Cryptography

Lecture 19

Arkady Yerukhimovich

November 4, 2024

Outline

1. Lecture 18 Review
2. Crypto Hardness Assumptions (Chapters 8.2, 8.3)
3. Assumptions in Cyclic Groups (Chapters 8.2, 8.3)

Lecture 18 Review

- The Group $\mathbb{Z}_N^*$
- Chinese Remainder Theorem
- Modular Arithmetic by Hand

Modular Arithmetic Without a Calculator

To evaluate exponentiation $\bmod N$ use the following steps:

- If N is not prime, apply the Chinese Remainder Theorem
- Reduce $\bmod \phi(N)$ in the exponent
- Reduce $\bmod N$ in the base

Useful Hints:

- Sometimes useful to use negative numbers
- look for things that are easy to compute (e.g., 1^{53})

Outline

1. Lecture 18 Review
2. Crypto Hardness Assumptions (Chapters 8.2, 8.3)
3. Assumptions in Cyclic Groups (Chapters 8.2, 8.3)

What Are Hardness Assumptions?

- As we've discussed before, all crypto primitives rely on computational hardness
- Thus, we need to assume that some problem is hard to compute
- We have seen such assumptions before: E.g., Existence of PRG, PRF, CRHF

What Are Hardness Assumptions?

- As we've discussed before, all crypto primitives rely on computational hardness
- Thus, we need to assume that some problem is hard to compute
- We have seen such assumptions before: E.g., Existence of PRG, PRF, CRHF
- Going forward, we will instead use hard problems from number theory and mathematics
  - Some of these problems have been studied for 1000s of years
  - Easy to state and widely understood
  - Still believed to be hard for all PPT machines

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

GenMod(1^n):

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

GenMod(1^n):

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\text{Factor}_{\mathcal{A},\text{GenMod}}(n)$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

GenMod(1^n):

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\text{Factor}_{\mathcal{A},\text{GenMod}}(n)$

- The challenger runs $(N, p, q) \leftarrow \text{GenMod}(1^n)$ and sends N to $\mathcal{A}$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

GenMod(1^n):

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\text{Factor}_{\mathcal{A},\text{GenMod}}(n)$

- The challenger runs $(N, p, q) \leftarrow \text{GenMod}(1^n)$ and sends N to $\mathcal{A}$
- $\mathcal{A}$ outputs two primes p', q'

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

GenMod(1^n):

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(n)$

- The challenger runs $(N, p, q) \leftarrow \mathsf{GenMod}(1^n)$ and sends N to $\mathcal{A}$
- $\mathcal{A}$ outputs two primes p', q'
- We say that $\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $p' \cdot q' = N$.

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

$\mathsf{GenMod}(1^n)$:

- Find n-bit primes p, q, compute $N = pq$
- Output (N, p, q)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(n)$

- The challenger runs $(N, p, q) \leftarrow \mathsf{GenMod}(1^n)$ and sends N to $\mathcal{A}$
- $\mathcal{A}$ outputs two primes p', q'
- We say that $\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $p' \cdot q' = N$.

Definition: Factoring is hard relative to GenMod if for all PPT $\mathcal{A}$ it holds that

$$\Pr[\mathsf{Factor}_{\mathcal{A},\mathsf{GenMod}}(n) = 1] \le \mathsf{negl}(n)$$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$
- No known way to factor in time polynomial in $||N||$ $= \log_2 N$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$
- No known way to factor in time polynomial in $||N||$
- Requires ability to efficiently sample n-bit primes

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$
- No known way to factor in time polynomial in $||N||$
- Requires ability to efficiently sample n-bit primes
    - Prime number theorem: The fraction of n-bit integers that are prime is at least $1/3n$

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$
- No known way to factor in time polynomial in $||N||$
- Requires ability to efficiently sample n-bit primes
  - Prime number theorem: The fraction of n-bit integers that are prime is at least $1/3n$
  - So, can sample integers at random, and test if they are prime

Factoring Assumption

Factoring Problem

Given $N = pq$ when p and q are n-bit primes, find p and q

Observations:

- Factoring can easily be done in $O(\sqrt{N})$ divisions – just try all numbers less than $\sqrt{N}$
- No known way to factor in time polynomial in $||N||$
- Requires ability to efficiently sample n-bit primes
  - Prime number theorem: The fraction of n-bit integers that are prime is at least $1/3n$
  - So, can sample integers at random, and test if they are prime
  - Miller-Rabin primality test – efficiently test if a number is prime

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

$f_e(x) = y = x^e \bmod N$

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

- Since $gcd(e, \phi(N)) = 1$, there is an integer $d = e^{-1} \bmod \phi(N)$

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

- Since $gcd(e, \phi(N)) = 1$, there is an integer $d = e^{-1} \bmod \phi(N)$
- The function $f_d(x) = x^d$ is also a permutation over $\mathbb{Z}_N^*$

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

- Since $gcd(e, \phi(N)) = 1$, there is an integer $d = e^{-1} \bmod \phi(N)$
- The function $f_d(x) = x^d$ is also a permutation over $\mathbb{Z}_N^*$
- Moreover, f_d is the inverse permutation of f_e

$$(x^e)^d = x^{[ed \bmod \phi(N)]} = x \bmod N$$

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

- Since $gcd(e, \phi(N)) = 1$, there is an integer $d = e^{-1} \bmod \phi(N)$
- The function $f_d(x) = x^d$ is also a permutation over $\mathbb{Z}_N^*$
- Moreover, f_d is the inverse permutation of f_e

$$(x^e)^d = x^{[ed \bmod \phi(N)]} = x \bmod N$$

- RSA problem is easy if know any of $d, \phi(N), p, q$

RSA Assumption (Rivest, Shamir, Adleman '77)

Given $N = pq$, Integer $e > 1$ s.t. $gcd(e, \phi(N)) = 1$, we know that $f_e(x) = x^e$ is a permutation over $\mathbb{Z}_N^*$

RSA Problem

Given (N, e) and $y \in \mathbb{Z}_N^*$, compute $[y^{1/e} \bmod N]$

Observations:

- Since $gcd(e, \phi(N)) = 1$, there is an integer $d = e^{-1} \bmod \phi(N)$
- The function $f_d(x) = x^d$ is also a permutation over $\mathbb{Z}_N^*$
- Moreover, f_d is the inverse permutation of f_e

$$(x^e)^d = x^{[ed \bmod \phi(N)]} = x \bmod N$$

- RSA problem is easy if know any of $d, \phi(N), p, q$
- RSA is potentially easier than factoring

RSA Assumption

GenRSA(1^n):

- $(N, p, q) \leftarrow$ GenMod(1^n), let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

RSA Assumption

GenRSA(1^n):

- $(N, p, q) \leftarrow \text{GenMod}(1^n)$, let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\text{RSAInv}_{\mathcal{A},\text{GenRSA}}(n)$

RSA Assumption

$\mathsf{GenRSA}(1^n)$:

- $(N, p, q) \leftarrow \mathsf{GenMod}(1^n)$, let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{RSAInv}_{\mathcal{A},\mathsf{GenRSA}}(n)$

- The challenger runs $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$, chooses $y \leftarrow \mathbb{Z}_N^*$, and sends (N, e, y) to $\mathcal{A}$

RSA Assumption

$\mathsf{GenRSA}(1^n)$:

- $(N, p, q) \leftarrow \mathsf{GenMod}(1^n)$, let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{RSAInv}_{\mathcal{A},\mathsf{GenRSA}}(n)$

- The challenger runs $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$, chooses $y \leftarrow \mathbb{Z}_N^*$, and sends (N, e, y) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_N^*$

RSA Assumption

GenRSA(1^n):

- $(N, p, q) \leftarrow$ GenMod(1^n), let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

RSAInv$_{\mathcal{A},\text{GenRSA}}(n)$

- The challenger runs $(N, e, d) \leftarrow$ GenRSA(1^n), chooses $y \leftarrow \mathbb{Z}_N^*$, and sends (N, e, y) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_N^*$
- We say that RSAInv$_{\mathcal{A},\text{GenRSA}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $x^e = y \bmod N$.

RSA Assumption

$\mathsf{GenRSA}(1^n)$:

- $(N, p, q) \leftarrow \mathsf{GenMod}(1^n)$, let $\phi(N) = (p-1)(q-1)$
- Choose $e > 1$ s.t. $gcd(e, \phi(N)) = 1$
- Compute $d = [e^{-1} \bmod \phi(N)]$
- Output (N, e, d)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{RSAInv}_{\mathcal{A},\mathsf{GenRSA}}(n)$

- The challenger runs $(N, e, d) \leftarrow \mathsf{GenRSA}(1^n)$, chooses $y \leftarrow \mathbb{Z}_N^*$, and sends (N, e, y) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_N^*$
- We say that $\mathsf{RSAInv}_{\mathcal{A},\mathsf{GenRSA}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $x^e = y \bmod N$.

Definition: RSA is hard relative to GenRSA if for all PPT $\mathcal{A}$ it holds that

$$\Pr[\mathsf{RSAInv}_{\mathcal{A},\mathsf{GenRSA}}(n) = 1] \leq \mathsf{negl}(n)$$

Outline

1. Lecture 18 Review
2. Crypto Hardness Assumptions (Chapters 8.2, 8.3)
3. Assumptions in Cyclic Groups (Chapters 8.2, 8.3)

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$G = \{g^0, g^1, g^2, g^3, \ldots, g^{|G|-1}\}$ $\quad |G| = q$

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

Gen(1^n):

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$\mathsf{Gen}(1^n)$:

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n)$

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$\mathsf{Gen}(1^n)$:

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $h \leftarrow G$, sends (G, q, g, h) to $\mathcal{A}$

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$\mathsf{Gen}(1^n)$:

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $h \leftarrow G$, sends (G, q, g, h) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_q$

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$\mathsf{Gen}(1^n)$:

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $h \leftarrow G$, sends (G, q, g, h) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_q$
- We say that $\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $g^x = h$.

Discrete Log Assumption

Let G be a cyclic group of order q with generator g

Discrete Log Problem

Given $h \in G$, find $0 \leq x \leq q-1$ s.t. $g^x = h$. We say $x = \log_g h$

$\mathsf{Gen}(1^n)$:

- Pick n-bit prime q, group G of order q, generator g. Output (G, q, g)

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

$\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $h \leftarrow G$, sends (G, q, g, h) to $\mathcal{A}$
- $\mathcal{A}$ outputs $x \in \mathbb{Z}_q$
- We say that $\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $g^x = h$.

Definition: DLog is hard relative to Gen if for all PPT $\mathcal{A}$ it holds that

$$\Pr[\mathsf{DLog}_{\mathcal{A},\mathsf{Gen}}(n) = 1] \leq \mathsf{negl}(n)$$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

$h_1 = g^x$

$h_1 = g^y$

$g^x \cdot g^y = g^{x+y}$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Computation Diffie-Hellman: $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n)$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Computation Diffie-Hellman: $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, sends $(G, q, g, h_1 = g^x, h_2 = g^y)$ to $\mathcal{A}$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Computation Diffie-Hellman: $\mathrm{CDH}_{\mathcal{A},\mathrm{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathrm{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, sends $(G, q, g, h_1 = g^x, h_2 = g^y)$ to $\mathcal{A}$
- $\mathcal{A}$ outputs $h_3 \in G$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Computation Diffie-Hellman: $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, sends $(G, q, g, h_1 = g^x, h_2 = g^y)$ to $\mathcal{A}$
- $\mathcal{A}$ outputs $h_3 \in G$
- We say that $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $h_3 = g^{xy}$.

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 1: Computational Diffie-Hellman

- Hard to find g^{xy}

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Computation Diffie-Hellman: $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, sends $(G, q, g, h_1 = g^x, h_2 = g^y)$ to $\mathcal{A}$
- $\mathcal{A}$ outputs $h_3 \in G$
- We say that $\mathsf{CDH}_{\mathcal{A},\mathsf{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $h_3 = g^{xy}$.

Definition: CDH is hard if for all PPT $\mathcal{A}$: $\Pr[\mathcal{A} \text{ wins}] \leq \mathsf{negl}(n)$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n)$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, $b \leftarrow \{0, 1\}$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, $b \leftarrow \{0, 1\}$
- If $b = 0$, send $(G, q, g, g^x, g^y, g^{xy})$ to $\mathcal{A}$.
  If $b = 1$, choose $z \leftarrow \mathbb{Z}_q$, and send (G, q, g, g^x, g^y, g^z) to $\mathcal{A}$

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, $b \leftarrow \{0, 1\}$
- If $b = 0$, send $(G, q, g, g^x, g^y, g^{xy})$ to $\mathcal{A}$.
  If $b = 1$, choose $z \leftarrow \mathbb{Z}_q$, and send (G, q, g, g^x, g^y, g^z) to $\mathcal{A}$
- $\mathcal{A}$ outputs bit b'

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathsf{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, $b \leftarrow \{0, 1\}$
- If $b = 0$, send $(G, q, g, g^x, g^y, g^{xy})$ to $\mathcal{A}$.
  If $b = 1$, choose $z \leftarrow \mathbb{Z}_q$, and send (G, q, g, g^x, g^y, g^z) to $\mathcal{A}$
- $\mathcal{A}$ outputs bit b'
- We say that $\mathsf{DDH}_{\mathcal{A},\mathsf{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $b = b'$.

Diffie-Hellman Assumptions

Let G be a cyclic group of order q with generator g

Diffie-Hellman Problem

Given $h_1 = g^x$, $h_2 = g^y$, find g^{xy}

Variant 2: Decisional Diffie-Hellman

- Hard to distinguish g^{xy} from a random element in G

Consider the following game between an adversary $\mathcal{A}$ and a challenger:

Decisional Diffie-Hellman: $\mathrm{DDH}_{\mathcal{A},\mathrm{Gen}}(n)$

- Challenger runs $(G, q, g) \leftarrow \mathrm{Gen}(1^n)$, $x, y \leftarrow \mathbb{Z}_q$, $b \leftarrow \{0, 1\}$
- If $b = 0$, send $(G, q, g, g^x, g^y, g^{xy})$ to $\mathcal{A}$.
  If $b = 1$, choose $z \leftarrow \mathbb{Z}_q$, and send (G, q, g, g^x, g^y, g^z) to $\mathcal{A}$
- $\mathcal{A}$ outputs bit b'
- We say that $\mathrm{DDH}_{\mathcal{A},\mathrm{Gen}}(n) = 1$ (i.e., $\mathcal{A}$ wins) if $b = b'$.

Definition: DDH is hard if for all PPT $\mathcal{A}$: $\Pr[\mathcal{A} \text{ wins}] \leq 1/2 + \mathrm{negl}(n)$

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

1. DLog vs. CDH

DLog: Given g^x find x

CDH: Given g^x, g^y find g^{xy}

x $(g^y)^x = g^{xy}$

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

1. DLog vs. CDH
   - If $\mathcal{A}$ can solve DLog, then given g^x, g^y, he can find x, y and compute g^{xy} (solve CDH)
   - The reverse direction doesn't seem true

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

1. DLog vs. CDH
   - If $\mathcal{A}$ can solve DLog, then given g^x, g^y, he can find x, y and compute g^{xy} (solve CDH)
   - The reverse direction doesn't seem true
2. CDH vs. DDH

CDH: $g^x, g^y \rightarrow g^{xy}$

DDH: g^x, g^y g^{xy} look random

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

1. DLog vs. CDH
   - If $\mathcal{A}$ can solve DLog, then given g^x, g^y, he can find x, y and compute g^{xy} (solve CDH)
   - The reverse direction doesn't seem true
2. CDH vs. DDH
   - If $\mathcal{A}$ given g^x, g^y can find g^{xy}, then he can distinguish this from g^z for a random $z \leftarrow \mathbb{Z}_q$ (solve DDH)
   - The reverse direction doesn't seem true

Relationship Between Cyclic Group Assumptions

Relative Hardness of The Problems

$DLog > CDH > DDH$

1. DLog vs. CDH
   - If $\mathcal{A}$ can solve DLog, then given g^x, g^y, he can find x, y and compute g^{xy} (solve CDH)
   - The reverse direction doesn't seem true
2. CDH vs. DDH
   - If $\mathcal{A}$ given g^x, g^y can find g^{xy}, then he can distinguish this from g^z for a random $z \leftarrow \mathbb{Z}_q$ (solve DDH)
   - The reverse direction doesn't seem true

Strength of Assumption

Since DDH is the easiest problem, assuming it is secure is the strongest assumption

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.
- DDH is easy in composite-order groups!!!

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.
- DDH is easy in composite-order groups!!!

Group of Quadratic Residues $\mod p$

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.
- DDH is easy in composite-order groups!!!

Group of Quadratic Residues $\mod p$

Let $p = 2q + 1$ with both p and q prime.

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p-1$

- $p-1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

x^2, y^2

$x^2 \cdot y^2 = (x \cdot y)^2$

$|\mathbb{Z}_p^*| = p-1 = 2q$

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.
- DDH is easy in composite-order groups!!!

Group of Quadratic Residues mod p

Let $p = 2q + 1$ with both p and q prime.

$$G = \{[h^2 \bmod p] | h \in \mathbb{Z}_p^*\}$$

is a group of prime order q

Preference for Prime-Order Groups

Note that $\mathbb{Z}_p^*$ for p prime, $p > 2$, has order $p - 1$

- $p - 1$ is not prime
- So $G = \mathbb{Z}_p^*$ is not a prime-order group

Reasons to prefer prime-order groups:

- Easy to find a generator, all $g \in G$ are generators.
- DDH is easy in composite-order groups!!!

Given

g, $h = g^x$ find x

$$g^x = \overbrace{g + g + g + \cdots + g}^{x} = gx$$

Group of Quadratic Residues mod p

Let $p = 2q + 1$ with both p and q prime.

$$G = \{[h^2 \bmod p] | h \in \mathbb{Z}_p^*\}$$

is a group of prime order q