

Diffie–Hellman Key Exchange and Public-Key Encryption

CS 171: Cryptography

Sanjam Garg

October 15, 2026

Central public-key primitives

- **Key exchange** (today)
 - **Public-key encryption** (today)
 - **Digital signatures** (later)
- All eliminate the need for a pre-shared secret key between the parties.

Overview

- 1 Assumptions
- 2 Key Exchange
- 3 Public-Key Encryption
- 4 Summary

Cryptographic Group

- $\mathbb{Z}_p^* = \{1, 2, \dots, p-1\}$ with $a \cdot b := ab \bmod p$, for a prime p .
- Pick primes $p = 2q + 1$, and any $g \in \mathbb{Z}_p^*$ with $g \neq 1$ and $g^q = 1$.
- Then $G = \langle g \rangle = \{g, g^2, \dots, g^q = 1\}$ is a cyclic group of **prime** order q .

Small Example: $p = 23$, $q = 11$

g	$\langle g \rangle$	order
1	$\{1\}$	1
2	$\{2, 4, 8, 16, 9, 18, 13, 3, 6, 12, 1\}$	11
5	all of $\mathbb{Z}_{23}^*$	22
22	$\{22, 1\}$	2

- Every order divides $22 = |\mathbb{Z}_{23}^*|$. Order 11 is prime: take $G = \langle 2 \rangle$.

Large Example: A 128-bit Prime

The group

$$p = 340282366920938463463374607431768196007 \quad (= 2^{128} - 15449, \text{ prime})$$

$$q = 170141183460469231731687303715884098003 \quad (= (p - 1)/2, \text{ prime})$$

$$g = 2, \quad \text{and } g^q \bmod p = 1, \text{ so } G = \langle 2 \rangle \text{ has prime order } q$$

- $x = 105477853248457859176235242282082391144$
 $h = g^x \bmod p = 176686941779824341518076277832209492233$
- Computing h from x is fast (repeated squaring). Finding x from (p, q, g, h) is the **discrete-log** problem.
- Even 128 bits is too small for security: real systems use p of 2048 bits (617 digits).

Recall: The Discrete-Log Problem

$\mathcal{G}(1^n)$ outputs a cyclic group G of order q ($|q| = n$) with generator g .

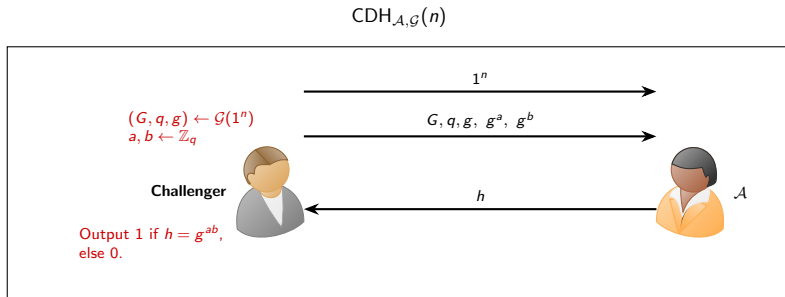
Game $\text{DLog}_{\mathcal{A}, \mathcal{G}}(n)$

- ① $(G, q, g) \leftarrow \mathcal{G}(1^n)$, and $h \leftarrow G$ uniform.
 - ② $\mathcal{A}$ is given 1^n and (G, q, g, h) , and outputs $x \in \mathbb{Z}_q$.
 - ③ Output 1 if $g^x = h$, and 0 otherwise.
- **Hard** relative to $\mathcal{G}$: every PPT $\mathcal{A}$ wins with probability $\leq \text{negl}(n)$.
 - Last time: DLog hard $\Rightarrow$ collision-resistant hashing, $H_s(x, r) = g^x h^r$.

The Diffie–Hellman Problems

- **Computational (CDH):** given g^a and g^b , compute g^{ab} .
- **Decisional (DDH):** given g^a and g^b , distinguish g^{ab} from a uniform group element.

CDH_{A,G}: Challenger vs. Adversary



Computational Diffie–Hellman Problem

Game $\text{CDH}_{\mathcal{A}, \mathcal{G}}(n)$

- 1 $(G, q, g) \leftarrow \mathcal{G}(1^n)$, and $a, b \leftarrow \mathbb{Z}_q$.
- 2 $\mathcal{A}$ is given 1^n and (G, q, g, g^a, g^b) , and outputs $h \in G$.
- 3 Output 1 if $h = g^{ab}$, and 0 otherwise.

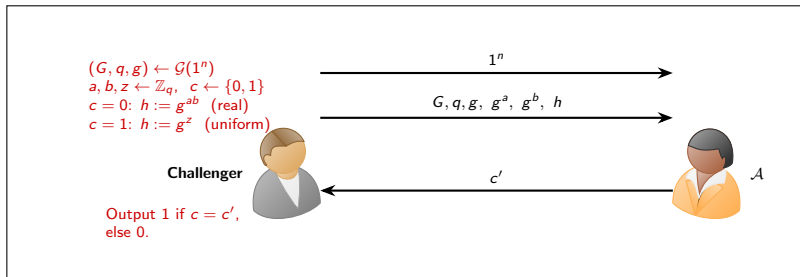
Definition 1 (CDH hardness)

CDH is **hard relative to** $\mathcal{G}$ if for all PPT $\mathcal{A}$:

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

DDH_{A,G}: Challenger vs. Adversary

DDH_{A,G}(n)



Decisional Diffie–Hellman Problem

Game $\text{DDH}_{\mathcal{A},\mathcal{G}}(n)$

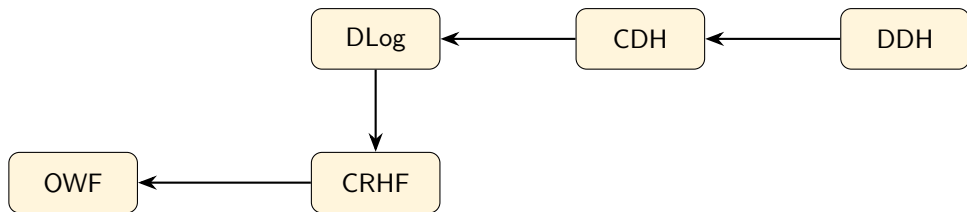
- 1 $(G, q, g) \leftarrow \mathcal{G}(1^n)$, and $a, b, z \leftarrow \mathbb{Z}_q$. Sample a uniform bit c .
- 2 If $c = 0$ set $h := g^{ab}$ (real); if $c = 1$ set $h := g^z$ (uniform).
- 3 $\mathcal{A}$ is given 1^n and (G, q, g, g^a, g^b, h) , and outputs c' .
- 4 Output 1 if $c' = c$, and 0 otherwise.

Definition 2 (DDH hardness)

DDH is **hard relative to** $\mathcal{G}$ if for all PPT $\mathcal{A}$:

$$\Pr [\text{DDH}_{\mathcal{A},\mathcal{G}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n).$$

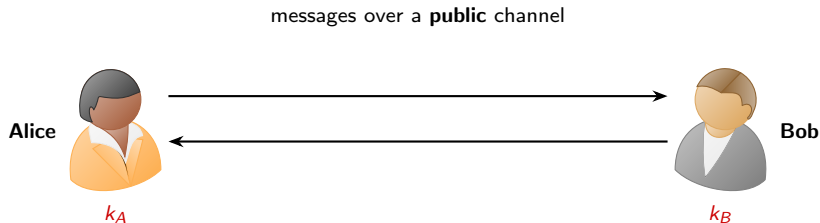
Diffie–Hellman Problems: Relations



$X \rightarrow Y$: “if X is hard, then Y is hard”. DDH is the **strongest** assumption.

- **DDH** $\rightarrow$ **CDH**: a CDH solver breaks DDH — compute g^{ab} and compare it with h .
- **CDH** $\rightarrow$ **DLog**: a DLog solver breaks CDH — find a from g^a , output $(g^b)^a = g^{ab}$.
- **DLog** $\rightarrow$ **CRHF**: the theorem we just proved. **CRHF** $\rightarrow$ **OWF**: a collision-resistant (compressing) hash function is one-way.

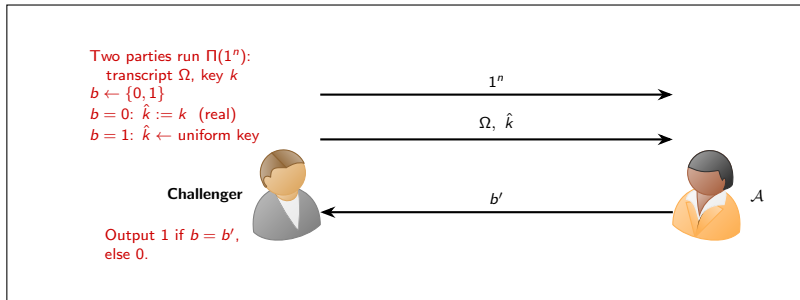
Key Exchange



- **Correctness:** $k_A = k_B =: k$.
- **Security (informally):** an eavesdropper who sees all the messages should not be able to tell k apart from a random key.

$\text{KE}_{\mathcal{A}, \Pi}^{\text{eav}}$: Challenger vs. Adversary

$\text{KE}_{\mathcal{A}, \Pi}^{\text{eav}}(n)$



Key Exchange: Security

Game $\text{KE}_{\mathcal{A}, \Pi}^{\text{eav}}(n)$

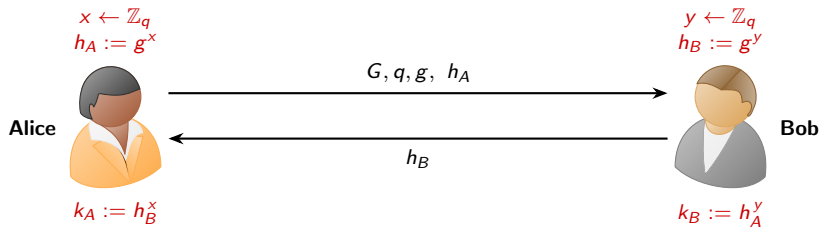
- 1 Two parties holding 1^n execute Π . This gives a transcript Ω of the messages and a key k output by each party.
- 2 Sample a uniform bit b . If $b = 0$ set $\hat{k} := k$; if $b = 1$ sample $\hat{k}$ uniformly.
- 3 $\mathcal{A}$ is given 1^n and $(\Omega, \hat{k})$, and outputs b' .
- 4 Output 1 if $b' = b$, and 0 otherwise.

Definition 3 (Secure key exchange)

A key-exchange protocol Π is **secure** (against eavesdroppers) if for all PPT $\mathcal{A}$:

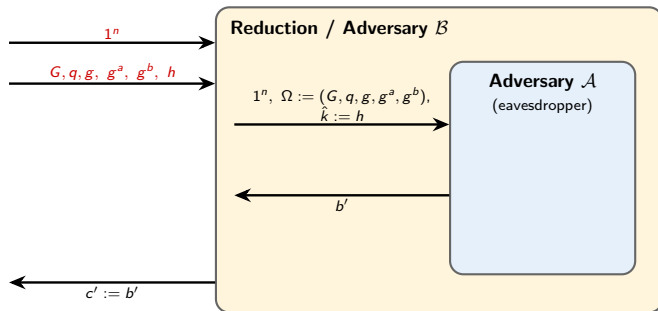
$$\Pr [\text{KE}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n).$$

The Diffie–Hellman Key-Exchange Protocol



- G, q, g can be fixed once and for all.
- **Correctness:** $k_A = (g^y)^x = g^{xy} = (g^x)^y = k_B$.
- **Security:** follows from the **DDH** assumption (next slide).
- **Subtle point:** the key is indistinguishable from a uniform *group element*, not from a uniform *bit string*.

Diffie–Hellman Is Secure: Proof by Reduction



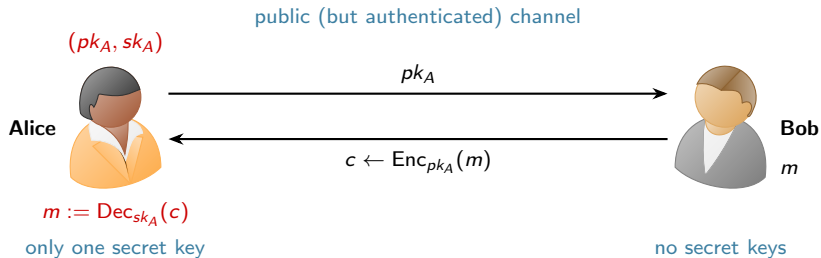
Why it works

(G, q, g, g^a, g^b) is exactly a DH transcript with $x = a, y = b$, whose key is g^{ab} . $c = 0$: $\hat{k}$ is the real key ($b = 0$). $c = 1$: $\hat{k}$ is a uniform group element ($b = 1$). So

$$\Pr [\text{KE}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] = \Pr [\text{DDH}_{\mathcal{B}, \mathcal{G}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n).$$

□

Public-Key Encryption



Setting

Alice publishes a **public key** pk_A and keeps her **secret key** sk_A . *Anyone* can send Alice a ciphertext $c \leftarrow Enc_{pk_A}(m)$; only Alice can decrypt it, using sk_A .

Public-Key Encryption (Syntax)

Definition 4 (Public-key encryption scheme)

A **public-key encryption scheme** is a triple of PPT algorithms $(\text{Gen}, \text{Enc}, \text{Dec})$:

- $\text{Gen}(1^n)$ outputs a key pair (pk, sk) .
- $\text{Enc}(pk, m)$ outputs a ciphertext c .
- $\text{Dec}(sk, c)$ outputs a message m (or $\perp$).

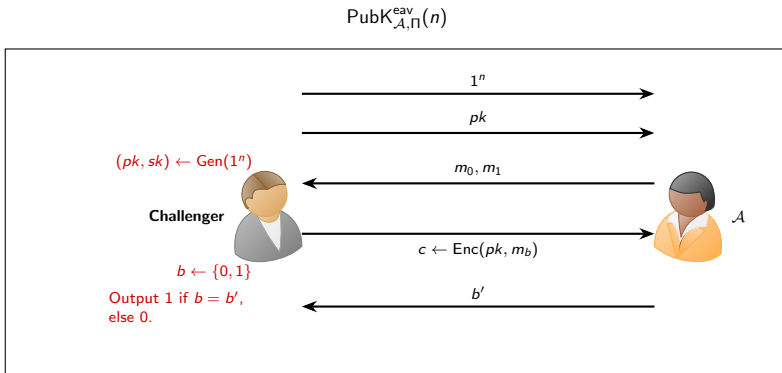
Correctness

For all (pk, sk) output by $\text{Gen}(1^n)$ and all (legal) m : $\text{Dec}(sk, \text{Enc}(pk, m)) = m$.

Security

EAV-security? CPA-security? CCA-security?

$\text{PubK}_{\mathcal{A}, \Pi}^{\text{eav}}$: Challenger vs. Adversary



Compare with $\text{PrivK}^{\text{eav}}$

The only change: $\mathcal{A}$ also gets the public key pk .

Game $\text{PubK}_{\mathcal{A}, \Pi}^{\text{eav}}(n)$

- 1 $(pk, sk) \leftarrow \text{Gen}(1^n)$. $\mathcal{A}$ is given 1^n and pk .
- 2 $\mathcal{A}$ outputs m_0, m_1 with $|m_0| = |m_1|$.
- 3 $b \leftarrow \{0, 1\}$, $c \leftarrow \text{Enc}(pk, m_b)$; c is given to $\mathcal{A}$, who outputs b' .
- 4 Output 1 if $b = b'$, and 0 otherwise.

Definition 5 (EAV-security)

$\Pi = (\text{Gen}, \text{Enc}, \text{Dec})$ is **EAV-secure** (indistinguishable in the presence of an eavesdropper) if for all PPT $\mathcal{A}$:

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

EAV Security vs. CPA Security

- In the **public-key** setting the two notions are **identical**.
- Why? Given pk , encryption can be performed by anyone, without any secret values: an eavesdropper can answer its own encryption-oracle queries.
- Hence encryption **must be randomized**: if $\text{Enc}(pk, \cdot)$ were deterministic, $\mathcal{A}$ would compute $\text{Enc}(pk, m_0)$ and compare it with c .

What About Security of Multiple Messages?

CPA-security implies security for encrypting **multiple messages** (as in the private-key setting).

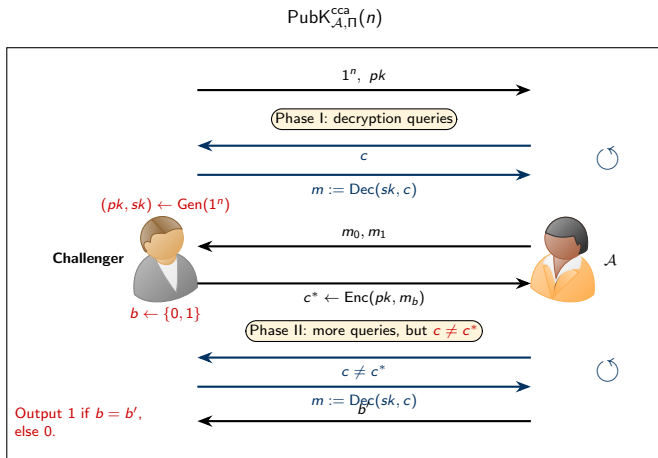
$$\text{Enc}(pk, m_1 \cdots m_t) := \text{Enc}(pk, m_1) \cdots \text{Enc}(pk, m_t).$$

- Proof via a direct **hybrid argument**, as for CPA $\Rightarrow$ mult-security in lecture 6. (Here $\mathcal{B}$ needs no oracle: it encrypts the other messages itself with pk .)

CCA Security (A Bigger Concern in the PKE Setting)

- An attacker can obtain **decryptions** of ciphertexts of its choice.
- An attacker can more easily come up with **illegitimate ciphertexts** (we cannot put a MAC on a ciphertext: the sender has no secret key).
- **Malleability**: given a ciphertext c of a message m , an attacker could obtain a ciphertext c' of a related message m' — *without knowing m' itself*.

PubK $_{\mathcal{A},\Pi}^{\text{cca}}$: Challenger vs. Adversary



Game $\text{PubK}_{\mathcal{A}, \Pi}^{\text{cca}}(n)$

- ① $(pk, sk) \leftarrow \text{Gen}(1^n)$. $\mathcal{A}$ is given 1^n , pk , and **oracle access** to $\text{Dec}(sk, \cdot)$.
- ② $\mathcal{A}$ outputs m_0, m_1 with $|m_0| = |m_1|$.
- ③ $b \leftarrow \{0, 1\}$, $c^* \leftarrow \text{Enc}(pk, m_b)$; c^* is given to $\mathcal{A}$, which may **not** query $\text{Dec}(sk, \cdot)$ on c^* .
- ④ $\mathcal{A}$ outputs b' . Output 1 if $b = b'$, and 0 otherwise.

Definition 6 (CCA-security)

Π is **CCA-secure** if for all PPT $\mathcal{A}$: $\Pr [\text{PubK}_{\mathcal{A}, \Pi}^{\text{cca}}(n) = 1] \leq \frac{1}{2} + \text{negl}(n)$.

Remark

Much harder to achieve in the PKE setting.

ElGamal Encryption

Gen(1^n)

$(G, q, g) \leftarrow \mathcal{G}(1^n)$; $x \leftarrow \mathbb{Z}_q$, $h := g^x$. Output $pk = (G, q, g, h)$ and $sk = x$.

Enc(pk, m), for $m \in G$

Sample $r \leftarrow \mathbb{Z}_q$; output
 $c = (c_1, c_2) := (g^r, m \cdot h^r)$.

Correctness?

$$\frac{c_2}{c_1^x} = \frac{m \cdot h^r}{(g^r)^x} = \frac{m \cdot g^{xr}}{g^{rx}} = m.$$

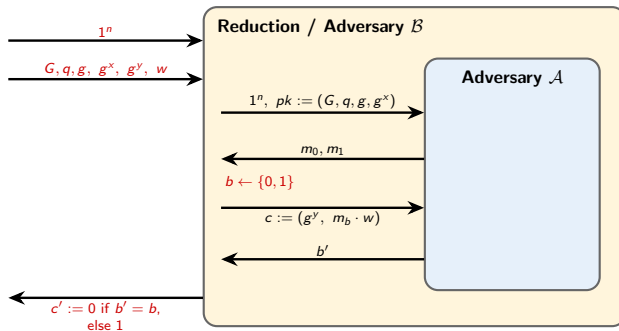
Dec(sk, c)

Parse $c = (c_1, c_2)$; output c_2/c_1^x .

Security?

Based on **DDH**: $(g^x, g^r, h^r = g^{xr})$ is a DH tuple, so h^r hides m like a one-time pad in G .

ElGamal Is EAV-Secure: Proof by Reduction



Given / To prove

Given: $\Pr[\text{PubK}_{\mathcal{A}, \Pi}^{\text{eav}}(n) = 1] \geq \frac{1}{2} + \epsilon(n)$ for non-negligible ϵ .

To prove: $\Pr[\text{DDH}_{\mathcal{B}, \mathcal{G}}(n) = 1] \geq \frac{1}{2} + \delta(n)$ for non-negligible δ .

ElGamal: Two Cases

Case $c = 0$: $w = g^{xy}$ (real)

$\mathcal{A}$'s view is exactly $\text{PubK}_{\mathcal{A}, \Pi}^{\text{eav}}(n)$:

$$\Pr[\mathcal{B} \text{ correct}] = \Pr[b' = b] \geq \frac{1}{2} + \epsilon(n).$$

pk has $sk = x$ uniform, and
 $(g^y, m_b \cdot g^{xy}) = (g^r, m_b \cdot h^r)$ with $r = y$
uniform: an honest encryption of m_b .

Case $c = 1$: $w = g^z$ (uniform)

$\mathcal{A}$ learns nothing about b :

$$\Pr[\mathcal{B} \text{ correct}] = \Pr[b' \neq b] = \frac{1}{2}.$$

w is uniform in G and independent of everything else, so $m_b \cdot w$ is uniform in G for either b — a one-time pad in the group.

ElGamal: Putting It Together

Success probability of $\mathcal{B}$

$$\begin{aligned}\Pr [\text{DDH}_{\mathcal{B}, \mathcal{G}}(n) = 1] &= \frac{1}{2} \Pr[\mathcal{B} \text{ correct} \mid c = 0] + \frac{1}{2} \Pr[\mathcal{B} \text{ correct} \mid c = 1] \\ &\geq \frac{1}{2} \left(\frac{1}{2} + \epsilon(n) \right) + \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{2} + \frac{\epsilon(n)}{2},\end{aligned}$$

contradicting the hardness of DDH. So ElGamal is EAV-secure, hence CPA-secure. $\square$

Remark (running time)

$\mathcal{B}$ runs $\mathcal{A}$ once and does one group multiplication: PPT.

Encrypting Long Messages

- Encrypting **block by block** is inefficient:
 - the ciphertext expands for each block;
 - public-key encryption is **2–3 orders of magnitude slower** than private-key.
- Anything better? Use public-key encryption only for a short **key**.

Hybrid Encryption

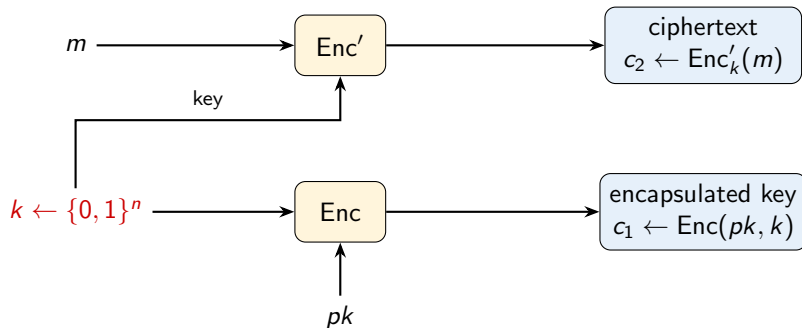
Idea

Use **public-key** encryption to set up a shared **secret key** k , which is then used to encrypt the message itself with fast **private-key** encryption.

Benefits

- The inefficiency of public-key encryption is not the bottleneck: we get **amortized efficiency** as the message grows large.
- The ciphertext expansion over the message is small.

Hybrid Encryption: The Construction



- **Decryption is natural:** $k := \text{Dec}(sk, c_1)$, then $m := \text{Dec}'_k(c_2)$.

Takeaway

The **functionality** of public-key encryption at the (asymptotic) **efficiency** of private-key encryption!

Summary

- **Cyclic groups:** prime-order subgroups $\langle g \rangle$ of $\mathbb{Z}_p^*$, e.g. $\langle 2 \rangle \subset \mathbb{Z}_{23}^*$ of order 11.
- **Hardness assumptions:** $\text{DLog} \Leftarrow \text{CDH} \Leftarrow \text{DDH}$.
- **Diffie–Hellman key exchange:** shared key g^{xy} , secure under DDH.
- **Public-key encryption** (Gen, Enc, Dec): anyone holding pk can encrypt; only sk decrypts.
- **Security notions:** $\text{PubK}^{\text{eav}} = \text{CPA}$ in the public-key setting, so encryption must be randomized; PubK^{cca} is much harder to achieve.
- **ElGamal:** $(g^r, m \cdot h^r)$. *Proved:* EAV-secure under DDH — with a real DH tuple $\mathcal{A}$ sees ElGamal, with a random one m_b is perfectly hidden.
- **Hybrid encryption:** encrypt a fresh key k with pk , the message with k : public-key functionality at private-key efficiency.