CSE 107: Introduction to Modern Cryptography

Nadia Heninger

UCSD

Winter 2025 Lecture 12

Last time: Number theory review

This time: Algebraic groups, modular exponentiation, discrete log, Diffie-Hellman

Groups

Group: $(S, \circ)$ with S a set and $\circ$ an operation

G is a group if:

- closed under operation $\circ$
- identity: $\exists e \in G$ s.t. $e \circ g = g = g \circ e \; \forall g \in G$
- inverses: $\forall g \in G \quad \exists h \in G$ s.t. $g \circ h = e = h \circ g$
- associative: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$

Groups

Group: $(S, \circ)$ with S a set and $\circ$ an operation

G is a group if:

- closed under operation $\circ$
- identity: $\exists e \in G$ s.t. $e \circ g = g = g \circ e \; \forall g \in G$
- inverses: $\forall g \in G \quad \exists h \in G$ s.t. $g \circ h = e = h \circ g$
- associative: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$

Abelian group:

- commutative: $\forall g, h : \quad g \circ h = h \circ g$

Groups

Group: $(S, \circ)$ with S a set and $\circ$ an operation

G is a group if:

- closed under operation $\circ$
- identity: $\exists e \in G$ s.t. $e \circ g = g = g \circ e \; \forall g \in G$
- inverses: $\forall g \in G \quad \exists h \in G$ s.t. $g \circ h = e = h \circ g$
- associative: $(g_1 \circ g_2) \circ g_3 = g_1 \circ (g_2 \circ g_3)$

Abelian group:

- commutative: $\forall g, h : \quad g \circ h = h \circ g$

Cyclic group:

- $G = (\langle a \rangle, \circ)$ (G is generated by one element)

Examples of groups

- $\mathbb{Z}$ abelian group with $+$
  identity $= 0$, inverse $= -g$, cyclic, generated by 1

Examples of groups

- $\mathbb{Z}$ abelian group with $+$
  identity $= 0$, inverse $= -g$, cyclic, generated by 1
- $\mathbb{Z}$ not a group with $\times$

Examples of groups

- $\mathbb{Z}$ abelian group with $+$
  identity $= 0$, inverse $= -g$, cyclic, generated by 1
- $\mathbb{Z}$ not a group with $\times$
- ($\mathbb{Z}$ mod $N, +$) is a group
  cyclic, generated by 1

Examples of groups

- $\mathbb{Z}$ abelian group with $+$
  identity $= 0$, inverse $= -g$, cyclic, generated by 1
- $\mathbb{Z}$ not a group with $\times$
- ($\mathbb{Z}$ mod $N, +$) is a group
  cyclic, generated by 1
- ($\mathbb{Z}$ mod $N, \times$) is not a group

Examples of groups

- $\mathbb{Z}$ abelian group with $+$
  identity $= 0$, inverse $= -g$, cyclic, generated by 1
- $\mathbb{Z}$ not a group with $\times$
- $(\mathbb{Z} \bmod N, +)$ is a group
  cyclic, generated by 1
- $(\mathbb{Z} \bmod N, \times)$ is not a group
- $(\{1, 2, \ldots, p-1\} \bmod p, \times)$ is a group if p prime
  "multiplicative group mod p" $\mathbb{Z}_p^*$

Multiplicative group mod p

$\mathbb{Z}_p^* = (\{1, 2, \ldots, p-1\} \bmod p, \times)$ is a group if p is prime

Is cyclic: $\exists a$ s.t. $G = \langle a \rangle = \{a, a^2, a^3, \ldots, a^{p-1}\}$.

Not every $a \in G$ generates G: $\langle g \rangle$ might be a subgroup of G.

Group orders in $\mathbb{Z}_p^*$

Group orders:

- $|G|$ is called the *order* of the group
- The order of an element g is $|\langle g \rangle|$

Theorem (Lagrange)

order$(g) \mid p-1$

Example:

2^0	2^1	2^2	2^3	2^4	2^5	2^6	
1	2	4	1	2	4	1	mod7
3^0	3^1	3^2	3^3	3^4	3^5	3^6	
1	3	2	6	4	5	1	mod7

$\exists$ efficent p.p.t alg. to find generator if factorization of $p-1$ is known

Fermat's little theorem

Theorem

G an abelian group with $|G| = m \implies g^m = 1 \; \forall g \in G$

Fermat's little theorem

Theorem

G an abelian group with $|G| = m \implies g^m = 1 \; \forall g \in G$

Corollary (Fermat's little theorem)

p prime, $a^{p-1} \equiv 1 \bmod p$

Computations with modular arithmetic

Efficient:

- Addition
- Subtraction
- Modular reduction
- Multiplication
- Multiplicative inverse (using GCD algorithm)
- Modular exponentiation

Reminder: For numerical algorithms, "efficient" means polynomial time in the number of bits of the input.

For cryptographic algorithms, we typically use numbers that are 256 to 4096 bits long.

Running time that is linear or quadratic in 256: feasible.

Running time that is linear in 2^{256}: not feasible.

Efficient modular exponentiation

Inefficient exponentiation:

$g^a = g \cdot g \ldots g$ $\quad$ a times: not poly-time in $\lg a$

Efficient exponentiation: Square and multiply (left-to-right)

Input: base b, exponent a, modulus m

Output: $b^a \bmod m$

Algorithm:

result = 1
for i from $\ell \ldots 0$ $\quad$ (a has ℓ bits)
  result = result^2 mod m
  if $a[i] = 1$: $\quad$ (bit i of a is 1)
    result = result $\cdot\, b$ mod m
return result

Discrete log

"Inverse operation" for modular exponentiation.

No general-purpose efficient algorithms.

Discrete log
Given y, g find x s.t. $g^x \equiv y \bmod p$

Current discrete log record mod p: 795 bits.

Current factoring record: 830 bits.

Best algorithm: Number field sieve, subexponential time.

Computations with modular arithmetic

Efficient:

- Addition
- Subtraction
- Multiplication
- Modular reduction
- Multiplicative inverse (using GCD algorithm)
- Modular exponentiation

Not known to be efficient:

- Discrete logarithm (for some groups)

"We stand today on the brink of a revolution in cryptography."

— Diffie and Hellman, 1976

New Directions in Cryptography

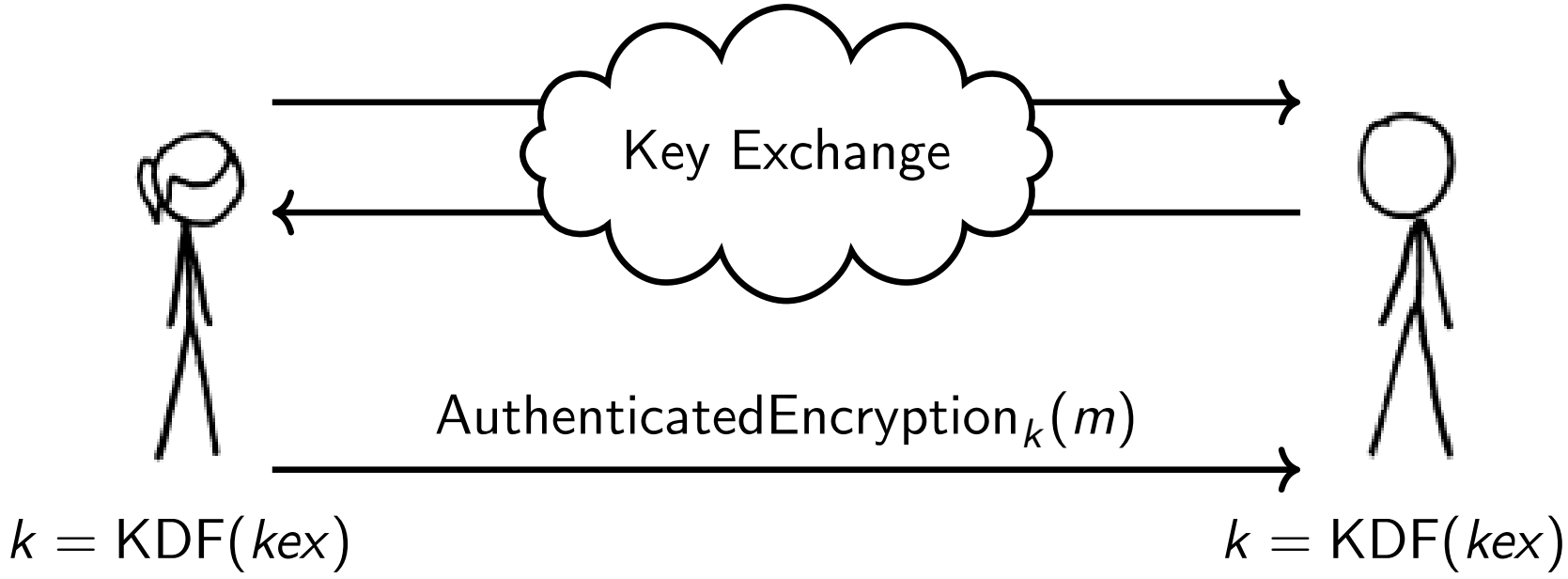
Invited Paper

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

The symmetric cryptography we just covered

Public key crypto idea # 1: Key exchange

Solving key distribution without trusted third parties



1. Alice, Bob exchange messages.
2. They derive a shared secret from the messages and use this to derive symmetric keys.
3. Passive eavesdropper can't recover shared secret from exchange messages.

Public key exchange

Symmetric cryptography existed for thousands of years.

In 1976 Diffie and Hellman wrote their paper outlining ideas behind public key cryptography.

They proposed an algorithm to solve the *key exchange* problem: How can Alice and Bob agree on a shared secret key over an insecure channel without the eavesdropper learning their secret key?

Textbook Diffie-Hellman

[Diffie Hellman 1976]

Public Parameters

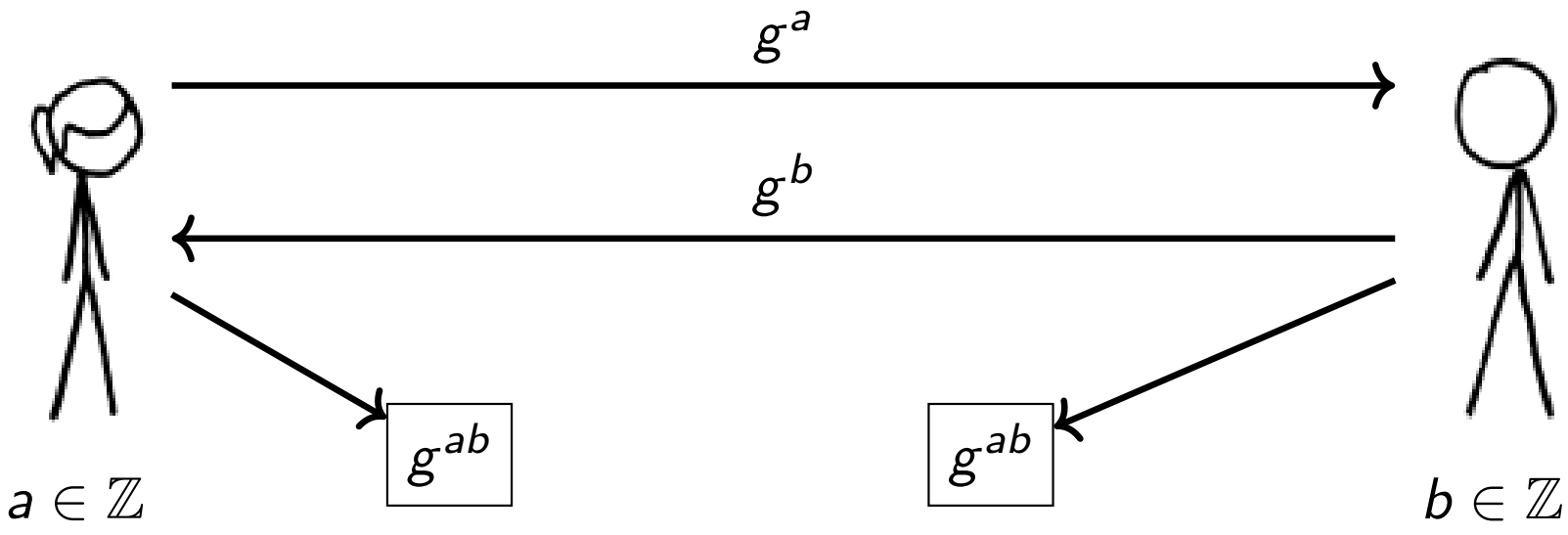
G a cyclic group

g group generator

Group desired properties:

- Efficient exponentiation
- Hard discrete log
- Commutativity in exponent

Key Exchange



Prime Field Diffie-Hellman

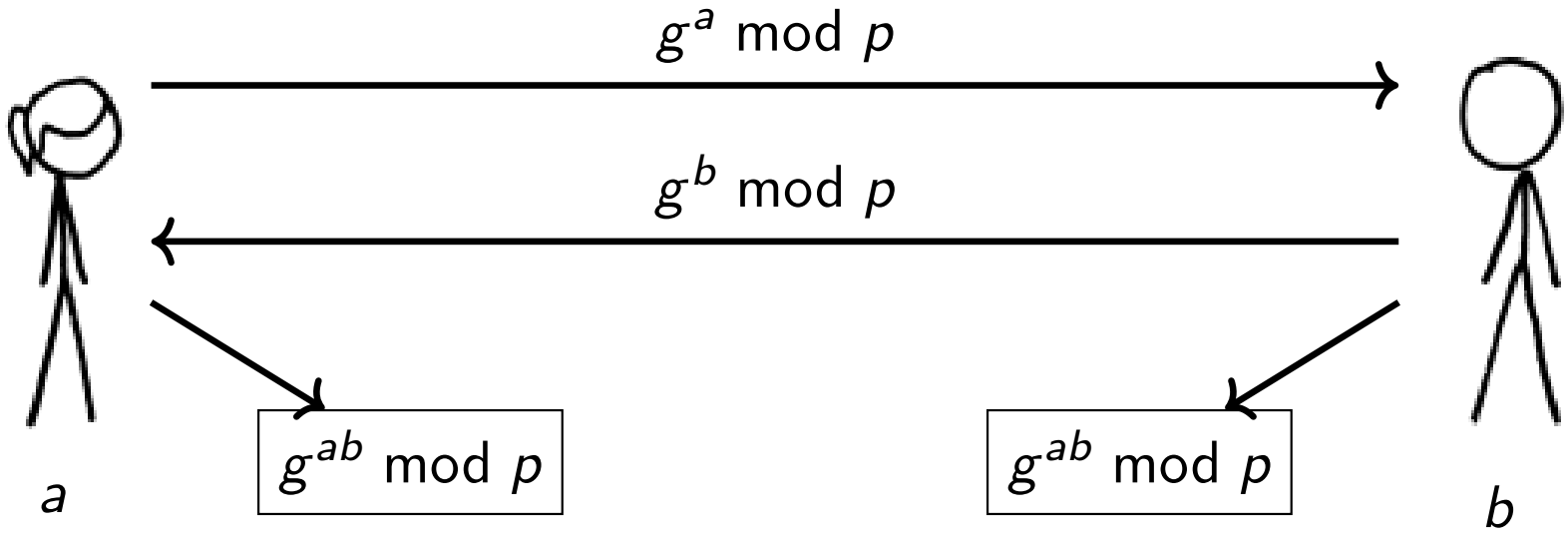
Public Parameters

p a prime

q a subgroup order; $q \mid (p-1)$

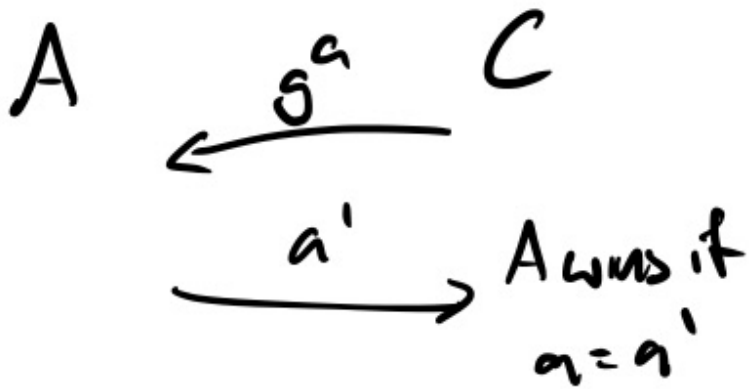
$g \in \mathbb{F}_p^*$ a generator of subgroup of order q

Key Exchange



The discrete log problem

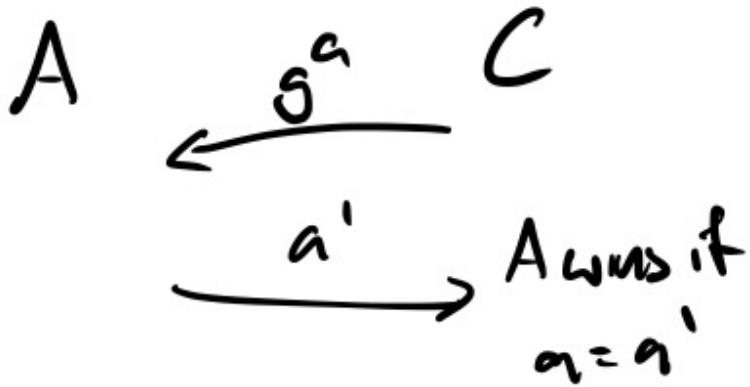
G a cyclic group with generator g.



Discrete log assumption:
Pr[A wins] is negligible.

The discrete log problem

G a cyclic group with generator g.



Discrete log assumption:
Pr[A wins] is negligible.

- Discrete log is easy $\implies$ Diffie-Hellman is easy to break.
  (Compute a and then compute public value.)
- But! Diffie-Hellman easy to break $\overset{?}{\implies}$ discrete log is easy.
- Discrete log is in NP and coNP $\implies$ not NP-complete
- Shor's algorithm solves discrete log with a quantum computer in polynomial time.

DiscreteLog as a formal game

Let $G = \langle g \rangle$ be a cyclic group of order m, and A an adversary.

Game $\mathrm{DL}_{G,g}$

procedure Initialize
$x \xleftarrow{\$} \mathbf{Z}_m$; $X \leftarrow g^x$
return X

procedure Finalize(x')
return $(x = x')$

The dl-advantage of A is

$$\mathbf{Adv}^{\mathrm{dl}}_{G,g}(A) = \Pr\left[\mathrm{DL}^{A}_{G,g} \Rightarrow \mathsf{true}\right]$$

Computational Diffie-Hellman Problem

G a cyclic group with generator g.

Let $G = \langle g \rangle$ be a cyclic group of order m, and A an adversary.

Game $\mathrm{CDH}_{G,g}$

procedure Initialize
$x, y \xleftarrow{\$} \mathbf{Z}_m$
$X \leftarrow g^x$; $Y \leftarrow g^y$
return X, Y

procedure Finalize(Z)
return $(Z = g^{xy})$

CDH assumption:
Pr[A wins] is negligible.

The cdh-advantage of A is

$$\mathbf{Adv}^{\mathrm{cdh}}_{G,g}(A) = \Pr\left[\mathrm{CDH}^{A}_{G,g} \Rightarrow \mathsf{true}\right]$$

- Equivalent to computing the shared Diffie-Hellman secret.
- Obvious algorithm: A computes discrete log of g^b, returns $(g^a)^b$.
- There could be some algorithm to solve that isn't discrete log: no such algorithm is known after 50 years.
- It is hard to even *verify* a correct solution (g^a, g^b, g^{ab}).

CDH Formally

Let $G = \langle g \rangle$ be a cyclic group of order m, and A an adversary.

Game $\mathrm{CDH}_{G,g}$

procedure Initialize
$x, y \xleftarrow{\$} \mathbf{Z}_m$
$X \leftarrow g^x$; $Y \leftarrow g^y$
return X, Y

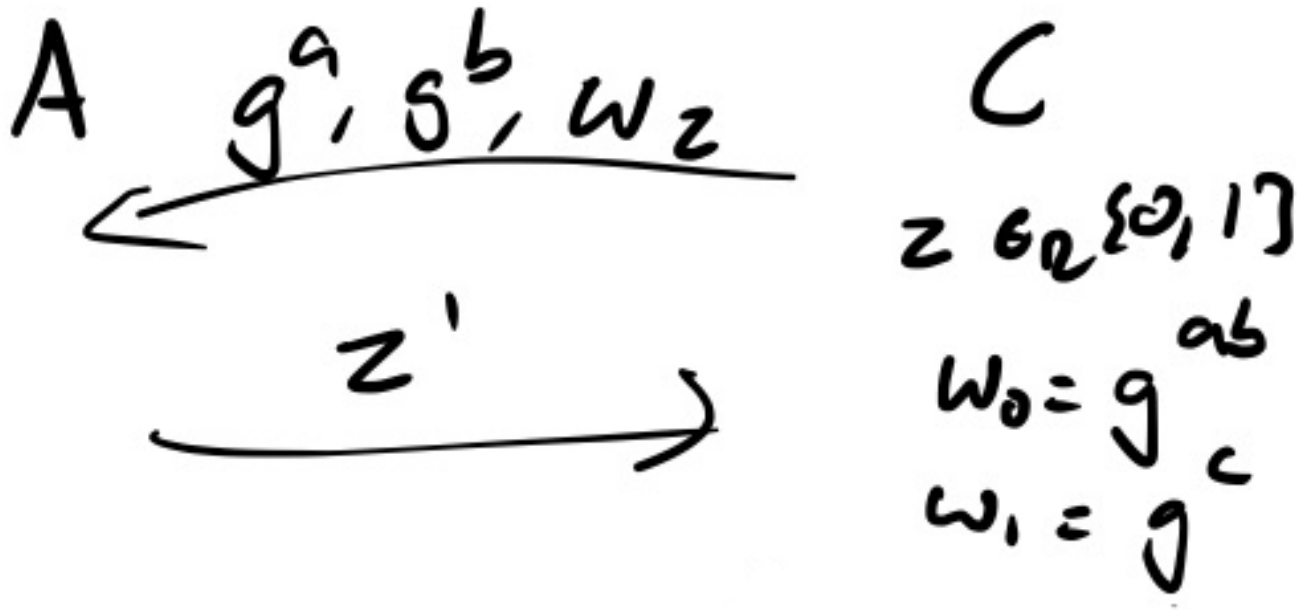
procedure Finalize(Z)
return $(Z = g^{xy})$

The cdh-advantage of A is

$$\mathbf{Adv}^{\mathrm{cdh}}_{G,g}(A) = \Pr\left[\mathrm{CDH}^{A}_{G,g} \Rightarrow \mathsf{true}\right]$$

Decisional Diffie-Hellman Assumption

G a cyclic group with generator g.



DDH assumption:
$|\Pr[A = 1 \mid z = 0] - \Pr[A = 1 \mid z = 1]|$ negligible.

- DDH hard $\implies$ difficult to distinguish Diffie-Hellman triples (g^a, g^b, g^{ab}) from random group elements (g^a, g^b, g^c).
- DDH hard $\implies$ CDH hard
- Assumption used for most security proofs.

Secure group choices for Diffie-Hellman

Why do we think discrete log is hard? Because there are groups for which there are no "good" algorithms known to compute them.

Group parameters are designed to avoid known cryptanalytic attacks.

Discrete log remains the best way to break Diffie-Hellman and related cryptosystems.

Types of groups used for Diffie-Hellman in practice now:

- "Prime-field Diffie-Hellman": Subgroups of $\mathbb{Z}_p^*$.
- "Elliptic curve Diffie-Hellman": Elliptic curve groups