Radboud University

Diffie-Hellman key agreement and ElGamal encryption

Cryptography, Autumn 2021

Lecturers: J. Daemen, B. Mennink

November 16, 2021

Institute for Computing and Information Sciences
Radboud University

Outline

Merkle-Diffie-Hellman key agreement

ElGamal encryption

Discrete log crypto security notions

Conclusions

Merkle-Diffie-Hellman key agreement

Ralph Merkle, Martin Hellman, Whitfield Diffie

Invented public-key cryptography in 1976!

Merkle-Diffie-Hellman key agreement

It does **public-key based establishment of a shared secret**

- It is a discrete-log based cryptosystem
- Key agreement:
  - Alice and Bob exchange information over a public channel
  - after the protocol they share a secret
- They can then derive from this secret one or more symmetric-cryptography keys
- . . . and use these keys to secure their communication
- There are several *flavors*

Discrete-log based cryptography: key material

- Domain parameters: specification of cyclic group we work in
  - non-secret information that is common to all users, here:
    - $p \in \mathbb{N}$: prime modulus
    - $g \in (\mathbb{Z}/p\mathbb{Z})^*$: generator (and its order q)
  - one always takes g with large prime order $\mathrm{ord}(g) = q$

    Note: q divides $p-1$ (due to Lagrange) so $\langle g \rangle \neq (\mathbb{Z}/p\mathbb{Z})^*$
- Users that participate in protocols make use of key pairs.
- E.g., Alice's key pair:
  - private key PrK that she keeps for herself: $a \in \mathbb{Z}/q\mathbb{Z}$
  - public key PK that she makes public: $A = g^a \in \langle g \rangle$

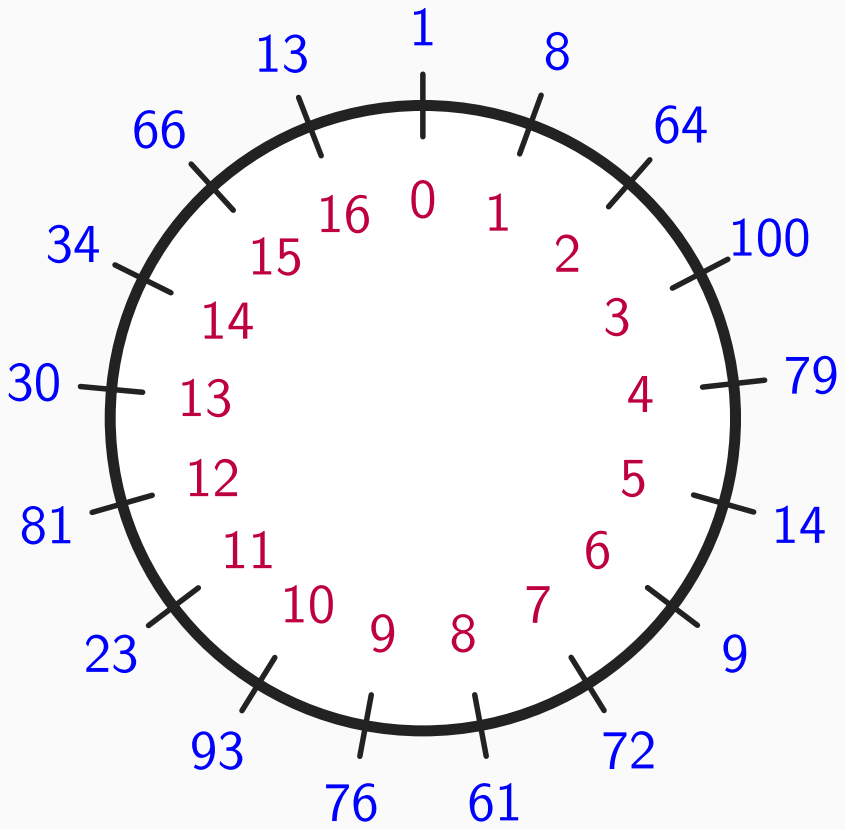
Key pair generation in discrete-log based crypto

(1) Random selection of the private key: $a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$

(2) Computation of the public key: $A \leftarrow g^a$

Note: both make use of the domain parameters

Toy example with prime order q: $p = 103, g = 8$

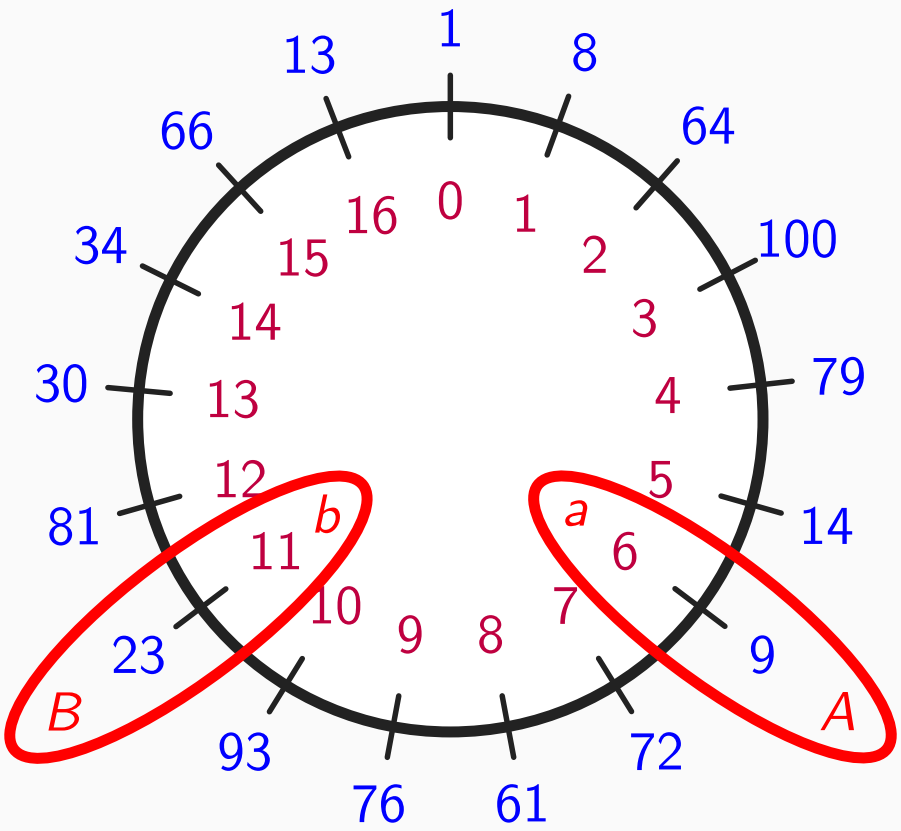


q, the order of 8 in $\mathbb{Z}/103\mathbb{Z}$, is 17

For 128 bits of security, NIST recommends $p \geq 2^{3072}$ and $q \geq 2^{256}$

. . . due to discrete log solving algorithms that we will discuss later

Key pairs in our toy example



Textbook (Merkle-)Diffie-Hellman key agreement

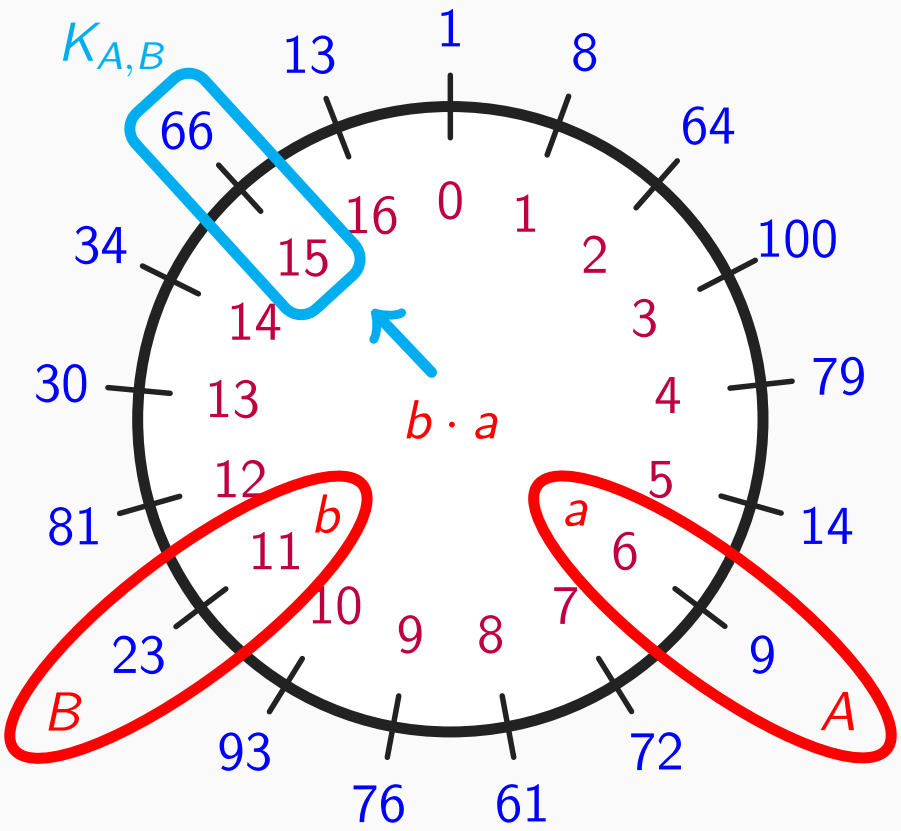
	Alice		Bob
have in advance:	p, g, q		p, g, q
	$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		$b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$
	$A \leftarrow g^a$		$B \leftarrow g^b$
		$\xrightarrow{\text{Alice},A}$	
		$\xleftarrow{\text{Bob},B}$	
	$K_{A,B} \leftarrow B^a$		$K_{B,A} \leftarrow A^b$

Alice and Bob arrive at the same shared secret $K_{A,B} = K_{B,A}$

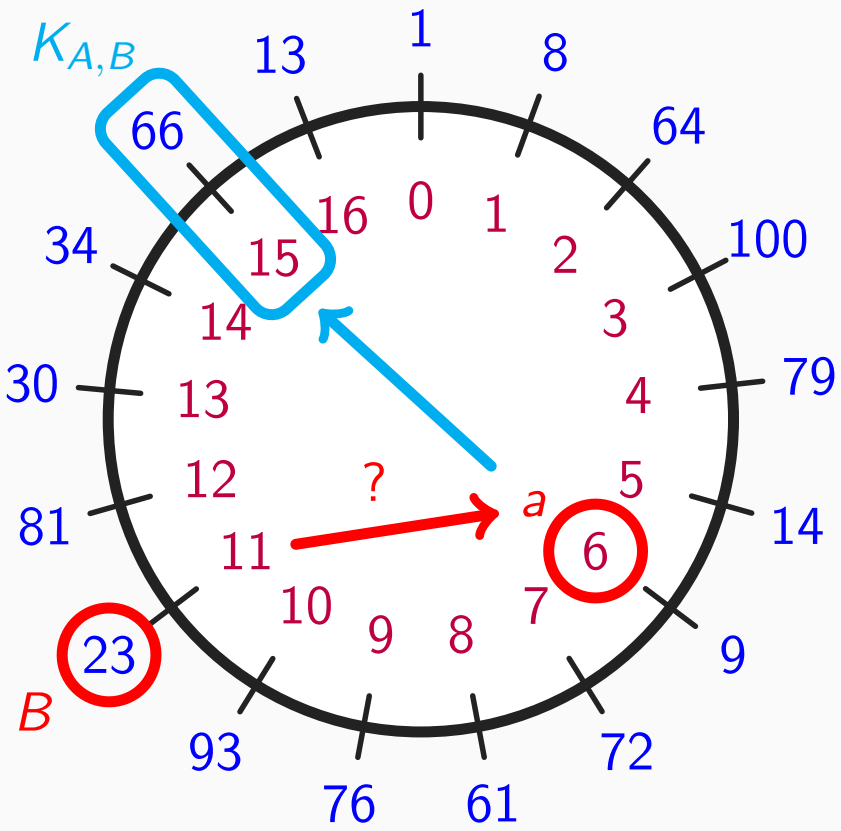
$$K_{A,B} = (B^a) = (g^b)^a = g^{b \cdot a} = g^{a \cdot b} = (g^a)^b = A^b = K_{B,A}$$

- Alice and Bob derive key(s) from secret: $K \leftarrow \mathrm{h}(\text{“KDF”}; K_{A,B})$
  - using key derivation function (KDF), in this example built from a cryptographic hash function
- This requires specifying how to encode elements of $\langle g \rangle$ as bitstrings
- They use K to encipher and/or MAC their communication

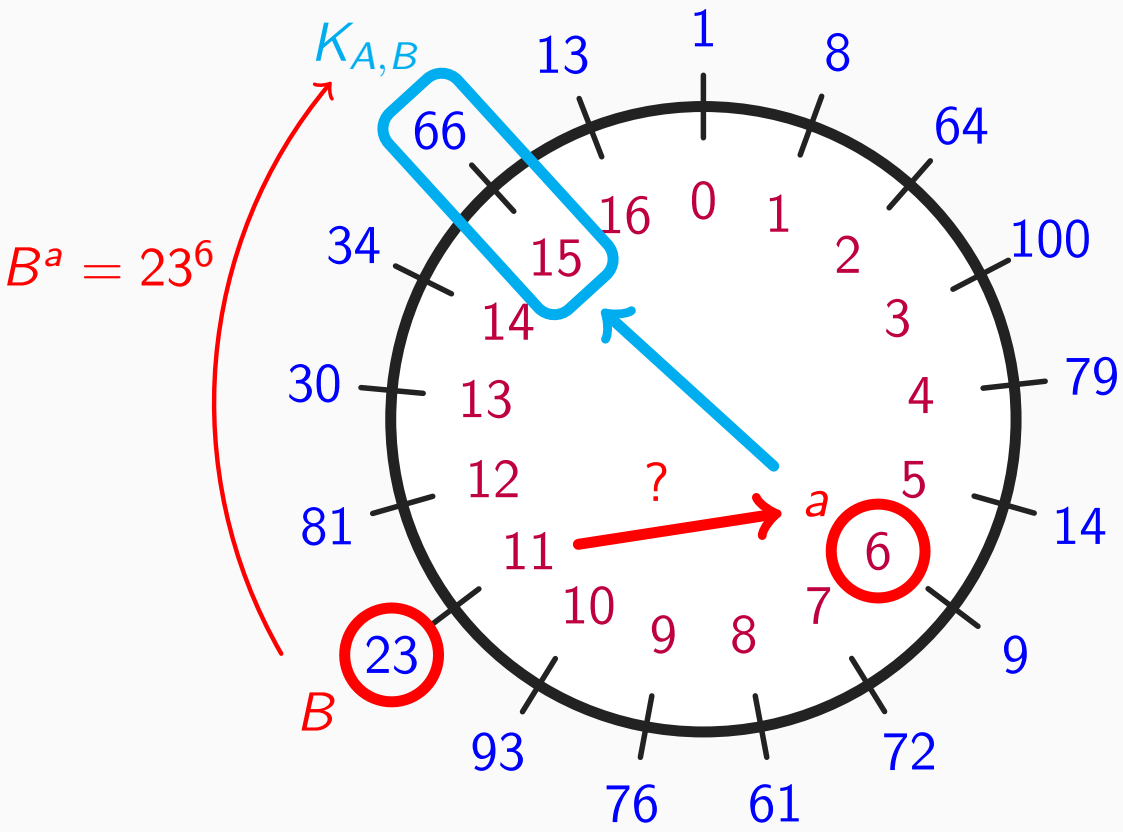
(Merkle-)Diffie-Hellman in our toy example



Alice's computation illustrated



Alice's computation illustrated



Man-in-the-middle attack

Alice		Eve		Bob
p, g, q		p, g, q		p, g, q
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		$e \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		$b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$
$A \leftarrow g^a$		$E \leftarrow g^e$		$B \leftarrow g^b$
	$\xrightarrow{\text{Alice}, A}$		$\xrightarrow{\text{Alice}, E}$	
			$\xleftarrow{\text{Bob}, B}$	
	$\xleftarrow{\text{Bob}, E}$			
$K \leftarrow E^a$		$K \leftarrow A^e$		
		$K' \leftarrow B^e$		$K' \leftarrow E^b$

Man-in-the-middle attack: discussion

- Alice and Bob both unknowingly share a secret with Eve
- In subsequent exchange protected with shared secrets
  - Eve decrypts, can read plaintext, and re-encrypts
  - Eve may modify/delete messages and compute tags
- Solution:
  - Alice must verify B really comes from Bob
  - Bob must verify A really comes from Alice

Public-key authentication is essential ! !! !!!

Diffie-Hellman key agreement: attention points

- Assume Alice authenticated Bob's public key and vice versa
- Security against eavesdropping Eve
  - Eve needs either a or b to compute $K_{A,B}$
  - given g, A and B, predicting $K_{A,B}$ should be hard
  - This is called (computational) Diffie-Hellman hardness assumption (CDH)
  - CDH seems as hard as discrete log problem but no proof of this
- Domain parameters: both need to work in same cyclic group $\langle g \rangle$
- Entity authentication?
  - can be done with challenge-response using key derived from shared secret
  - along with encryption, message origin authentication

DH: mutual and unilateral public-key authentication

- Mutual PK authentication: both parties authenticate public keys
  - If Alice validated Bob's public key, she knows only Bob has $K_{A,B}$
  - If Bob validated Alice's public key, he knows only Alice has $K_{A,B}$
- Unilateral authentication of the public key
  - Alice authenticates Bob's public key but not vice versa
  - Alice still has guarantee that only Bob knows $K_{A,B}$ and so
    - only Bob can decipher what she enciphers with $K_{A,B}$
    - only Bob can generate tags with $K_{A,B}$
  - TLS (https) almost always uses unilateral authentication: website does not authenticate public key of browser

Note: even if there is public-key authentication, DH as such does not achieve entity or message origin authentication

DH key agreement: forward secrecy

Static Diffie-Hellman: Alice and Bob have long-term key pairs

- advantage: public key only needs to be authenticated once (instead of each time)
- limitation: $K_{A,B}$ is always the same
- leakage of $K_{A,B}$, a or b allows decryption of all past messages
- this is called: lack of *forward secrecy*

Forward secrecy

is the property that the compromise of keys in a device does not compromise encrypted communication of the past

Consider textbook Diffie-Hellman

- Alice and Bob generate fresh key pairs (a, A) and (b, B) per session/message
- these are called ephemeral key pairs
- leaking $K_{A,B}$, a or b only affects single session/message
- How to protect against man-in-the-middle?

Diffie-Hellman key agreement with forward secrecy

- forward secrecy by textbook Diffie-Hellman
  - Alice generates ephemeral key pair (a, A)
  - Bob generates ephemeral key pair (b, B)
  - They do a Diffie-Hellman key agreement with these keys
  - Each destroys her/his private key and shared secret after establishment of symmetric key(s) K
  - At the end of the session both destroy K
- protection against man-in-the middle by signing public keys
  - both Alice and Bob have long-term signing keys they authenticate from each other
  - can be done manually or via a PKI
  - they use these to sign the ephemeral public keys

ElGamal encryption

ElGamal encryption

- One of the earliest public-key encryption schemes is ElGamal, invented by Taher ElGamal in 1985
- Encryption based on Diffie-Hellman key agreement
- Interesting because often used as building block in cryptographic protocols
- Alice encrypt a message M to cryptogram (C, A) for Bob like this:

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M' \leftarrow C \times A^{q-b}$

$$M' = C \times A^{q-b} = M \times B^a \times A^{-b} = M \times (g^b)^a \times (g^a)^{-b} = M \times g^{ba} \times g^{-ab} = M$$

ElGamal encryption: attention points

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M \leftarrow C \times A^{q-b}$

- Message M must be an element of $\langle g \rangle$
  - requires encoding function mapping m to $M \in \langle g \rangle$
  - note: must be efficiently decodable for Bob to decrypt
  - existence of such a function depends on the group $\langle g \rangle$
- As first step, Alice generates an ephemeral key pair (a, A)
  - for security, a must be randomly generated for each encryption
  - re-use leads to leakage, like in one-time pad
- Encryption costs 2 exponentiations, decryption a single one

Security of ElGamal encryption and DDH

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$C \leftarrow M \times B^a$	$\xrightarrow{\text{Alice},(C,A)}$	$M \leftarrow C \times A^{q-b}$

- Encryption works by multiplication with a one-time secret B^a
- Secure if this secret is indistinguishable from random element in $\langle g \rangle$
- Leads to *decisional Diffie-Hellman* (DDH) security notion for $\langle g \rangle$
  - with what Eve knows, she cannot distinguish B^a from an element randomly chosen from $\langle g \rangle$, that is:
  - given (g^a, g^b, D), it is hard to determine whether $D = g^{ab}$
- Don't forget: before you encrypt, verify that B is indeed Bob's public key!

IND-CPA security of ElGamal encryption

- Security notion for (public-key) encryption: indistinguishability under chosen-plaintext attacks (IND-CPA)
- For Enc_{PK}, we play a game between *challenger* and *adversary*

Challenger		Adversary
$p, g, (q)$		$p, g, (q)$
$b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$PK \leftarrow g^b$	$\xrightarrow{PK}$	Repeat: $\mathsf{Enc}_{PK}(M)$
	$\xleftarrow{M_0, M_1}$	M_0, M_1 messages
$i \xleftarrow{\$} \{0, 1\}$		
$CT \leftarrow \mathsf{Enc}_{PK}(M_i)$	$\xrightarrow{CT}$	Repeat: $\mathsf{Enc}_{PK}(M)$

- Adversary must guess which message was encrypted: M_0 or M_1
- Secure if adversary has negligible advantage
- This is the case if the decisional Diffie-Hellman problem is hard (see homework)

Note: deterministic encryption schemes can't be IND-CPA secure

Key encapsulation mechanism (KEM) from ElGamal

KEM transports a key from Alice to Bob without interaction

- allows sending arbitrary encrypted message in one transmission
  - public-key crypto is used to *transport* a shared secret
  - in same transmission, (symmetrically) encrypted message

Alice		Bob
$p, g, (q), B$		$p, g, (q), b, B(= g^b)$
$a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$		
$A \leftarrow g^a$		
$K \leftarrow \mathsf{h}(\text{“KDF”}; B^a)$		
$CT \leftarrow \mathsf{Enc}_K(m)$	$\xrightarrow{\text{Alice},(A,CT)}$	$K \leftarrow \mathsf{h}(\text{“KDF”}; A^b)$
		$m \leftarrow \mathsf{Dec}_K(CT)$

(Basically: Diffie-Hellman with static B and ephemeral A)

Discrete log crypto security notions

Security notions: computational variants

Discrete log (DL) problem

Let $a \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$ and $A \leftarrow g^a$

Given $\langle g \rangle$ and A, determine a

Computational Diffie-Hellman (CDH) problem

Let $a, b \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, $A \leftarrow g^a$ and $B \leftarrow g^b$

Given $\langle g \rangle$ and A, B, determine g^{ab}

Computational hardness assumption

Let s be the security strength

A problem is computationally hard to solve with respect to s, if for all algorithms that solve it with computational complexity N and success probability p, it holds that $N/p \geq 2^s$

Security notions: indistinguishability variant

Decisional Diffie-Hellman (DDH) problem

Let $a, b, c \xleftarrow{\$} \mathbb{Z}/q\mathbb{Z}$, and $A \leftarrow g^a$, and $B \leftarrow g^b$
With probability $\frac{1}{2}$, set $C \leftarrow g^c$, and otherwise $C \leftarrow g^{ab}$.

Given $\langle g \rangle$ and A, B, C, determine whether $C = g^{ab}$ holds.

- Just guessing $C = g^{ab}$ (or $C \leftarrow g^c$) has success probability $\frac{1}{2}$
- Breaking DDH is defined as doing better than random guessing
- Therefore, we consider the advantage of a distinguisher $\mathcal{A}$ over random guessing:

$$\mathrm{Adv}_{\mathcal{A}} = |\Pr(\mathcal{A} = 1 \mid C = g^{ab}) - \Pr(\mathcal{A} = 1 \mid C = g^c)|$$

Indistinguishability hardness assumption

Let s be the security strength
An indistinguishability problem is hard with respect to s, if for all distinguishers $\mathcal{A}$ with computational complexity N and advantage $\mathrm{Adv}_{\mathcal{A}}$, it holds that $N/\mathrm{Adv}_{\mathcal{A}} \geq 2^s$

Relations between security notions

$$\text{DDH is hard} \Rightarrow \text{ CDH is hard } \Rightarrow \text{ DL is hard}$$

- If DDH hardness assumption holds in $\langle g \rangle$, CDH hardness holds too
  - determining shared secret allows distinguishing it from random
- If CDH is hard in $\langle g \rangle$, DL is hard too
  - solving discrete log allows determining the shared secret
- Implications for cryptographic schemes
  - ElGamal encryption is secure if DDH is hard
  - Diffie-Hellman is secure if CDH is hard
  - Any discrete-log based crypto requires DL to be hard
- Strength: $\log_2(N/\Pr(\text{success}))$ or $\log_2(N/\mathrm{Adv})$ with N workload
- Achieved security strength depends on $\langle g \rangle$
  - for s bits of security $\mathrm{ord}(g)$ must be at least 2^{2s}
  - if $\langle g \rangle \subset (\mathbb{Z}/p\mathbb{Z})^*$, it is required that $p \ggg 2^{2s}$
  - $\exists$ groups where CDH is hard and DDH not, e.g., if $\mathrm{ord}(g)$ is not prime

Conclusions

Conclusions

- Two very simple discrete-log based cryptosystems:
  - (Merkle)-Diffie-Hellman allows establishing a shared secret
  - ElGamal allows encrypting a message $M \in \langle g \rangle$
- We treated $\langle g \rangle \subset (\mathbb{Z}/p\mathbb{Z})^*$ but there are other choices for $\langle g \rangle$
  - "Diffie-Hellman is secure" said scientifically: CDH is hard
  - "ElGamal is secure" said scientifically: DDH is hard
  - Both require that DL is hard for $\langle g \rangle$
- Both require parties to authenticate public keys of the other party