

Computation problems over cyclic groups of finite order.

Let G be a group generation algorithm.

G is a PPT algorithm.

On input n (security parameter), G will output a cyclic group G along with its "description", the order q , where $\|q\| = n$, and a generator $g \in G$.

Description of a cyclic group specifies how the group elements are represented. Typically, the group elements are represented as fixed length binary string.

- To work with the cyclic groups, there must be efficient (polynomial time) algorithms for the following:

- (a) "Computation with the group elements"
- (b) "Sampling uniformly a group element"

Therefore, to define computational problems over finite order cyclic groups, we need to specify three things.

- (i) Efficient group generation algorithm ' G '.
- (ii) Efficient algorithm for computation with group elements.
- (iii) Efficient algorithm for uniform sampling of group elements.

Discrete-Logarithm problem:

Let G be a cyclic group of order q .

Let $g \in G$ be a generator.

Then for any $h \in G$, $\exists$ a unique $x \in \mathbb{Z}_q$ s.t. $h = g^x$

$x := \log_g h$ is called the "discrete logarithm" of h base g .

$\log_g 1 = 0$ where 1 is the identity of G .

Discrete Logarithm problem:

$$\underline{\text{DLOG}}_{A, G}^{(n)}.$$

- ✓ 1. Run $G(1^n)$ to obtain (G, q, g) , where $|G|=q$, and $g \in G$ is a generator.
- ✓ 2. Choose $h \xleftarrow{\$} G$.
- 3. A is given (G, q, g, h) and output $x \in \mathbb{Z}_q$.
- ✓ 4. output 1 if $h = g^x$

Defn. We say DLOG is hard with respect to the group generation algorithm G if for every PPT algorithm A , $\exists$ a negligible fn. $\text{negl}(\cdot)$, such that.

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

Computational Diffie-Hellman problem:-

Given a finite cyclic group G of order q .
and two uniform group elements $h_1, h_2 \in G$.

We define

$$\text{DH}(h_1, h_2) := (h_1)^{\log h_2} = (h_2)^{\log h_1}$$

In other words, $\exists! x_1, x_2 \in \mathbb{Z}_q$ s.t. $h_1 = g^{x_1}$ and $h_2 = g^{x_2}$

$$\text{DH}(g^{x_1}, g^{x_2}) = g^{x_1 \cdot x_2}$$

Computational Diffie-Hellman (CDH).

$CDH_{A,G}(1^n)$

1. Run $G(1^n)$ to obtain (G, g, g)

2. $x_1, x_2 \xleftarrow{\$} \mathbb{Z}_q$. compute $h_1 = g^{x_1}, h_2 = g^{x_2}$.

3. A is given (G, g, g, h_1, h_2) & outputs z .

4. output 1 if $z = h_1^{x_2}$ or equivalently, $z = h_2^{x_1}$

Defn. We say that CDH problem is hard with respect to the group generation algorithm G , if for every PPT algorithm A , $\exists$ a negligible $\text{fn. } \text{negl}()$, such that

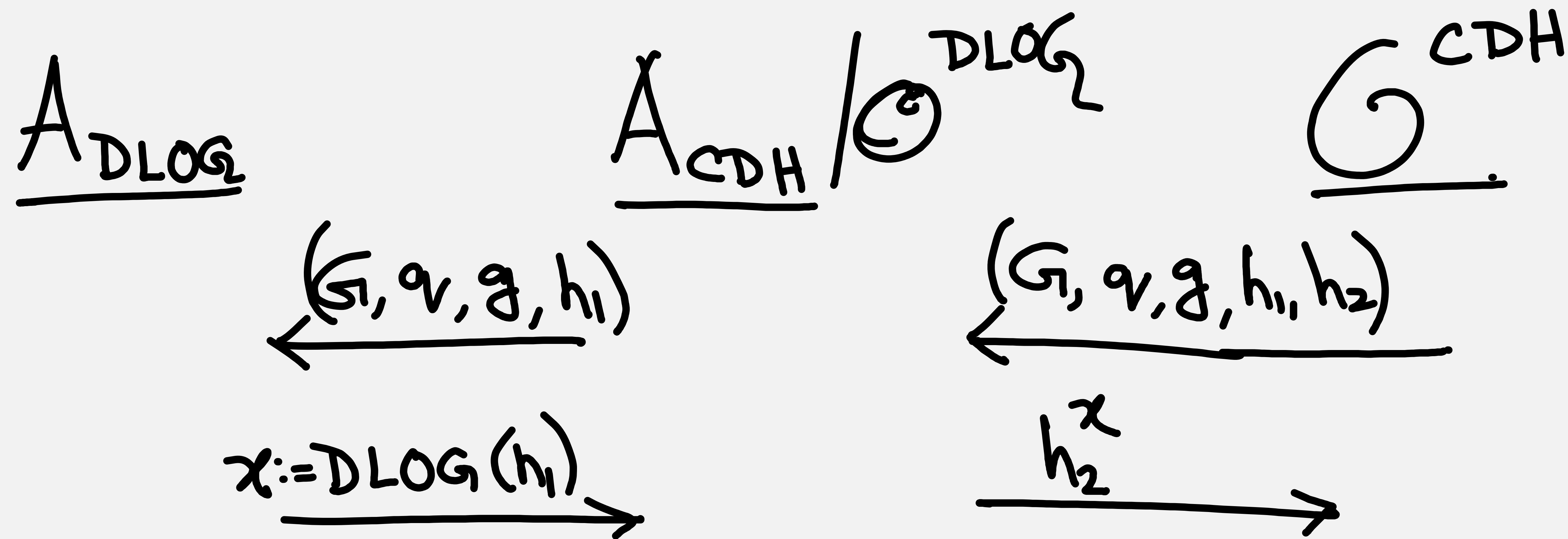
$$\Pr[CDH_{A,G}(1^n) = 1] \leq \text{negl}(n).$$

Lemma: If x_1, x_2 are chosen uniformly at random from $\mathbb{Z}_q$, then $h_1 := g^{x_1}, h_2 := g^{x_2}$ are also uniform over G .

Lemma:- CDH is hard in $G \Rightarrow \text{DLOG}_2$ is hard in G .

Pf: $\exists A_{\text{DLOG}_2}$ that can solve DLOG_2 with prob at least $\epsilon_{\text{DLOG}_2}^{(n)}$.

We need to construct an adversary A_{CDH} .



A_{DLOG} wins $\Rightarrow$

A_{CDH} wins

$A \Rightarrow B$

$\Pr[A] \leq \Pr[B]$

$$\Pr[\text{CDH}_{A_G}(1^n) = -1] = \Pr[A_{\text{CDH}} \text{ wins}] \geq \Pr[A_{\text{DLOG}} \text{ wins}] \geq \epsilon_{\text{DLOG}}$$

Decisional Diffie-Hellman problem:-

CDH: given $h_1 = g^{x_1}$, $h_2 = g^{x_2}$, compute $g^{x_1 x_2}$.

DDH: given $(g^{x_1}, g^{x_2}, g^{x_1 x_2})$ or (g^{x_1}, g^{x_2}, g^z) , where $z \leftarrow \mathbb{Z}_q$.

DDH_{D, G}(1ⁿ).

1. Run G on input 1^n to obtain (G, q, g) .
2. Choose $x_1 \leftarrow \mathbb{Z}_q$, $x_2 \leftarrow \mathbb{Z}_q$.
3. Compute $h_1 = g^{x_1}$, $h_2 = g^{x_2}$.
4. Sample a bit $b \leftarrow \{0, 1\}$.
5. if $b = 0$, then $h' = g^{x_1 x_2}$
if $b = 1$ then sample $z \leftarrow \mathbb{Z}_q$, compute $h' = g^z$.
6. D is given (G, q, g, h_1, h_2, h') and output $\hat{b}$.
7. output 1 if $\hat{b} = b$.

Defn. DDH problem is hard relative to the group generation algorithm G , if for every PPT distinguisher D , $\exists$ a negligible $\text{fn } \text{negl}()$, such that

$$\Pr[DDH_{D, G}(1^n) = 1] \leq \text{negl}(n)$$

$$\text{Adv}_G^{\text{DDH}}(D) = \left[\Pr[D(G, q, g, g^{x_1}, g^{x_2}, g^{x_1 x_2}) = 1] - \Pr[D(G, q, g, g^{x_1}, g^{x_2}, g^z) = 1] \right] \leq \text{negl}(n).$$

Lemma:- DDH problem is hard in $G \Rightarrow$ CDH problem is hard in G

Corollary:- DDH problem is hard in $G \Rightarrow$ DLOG _{G} is hard in G .

Importance of prime order group in cryptography:-

- Pohlig-Hellman algorithm, discrete log is easy to solve if the group order q has small prime factors.
- Easy to choose generator
- In the cryptographic proof, we often need the multiplicative inverse of the exponent.

$\mathbb{Z}_p^*$, where p is prime is a cyclic group.

Theorem: Let p and q be two primes such that $q \mid p-1$.

Then $G = \{h^r : h \in \mathbb{Z}_p^*\}$, where $(p-1) = q \cdot r$

- G is a subgroup of $\mathbb{Z}_p^*$.
- $|G| = q$.

Prf: To prove the second part,

$f_r: \mathbb{Z}_p^* \rightarrow G$ is a r-to-1 fn, defined as

$$f_r(h) = h^r \pmod{p}.$$

$$(g^i)^r = (g^j)^r, \quad i \in \{0, \dots, p-2\}, \quad j \in \{0, \dots, p-2\}.$$

for a fixed $j \in \{0, \dots, p-2\}$, i can be $\{j, j+q, j+2q, \dots, j+(r-1)q\}$.

$$\begin{aligned} i^r &\equiv j^r \pmod{p-1} \\ (p-1) \mid r(i-j) &\Rightarrow (p-1) = q \cdot r \\ q \mid (p-1) &\Rightarrow (p-1) = q \cdot r \\ q \cdot r \mid r(i-j) &\Rightarrow q \mid (i-j) \end{aligned}$$

Collision-Resistance Hash fn from DLOG Assumption.

- Run group generation algorithm G on input 1^n , to obtain (G, q, g) , where the group order q is prime.
- Then choose a uniform group element h from G .
- set $k = (G, q, g, h)$ as the key.

Define Hash fn $H: \mathbb{Z}_q \times \mathbb{Z}_q \rightarrow G$.

$$H_k(x_1, x_2) := g^{x_1} h^{x_2}$$

Thm: If DLOG_2 is hard in $\mathbb{G}$, then the constructed hash $\underline{H}$ is collision-resistant.

$$\exists (x_1, x_2) \neq (x'_1, x'_2) : H_k(x_1, x_2) = H_k(x'_1, x'_2).$$

$$\in \mathbb{Z}_q \times \mathbb{Z}_q \quad \in \mathbb{Z}_q \times \mathbb{Z}_q$$

Therefore, discrete log of h is $(x_1 - x'_1)(x'_2 - x_2)^{-1}$

$$g^{x_1} h^{x_2} = g^{x'_1} h^{x'_2}$$

$$\Rightarrow g^{x_1 - x'_1} = h^{x'_2} \cdot h^{-x_2}$$

$$\Rightarrow g^{x_1 - x'_1} = h^{(x'_2 - x_2)}$$

$$\Rightarrow g^{(x_1 - x'_1)(x'_2 - x_2)^{-1}} = h.$$

Exercise:

Define (Gen, H) as follows:-

Gen : on input 1^n , run $\text{GenRSA}(1^n)$ to obtain (N, e, d) .
select $y \leftarrow \mathbb{Z}_N^*$.

key $k = (N, e, y)$.

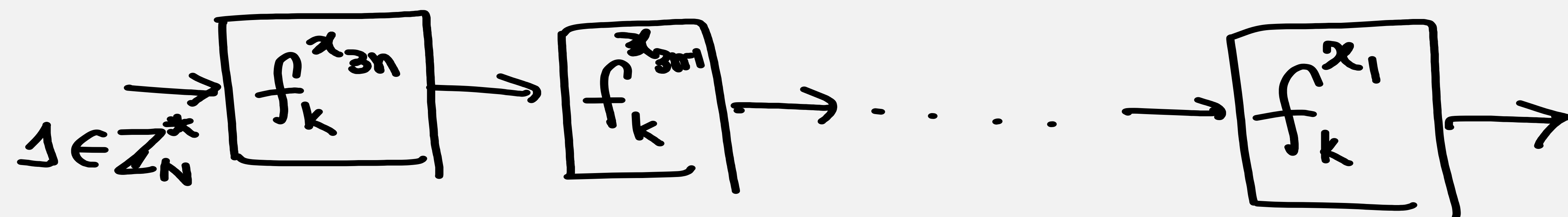
$H: \{0, 1\}^{3n} \rightarrow \mathbb{Z}_N^*$ as follows

Let $f_k^0(\hat{x}) := \hat{x}^e \bmod N$, where $f^b: \mathbb{Z}_N^* \rightarrow \mathbb{Z}_N^*$, for $b \in \{0, 1\}$.

" $f_k^1(\hat{x}) := y \hat{x}^e \bmod N$

Let $x = (x_1, \dots, x_{3n}) \in \{0, 1\}^{3n}$

$$H_k(x) = f_k^{x_1} \left(f_k^{x_2} \left(\dots (1) \right) \dots \right).$$



Thm:

If RSA is hard with respect to GenRSA , then the constructed hash f_{in} is collision-resistant.