b = b'$ and 0 otherwise.

Definition (11.11)

A KEM Π is CPA-secure if for all ppt $\mathcal{A}$,

$$\Pr[\mathsf{KEM}^{\mathsf{cpa}}_{\mathcal{A},\Pi}(n) = 1] \leq \frac{1}{2} + \mathsf{negl}(n).$$

CPA-security (2)

Theorem (11.12)

If Π is a CPA-secure KEM and Π' is a private encryption scheme that is EAV-secure, then Π^{hy} (Construction 11.10) is a CPA-secure public encryption scheme.

CPA security (3)

Adversary $\mathcal{A}_1$:

1. $\mathcal{A}_1$ is given $(pk, \mathrm{c}, \hat{\mathrm{k}})$
2. $\mathcal{A}_1$ runs $\mathcal{A}^{\mathsf{hy}}(pk)$ to obtain $\mathrm{m}_0, \mathrm{m}_1$. Then $\mathcal{A}_1$ computes $\mathrm{c}' \leftarrow \mathrm{Enc}'_{\hat{\mathrm{k}}}(\mathrm{m}_0)$, gives $(\mathrm{c}, \mathrm{c}')$ to $\mathcal{A}^{\mathsf{hy}}$ and outputs the bit b' that $\mathcal{A}^{\mathsf{hy}}$ outputs.

Adversary $\mathcal{A}'$:

1. $\mathcal{A}'(1^{\mathrm{n}})$ runs $\mathrm{Gen}(1^{\mathrm{n}})$ to generate (pk, sk). $\mathcal{A}'$ computes $\mathrm{c} \leftarrow \mathrm{Encaps}^{(1)}_{pk}(1^{\mathrm{n}})$.
2. $\mathcal{A}'$ runs $\mathcal{A}^{\mathsf{hy}}(pk)$ to obtain messages $\mathrm{m}_0, \mathrm{m}_1$. $\mathcal{A}'$ outputs these messages and gets a challenge c'.
3. $\mathcal{A}'$ gives $(\mathrm{c}, \mathrm{c}')$ to $\mathcal{A}^{\mathsf{hy}}$ and outputs the bit b' that is output by $\mathcal{A}^{\mathsf{hy}}$.

CCA-security

Let $\Pi = (\text{Gen}, \text{Encaps}, \text{Decaps})$ be a KEM:

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

1. $\text{Gen}(1^n)$ is run to obtain keys (pk, sk). Then $\text{Encaps}_{pk}(1^n)$ is run to generate (c, k) with $k \in \{0,1\}^n$
2. Choose $b \in \{0,1\}$ uniformly at random. If $b = 0$ set $\hat{k} := k$ Else choose $\hat{k} \in \{0,1\}^n$ uniformly at random.
3. Give $(pk, c, \hat{k})$ to $\mathcal{A}$ and access to the oracle $\text{Decaps}_{sk}(.)$. $\mathcal{A}$ may not query c itself.
4. $\mathcal{A}$ outputs a bit b'. The outcome of the experiment is 1 if $b = b'$ and 0 otherwise.

CCA-security (2)

Definition (11.13)

A KEM Π is CCA-secure if for all ppt $\mathcal{A}$,

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

Theorem (11.14)

If Π is a CCA-secure KEM and Π' is a CCA-secure private key encryption scheme, then Π^{hy} (Construction 11.10) is a CCA-secure public key encryption scheme.

El-Gamal encryption

We need to generate a group G of order q, where q is an n-bit number, and a generator g.

Construction 11.16:

- Gen: Generate (G, q, g). Choose $x \in \mathbb{Z}_q$ uniformly at random and compute $h := g^x$. pk is (G, q, g, h) and sk is (G, q, g, x). The message space is G.
- Enc: given $pk = (G, q, g, h)$ and $m \in G$ choose a uniform $y \in \mathbb{Z}_q$ and output $\langle g^y, h^y \cdot m \rangle$.
- Dec: on input $sk = (G, q, g, x)$ and ciphertext $\langle c_1, c_2 \rangle$ output $\hat{m} = c_2 / c_1^x$.

It works, because:

$$\hat{m} = \frac{c_2}{c_1^x} = \frac{h^y m}{(g^y)^x} = \frac{(g^x)^y m}{g^{xy}} = m$$

El-Gamal encryption (2)

Lemma (11.15)

Let G *be a finite group and let* $\mathrm{m} \in \mathrm{G}$. *Then the elements* $\mathrm{k} \cdot \mathrm{m}$ *are uniformly distributed, that is, for any fixed* $\mathrm{g} \in \mathrm{G}$,

$$\mathrm{Pr}_{\mathrm{k} \in \mathrm{G}}[\mathrm{km} = \mathrm{g}] = 1/|\mathrm{G}|.$$

Theorem (11.18)

If DDH is hard relative to the group generation algorithm, then the El-Gamal scheme is CPA-secure.

El-Gamal encryption (3)

Algorithm D**:**

D gets $(\mathrm{G}, \mathrm{q}, \mathrm{g}, \mathrm{h}_1, \mathrm{h}_2, \mathrm{h}_3)$ as input

1. Set $pk = (\mathrm{G}, \mathrm{h}, \mathrm{q}, \mathrm{h}_1)$ and run $\mathcal{A}(pk)$ to obtain messages $\mathrm{m}_0, \mathrm{m}_1 \in \mathrm{G}$.
2. Choose a bit b uniformly at random and set $\mathrm{c}_1 := \mathrm{h}_2$ and $\mathrm{c}_2 := \mathrm{h}_3 \cdot \mathrm{m}_\mathrm{b}$.
3. Give $\langle \mathrm{c}_1, \mathrm{c}_2 \rangle$ to $\mathcal{A}$ and obtain its bit b'.
4. If $\mathrm{b} = \mathrm{b}'$, then output 1. Otherwise, output 0.

DDH-based key encapsulation

Construction 11.19

- Gen: generate (G, q, g). Choose a uniform $x \in \mathbb{Z}$ and set $h := g^x$. Choose a function $H : G \to \{0,1\}^{\ell(n)}$. pk is (G, q, g, h, H) and sk is (G, q, g, x)
- Encaps: on input pk, choose a uniform $y \in \mathbb{Z}_q$ and output the ciphertext g^y and the key $H(h^y)$.
- Decaps: on input sk and ciphertext $c \in G$, output the key $H(c^x)$.

$H : G \to \{0,1\}^{\ell(n)}$ should distribute the group elements "almost evenly" among the strings.

DDH-based key encapsulation (2)

Theorem (11.20)

If DDH is hard relative to the group generation algorithm, then Construction 11.19 is a CPA-secure KEM.

Theorem (11.21)

If the CDH problem is hard and H *is modelled as a random oracle, then Construction 1.19 is CPA-secure.*

The gap-CDH assumption

- Let G be a cyclic group with generator g.
- Let $h_1 = g^{x_1}$ and $h_2 = g^{x_2}$. We define

$$DH_g(h_1, h_2) := g^{x_1 \cdot x_2} = h_1^{x_2} = h_2^{x_1}.$$

- Computational Diffie-Hellman experiment: Compute $DH(h_1, h_2)$ from uniform h_1 and h_2 (not knowing x_1 or x_2).

Now: We add an oracle $=$ to the experiment: $O(u, v) = 1$ if $u = v^y$ and 0 otherwise.

Theorem (11.22)

If the gap-CDH problem is hard relative to the group generation algorithm and H is modeled as a random oracle, then Construction 11.19 is a CCA-secure KEM.

CCA-security

Construction 11.23

Let $\Pi_E = (\mathrm{Enc}', \mathrm{Dec}')$ be a private-key encryption scheme and let $\Pi_M = (\mathrm{Mac}, \mathrm{Vrfy})$ be a message authentication code.

- Gen: Generate (G, q, g). Choose $c \in \mathbb{Z}_q$ uniformly at random, set $h := g^x$ and choose an $H : G \to \{0,1\}^{2n}$. pk is (G, q, g, h, H) and sk is (G, q, g, x, H).
- Enc: Choose $y \in \mathbb{Z}_q$ and set $k_E \| k_M = H(h^y)$. Compute $c' \leftarrow \mathrm{Enc}'_{k_E}(m)$ and output the cipher text $(g^y, c', \mathrm{Mac}_{k_M}(c'))$.
- Dec: On ciphertext (c, c', t) output $\perp$ if $c \notin G$. Else compute $k_E \| k_M := H(c^x)$. If $\mathrm{Vrfy}_{k_M}(c', t) \neq 1$ then output $\perp$. Otherwise output $\mathrm{Dec}'_{k_E}(c')$.

DHIES/ECIES

Corollary (11.24)

Let Π_E be CPA-secure and Π_M be a strongly secure MAC. If gap-CDH is hard and H is modeled as a random oracle, then Construction 11.23 is a CCA-secure public key encryption scheme

Encryption of a message:

$$(g^y, \mathrm{Enc}'_{k_E}(m), \mathrm{Mac}_{k_M}(c')).$$

- Diffie–Hellman Integrated Encryption Scheme: G is a cyclic subgroup of a finite field.
- Elliptic Curve Integrated Encryption Scheme: G is an elliptic curve group.