# Lecture 6. Key exchange

## Introduction to cryptology


Bruno Grenet

M1 INFO, MOSIG & AM

Université Grenoble Alpes – IM²AG


https://membres-ljk.imag.fr/Bruno.Grenet/IntroCrypto.html

# Introduction

## Up to now: Symmetric cryptography

- Symmetric encryption *confidentiality*
- Message authentication codes *authenticity/integrity*

$\rightarrow$ Both require parties to share a common secret

How do two parties agree on a common secret?

## Bad solution

- Any pair of parties agree on a common key
  - If $N$ parties, it requires $N^2$ keys!
  - To share a key, the parties must meet

# One possible solution: key distribution centers (KDCs)

## Idea

- Each party shares a (secret) key with the KDC
- If Alice wants to talk to Bob:
  - Alice gets an encrypted *session key* $k$: $\mathsf{Enc}_{k_a}(k)$
  - Bob gets the same encrypted temporary key: $\mathsf{Enc}_{k_b}(k)$
  - Alice and Bob decrypt $k$ and use it to communicate

## Advantages

- Each party retains (in the long run) only one key
- Each party only needs to meet the KDC, once

## Disadvantages

- The KDC is the central security point:
  - If it is attacked, all security falls
  - If it fails, no communication is possible
- Does not work in *open system* like Internet

# Public-key cryptography

## Key-exchange protocols

- Two parties discuss publicly
- At the end, both parties know a same secret $k$
- External observers do not learn the secret, even after reading all exchanged messages

## Public-key encryption and signatures

- Direct protocols to ensure confidentiality, authenticity and/or integrity
- Based on a pair (public key, private key) $\rightarrow$ no common secret

## In this course

- This lecture: Key-exchange protocols
- Lecture 7: Public-key encryption
- Lecture 8: Signatures *public-key equivalent to MACs*

# The goal of a key exchange

Allow two parties to agree on a key, remotely

## Objective

- Alice and Bob exchange messages
- At the end of the exchange, they both know the same key $k$
- An attacker who sees all the messages has no information about $k$

## Is this possible?

- The attacker sees as much as Alice and Bob ‽
- No information $\rightarrow$ computational security

# New Directions in Cryptography

644 IEEE TRANSACTIONS ON INFORMATION THEORY, VOL. IT-22, NO. 6, NOVEMBER 1976

# New Directions in Cryptography

*Invited Paper*

WHITFIELD DIFFIE AND MARTIN E. HELLMAN, MEMBER, IEEE

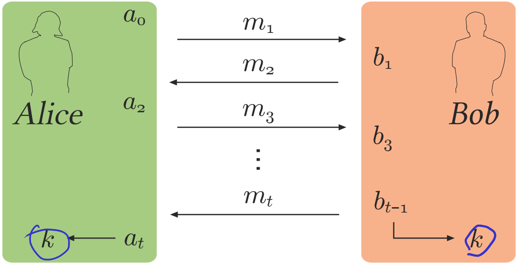
***Abstract*—Two kinds of contemporary developments in cryptography are examined. Widening applications of teleprocessing have given rise to a need for new types of cryptographic systems, which minimize the need for secure key distribution channels and supply the equivalent of a written signature. This paper suggests ways to solve these currently open problems. It also discusses how the theories of communication and computation are beginning to provide the tools to solve cryptographic problems of long standing.**

## I. Introduction

WE STAND TODAY on the brink of a revolution in cryptography. The development of cheap digital hardware has freed it from the design limitations of me-

The best known cryptographic problem is that of privacy: preventing the unauthorized extraction of information from communications over an insecure channel. In order to use cryptography to insure privacy, however, it is currently necessary for the communicating parties to share a key which is known to no one else. This is done by sending the key in advance over some secure channel such as private courier or registered mail. A private conversation between two people with no prior acquaintance is a common occurrence in business, however, and it is unrealistic to expect initial business contacts to be postponed long enough for keys to be transmitted by some physical means. The cost and delay imposed by this key distribution problem is a major barrier to the transfer of business communications to large teleprocessing networks.

# Definition of a protocol




## Key exchange protocol

- Public : messages $m_1, \ldots, m_t$ ; key space $\mathcal{K}$
- Private : the $a_i$ known only to Alice, the $b_i$ only to Bob
- Correct protocol if Alice and Bob compute the same key $k \in \mathcal{K}$

## Vocabulary

- $m_1, \ldots, m_t$: the *transcript*

Freely inspired by: J. Katz, Y. Lindell. Introduction to modern cryptography. 3rd ed, CRC Press, 2021.

# Security of a protocol

> Secure key exchange protocol: given $m_1, \ldots, m_t$, it is difficult to compute $k$

## Key exchange indistinguishability experiment $\mathrm{Exp}_{\mathrm{KE}}^{\mathrm{IND-EAV}}(A)$

**Challenger** simulates the protocol $\to$ transcript $m_1, \ldots, m_t$ and key $k \in \mathcal{K}$
draws $b \twoheadleftarrow \{0, 1\}$ and returns $\hat{k} = k$ if $b = 1$ and $\hat{k} \twoheadleftarrow \mathcal{K}$ otherwise

**Adversary** sees the transcript and $\hat{k}$, and returns a bit $b'$

## Advantages

- $\mathrm{Adv}_{\mathrm{KE}}^{\mathrm{IND-EAV}}(A) = |\Pr[b' = 1 | b = 1] - \Pr[b' = 1 | b = 0]|$
- $\mathrm{Adv}_{\mathrm{KE}}^{\mathrm{IND-EAV}}(t) = \max_{A_t} \mathrm{Adv}^{\mathrm{IND-EAV}}(A_t)$ where $A_t$ denotes an algorithm with running time $\leq t$

# Eavesdropper security and *man-in-the-middle* attack

## Indistinguishability in the presence of an eavesdropper

- Security definition assumes a ~~secure~~ authenticated channel between Alice and Bob
- The adversary is only *passive*

## Man-in-the-middle attack

- Charlie intercepts messages between Alice and Bob
- He impersonates both Alice and Bob
- He creates a common secret with Alice, and another one with Bob
- Alice and Bob incorrectly think they share a common secret

## Key exchange is not enough

Combine with authentication signatures

- *Authenticated* key exchange

# A glimpse of Diffie-Hellman protocol

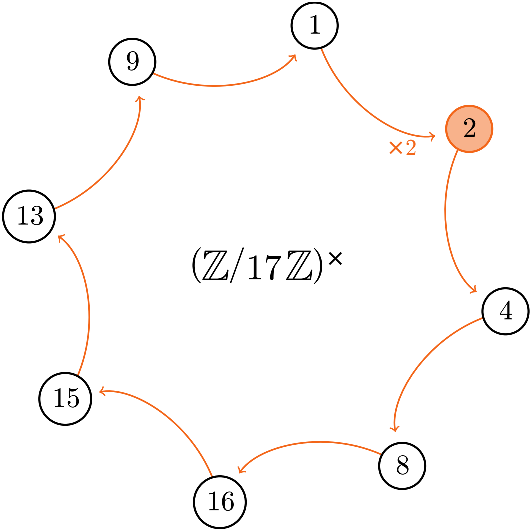
## Protocol sketch

| | |
|---|---|
| Public: | a *number* $g$ |
| Alice | chooses a *random* $a$, computes $g^a$ and sends $g^a$ to Bob |
| Bob | chooses a *random* $b$, computes $g^b$ and sends $g^b$ to Alice |
| Alice | computes $k = (g^b)^a = g^{ab}$ |
| Bob | computes $k = (g^a)^b = g^{ab}$ |

## Outstanding issues

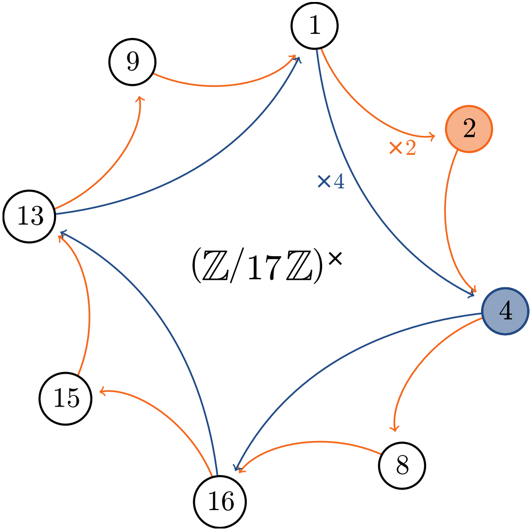
- How to choose $g$?
  - If it is an integer, $g^a$, $g^b$ and $g^{ab}$ are *huge* integers
- Why is this scheme secure?

# The multiplicative group of $\mathbb{Z}/p\mathbb{Z}$



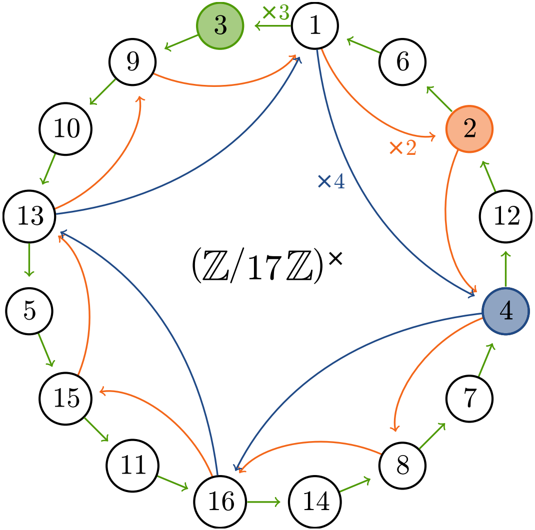

- 2 has order 8

# The multiplicative group of $\mathbb{Z}/p\mathbb{Z}$



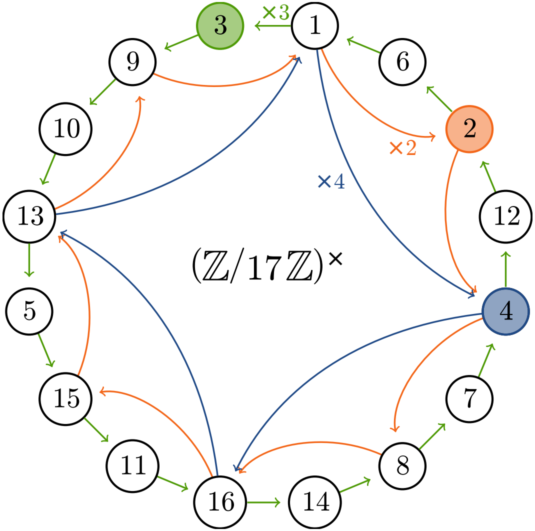

- 2 has order 8
- 4 has order 4

# The multiplicative group of $\mathbb{Z}/p\mathbb{Z}$




- 2 has order 8
- 4 has order 4
- 3 has order 16 $\rightarrow$ generator

# The multiplicative group of $\mathbb{Z}/p\mathbb{Z}$




- 2 has order 8
- 4 has order 4
- 3 has order 16 → generator

*Theorem*

For every $p$, $(\mathbb{Z}/p\mathbb{Z})^\times$ is a cyclic group: there exists a generator $g \in (\mathbb{Z}/p\mathbb{Z})^\times$ such that

$$\begin{aligned}(\mathbb{Z}/p\mathbb{Z})^\times &= \{g^n : n \in \mathbb{Z}\} \\ &= \{g^n : 0 \le n < p-1\}\end{aligned}$$

**Remark**

The generator is *not* unique
(ex.: 3, 5, 6, 7, 10, 11, 12, 14)

# Cyclic groups

## Definition

A multiplicative cyclic group with generator $g$ is a set $G$ such that $G = \{g^n : n \in \mathbb{Z}\}$

## Remarks

- The generator is not unique
- If $G$ is finite with generator $g$, $G = \{g^t : 0 \leq t < n\}$ — $n = |G|$: *order*
  - if $m = nq + r$, $g^m = g^{nq+r} = (g^n)^q \cdot g^r = g^r$
  - $\Rightarrow$ for all $x \in G$, there exists a *unique* $t \in \{0, \ldots, n-1\}$ s.t. $x = g^t$
- Each element $x \in G$ defines a *subgroup* $G_x = \{x^t : t \in \mathbb{Z}\} \subset G$
  - if $x$ has order $s$, $G_x$ contains $s$ elements
  - if $x$ has order $s$, $x^r$ has order $s/\textsc{gcd}(s, r)$
  - $g^t$ has order $n/\textsc{gcd}(t, n)$

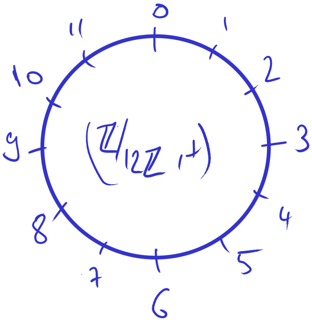
## More general definitions

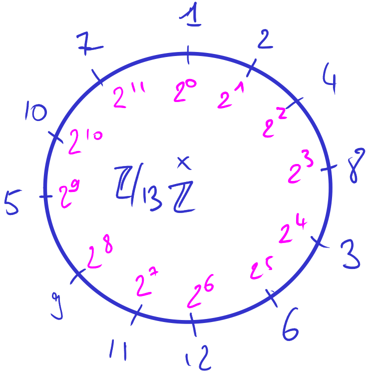
- Cyclic group $(G, \star)$ with any binary operation $\star$
  - Additive cyclic group with generator $g$: $G = \{n \cdot g : n \in \mathbb{Z}\}$
  - Note $(G, \times)$, $(G, +)$ or $(G, \star)$ to specify the type of cyclic group
- General (non-cyclic) groups — *cf.* Lecture 9

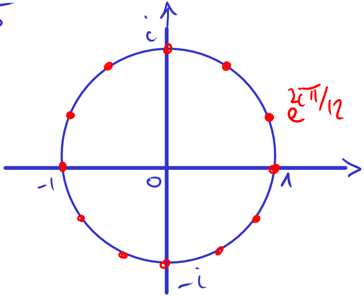
## Examples and counterexamples

| | Additive | Multiplicative |
|---|---|---|
| Infinite | $(\mathbb{Z}, +)$ gen. $1$ | $(\{2^n : n \in \mathbb{Z}\}, \times)$ gen $2$ |
| Finite | $(\mathbb{Z}/n\mathbb{Z}, +)$ gen. $1$ or any $k$ s.t. $\gcd(k,n)=1$ | $(\mathbb{Z}/p\mathbb{Z}^{\times}, \times)$ with $p$ prime<br>$(\{-1, 1\}, \times)$ with gen. $-1$<br>$(\{e^{\frac{2i\pi k}{n}} : k \in \mathbb{Z}\}, \times)$ gen $e^{2i\pi/n}$. |

# Graphical representation



# Discrete logarithm problem

## Definitions

Given a cyclic group $G$ with generator $g$,

- the discrete logarithm of $x$ in base $g$ is the unique $0 \leq t < |G|$ such that $x = g^t$
- the discrete logarithm problem is, given $x$, to compute $t$

## The naive algorithm

- Compute $g^0, g^1, g^2, \ldots$ until we get $g^t = x$
- Complexity $O(t) = O(|G|)$ operations in $G$

Case $\mathbb{Z}/p\mathbb{Z}^{\times}$

Input size: $\Theta(\log p)$

Complexity: $\Theta(p)$

## Easy case: $(\mathbb{Z}/n\mathbb{Z}, +)$

- Generators: 1 or any $g$ such that $\text{GCD}(g, n) = 1$
- Discrete logarithm of $x$: $t$ s.t. $x = t \cdot g \bmod n$
- Case $g = 1$: nothing to do!
- General case:

  1. Compute $u, v$ s.t. $u \cdot g + v \cdot n = 1$
  2. Return $t = u \cdot x \bmod n$

Input size: $\Theta(\log n)$

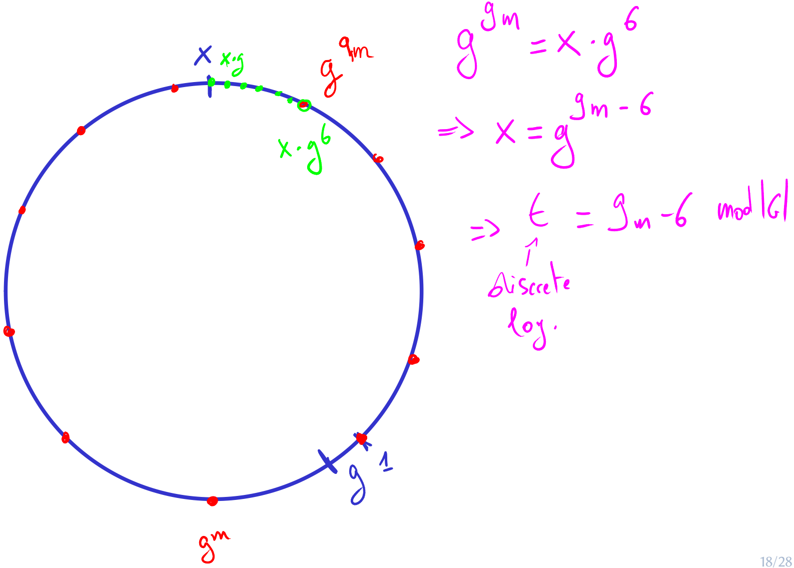
Complexity: $\Theta(\log^2 n)$

Extended Euclidean Algorithm

$t \cdot g = u \cdot x \cdot g = x \bmod n$

# *Baby step-giant step*: A picture is worth a thousand words

(Shanks, 1971)



# *Baby step-giant step*: the algorithm

(Shanks, 1971)

**Input:** a cyclic group $G$ of order $n$, with generator $g$, and $x \in G$

**Output:** the discrete logarithm $t$ of $x$ in base $g$

1. $m \leftarrow \lceil \sqrt{n} \rceil$
2. $B \leftarrow [1, g, g^2, \ldots, g^{m-1}]$ — *Baby steps*
3. $(h, y, j) \leftarrow (g^m, x, 0)$
4. while $y \notin B$: $(y, j) \leftarrow (y \cdot h, j + 1)$ — *Giant steps*: $y = x \cdot g^{m \cdot j}$
5. $i \leftarrow$ index such that $y = g^i$ — *Collision found*: $x \cdot g^{m \cdot j} = g^i$
6. return $(i - m \cdot j) \bmod n$

## Analysis

**Correction:** by Euclidean division, there exist $i, j < m$ such that $t = i - mj \bmod n$

**Complexity:** Baby steps & giant steps: $O(\sqrt{n})$

Collision search: $O(\sqrt{n})$ (naive), $O(\log n)$ (dichotomy), $O(1)$ (hash tables)

$\Rightarrow O(\sqrt{n})$ (same in space)

# DLP hardness

*Theorem* (baby-step giant-step)

In any cyclic group $G$, the discrete logarithm problem can be computed in time $O(\sqrt{|G|})$

- Easily parallelizable
- Variants with better space complexity: Pollard's $\rho$ or kangaroos (*a.k.a.* $\lambda$) algorithms
- Pohlig-Hellman (1978): $O(\sqrt{p})$ *largest prime divisor of* $|G|$

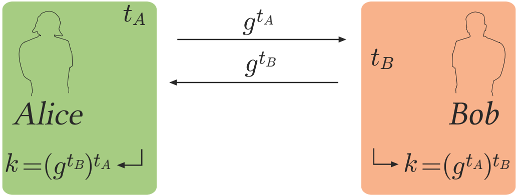
## Choice of $G$

- $(\mathbb{Z}/n\mathbb{Z}, +)$ or $|G|$ small: easy DLP!
- $(\mathbb{Z}/p\mathbb{Z}^\times, \times)$: usually hard, though not *maximally* hard $\ll \sqrt{n}$
  - record: $p$ of 795 bits, 3100 core-year — Boudot *et al.*, 2019
- Points of an elliptic curve over a finite field: maximally hard $O(\sqrt{n})$
  - record: group of 114 bits, 13 days on GPU — Zieniewicz & Pons, 2020

## Additional remarks

- One should use prime order groups
- Algorithm polynomial in log $n$ on a quantum computer — Shor, 1997

# The protocol




## Diffie-Hellman protocol

**Input :** Group $G$ of order $n$ and generator $g \in G$

*Alice* draws $t_A \twoheadleftarrow \{0, \dots, n-1\}$, computes $h_A = g^{t_A}$ and sends $h_A$ to Bob

*Bob* draws $t_B \twoheadleftarrow \{0, \dots, n-1\}$, computes $h_B = g^{t_B}$ and sends $h_B$ to Alice

*Alice* computes $k_A = h_B^{t_A}$

*Bob* computes $k_B = h_A^{t_B}$.

## Correctness

The protocol is correct: $k_A = (g^{t_B})^{t_A} = g^{t_A t_B} = (g^{t_A})^{t_B} = k_B$

Freely inspired from: J. Katz, Y. Lindell. Introduction to modern cryptography. 3rd ed, CRC Press, 2021.

# Use in practice

## Where do the secret lives?

- Shared secret $k \in G$, while usually one needs it in $\{0,1\}^*$
- *Key derivation function* $\mathrm{KDF} : G \to \{0,1\}^*$ $\simeq$ hash function

## *Man-in-the-middle* attack

- Requires an authentication between Alice and Bob
- Out-of-scope of this lecture $\to$ *c.f.* signatures Lecture 8

## Cost of the protocol

- Requires two exponentiations in $G$
- $O(\log |G|)$ operations in $G$ binary powering
- $O(\log^2 p \log\log p)$ bit-operations for $\mathbb{Z}/p\mathbb{Z}$

# Binary powering

$$g^t = \begin{cases} g^{\lfloor t/2 \rfloor} \cdot g^{\lfloor t/2 \rfloor} & \text{if } t \text{ is even} \\ g \cdot g^{\lfloor t/2 \rfloor} \cdot g^{\lfloor t/2 \rfloor} & \text{if } t \text{ is odd} \end{cases}$$

**Input:** $g \in G$, $t \in \mathbb{Z}_{\geq 0}$
**Output:** $g^t$

1. $h \leftarrow 1$
2. while $t \neq 0$:
3. if $t$ is odd: $h \leftarrow h \cdot g$
4. $g \leftarrow g \cdot g$
5. $t \leftarrow \lfloor t/2 \rfloor$
6. return $h$

## Complexity

- ~~$O(t)$~~ multiplications in $G$

$\Theta(\log t)$

## Correctness

- Invariant: $h \cdot g^t = g^{t_{init}}$

# The Computational Diffie-Hellman (CDH) hypothesis

## Experiment $\mathrm{Exp}_G^{\mathrm{CDH}}(A)$

Challenger simulates the DH protocol $\rightarrow$ transcript $g, x_1, x_2 \in G$

Adversary is given the transcript and outputs $y \in G$

Success of the adversary if $y = g^{t_1 t_2}$ where $x_1 = g^{t_1}$ and $x_2 = g^{t_2}$

## Advantages

- $\mathrm{Adv}_G^{\mathrm{CDH}}(A) = \Pr[\textsc{success}(A)]$
- $\mathrm{Adv}_G^{\mathrm{CDH}}(t) = \max_{A_t} \Pr[\textsc{success}(A_t)]$ where $A_t$ is an algorithm that runs in time $\leq t$

CDH hypothesis: $\mathrm{Adv}_G^{\mathrm{CDH}}(t)$ is *negligible* for *reasonable* $t$ in particular $\ll \sqrt{|G|}$

## Remarks

- CDH for $G \Rightarrow$ the discrete log. is *hard* in $G$ *cf contrapositive*
- $g^{t_1 t_2} = (g^{t_1})^{t_2} = (g^{t_2})^{t_1} \neq g^{t_1} g^{t_2}$

# The *decisional* Diffie-Hellman (DDH) hypothesis

## Experiment $\mathrm{Exp}_G^{\mathrm{DDH}}(A)$

**Challenger** simulates the DH protocol $\to (x_1, x_2, k) \leftarrow (g^{t_1}, g^{t_2}, g^{t_1 t_2})$
draws $b \twoheadleftarrow \{0, 1\}$ and sets $\hat{k} \leftarrow k$ if $b = 1$, $\hat{k} \twoheadleftarrow G$ if $b = 0$

**Adversary** is given the transcript $(g, x_1, x_2)$ and $\hat{k}$ and outputs $b'$

## Advantages

- $\mathrm{Adv}_G^{\mathrm{DDH}}(A) = |\Pr[b' = 1 | b = 1] - \Pr[b' = 0 | b = 1]|$
- $\mathrm{Adv}_G^{\mathrm{DDH}}(t) = \max_{A_t} \mathrm{Adv}_G^{\mathrm{DDH}}(A_t)$ where $A_t$ runs in time $\leq t$

CDH hypothesis: $\mathrm{Adv}_G^{\mathrm{CDH}}(t)$ is *negligible* for *reasonable* $t$ in particular $\ll \sqrt{|G|}$

## Relation with other hypotheses

DDH for $G \Rightarrow$ CDH for $G \Rightarrow$ hardness of DLP

# Security of the Diffie-Hellman protocol

*Theorem*

If the DDH hypothesis holds for $G$, then the Diffie-Hellman protocol with group $G$ is IND-EAV secure

*Proof.* $\mathrm{Adv}^{\mathrm{IND-EAV}}_{\mathrm{DH}(G)}(t) = \mathrm{Adv}^{\mathrm{DDH}}_{G}(t)$ by definition!

# Conclusion

## 50 shades of Diffie-Hellman

- The DH protocol is essentially the only key exchange protocol
- But many choices of cyclic group $G$ : $(\mathbb{Z}/p\mathbb{Z})^\times$, elliptic curve, isogenies, ...
- Key derivation function to go from $G$ to $\{0,1\}^*$

## Security of the protocol

- Three hypotheses: DLP hardness, CDH, DDH
- DDH $\iff$ IND-EAV security
- DDH $\Rightarrow$ CDH $\Rightarrow$ DLP hardness

## Inherent vulnerability: *man-in-the-middle*

- Charlie stands between Alice and Bob, and intercepts, modifies, etc. all messages between Alice and Bob
- Requires *authentication* between Alice and Bob *signatures*