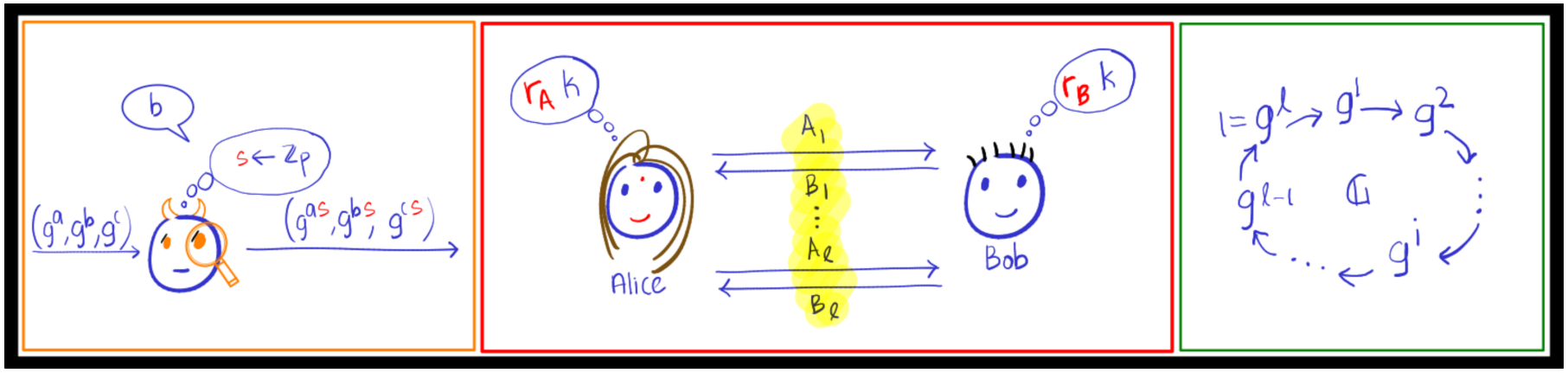
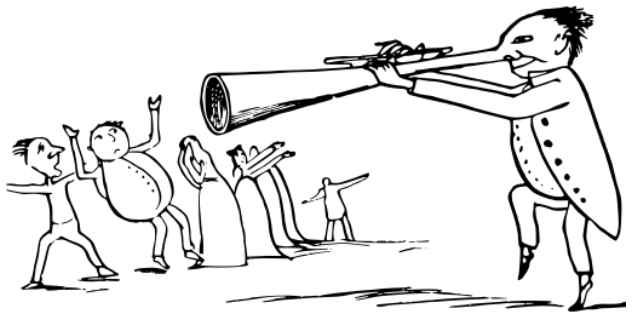


CS409m: Introduction to Cryptography

Lecture 12 (24/Sep/25)

Instructor: Chethan Kamath

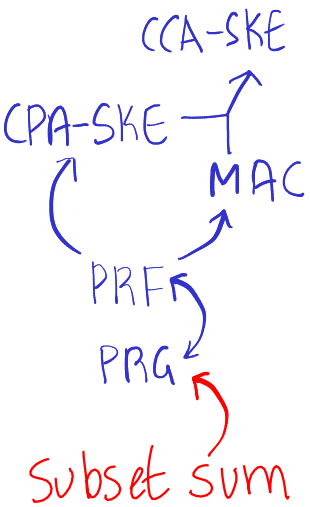
Announcements



- Mid-sem cribs session 12:30-14:30 Monday (29/Sep)
  - View your answer sheet ~~14:00-16:00~~ on ~~Friday~~ (~~26~~/Sep) in CC305
  - Submit cribs online by ~~Monday (29/Sep~~, 23:59) Wednesday 01/Oct
- Assignment 4 (ungraded) will be released on Friday (26/Sep)

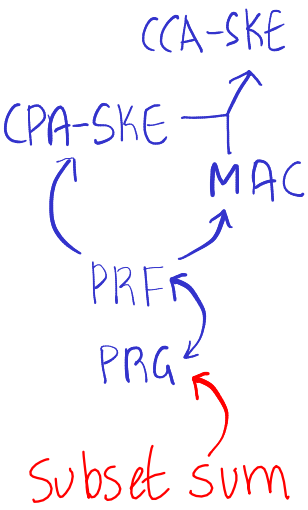
Recall from Last Module

- We learnt: secure communication in the shared-key setting
- Primitives encountered: PRG, PRF, MAC
- Computational hardness assumptions: subset-sum problem

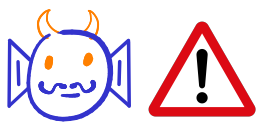


Recall from Last Module

- We learnt: secure communication in the shared-key setting
- Primitives encountered: PRG, PRF, MAC
- Computational hardness assumptions: subset-sum problem

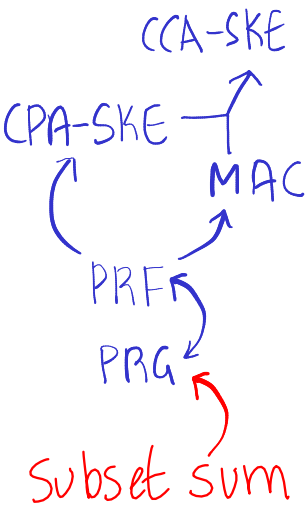


- Key conceptual takeaways:
  - Threat modelling
  - Computational security
  - What design choices lead to vulnerability?

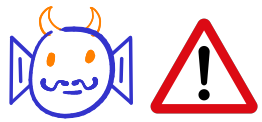


Recall from Last Module

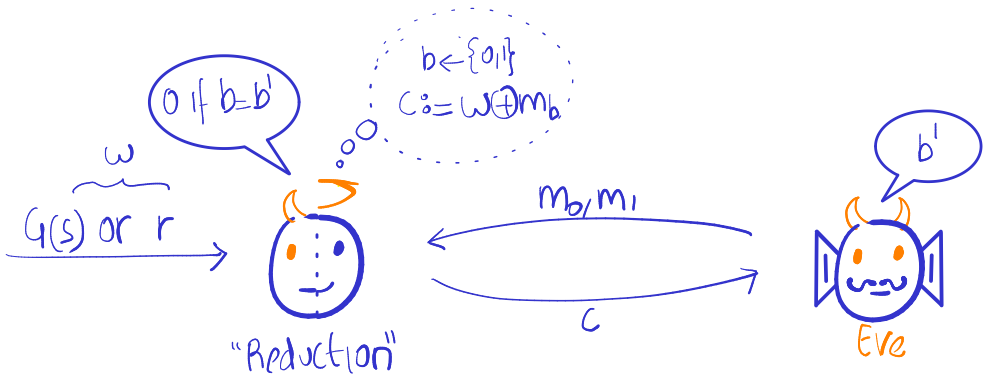
- We learnt: secure communication in the shared-key setting
- Primitives encountered: PRG, PRF, MAC
- Computational hardness assumptions: subset-sum problem



- Key conceptual takeaways:
  - Threat modelling
  - Computational security
  - What design choices lead to vulnerability?

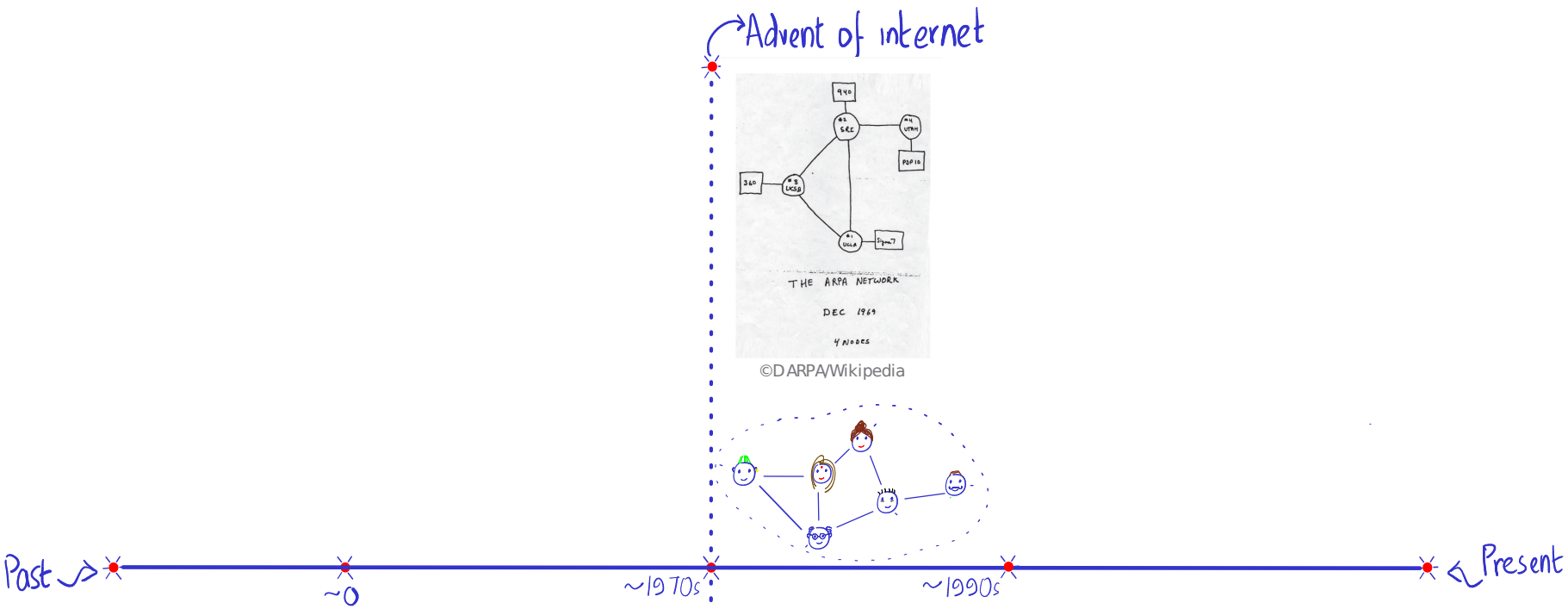


- Key tools: security reduction, hybrid argument



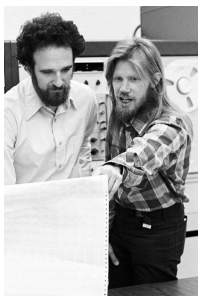
This Module...

MODULE 2
(Public keys)

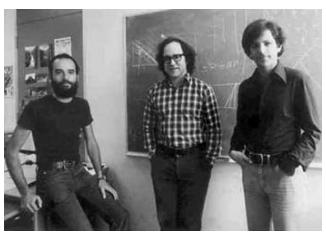


This Module...

MODULE 2
(Public keys)

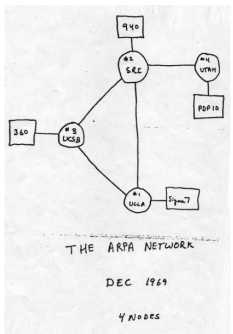


©icour.fr

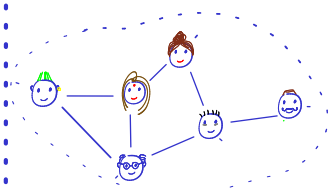


©cs.miami.edu (Rosenberg)

Advent of internet



©DARPA/Wikipedia

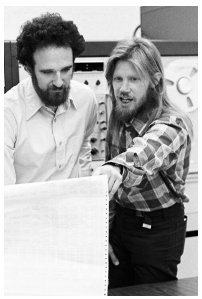


Past → ~0 ~1970s ~1990s ← Present

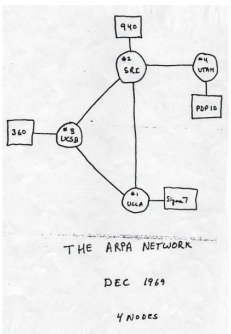
This Module...

MODULE 2
(Public keys)

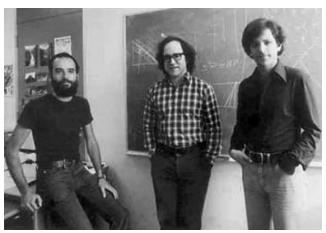
Advent of internet



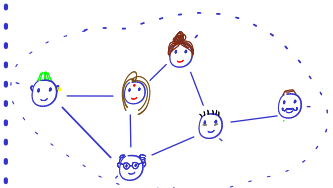
©icour.fr



©DARPA/Wikipedia



©cs.miami.edu (Rosenberg)



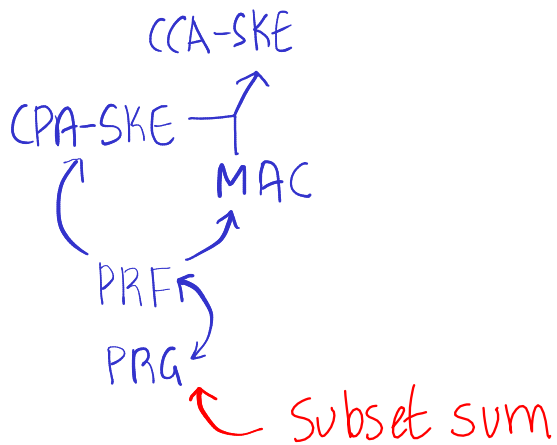
Past → ~0 ~1970s ~1980s ~1990s ← Present

Birth of "provable security"

©Oded Goldreich

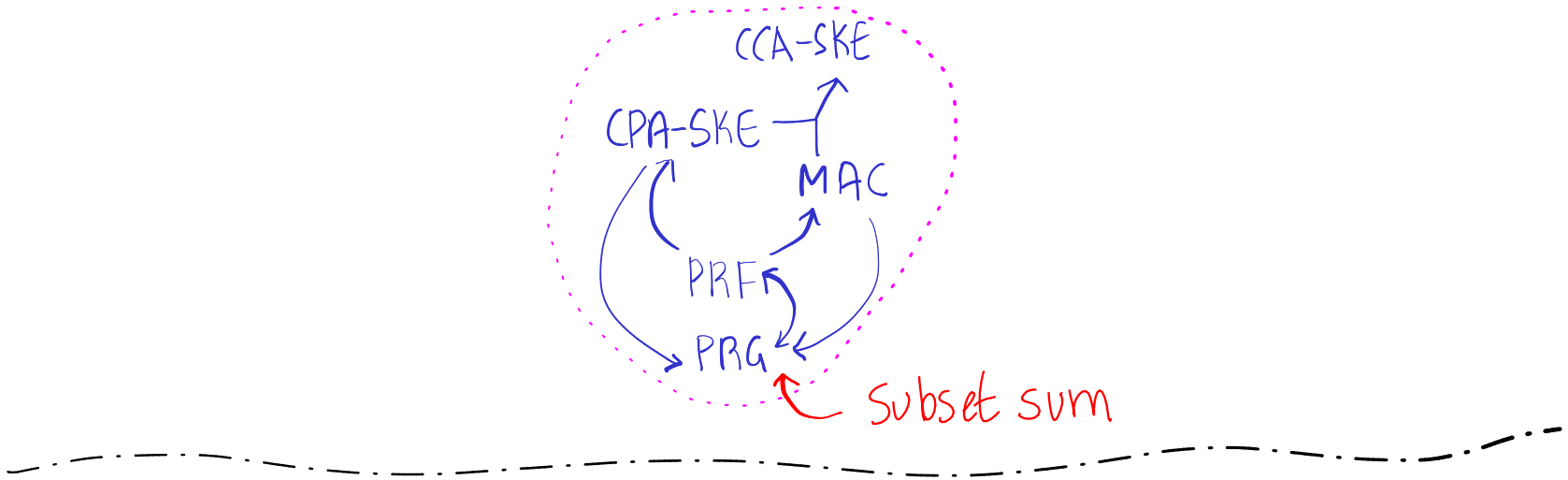
This Module...

- Minicrypt to Cryptomania



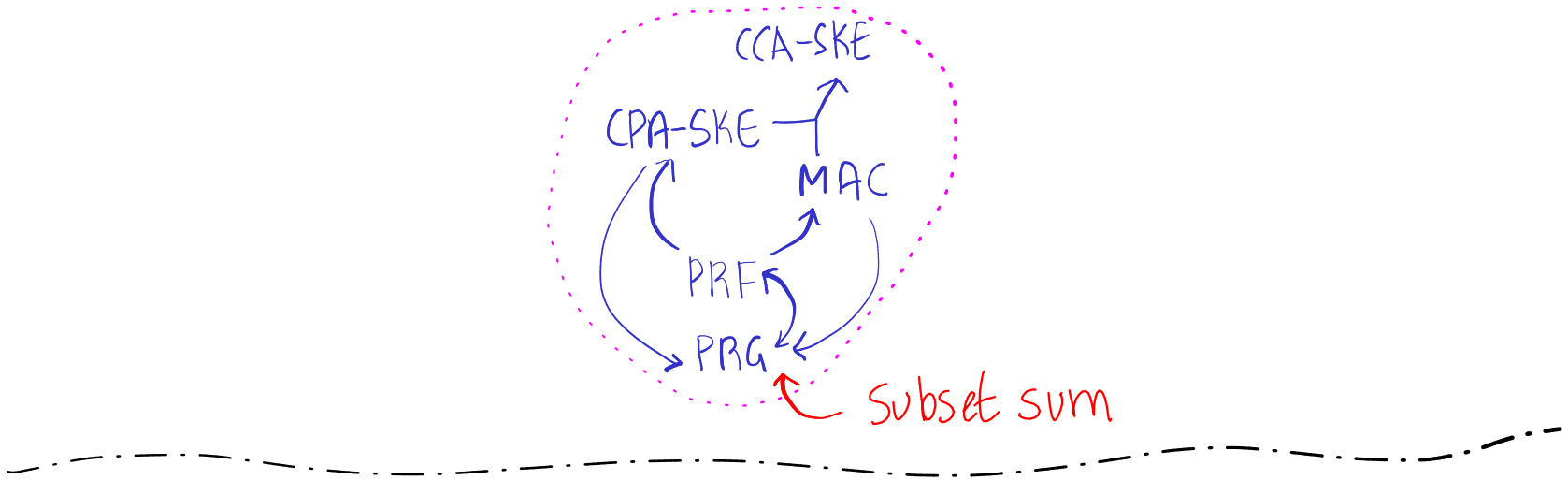
This Module...

- Minicrypt to Cryptomania



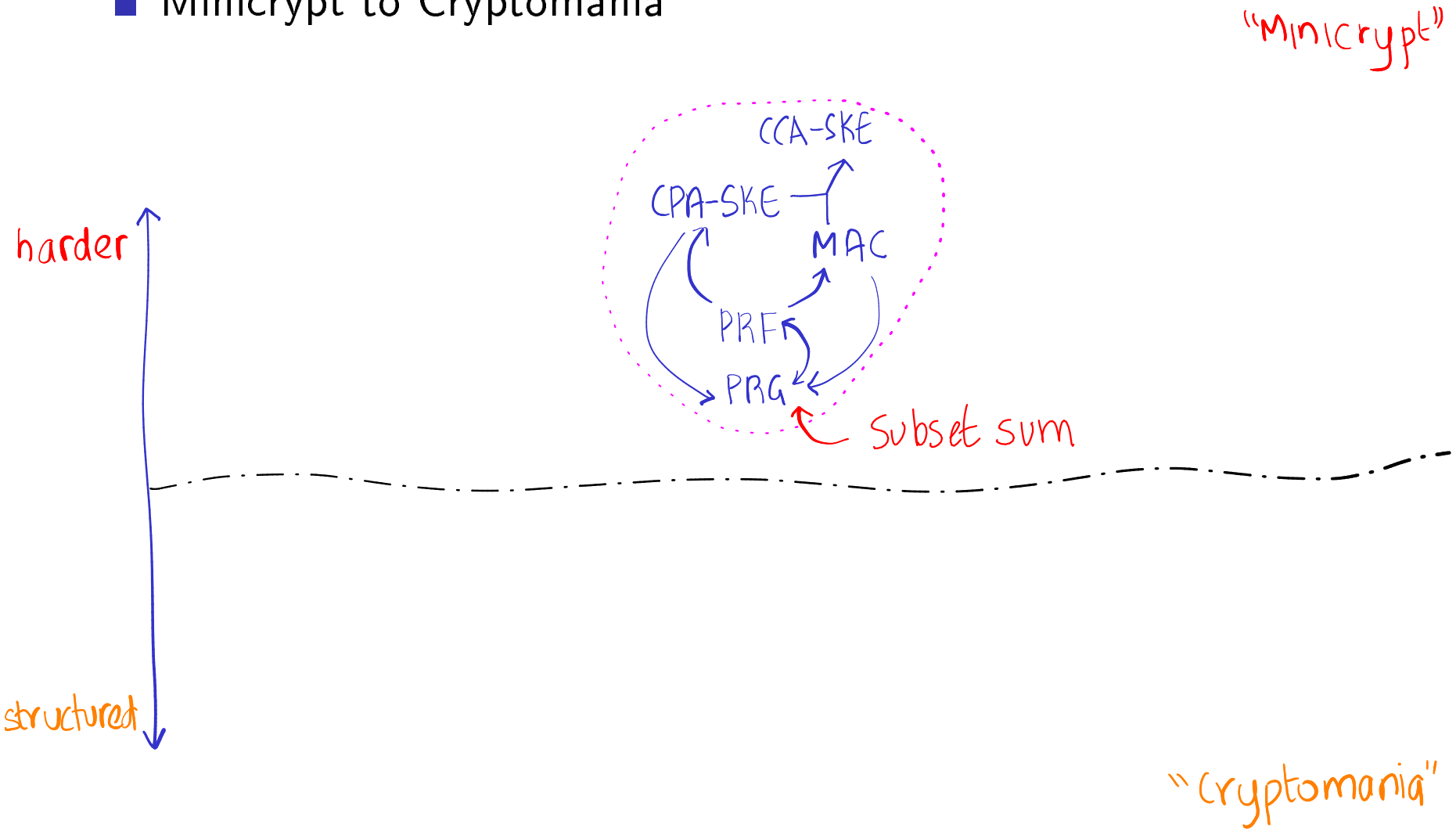
This Module...

- Minicrypt to Cryptomania



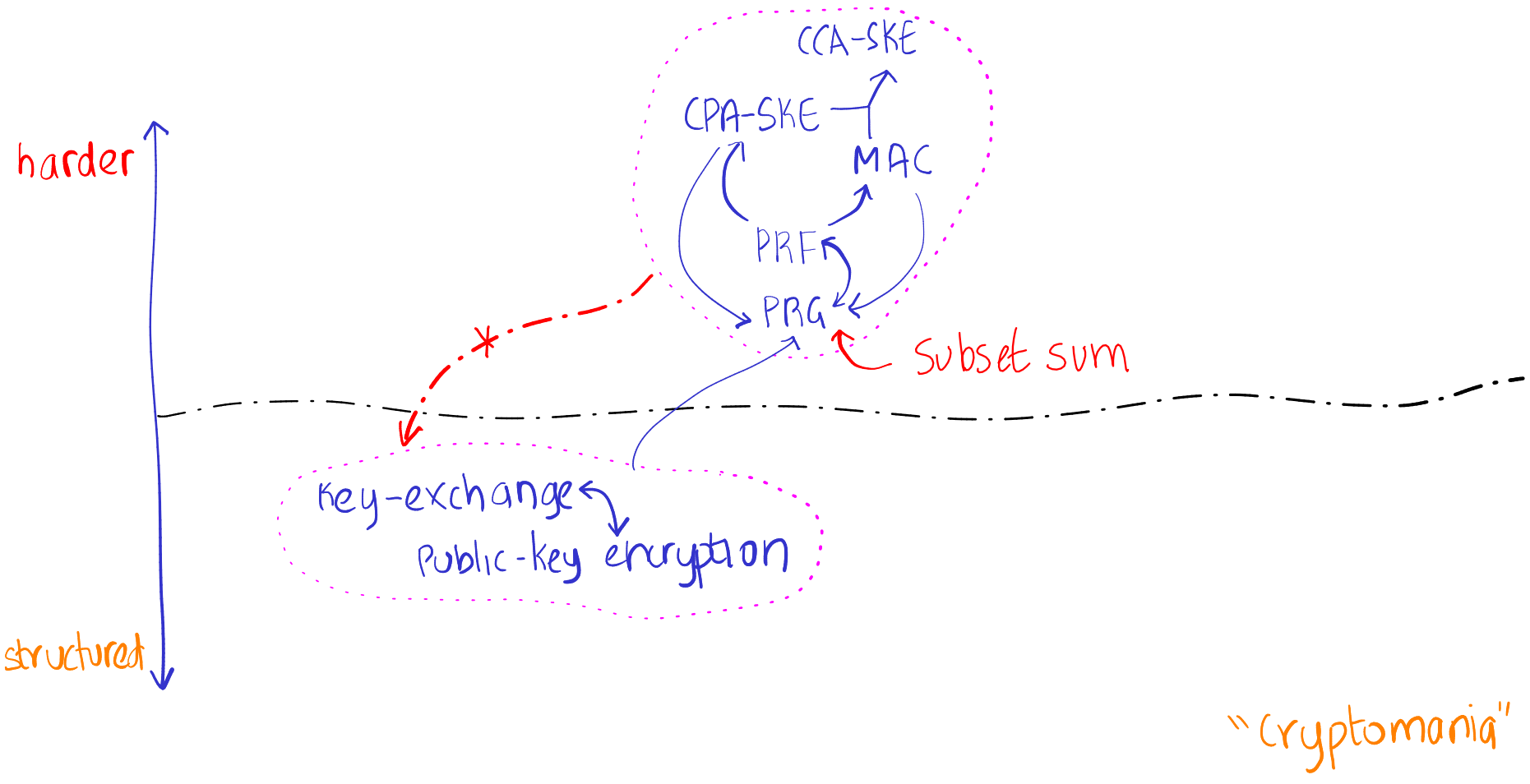
This Module...

- Minicrypt to Cryptomania



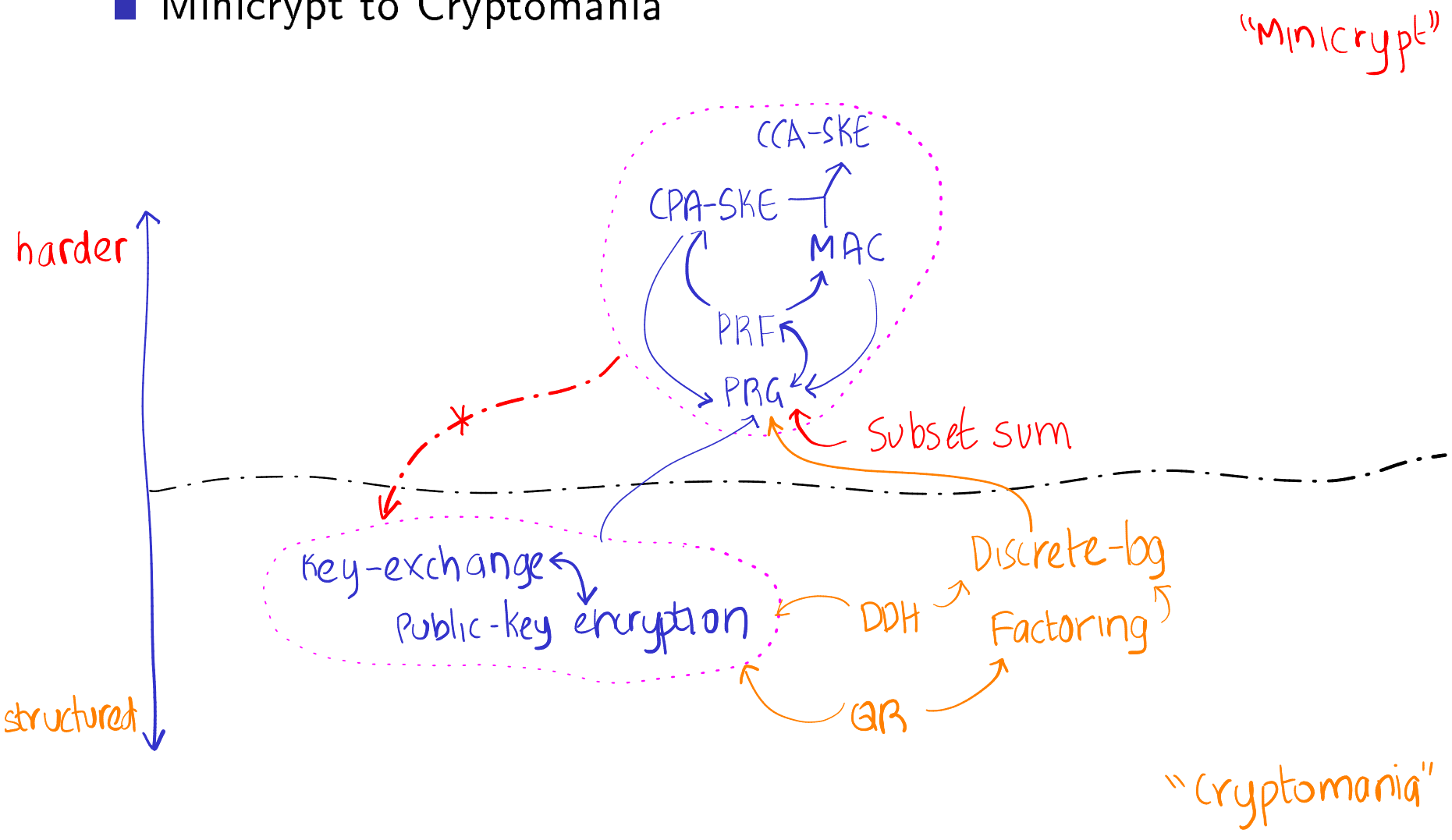
This Module...

- Minicrypt to Cryptomania



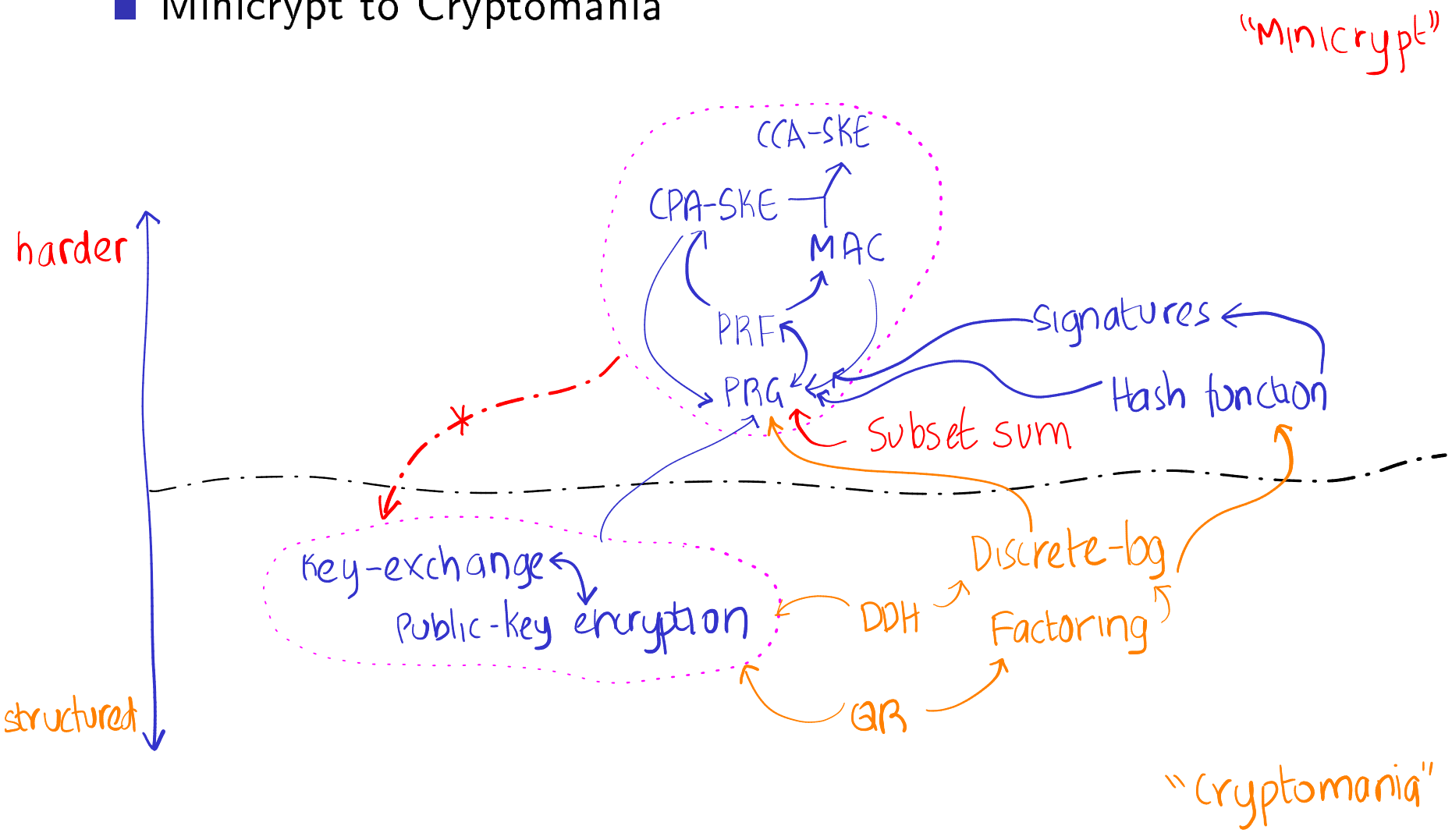
This Module...

- Minicrypt to Cryptomania



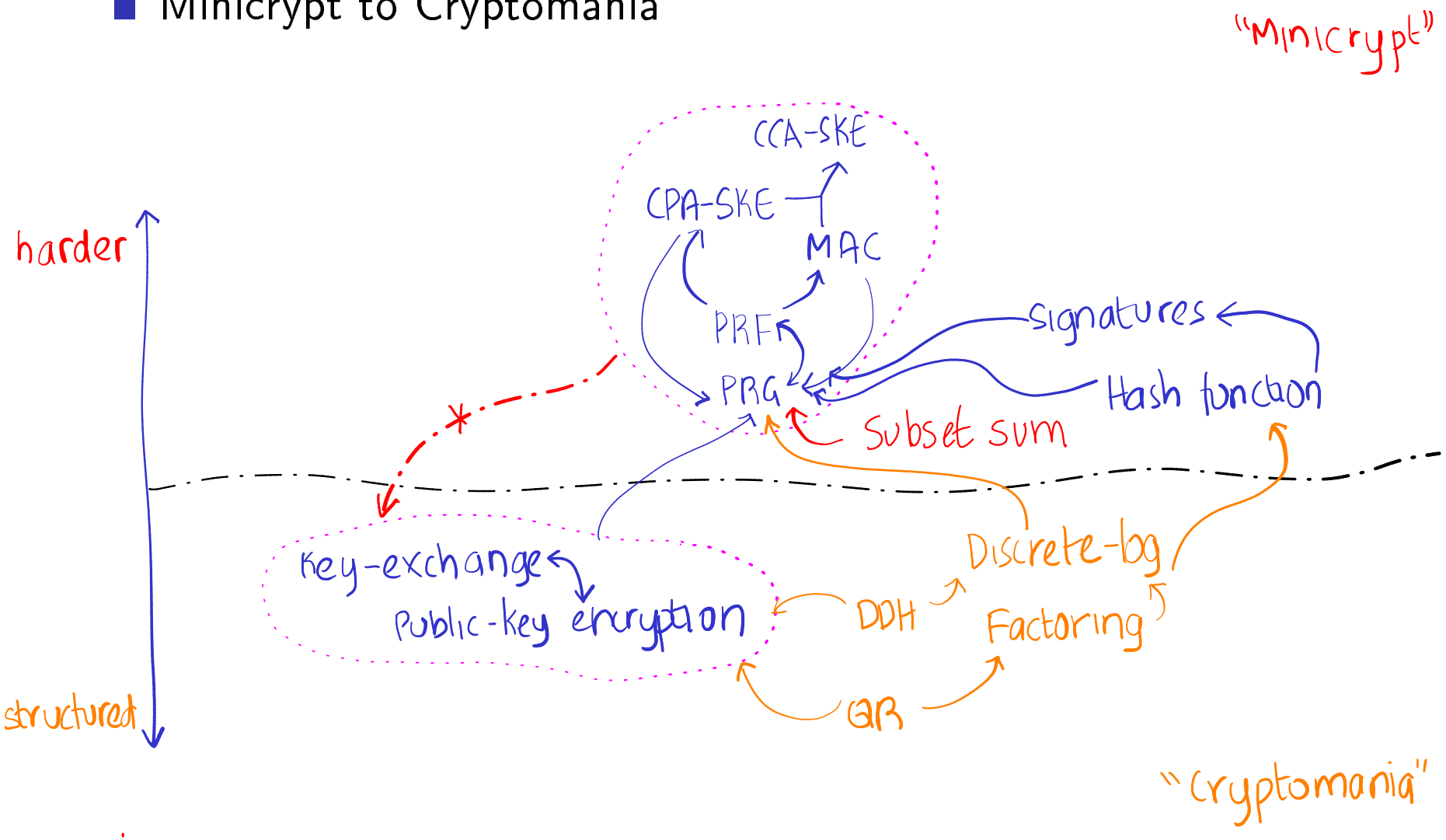
This Module...

- Minicrypt to Cryptomania



This Module...

- Minicrypt to Cryptomania



Today: how does one establish a shared key in the first place?

Plan for Today's Lecture

- Task: key exchange
- Threat model: computational secrecy against eavesdroppers

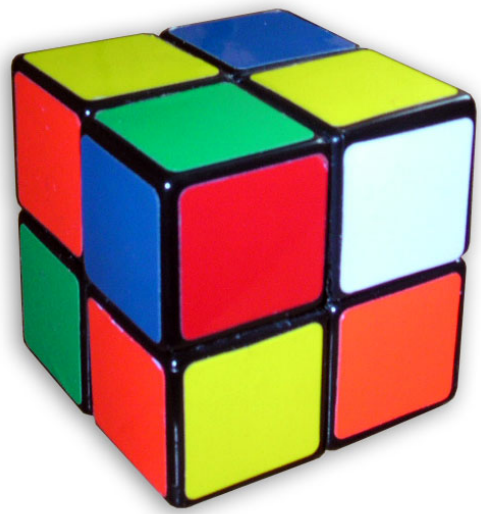
Plan for Today's Lecture

- Task: key exchange 
- Threat model: computational secrecy against eavesdroppers 

Key Exchange

©Wikipedia

Basic Group Theory



©Mike Gonzalez/Wikipedia

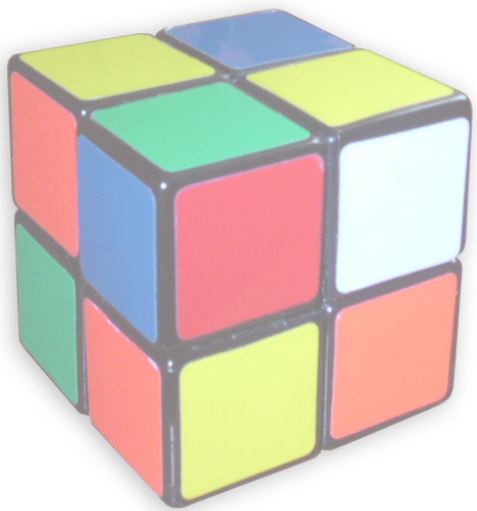
Plan for Today's Lecture

- Task: key exchange
- Threat model: computational secrecy against eavesdroppers

Key Exchange

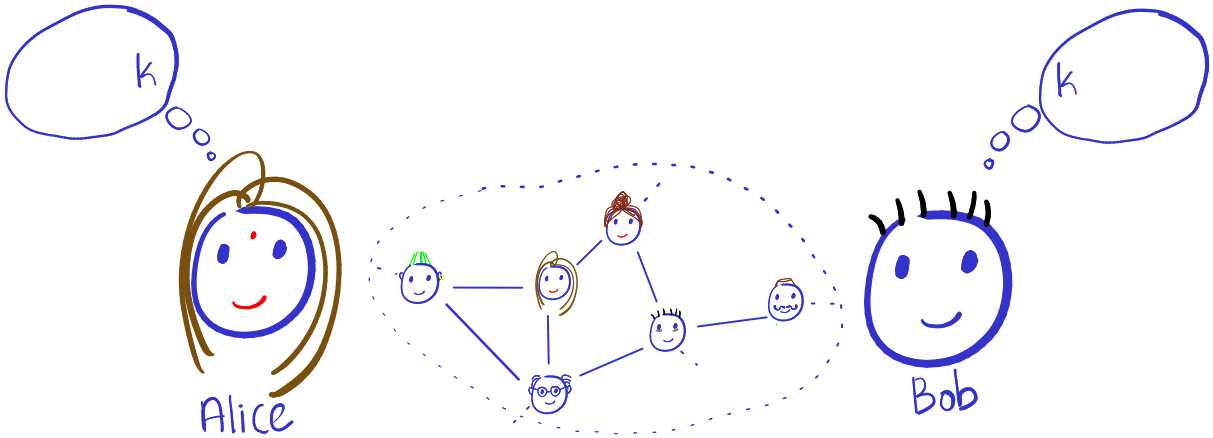
©Wikipedia

Basic Group Theory

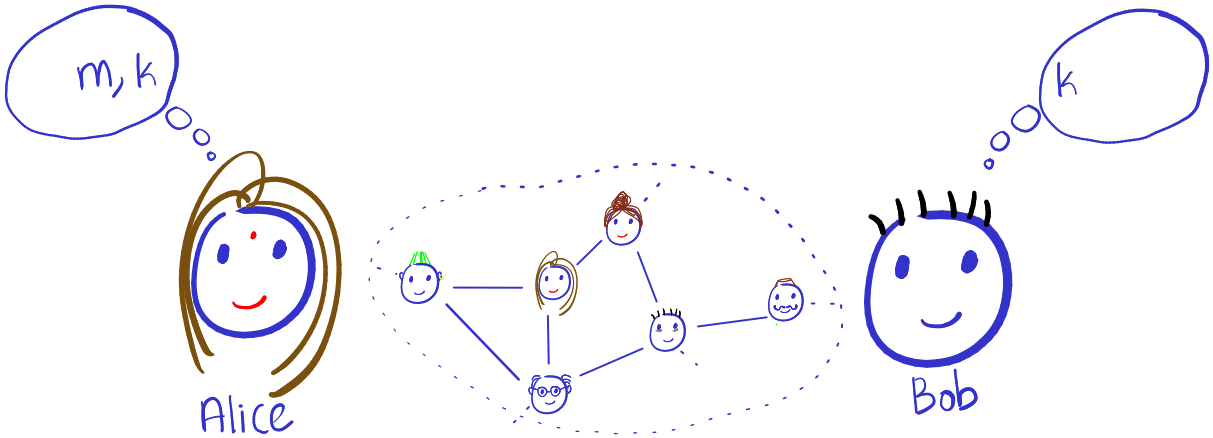


©Mike Gonzalez/Wikipedia

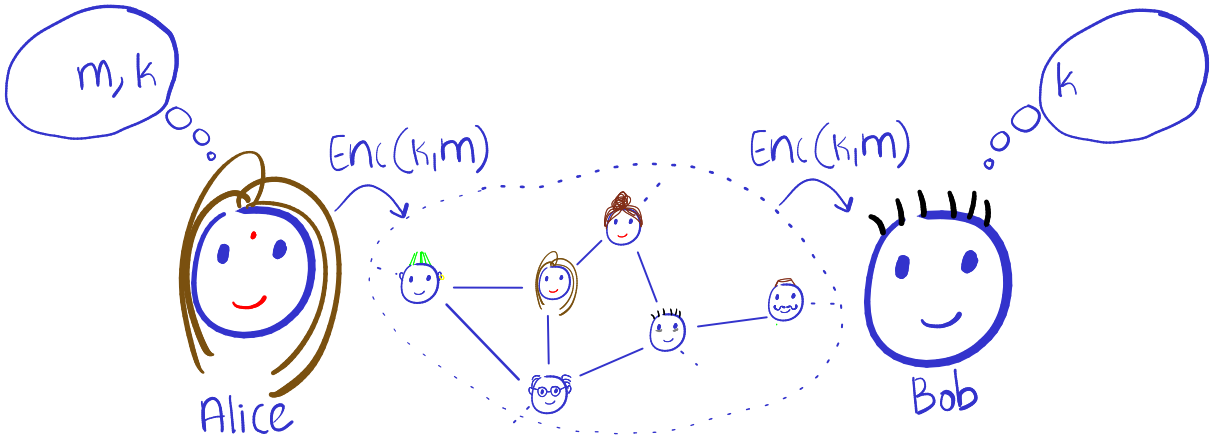
How To Establish a Shared Key in the First Place?



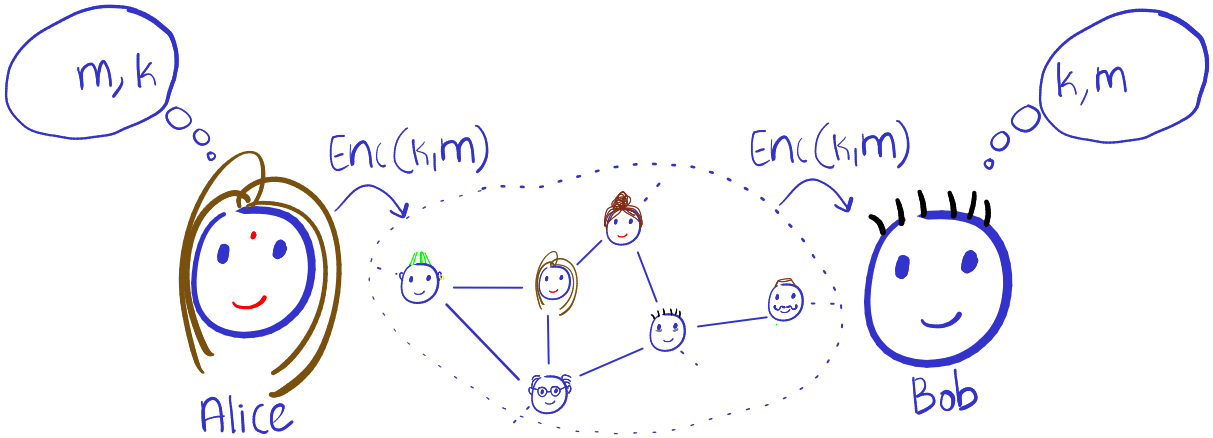
How To Establish a Shared Key in the First Place?



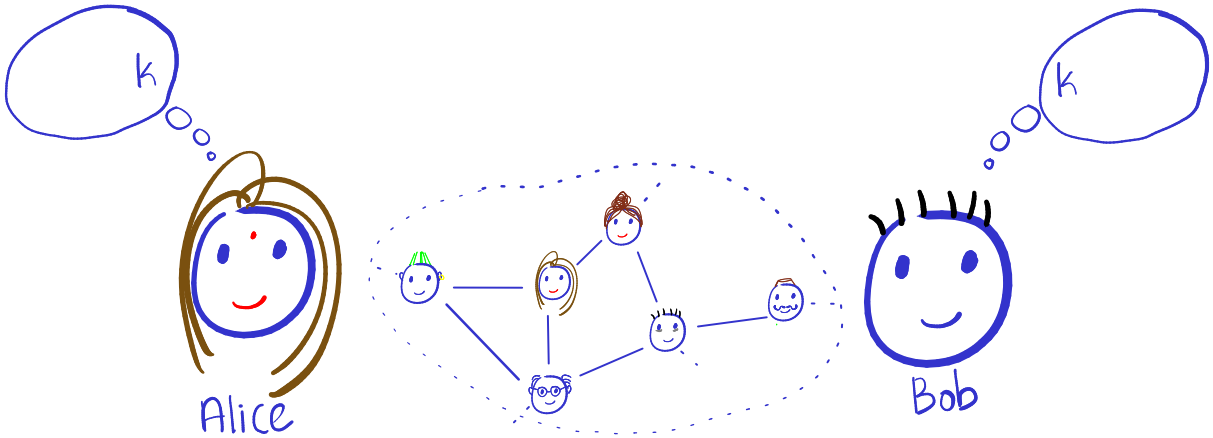
How To Establish a Shared Key in the First Place?



How To Establish a Shared Key in the First Place?

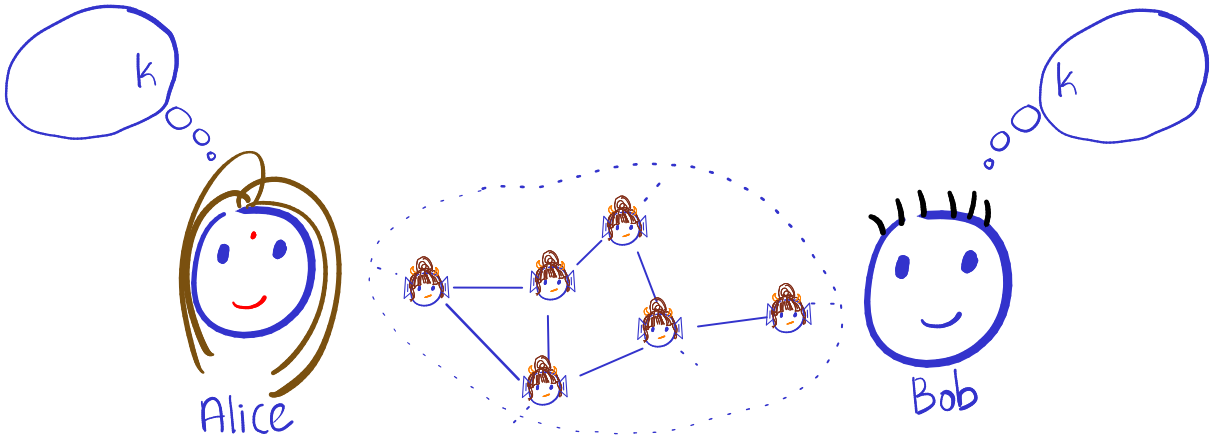


How To Establish a Shared Key in the First Place?



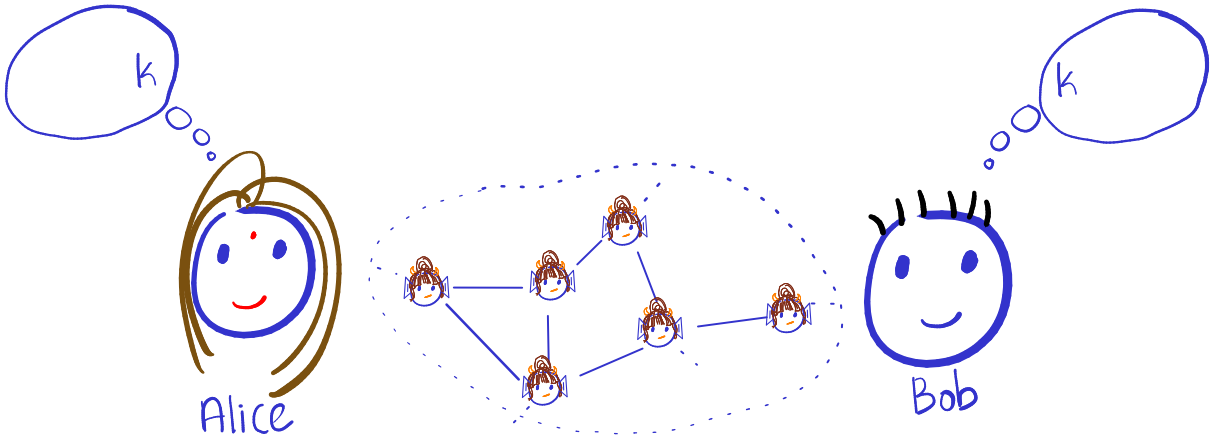
- The setting: Alice and Bob want to establish a shared key $k \in \{0,1\}^n$

How To Establish a Shared Key in the First Place?



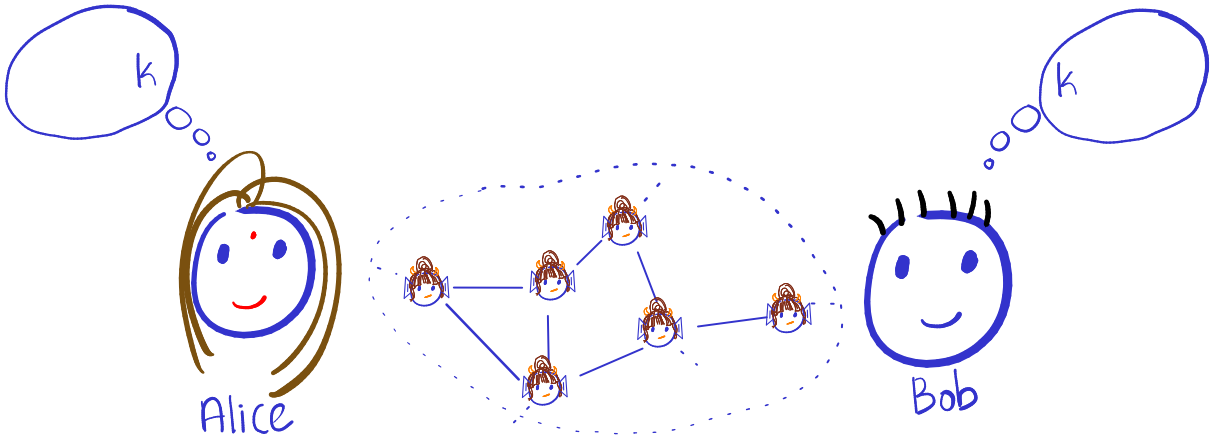
- The setting: Alice and Bob want to establish a shared key $k \in \{0,1\}^n$ in presence of an *eavesdropper* Eve 

How To Establish a Shared Key in the First Place?



- The setting: Alice and Bob want to establish a shared key $k \in \{0,1\}^n$ in presence of an *eavesdropper* Eve 
- Alice and Bob execute a *protocol*, at the end of which they will have established a key

How To Establish a Shared Key in the First Place?



- The setting: Alice and Bob want to establish a shared key $k \in \{0,1\}^n$ in presence of an *eavesdropper* Eve 
- Alice and Bob execute a *protocol*, at the end of which they will have established a key
- Key Exchange IRL: HTTPs, TLS, SSH

Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.

Syntax of Key Exchange Protocol

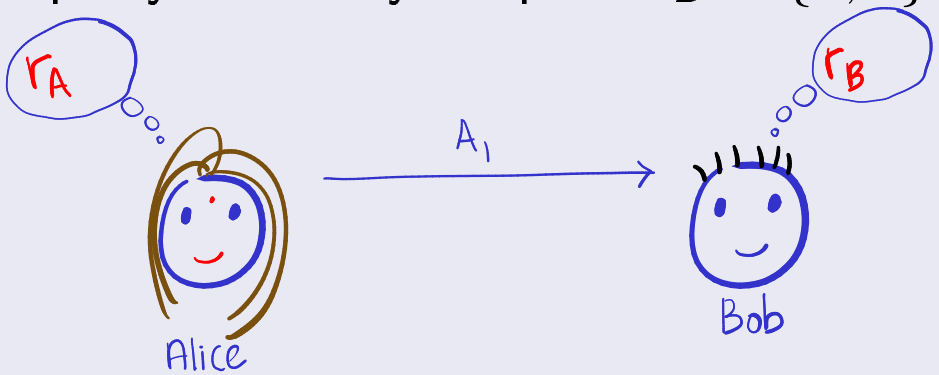
Definition 1 (Key Exchange Protocol)

A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.

Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

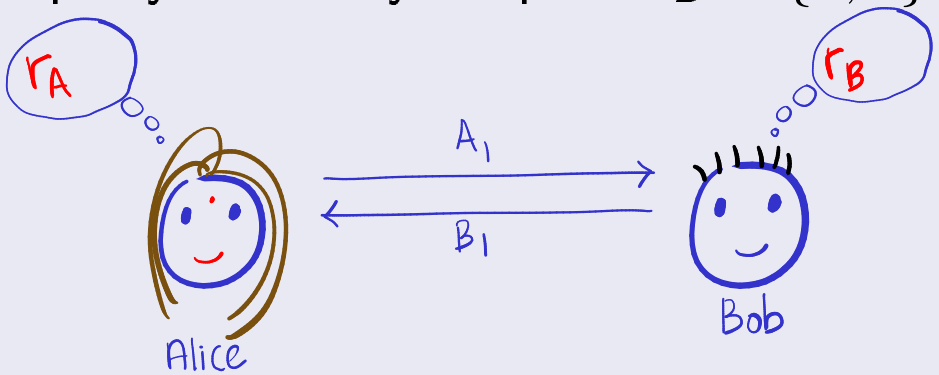
A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.



Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

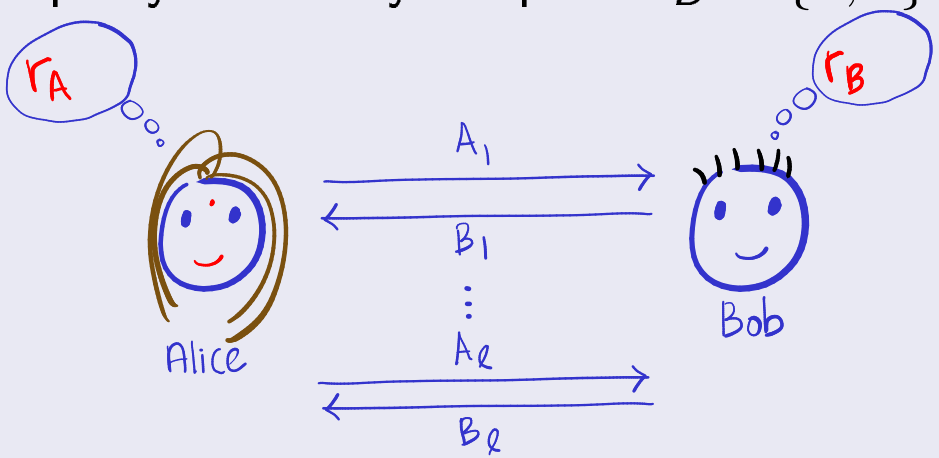
A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.



Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

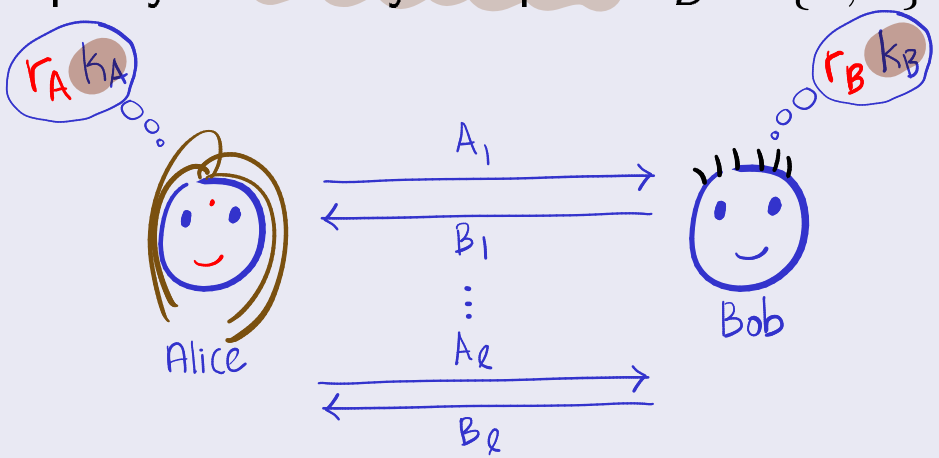
A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.



Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

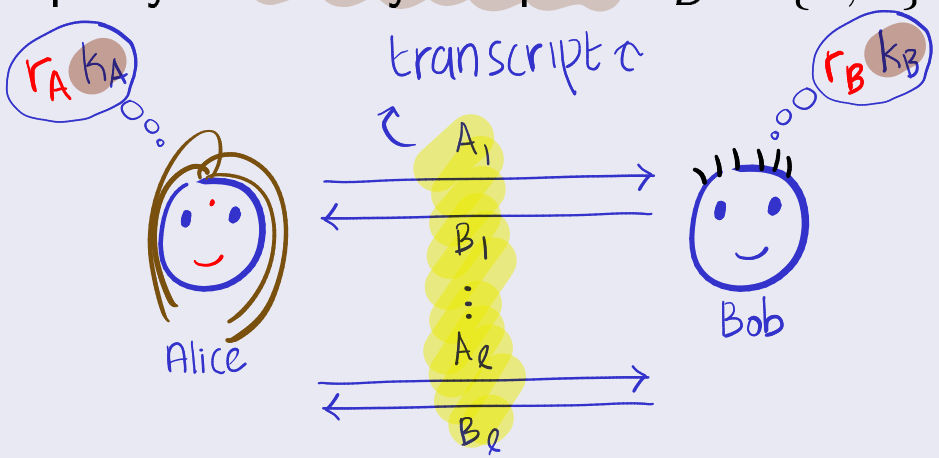
A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.



Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

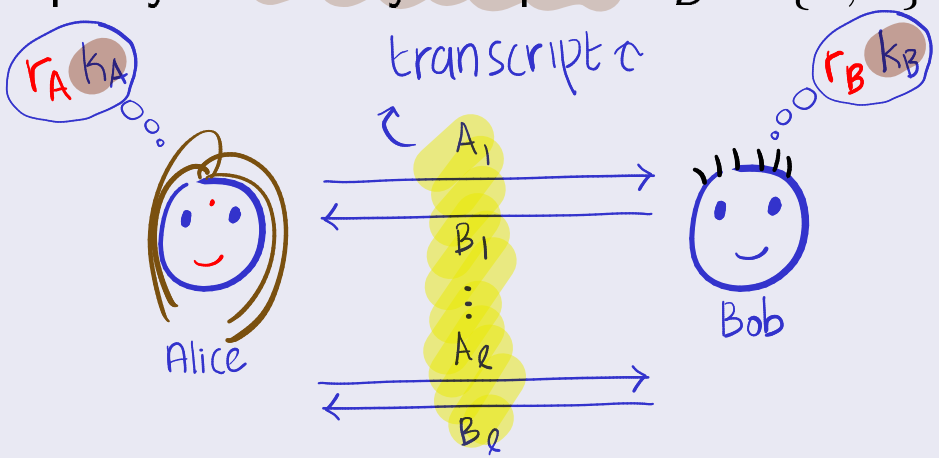
A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.



Syntax of Key Exchange Protocol

Definition 1 (Key Exchange Protocol)

A (two-party) key-exchange protocol Π is a probabilistic protocol between two parties A and B at the end of which party A locally outputs $k_A \in \{0,1\}^n$ and party B locally outputs $k_B \in \{0,1\}^n$.

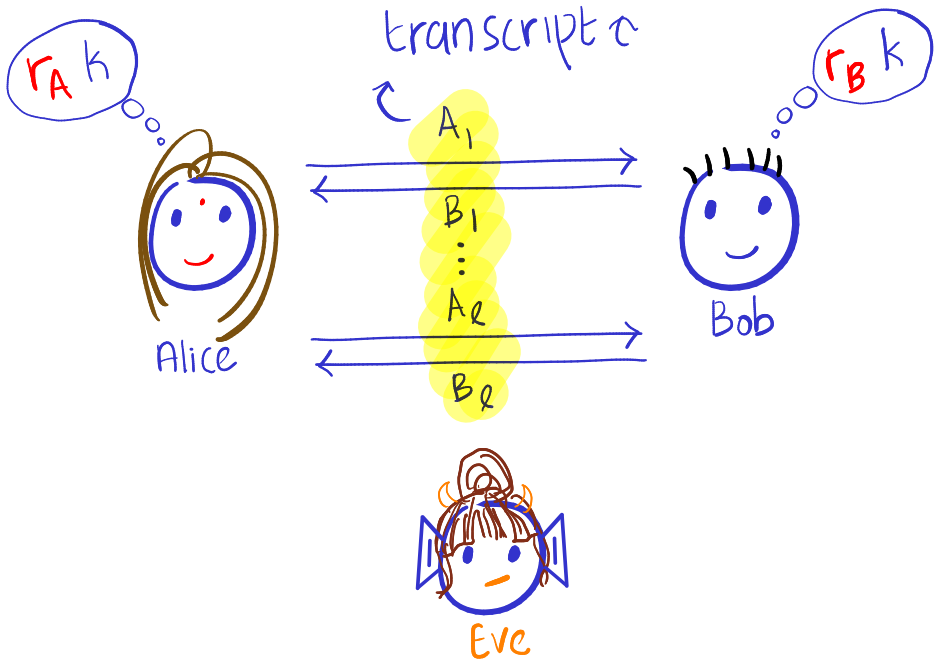


- Correctness of key exchange: for every $n \in \mathbb{N}$

$$\Pr_{(k_A,k_B,\tau)\leftarrow\Pi(1^n)}[k_A = k_B] = 1$$

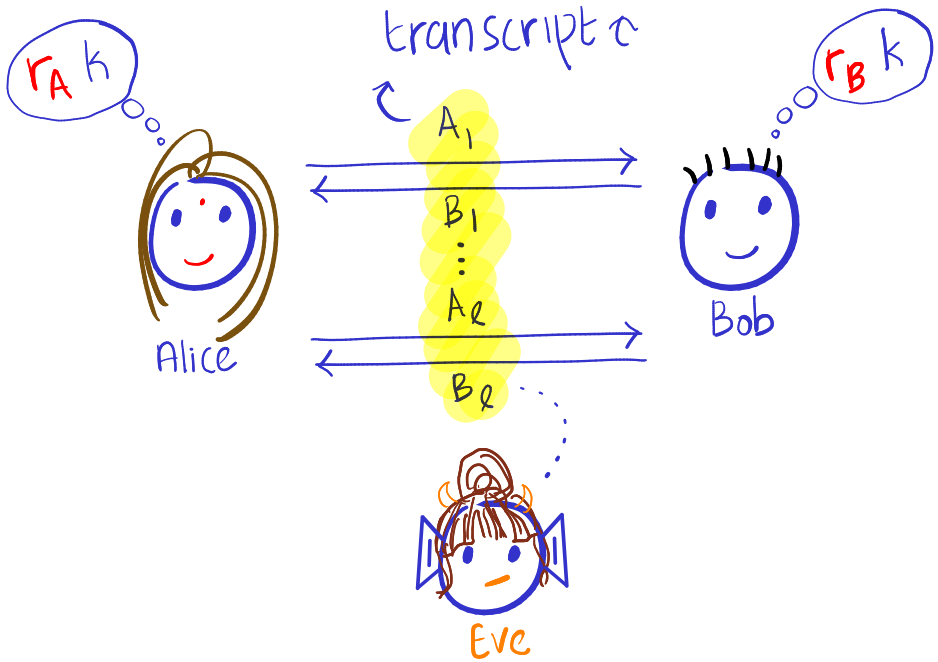
How to Define Security?...

- Intuitively, what is the security requirement?



How to Define Security?...

- Intuitively, what is the security requirement?
  - Key k should be "hidden" given only *transcript* τ of the protocol



How to Define Security?...

- Intuitively, what is the security requirement?
  - Key k should be "hidden" given only *transcript* τ of the protocol

Definition 2 (Secrecy Against Eavesdroppers)

A key-exchange protocol Π is computationally secret against eavesdroppers if for every PPT eavesdropper Eve the following is negligible.

$$\delta(n) := \left| \Pr_{(k,\tau)\leftarrow\Pi(1^n)}[\mathsf{Eve}(\tau, k) = 0] - \Pr_{\substack{(k,\tau)\leftarrow\Pi(1^n) \\ r\leftarrow\{0,1\}^n}}[\mathsf{Eve}(\tau, r) = 0] \right|$$

How to Define Security?...

- Intuitively, what is the security requirement?
  - Key k should be "hidden" given only *transcript* τ of the protocol

Definition 2 (Secrecy Against Eavesdroppers)

A key-exchange protocol Π is computationally secret against eavesdroppers if for every PPT eavesdropper Eve the following is negligible.

$$\delta(n) := \left| \Pr_{(k,\tau)\leftarrow\Pi(1^n)}[\mathrm{Eve}(\tau, k) = 0] - \Pr_{\substack{(k,\tau)\leftarrow\Pi(1^n) \\ r\leftarrow\{0,1\}^n}}[\mathrm{Eve}(\tau, r) = 0] \right|$$

"Real world"

"Random world"

How to Define Security?...

- Intuitively, what is the security requirement?
  - Key k should be "hidden" given only *transcript* τ of the protocol

Definition 2 (Secrecy Against Eavesdroppers)

A key-exchange protocol Π is computationally secret against eavesdroppers if for every PPT eavesdropper Eve the following is negligible.

$$\delta(n) := \left| \Pr_{(k,\tau)\leftarrow\Pi(1^n)}[\mathrm{Eve}(\tau, k) = 0] - \Pr_{\substack{(k,\tau)\leftarrow\Pi(1^n) \\ r\leftarrow\{0,1\}^n}}[\mathrm{Eve}(\tau, r) = 0] \right|$$

"Real world"

"Random world"

Exercise 1

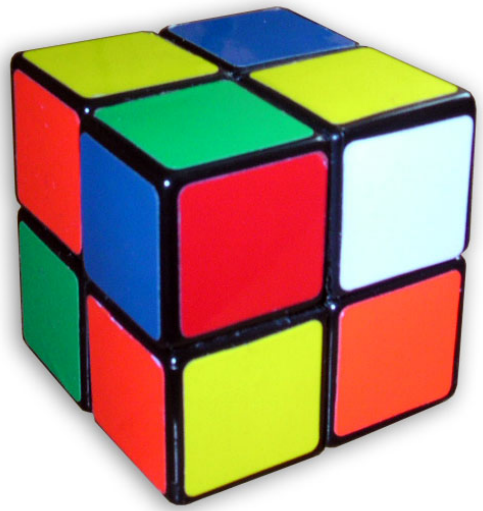
How can an unbounded eavesdropper Eve break secrecy?

Plan for Today's Lecture

- Task: key exchange
- Threat model: computational secrecy against eavesdroppers

Key Exchange

Basic Group Theory



Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Definition 3 (Group axioms) $\forall g_1, g_2 \in \mathcal{G} : g_1 \cdot g_2 \in \mathcal{G}$

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Definition 3 (Group axioms) $\forall g_1, g_2 \in \mathcal{G} : g_1 \cdot g_2 \in \mathcal{G}$

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.

$\forall g_1, g_2, g_3 \in \mathcal{G} : (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Definition 3 (Group axioms) $\forall g_1, g_2 \in \mathcal{G} : g_1 \cdot g_2 \in \mathcal{G}$

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.

$\exists 1 \in \mathcal{G}\ \forall g \in \mathcal{G} : 1 \cdot g = g \cdot 1 = g$

$\forall g_1, g_2, g_3 \in \mathcal{G} : (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Definition 3 (Group axioms) $\forall g_1, g_2 \in \mathcal{G} : g_1 \cdot g_2 \in \mathcal{G}$

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.

$\exists 1 \in \mathcal{G}\ \forall g \in \mathcal{G} : 1 \cdot g = g \cdot 1 = g$

$\forall g_1, g_2, g_3 \in \mathcal{G} : (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$

$\forall g \exists g^{-1} : g \cdot g^{-1} = 1$

Basic Group Theory...

- What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?
  - Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

Definition 3 (Group axioms) $\forall g_1, g_2 \in G : g_1 \cdot g_2 \in G$

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.
$\mathbb{G}$ Abelian if it additionally satisfies 5) commutativity.

$\exists 1 \in G \; \forall g \in G : 1 \cdot g = g \cdot 1 = g$

$\forall g_1, g_2, g_3 \in G : (g_1 \cdot g_2) \cdot g_3 = g_1 \cdot (g_2 \cdot g_3)$

$\forall g \exists g^{-1} : g \cdot g^{-1} = 1$

$\forall g_1, g_2 \in G : g_1 \cdot g_2 = g_2 \cdot g_1$

Basic Group Theory...

What are some properties of $(\{0,1\}^n, \oplus)$ we have exploited?

- Closure of $\oplus$, self-inverse ($k \oplus k = 0^n$), associativity?

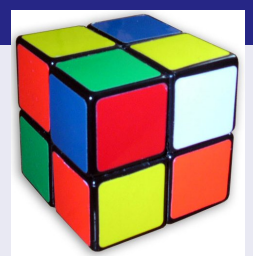
Definition 3 (Group axioms)

A group $\mathbb{G}$ is a set $\mathcal{G}$ with a binary operation $\cdot$ satisfying: 1) closure 2) associativity, 3) existence of identity and 4) existence of inverse.
$\mathbb{G}$ Abelian if it additionally satisfies 5) commutativity.

Abstracts properties of integers $\mathbb{Z} := (\{\cdots, -1, 0, 1, \cdots\}, +)$

Exercise 2

- Show that $\mathbb{Z} := (\{\cdots, -1, 0, 1, \cdots\}, +)$ is a group
- What is the group corresponding to $2 \times 2 \times 2$ Rubik's cube?
  - Describe the set $\mathcal{G}$ and the operation $\cdot$



Definition 4 (Group terminology)

- Order of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer

Definition 4 (Group terminology)

- Order of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- Order of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$

(ℓ-1 times)

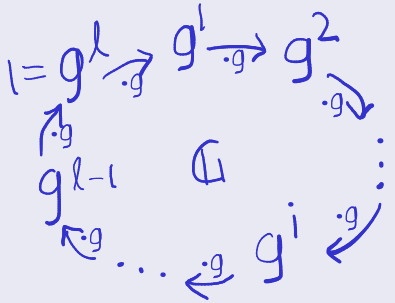
Basic Group Theory...

Definition 4 (Group terminology)

- Order of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- Order of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$ (ℓ-1 times)
- Cyclic group: there exists a "generator" $g \in \mathcal{G}$ with order $\ell = |\mathcal{G}|$
  - That is $\{g^1 = g, g^2, \ldots, g^{\ell-1}, g^\ell = 1\} = \mathcal{G}$

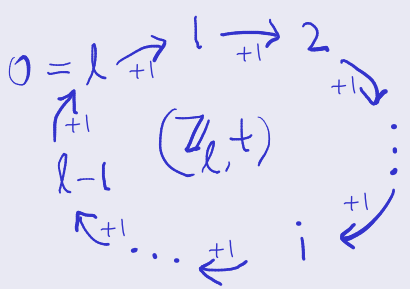
Definition 4 (Group terminology)

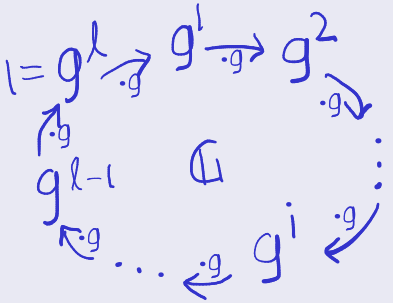
- Order of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- Order of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$ (ℓ-1 times)
- Cyclic group: there exists a "generator" $g \in \mathcal{G}$ with order $\ell = |\mathcal{G}|$
  - That is $\{g^1 = g, g^2, \ldots, g^{\ell-1}, g^\ell = 1\} = \mathcal{G}$



Definition 4 (Group terminology)

- **Order** of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- **Order** of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$ ($\ell - 1$ times)
- Cyclic group: there exists a "generator" $g \in \mathcal{G}$ with order $\ell = |\mathcal{G}|$
  - That is $\{g^1 = g, g^2, \ldots, g^{\ell-1}, g^\ell = 1\} = \mathcal{G}$



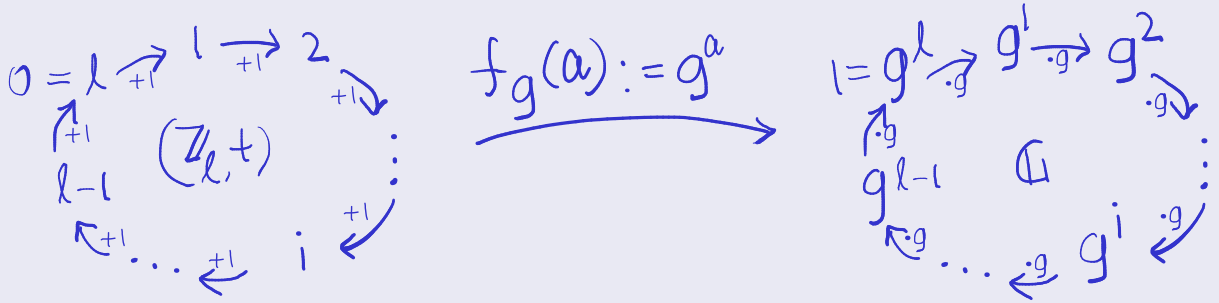


  - "Isomorphism" between $(\mathbb{Z}_\ell, +)$ and $\mathbb{G}$

Is $(\{0,1\}^n, \oplus)$ cyclic? What is the maximum order of any element?

Definition 4 (Group terminology)

- **Order** of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- **Order** of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$ ($\ell - 1$ times)
- Cyclic group: there exists a "generator" $g \in \mathcal{G}$ with order $\ell = |\mathcal{G}|$
  - That is $\{g^1 = g, g^2, \ldots, g^{\ell-1}, g^\ell = 1\} = \mathcal{G}$

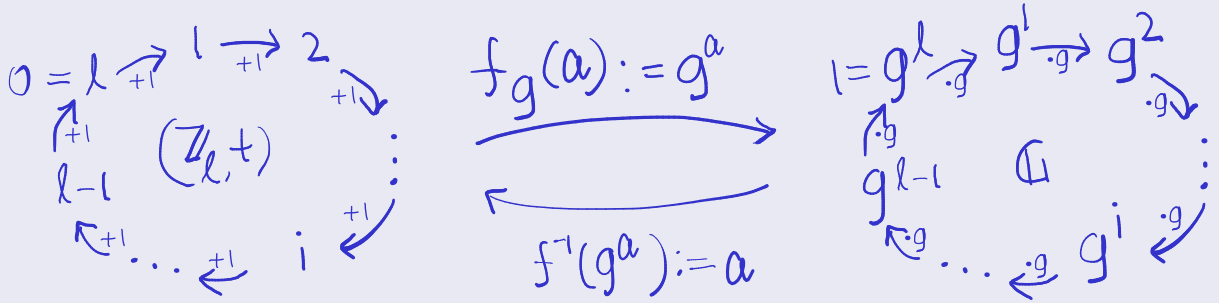


  - "Isomorphism" between $(\mathbb{Z}_\ell, +)$ and $\mathbb{G}$

Is $(\{0,1\}^n, \oplus)$ cyclic? What is the maximum order of any element?

Definition 4 (Group terminology)

- Order of the group, $|\mathcal{G}|$.
  - We're interested in groups of *finite* order
  - Can be represented on a digital computer
- Order of an element g: smallest $\ell \in \mathbb{N}$ such that $g^\ell := g \cdot \ldots \cdot g = 1$ ($\ell - 1$ times)
- Cyclic group: there exists a "generator" $g \in \mathcal{G}$ with order $\ell = |\mathcal{G}|$
  - That is $\{g^1 = g, g^2, \ldots, g^{\ell-1}, g^\ell = 1\} = \mathcal{G}$



  - "Isomorphism" between $(\mathbb{Z}_\ell, +)$ and $\mathbb{G}$

Is $(\{0,1\}^n, \oplus)$ cyclic? What is the maximum order of any element?

Basic Group Theory...

Exercise 3 (Lagrange's theorem)

Prove that the order of an element divides order of the (finite) group.

Exercise 4

For a group $\mathbb{G}$ of order ℓ with generator g, show using group axioms that for all $a, b \in \mathbb{Z}_\ell$, $(g^a)^b = g^{ab} = (g^b)^a$

Exercise 5

Prove that a prime-order group is cyclic. Are all cyclic groups of prime order?

Some Examples of (Finite) Groups

Addition modulo prime p

$(\mathbb{Z}_p, +)$

$\{0, \cdots p-1\}$ $\quad g_1 + g_2 := g_1 + g_2 \pmod p$

Some Examples of (Finite) Groups

Addition modulo prime p

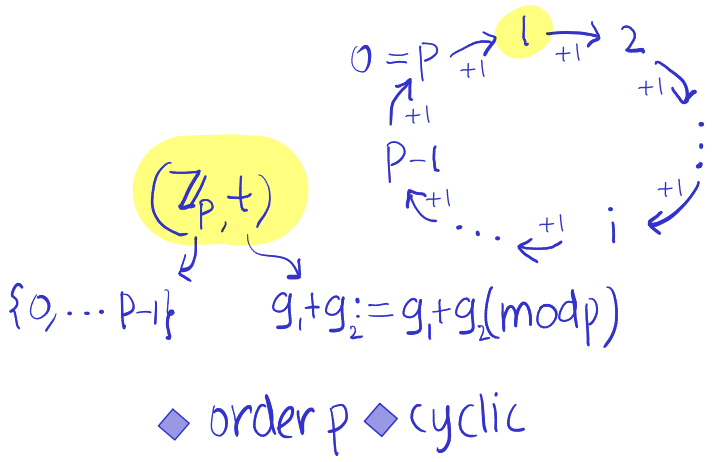
$(\mathbb{Z}_p, +)$

$\{0, \cdots p-1\}$ $\quad g_1 + g_2 := g_1 + g_2 \pmod p$

- order p
- cyclic

Some Examples of (Finite) Groups

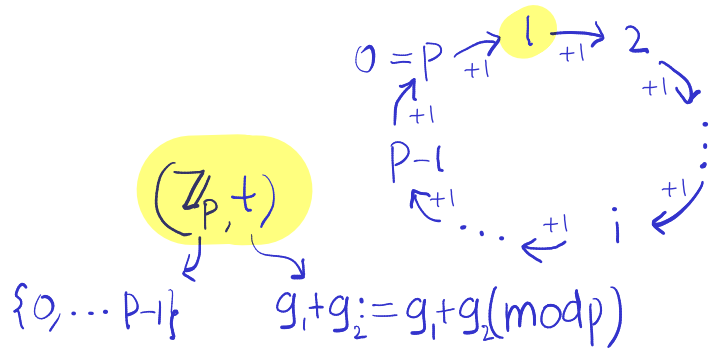
Addition modulo prime p



- order p
- cyclic

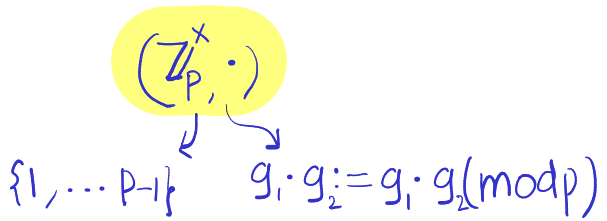
Some Examples of (Finite) Groups

Addition modulo prime p



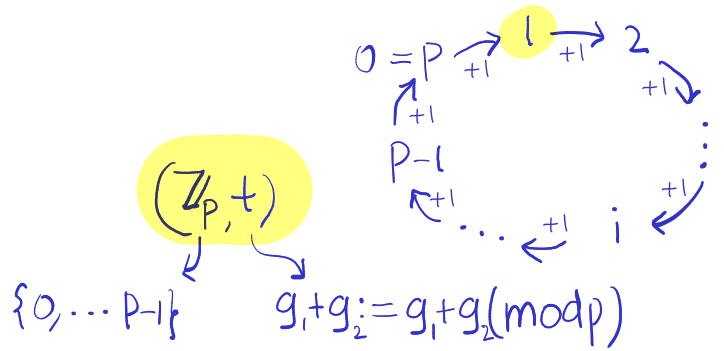
- order p
- cyclic

Multiplication modulo prime p



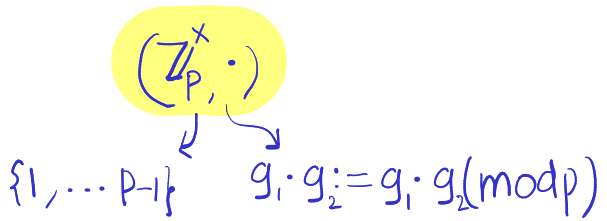
Some Examples of (Finite) Groups

Addition modulo prime p



- order p
- cyclic

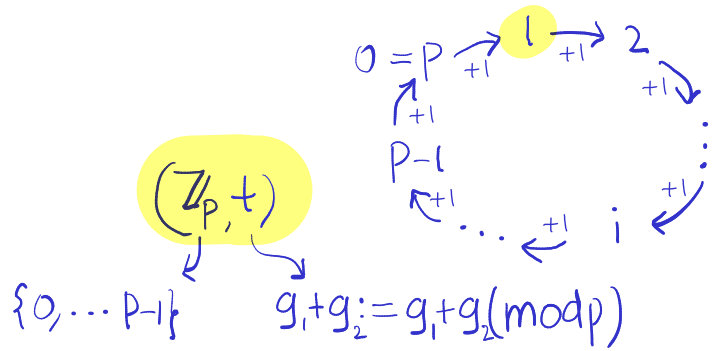
Multiplication modulo prime p



- order $p-1$
- cyclic

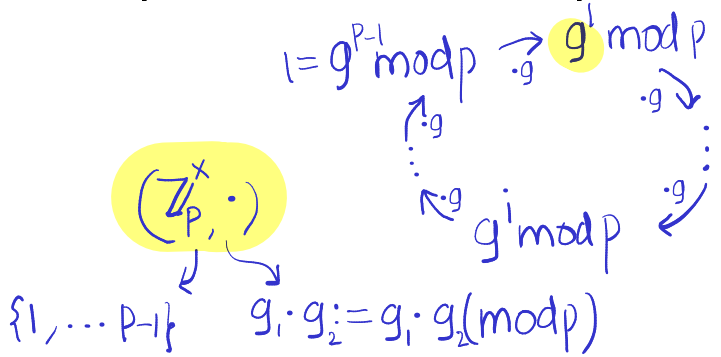
Some Examples of (Finite) Groups

Addition modulo prime p



- order p
- cyclic

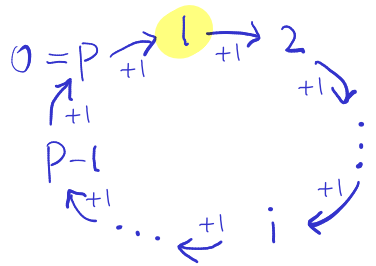
Multiplication modulo prime p



- order $p-1$
- cyclic

Some Examples of (Finite) Groups

Addition modulo prime p

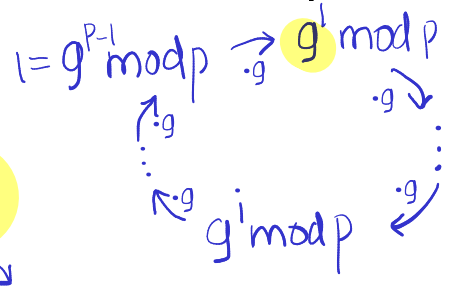


$(\mathbb{Z}_p, +)$

$\{0, \dots p-1\}$ $\quad g_1 + g_2 := g_1 + g_2 \pmod p$

- order p
- cyclic

Multiplication modulo prime p



$(\mathbb{Z}_p^\times, \cdot)$

$\{1, \dots p-1\}$ $\quad g_1 \cdot g_2 := g_1 \cdot g_2 \pmod p$

- order $p-1$
- cyclic

Multiplication modulo $N = pq$

primes

$(\mathbb{Z}_N^\times, \cdot)$

$\{1, \dots p-1,$
$p+1, \dots, 2p-1,$
$\dots pq-1\}$

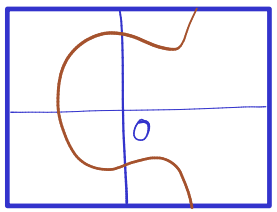
$g_1 \cdot g_2 := g_1 \cdot g_2 \pmod N$

- order $(p-1)(q-1)$
- not cyclic

Elliptic curves modulo prime p

$(E_p, +)$ "curve points addition"

solutions to
$y^2 = x^3 + Ax + B \pmod p$



- $|p+1 - \text{order}| \leq 2\sqrt{p}$
- cyclic

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How? 

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How? 
    1. Compute $g := g_1 \cdot g_2$ (over $\mathbb{Z}$) using integer multiplication
    2. *Reduce* g mod p using integer division


Computable in $\tilde{O}(n)$ time, where $n := \|p\|$

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How?
    1. Compute $g := g_1 \cdot g_2$ (over $\mathbb{Z}$) using integer multiplication
    2. *Reduce* g mod p using integer division

  Computable in $\tilde{O}(n)$ time, where $n := \|p\|$
- Modular exponentiation: $g^a \bmod p$
  - Reduces to modular multiplication. How?

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How?
    1. Compute $g := g_1 \cdot g_2$ (over $\mathbb{Z}$) using integer multiplication
    2. *Reduce* g mod p using integer division

  Computable in $\tilde{O}(n)$ time, where $n := \|p\|$
- Modular exponentiation: $g^a \bmod p$
  - Reduces to modular multiplication. How?
    - Square and multiply (and reduce) algorithm

  Computable in $\tilde{O}(n^2)$ time
- Modular inverse: $g^{-1} \bmod p$
  - Claim: reduces to finding $a, b \in \mathbb{Z}$ such that $ag + bp = 1$. Why?

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How?
    1. Compute $g := g_1 \cdot g_2$ (over $\mathbb{Z}$) using integer multiplication
    2. *Reduce* g mod p using integer division

  Computable in $\tilde{O}(n)$ time, where $n := \|p\|$
- Modular exponentiation: $g^a \bmod p$
  - Reduces to modular multiplication. How?
    - Square and multiply (and reduce) algorithm

  Computable in $\tilde{O}(n^2)$ time
- Modular inverse: $g^{-1} \bmod p$
  - Claim: reduces to finding $a, b \in \mathbb{Z}$ such that $ag + bp = 1$. Why?
  - Can use Extended Euclidean Algorithm to compute a and b

    Computable in $\tilde{O}(n^2)$ time

What is Easy to Compute Over Cyclic Groups?...

Let's focus on $\mathbb{Z}_p^* = (\{0, \ldots, p-1\}, \cdot)$

- Modular multiplication $g_1 \cdot g_2 \bmod p$
  - Reduces to integer operations. How?
    1. Compute $g := g_1 \cdot g_2$ (over $\mathbb{Z}$) using integer multiplication
    2. *Reduce* g mod p using integer division

  Computable in $\tilde{O}(n)$ time, where $n := \|p\|$
- Modular exponentiation: $g^a \bmod p$
  - Reduces to modular multiplication. How?
    - Square and multiply (and reduce) algorithm

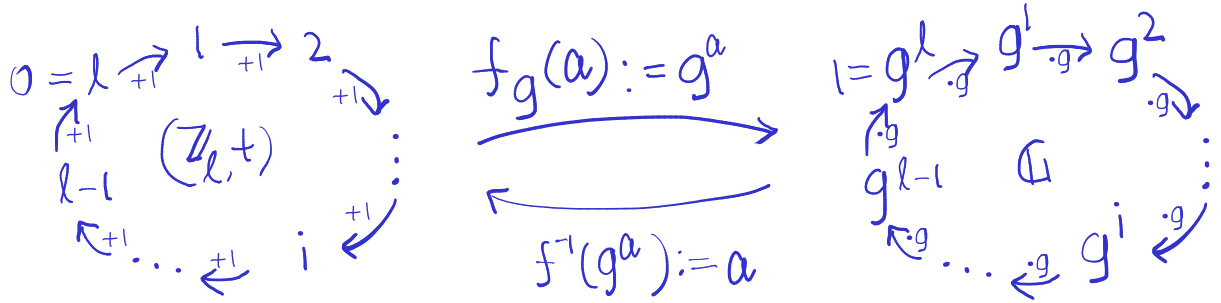
  Computable in $\tilde{O}(n^2)$ time
- Modular inverse: $g^{-1} \bmod p$
  - Claim: reduces to finding $a, b \in \mathbb{Z}$ such that $ag + bp = 1$. Why?
  - Can use Extended Euclidean Algorithm to compute a and b

    Computable in $\tilde{O}(n^2)$ time

In general: group operation, exponentiation and inverse efficient

What is Hard to Compute Over Cyclic Groups?...

- Recall the exponentiation map for cyclic group



What is Hard to Compute Over Cyclic Groups?...

Recall the exponentiation map for cyclic group

Definition 5 (Discrete logarithm (DLog) problem in $\mathbb{G}$ w.r.to S)

- Input: order, generator
  1. $(\mathbb{G}, \ell, g)$ sampled by a group sampler $\mathsf{S}(1^n)$
  2. $h := g^a$ for $a \leftarrow \mathbb{Z}_\ell$
- Solution: a

What is Hard to Compute Over Cyclic Groups?...

Recall the exponentiation map for cyclic group

Definition 5 (Discrete logarithm (DLog) problem in $\mathbb{G}$ w.r.to S)

- Input: (order, generator)
  1. $(\mathbb{G}, \ell, g)$ sampled by a group sampler $\mathsf{S}(1^n)$
  2. $h := g^a$ for $a \leftarrow \mathbb{Z}_\ell$
- Solution: a

Assumption 1 (DLog assumption in $\mathbb{G}$ w.r.to S...)

... holds if solving the DLog problem in $\mathbb{G}$ w.r.to S is hard for all PPT inverters Inv. That is, for all Inv, the following is negligible:

$$\delta(n) := \Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow\mathsf{S}(1^n) \\ a\leftarrow\mathbb{Z}_\ell}}[\mathsf{Inv}((\mathbb{G},\ell,g), g^a) = a]$$

What is Hard to Compute Over Cyclic Groups?...

Recall the exponentiation map for cyclic group

Definition 5 (Discrete logarithm (DLog) problem in $\mathbb{G}$ w.r.to S)

- Input:
  1. $(\mathbb{G}, \ell, g)$ sampled by a group sampler $\mathsf{S}(1^n)$
  2. $h := g^a$ for $a \leftarrow \mathbb{Z}_\ell$
- Solution: a

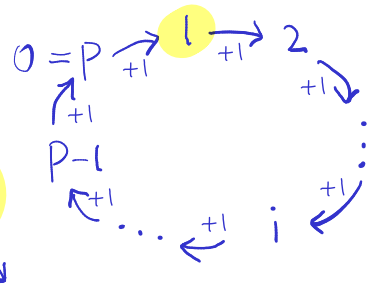
Assumption 1 (DLog assumption in $\mathbb{G}$ w.r.to S...)

... holds if solving the DLog problem in $\mathbb{G}$ w.r.to S is hard for all PPT inverters Inv. That is, for all Inv, the following is negligible:

$$\delta(n) := \Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow\mathsf{S}(1^n) \\ a\leftarrow\mathbb{Z}_\ell}}[\mathsf{Inv}((\mathbb{G},\ell,g), g^a) = a]$$

What is Hard to Compute Over Cyclic Groups?...

Addition modulo prime p

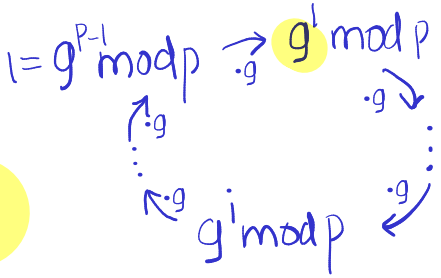


$(\mathbb{Z}_p, +)$

$\{0, \dots p-1\}$ $\quad g_1 + g_2 := g_1 + g_2 \pmod p$

◆ order p ◆ cyclic

Multiplication modulo prime p



$(\mathbb{Z}_p^\times, \cdot)$

$\{1, \dots p-1\}$ $\quad g_1 \cdot g_2 := g_1 \cdot g_2 \pmod p$

◆ order $p-1$ ◆ cyclic

Multiplication modulo $N = pq$

primes

$(\mathbb{Z}_N^\times, \cdot)$

$\{1, \dots p-1, p+1, \dots, 2p-1, \dots pq-1\}$ $\quad g_1 \cdot g_2 := g_1 \cdot g_2 \pmod N$

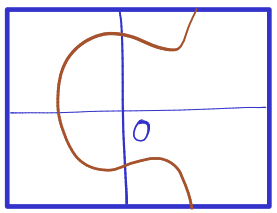
◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

$(E, +)$ "curve points addition"

solutions to

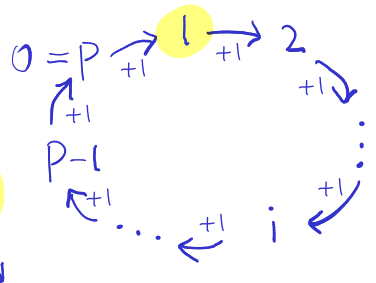
$y^2 = x^3 + Ax + B \pmod p$



◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

What is Hard to Compute Over Cyclic Groups?...

Addition modulo prime p



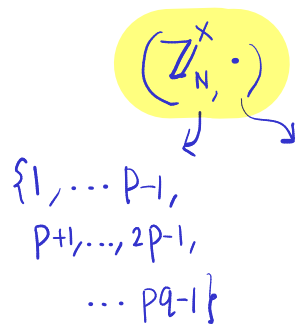
- order p
- cyclic

Multiplication modulo prime p

- order $p-1$
- cyclic

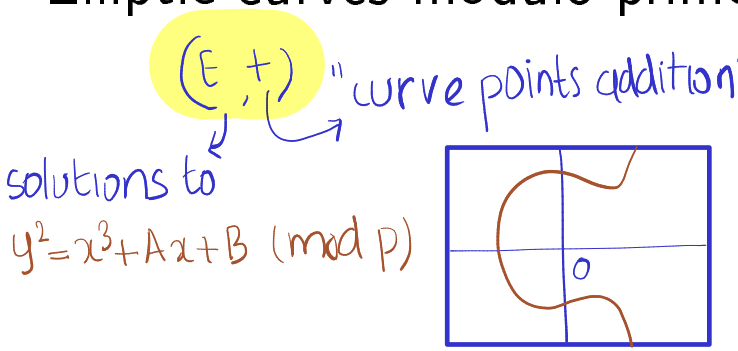
Multiplication modulo $N = pq$

primes



- order $(p-1)(q-1)$
- not cyclic

Elliptic curves modulo prime p

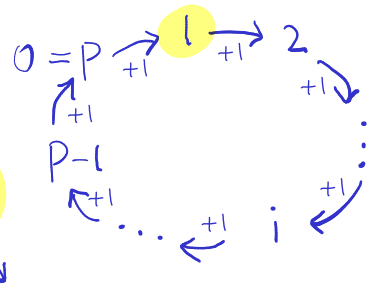


- $|p+1 - \text{order}| \leq 2\sqrt{p}$
- cyclic

What is Hard to Compute Over Cyclic Groups?...

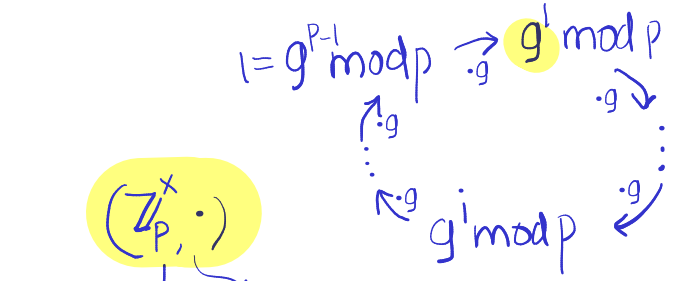
? Easy! Division modulo p !

Addition modulo prime p



◆ order p ◆ cyclic

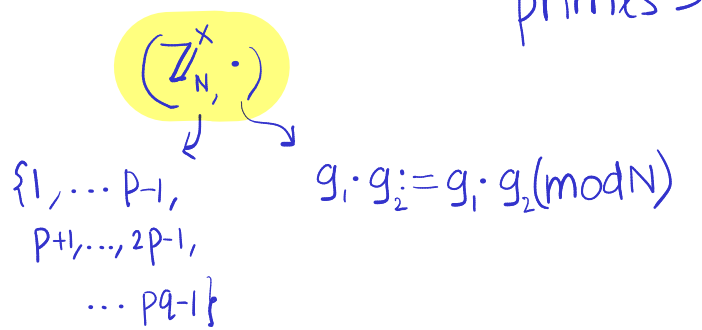
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

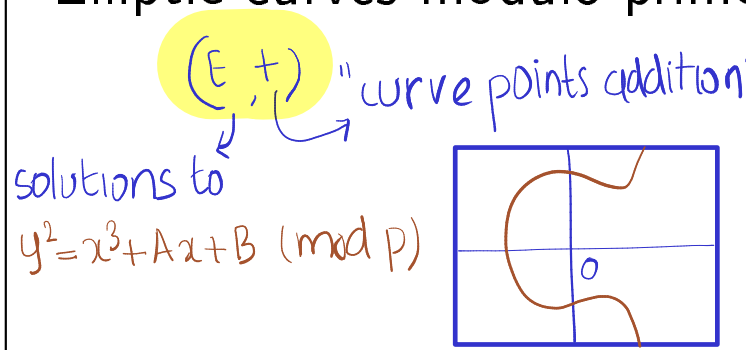
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

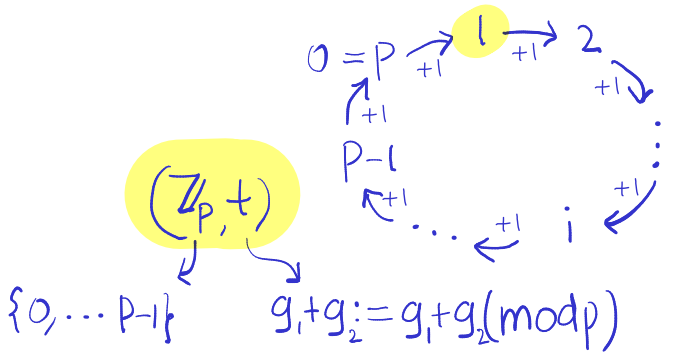


◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

What is Hard to Compute Over Cyclic Groups?...

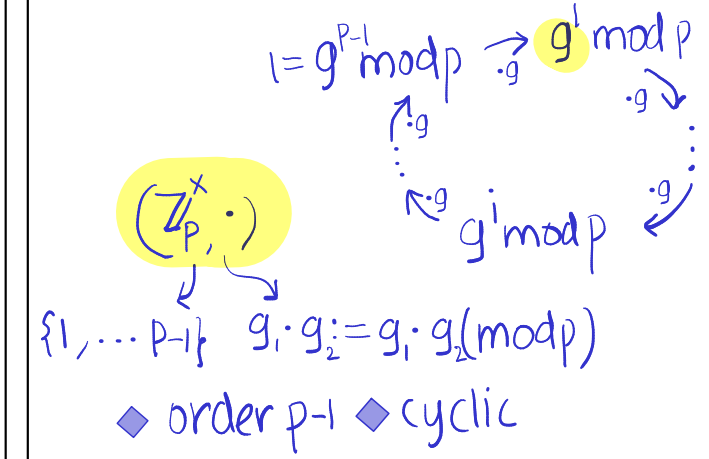
? Easy! Division modulo p ⚠ ?

Addition modulo prime p



◆ order p ◆ cyclic

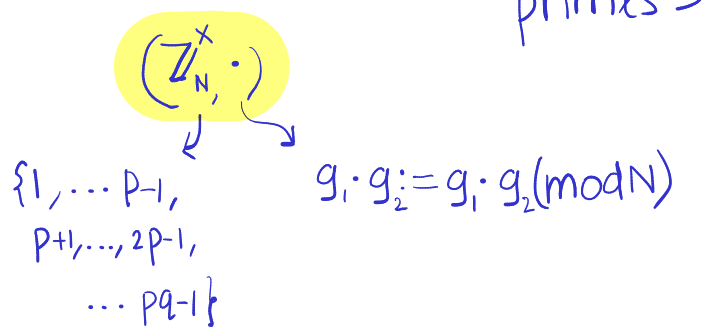
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

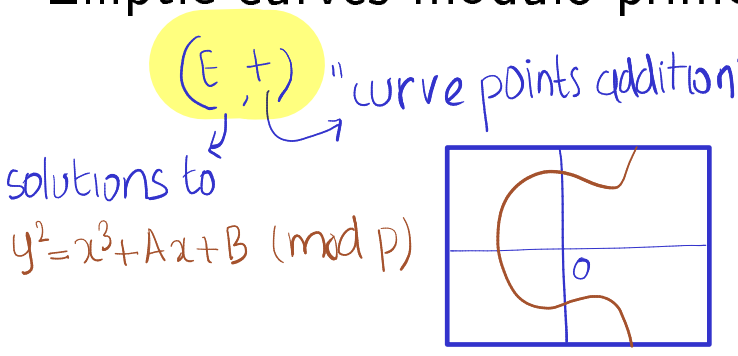
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p



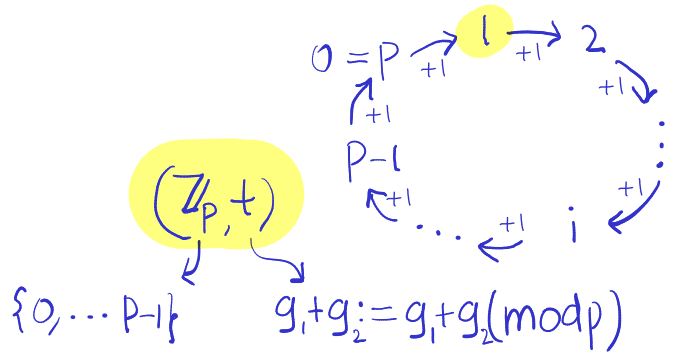
◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

What is Hard to Compute Over Cyclic Groups?...

? Easy! Division modulo p ⚠

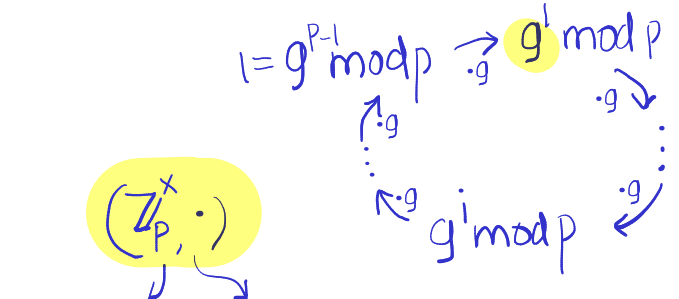
? Believed hard; ∃ sub-exponential algos

Addition modulo prime p



◆ order p ◆ cyclic

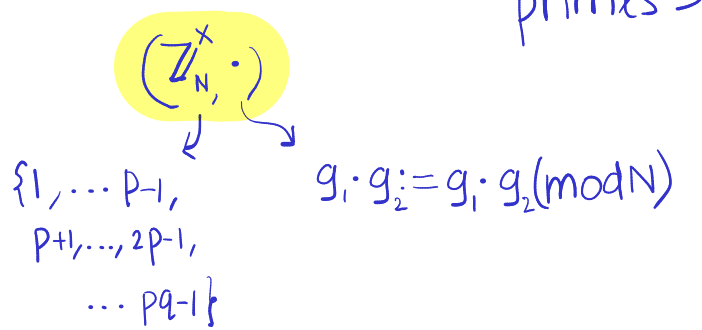
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

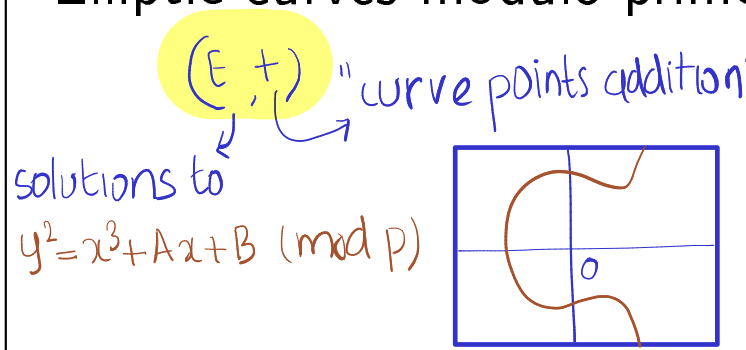
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

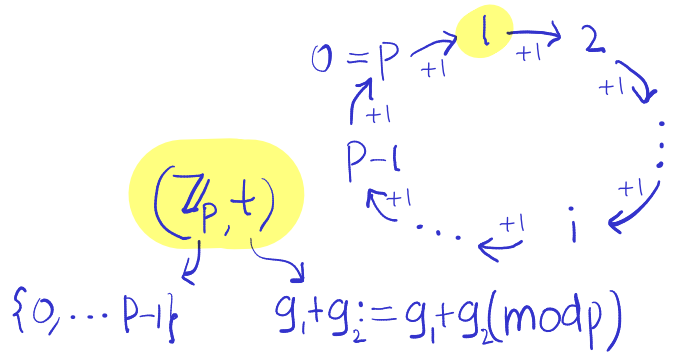


◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

What is Hard to Compute Over Cyclic Groups?...

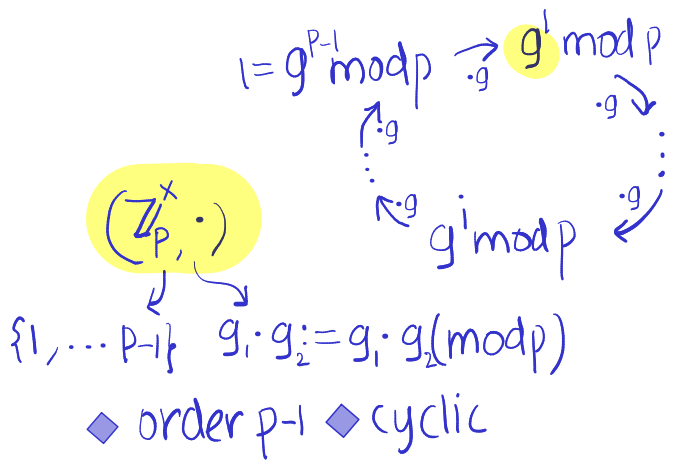
? Easy! Division modulo p ⚠ ? Believed hard; ∃ sub-exponential algos

Addition modulo prime p



◆ order p ◆ cyclic

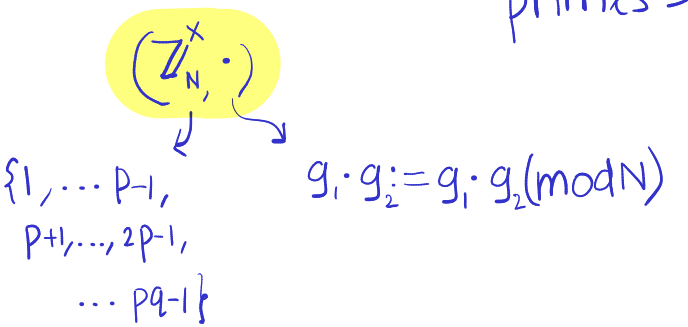
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

Multiplication modulo $N = pq$

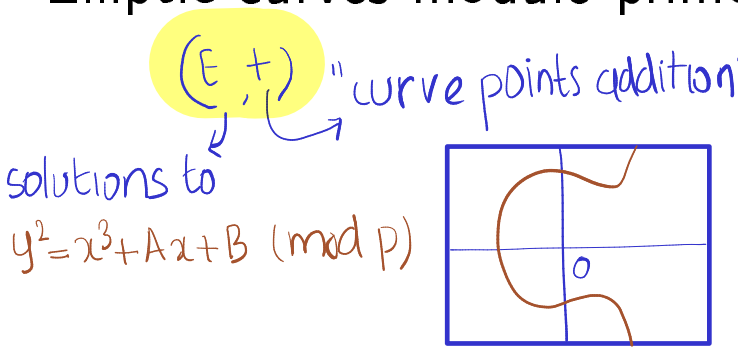
primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Hard in its cyclic subgroup

Elliptic curves modulo prime p



◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

Believed hard; no known sub-exp. algo

Efficient Group Samplers Exist

E.g.: $(\mathbb{Z}_p^*, p, g) \leftarrow \mathsf{S}(1^n)$

1. Sample random prime p such that $\|p\| \approx n$
   1. Sample random *integer* p such that $\|p\| \approx n$
   2. Test whether p prime using (say) Miller-Rabin text
      - ⏱ Randomised test, runs in roughly $\tilde{O}(n^3)$ time for negligible error

Efficient Group Samplers Exist

$$\text{E.g.: } (\mathbb{Z}_p^*, p, g) \leftarrow \mathsf{S}(1^n)$$

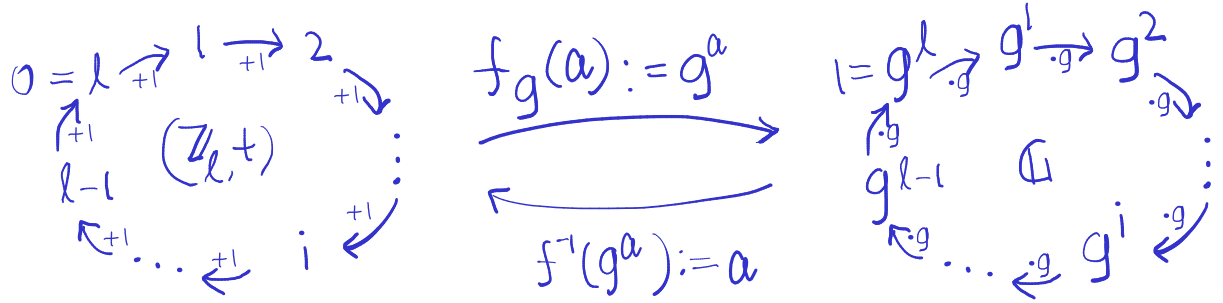
1. Sample random prime p such that $\|p\| \approx n$
   1. Sample random *integer* p such that $\|p\| \approx n$
   2. Test whether p prime using (say) Miller-Rabin text
      - ⏱ Randomised test, runs in roughly $\tilde{O}(n^3)$ time for negligible error
2. Sample a random generator
   1. Sample a random $g \in \{0, \ldots, p-1\}$
   2. Test whether g is a generator
      - ⏱ Efficient if factorisation of $p-1$ known

Efficient Group Samplers Exist

$$\text{E.g.: } (\mathbb{Z}_p^*, p, g) \leftarrow \mathsf{S}(1^n)$$

1. Sample random prime p such that $\|p\| \approx n$
   1. Sample random *integer* p such that $\|p\| \approx n$
   2. Test whether p prime using (say) Miller-Rabin text
      - ⏱ Randomised test, runs in roughly $\tilde{O}(n^3)$ time for negligible error
2. Sample a random generator
   1. Sample a random $g \in \{0, \ldots, p-1\}$
   2. Test whether g is a generator
      - ⏱ Efficient if factorisation of $p-1$ known

- Note: sample random generator $\implies$ sample random group element via "isomorphism"



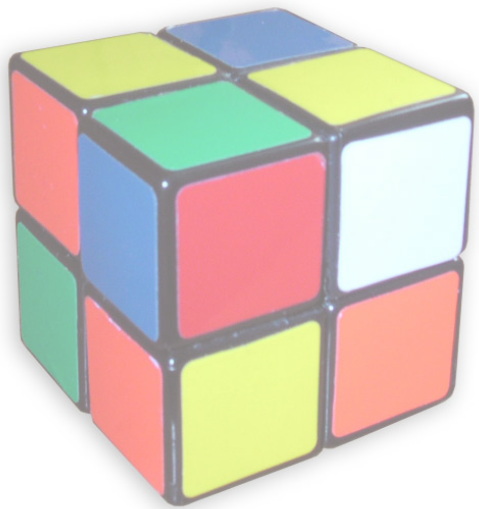
Plan for Today's Lecture

- Task: key exchange
- Threat model: computational secrecy against eavesdroppers

Key Exchange

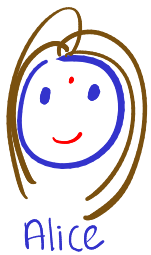
©Wikipedia

Basic Group Theory



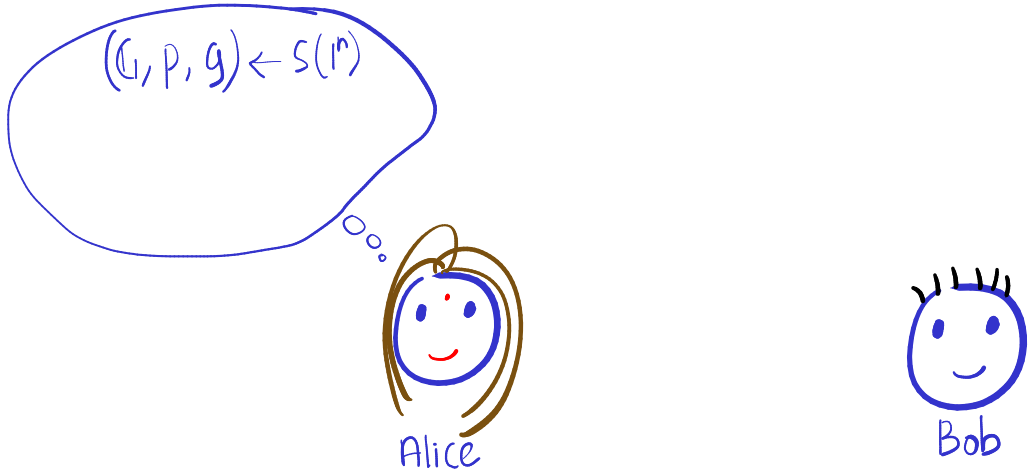
©Mike Gonzalez/Wikipedia

Diffie-Hellman Key-Exchange Protocol



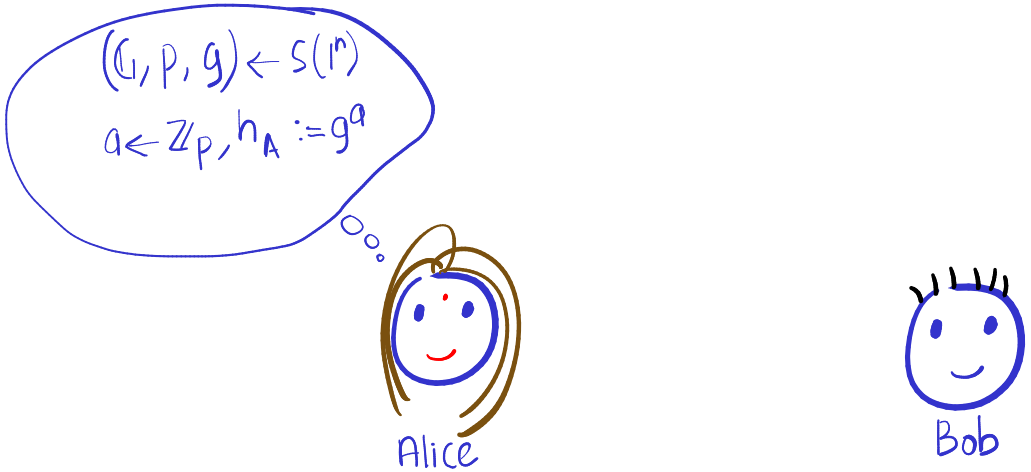
Protocol 1

Diffie-Hellman Key-Exchange Protocol



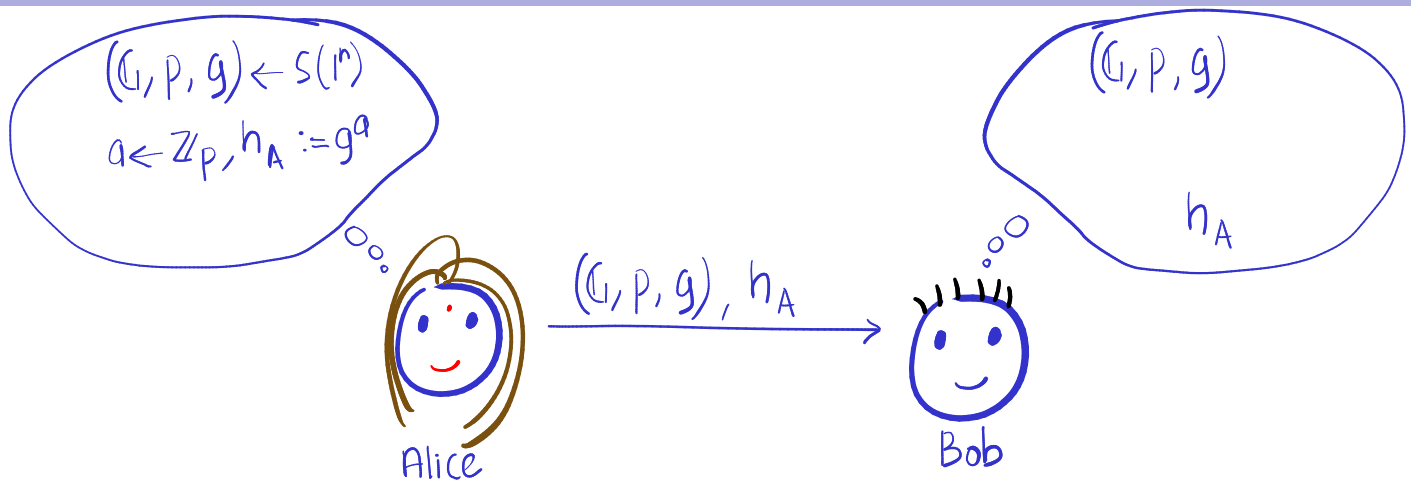
Protocol 1

Diffie-Hellman Key-Exchange Protocol



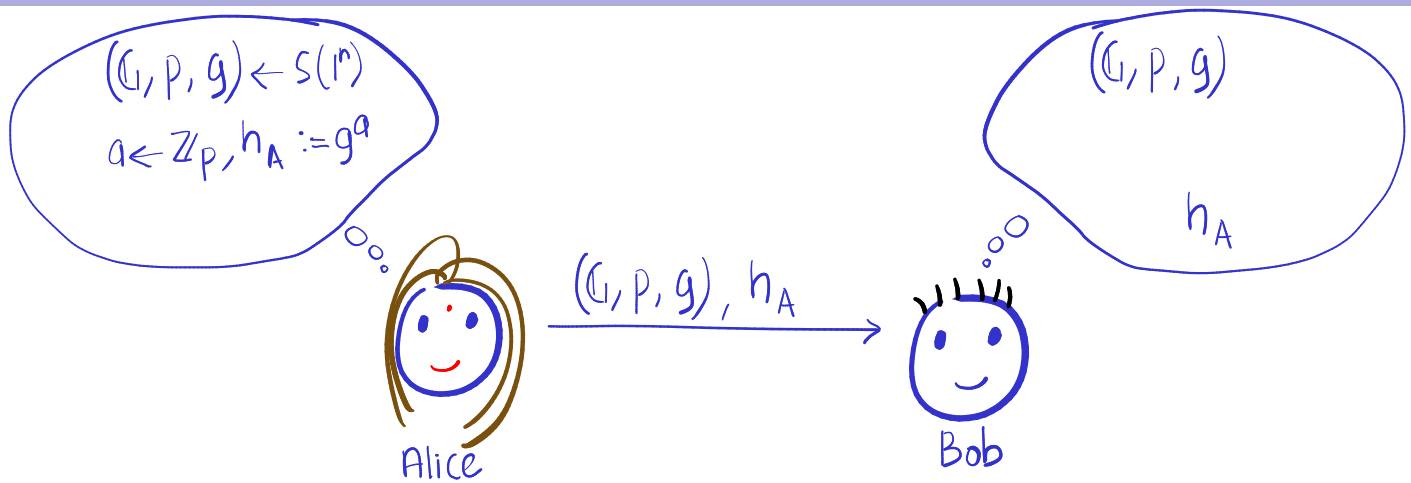
Protocol 1

Diffie-Hellman Key-Exchange Protocol



Protocol 1

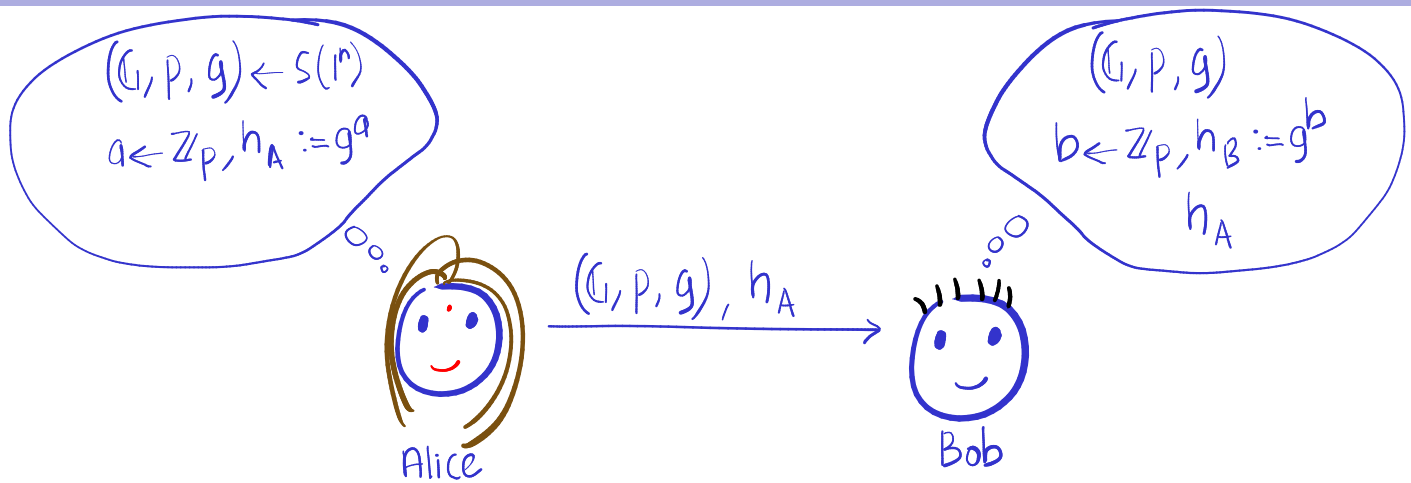
Diffie-Hellman Key-Exchange Protocol



Protocol 1

1. Alice→Bob: Send $((\mathbb{G}, \ell, g), h_A := g^a)$, where $(\mathbb{G}, \ell, g) \leftarrow S(1^n)$ and $a \leftarrow \mathbb{Z}_\ell$

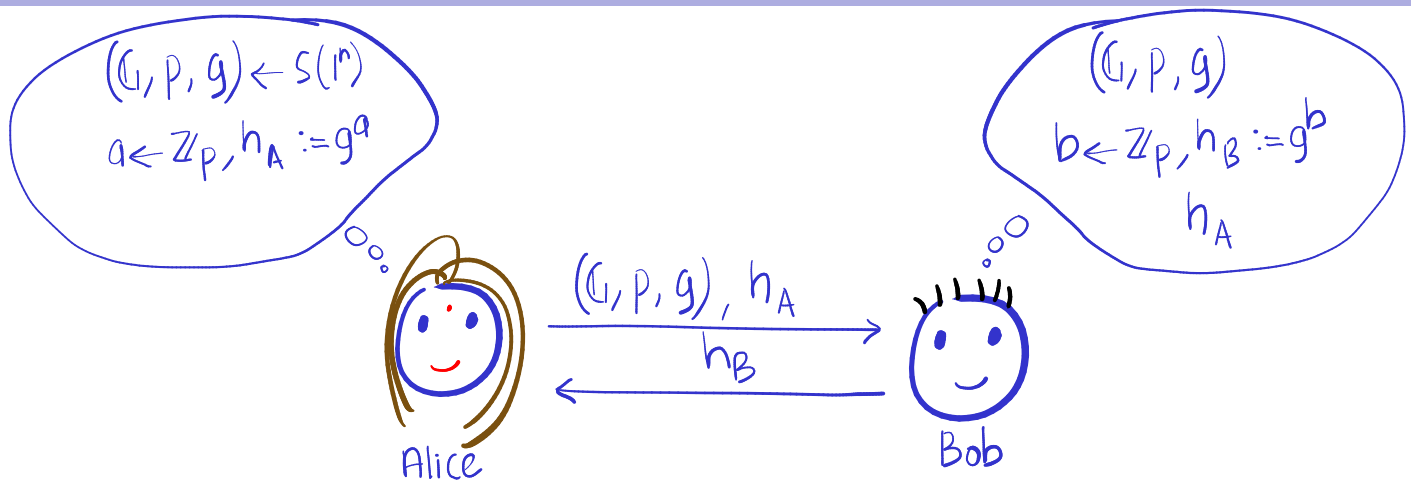
Diffie-Hellman Key-Exchange Protocol



Protocol 1

1. Alice→Bob: Send $((\mathbb{G}, \ell, g), h_A := g^a)$, where $(\mathbb{G}, \ell, g) \leftarrow S(1^n)$ and $a \leftarrow \mathbb{Z}_\ell$

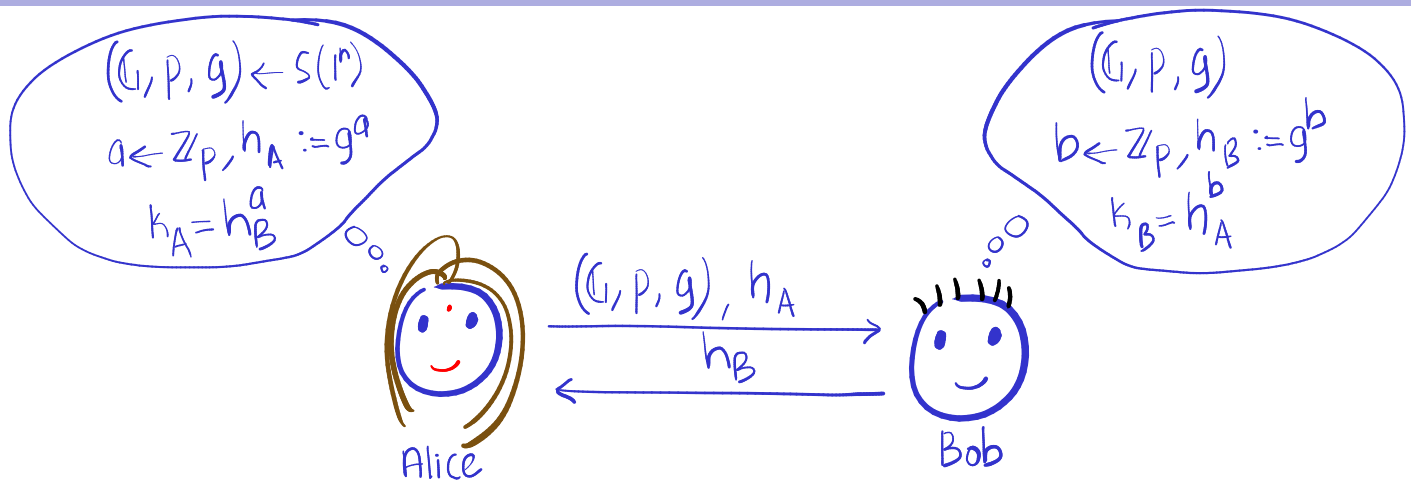
Diffie-Hellman Key-Exchange Protocol



Protocol 1

1. Alice→Bob: Send $((\mathbb{G}, \ell, g), h_A := g^a)$, where $(\mathbb{G}, \ell, g) \leftarrow \mathsf{S}(1^n)$ and $a \leftarrow \mathbb{Z}_\ell$

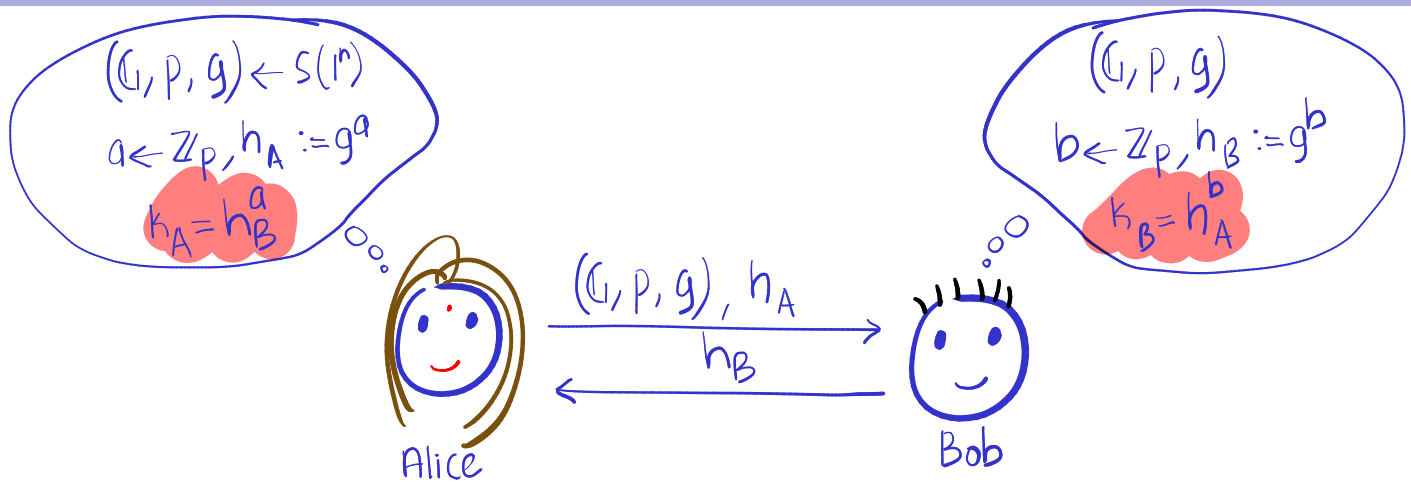
Diffie-Hellman Key-Exchange Protocol



Protocol 1

1. Alice→Bob: Send $((\mathbb{G}, \ell, g), h_A := g^a)$, where $(\mathbb{G}, \ell, g) \leftarrow \mathsf{S}(1^n)$ and $a \leftarrow \mathbb{Z}_\ell$
2. Alice←Bob: Send $h_B := g^b$ for $b \leftarrow \mathbb{Z}_\ell$

Diffie-Hellman Key-Exchange Protocol



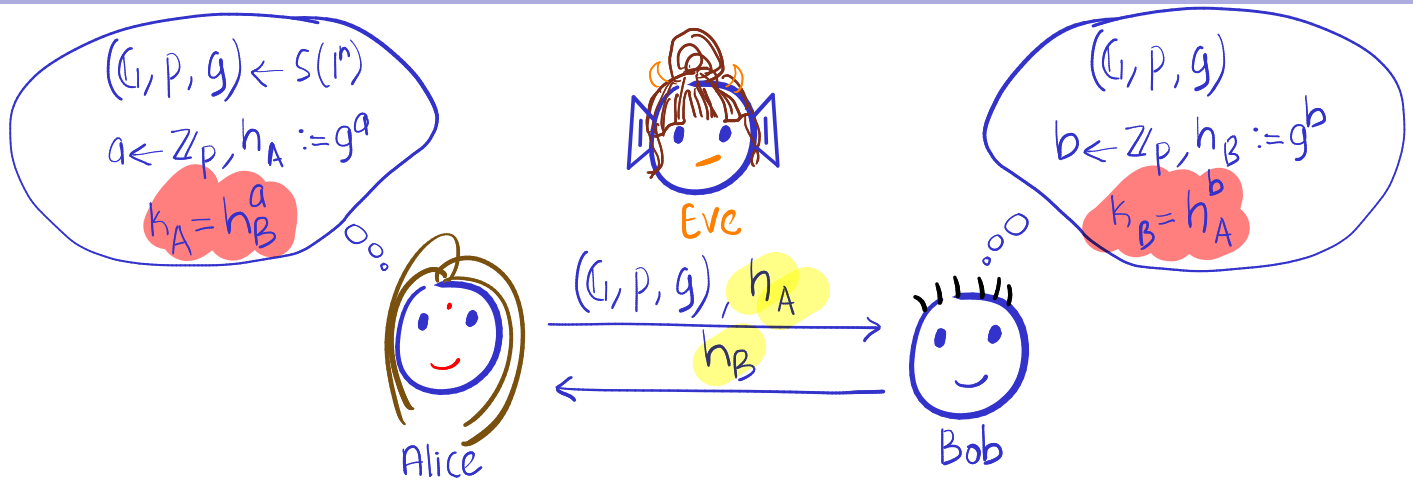
Protocol 1

1. Alice→Bob: Send $((\mathbb{G}, \ell, g), h_A := g^a)$, where $(\mathbb{G}, \ell, g) \leftarrow S(1^n)$ and $a \leftarrow \mathbb{Z}_\ell$
2. Alice←Bob: Send $h_B := g^b$ for $b \leftarrow \mathbb{Z}_\ell$
3. Alice outputs $k_A := (h_B)^a$; Bob outputs $k_B := (h_A)^b$

- Correctness of key generation:

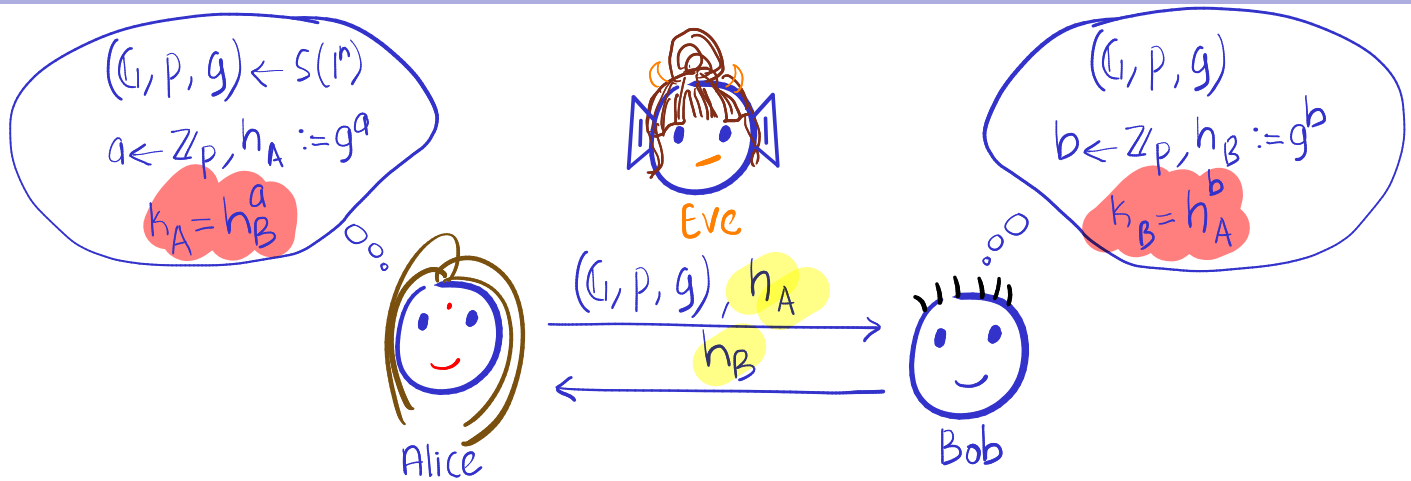
$$k_A = h_B^a = (g^b)^a = g^{ab} = (g^a)^b = h_A^b = k_B$$

When is it Secret Against Eavesdroppers?...



- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

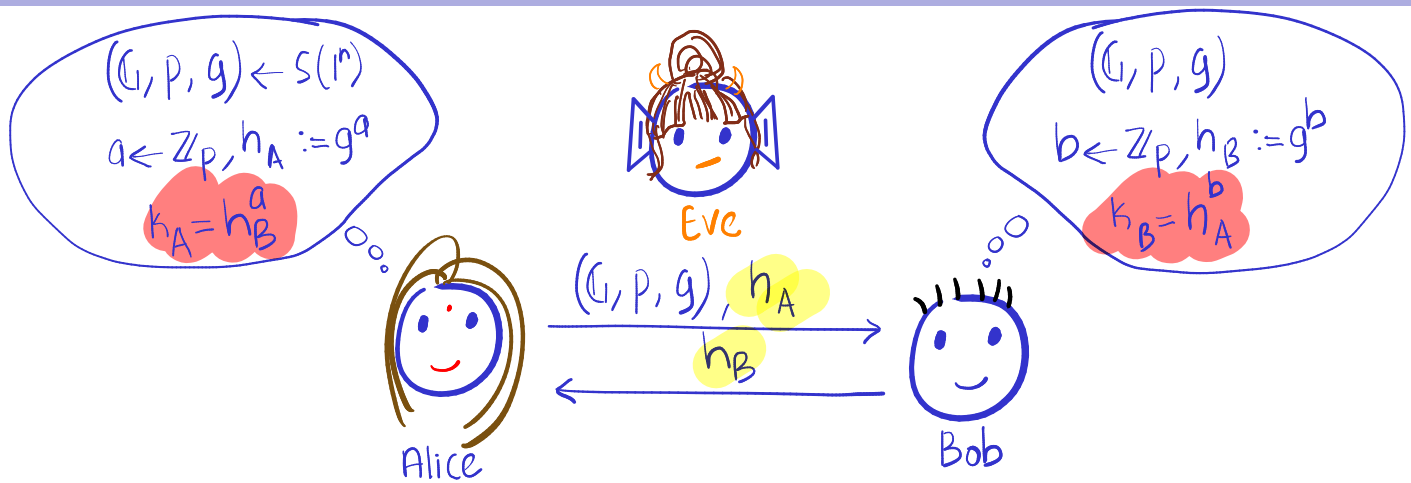
When is it Secret Against Eavesdroppers?...



- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

? What if DLog problem is easy over $\mathbb{G}$?

When is it Secret Against Eavesdroppers?...

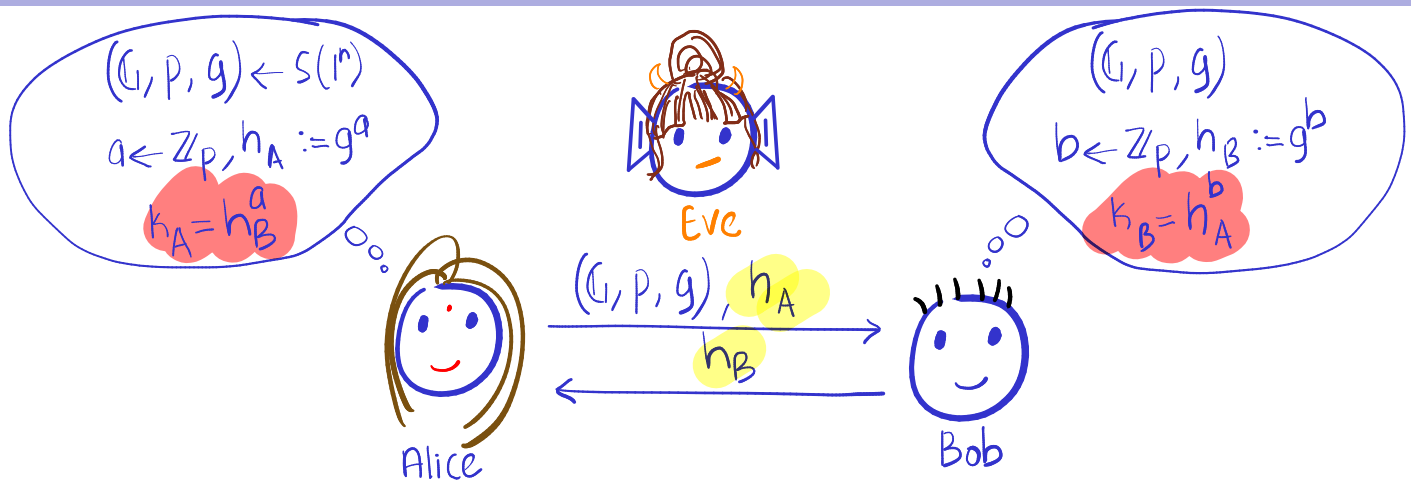


- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

? What if DLog problem is easy over $\mathbb{G}$?

⚠ Then Eve can invert h_A to get a and compute $k = h_B^a$

When is it Secret Against Eavesdroppers?...



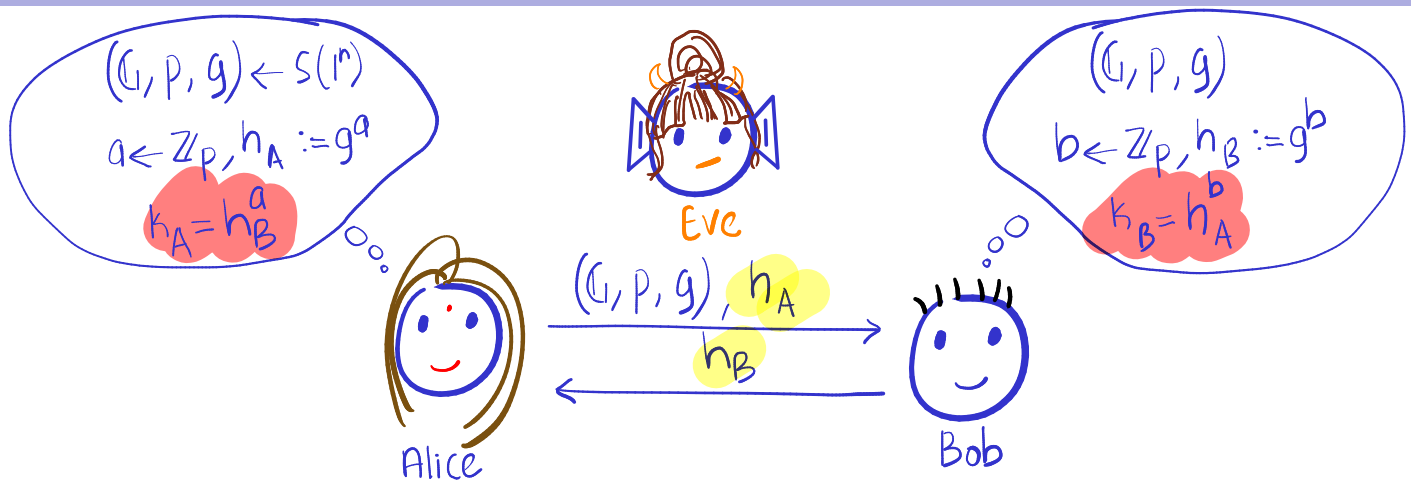
- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

? What if DLog problem is easy over $\mathbb{G}$?

⚠ Then Eve can invert h_A to get a and compute $k = h_B^a$

? Is DLog problem being hard sufficient?

When is it Secret Against Eavesdroppers?...



- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

❓ What if DLog problem is easy over $\mathbb{G}$?

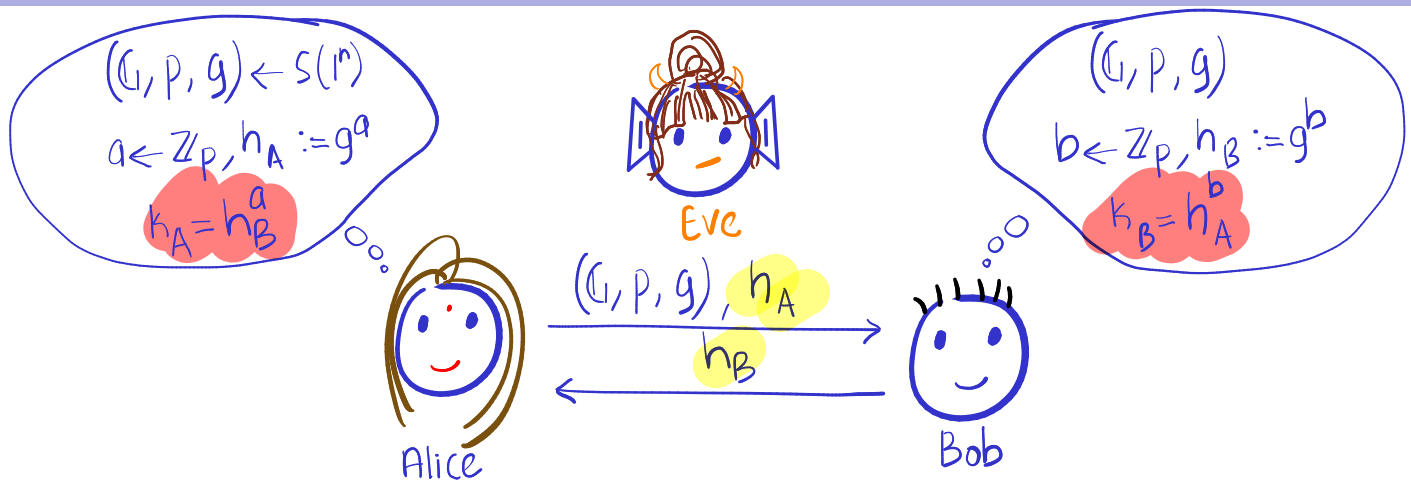
⚠ Then Eve can invert h_A to get a and compute $k = h_B^a$

❓ Is DLog problem being hard sufficient?

⚠ No, what if Eve can compute g^{ab} given g^a and g^b?

- This is the "computational Diffie-Hellman" (CDH) problem

When is it Secret Against Eavesdroppers?...



- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

❓ What if DLog problem is easy over $\mathbb{G}$?

⚠ Then Eve can invert h_A to get a and compute $k = h_B^a$

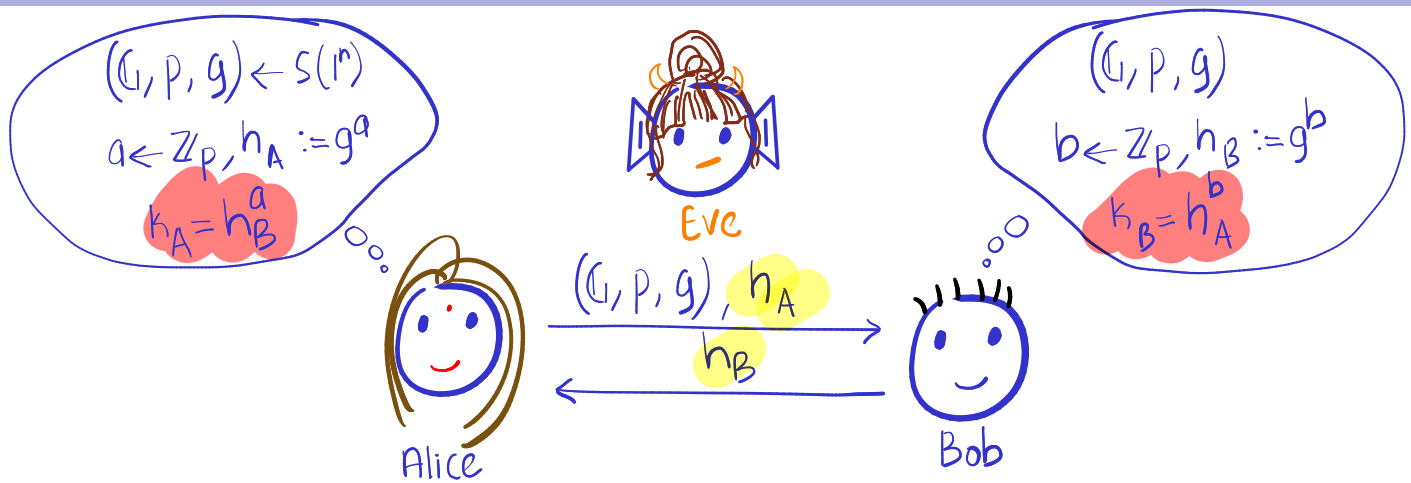
❓ Is DLog problem being hard sufficient?

⚠ No, what if Eve can compute g^{ab} given g^a and g^b?

- This is the "computational Diffie-Hellman" (CDH) problem

❓ Is CDH problem being hard sufficient?

When is it Secret Against Eavesdroppers?...



- What does Eve see? The transcript is $(h_A := g^a, h_B := g^b)$

❓ What if DLog problem is easy over $\mathbb{G}$?

⚠ Then Eve can invert h_A to get a and compute $k = h_B^a$

❓ Is DLog problem being hard sufficient?

⚠ No, what if Eve can compute g^{ab} given g^a and g^b?

- This is the "computational Diffie-Hellman" (CDH) problem

❓ Is CDH problem being hard sufficient?

⚠ What if Eve can distinguish g^{ab} from random group elements?

- There exist such groups!

When is it Secret Against Eavesdroppers?...

Assumption 2 (Decisional DH (DDH) assumption in $\mathbb{G}$ w.r.to S $\cdots$)

$\cdots$ holds if for all PPT distinguishers D, the following is negligible:

$$\Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^{ab}) = 0] - \Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b,r\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^r) = 0]$$

"Real world" "Random world"

When is it Secret Against Eavesdroppers?...

Assumption 2 (Decisional DH (DDH) assumption in $\mathbb{G}$ w.r.to S $\cdots$)

$\cdots$ holds if for all PPT distinguishers D, the following is negligible:

$$\Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^{ab}) = 0] - \Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b,r\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^r) = 0]$$

"Real world" "Random world"

Theorem 1

Diffie-Hellman key-exchange is computationally secret against eavesdroppers under the DDH assumption in $\mathbb{G}$ w.r.to S.

Proof.

Secrecy requirement is same as the assumption! □

When is it Secret Against Eavesdroppers?...

Assumption 2 (Decisional DH (DDH) assumption in $\mathbb{G}$ w.r.to S $\cdots$)

$\cdots$ holds if for all PPT distinguishers D, the following is negligible:

$$\Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^{ab}) = 0] - \Pr_{\substack{(\mathbb{G},\ell,g)\leftarrow \mathsf{S}(1^n)\\ a,b,r\leftarrow\mathbb{Z}_\ell}}[\mathsf{D}(g^a, g^b, g^r) = 0]$$

"Real world" "Random world"

Theorem 1

Diffie-Hellman key-exchange is computationally secret against eavesdroppers under the DDH assumption in $\mathbb{G}$ w.r.to S.

Proof.

Secrecy requirement is same as the assumption! □

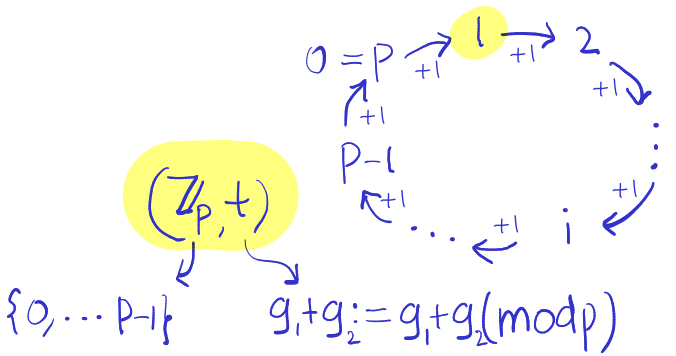
Exercise 6

But I did slightly cheat! Figure out where.

Where is DDH Assumption Known to Hold?

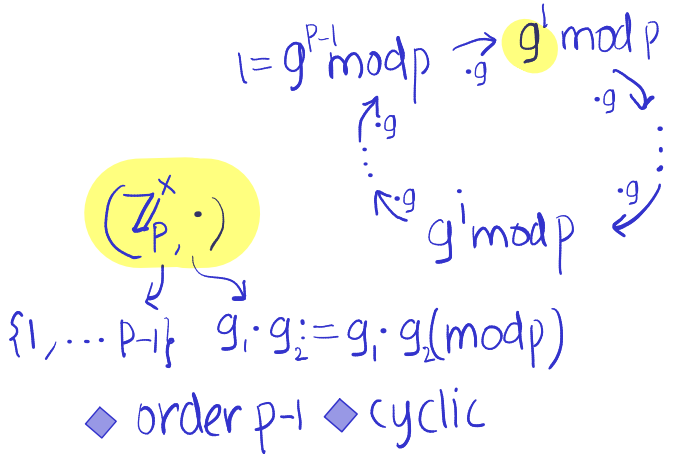
?

Addition modulo prime p



◆ order p ◆ cyclic

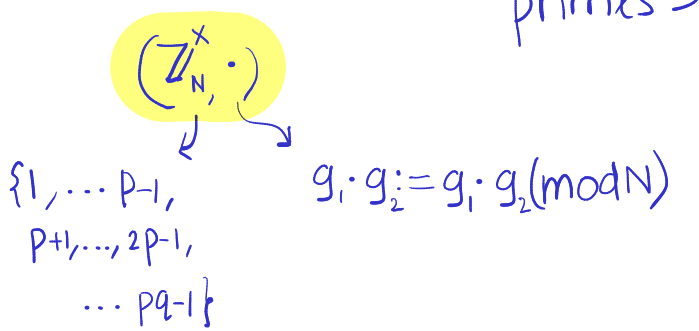
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

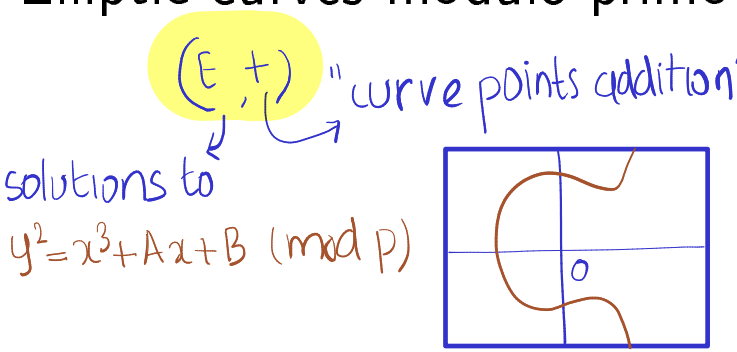
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

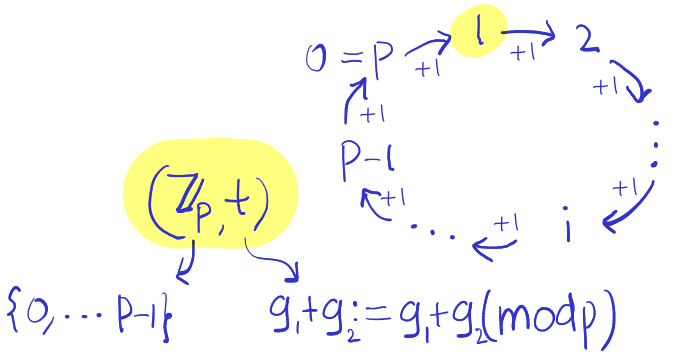


◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

Where is DDH Assumption Known to Hold?

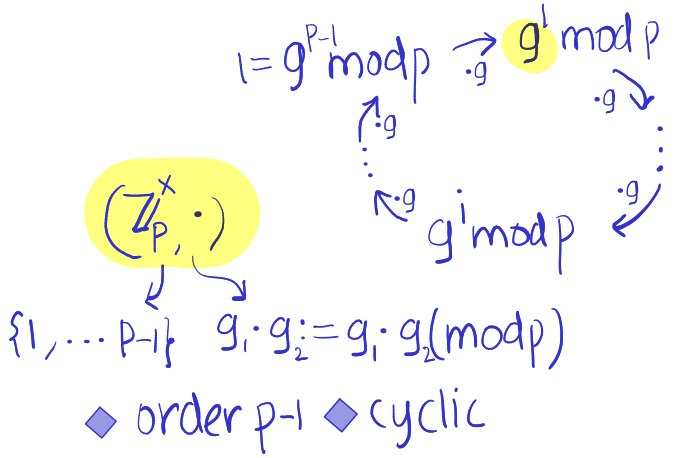
? Easy! Since DLog is easy ⚠

Addition modulo prime p



◆ order p ◆ cyclic

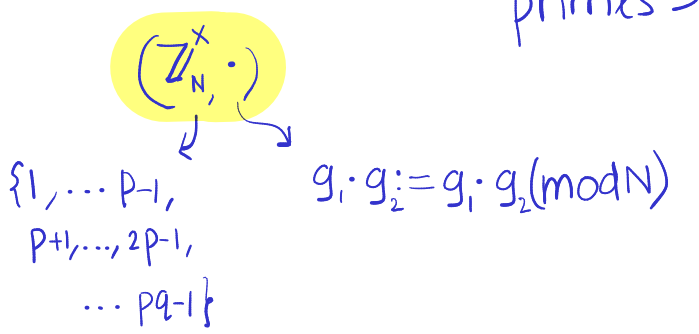
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

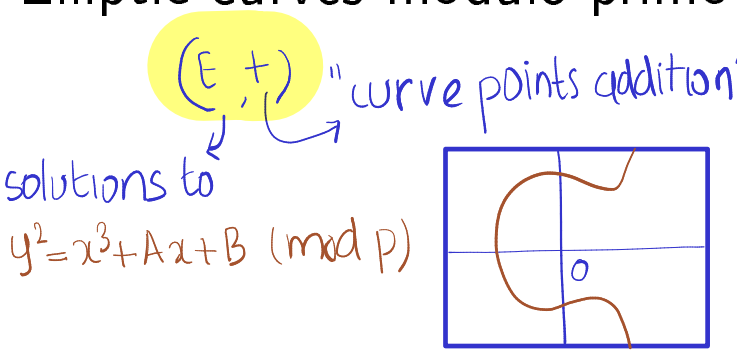
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

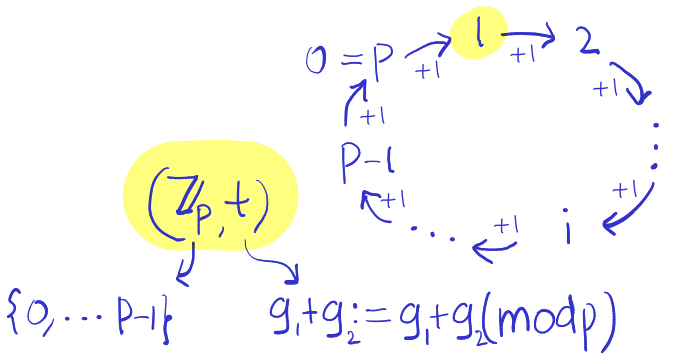


◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

Where is DDH Assumption Known to Hold?

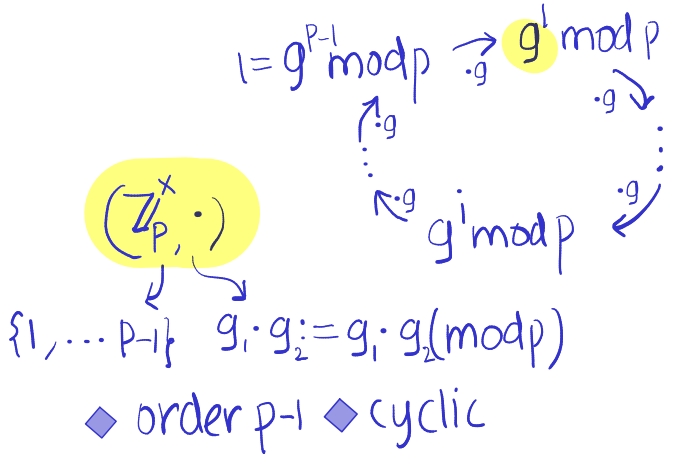
Easy! Since DLog is easy

Addition modulo prime p



- order p
- cyclic

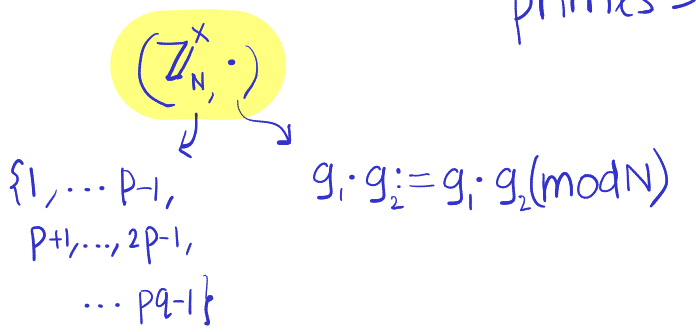
Multiplication modulo prime p



- order $p-1$
- cyclic

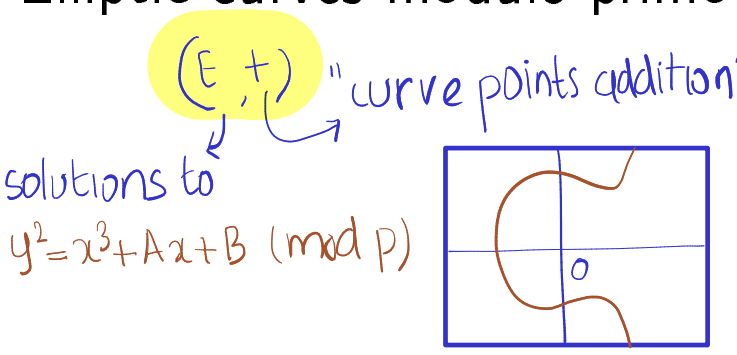
Multiplication modulo $N = pq$

primes



- order $(p-1)(q-1)$
- not cyclic

Elliptic curves modulo prime p

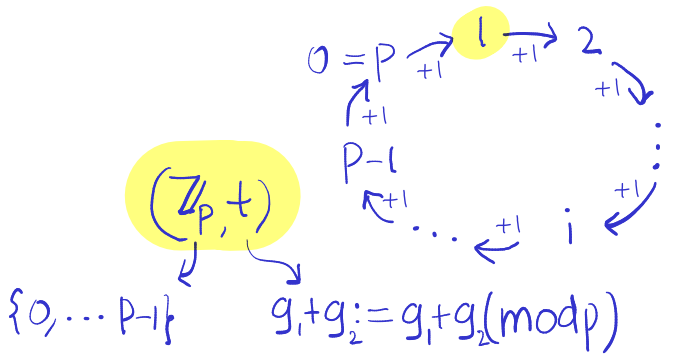


- $|p+1 - \text{order}| \leq 2\sqrt{p}$
- cyclic

Where is DDH Assumption Known to Hold?

Easy! Since DLog is easy ⚠

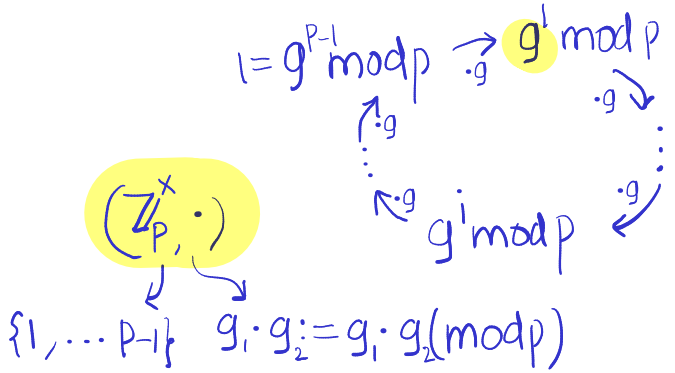
Addition modulo prime p



◆ order p ◆ cyclic

Easy! See Assignment 4 ⚠

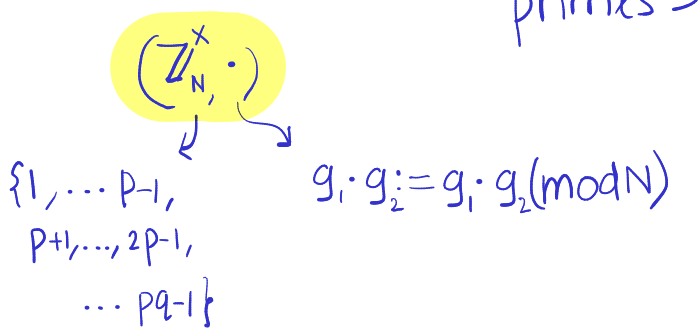
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

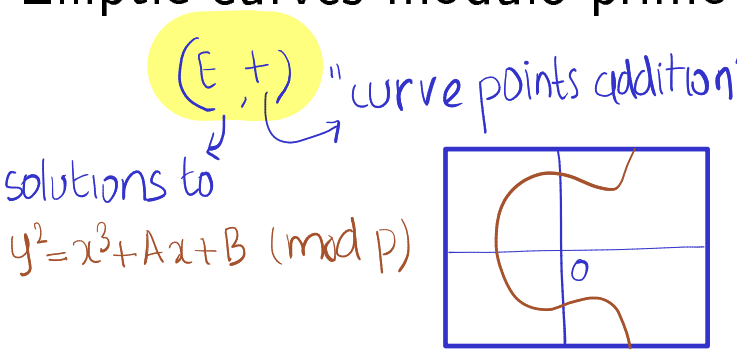
Multiplication modulo $N = pq$

primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Elliptic curves modulo prime p

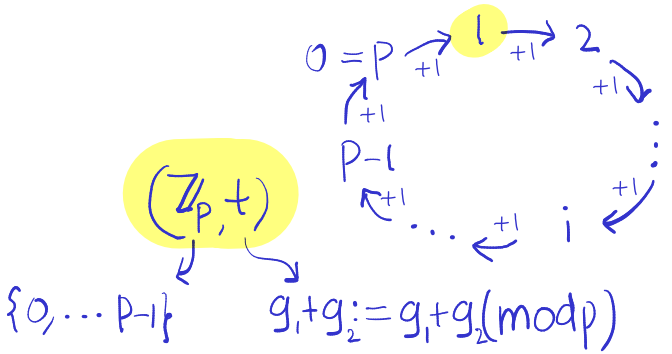


◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

Where is DDH Assumption Known to Hold?

Easy! Since DLog is easy ⚠

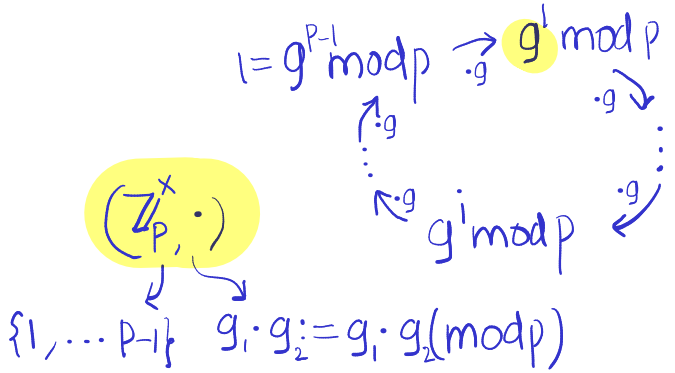
Addition modulo prime p



◆ order p ◆ cyclic

Easy! See Assignment 4 ⚠

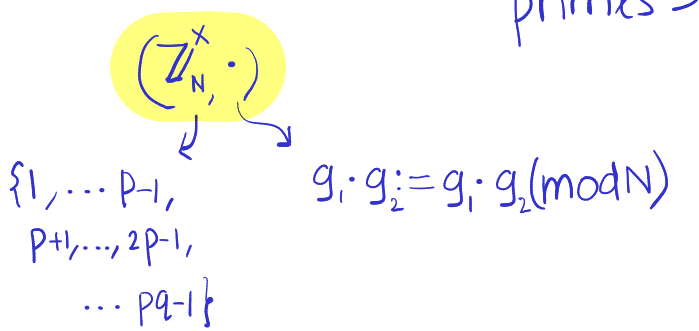
Multiplication modulo prime p



◆ order $p-1$ ◆ cyclic

Multiplication modulo $N = pq$

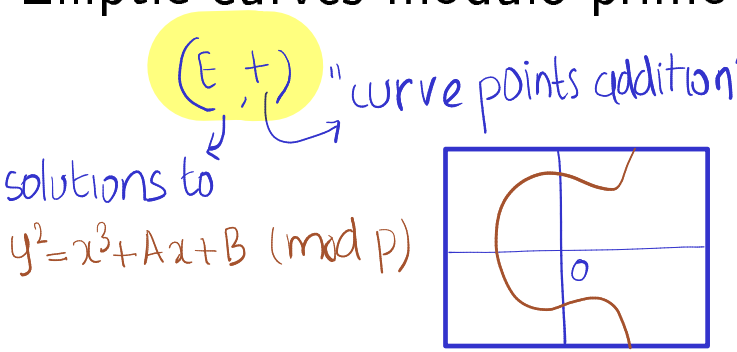
primes



◆ order $(p-1)(q-1)$ ◆ not cyclic

Hard in its cyclic subgroup

Elliptic curves modulo prime p



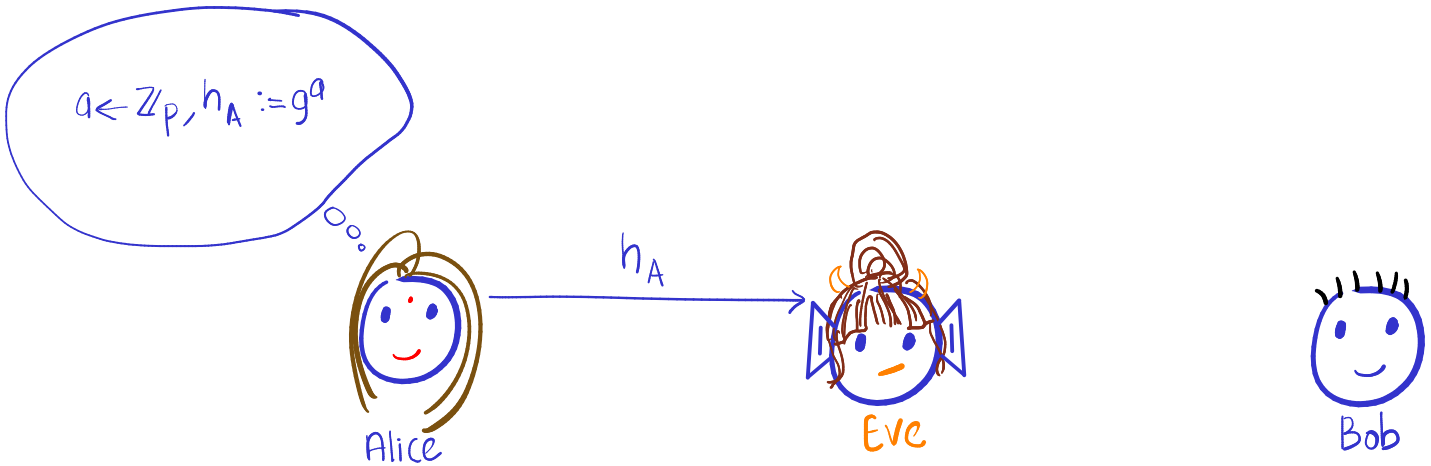
◆ $|p+1 - \text{order}| \leq 2\sqrt{p}$ ◆ cyclic

Believed hard

What About Secrecy Against Active Eve?

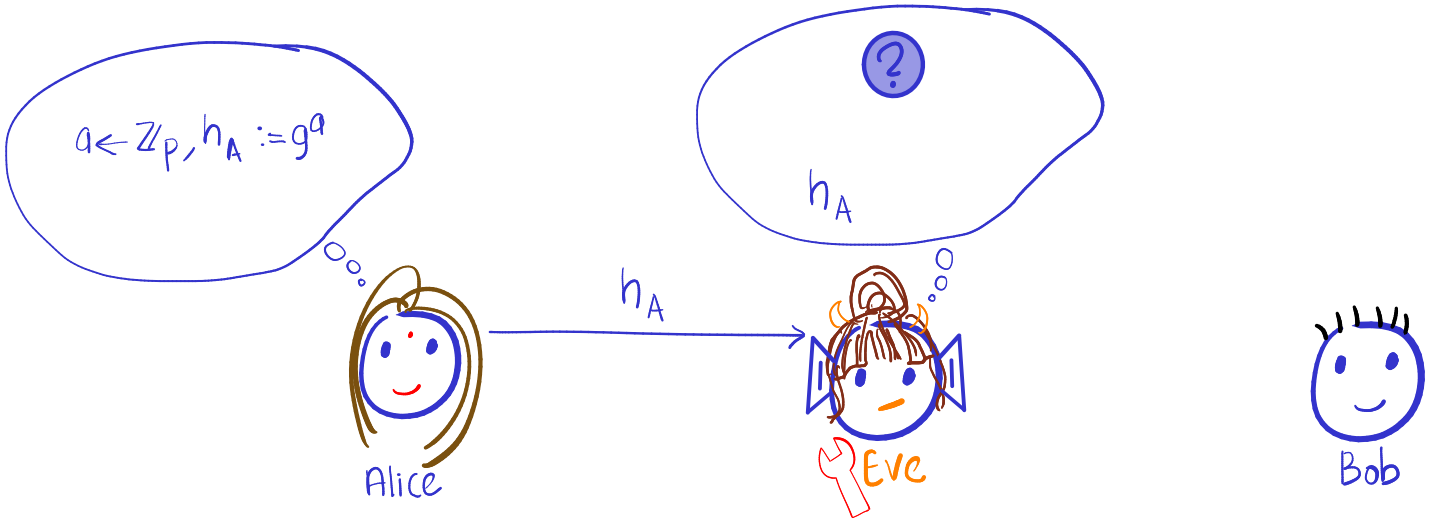
- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

What About Secrecy Against Active Eve?



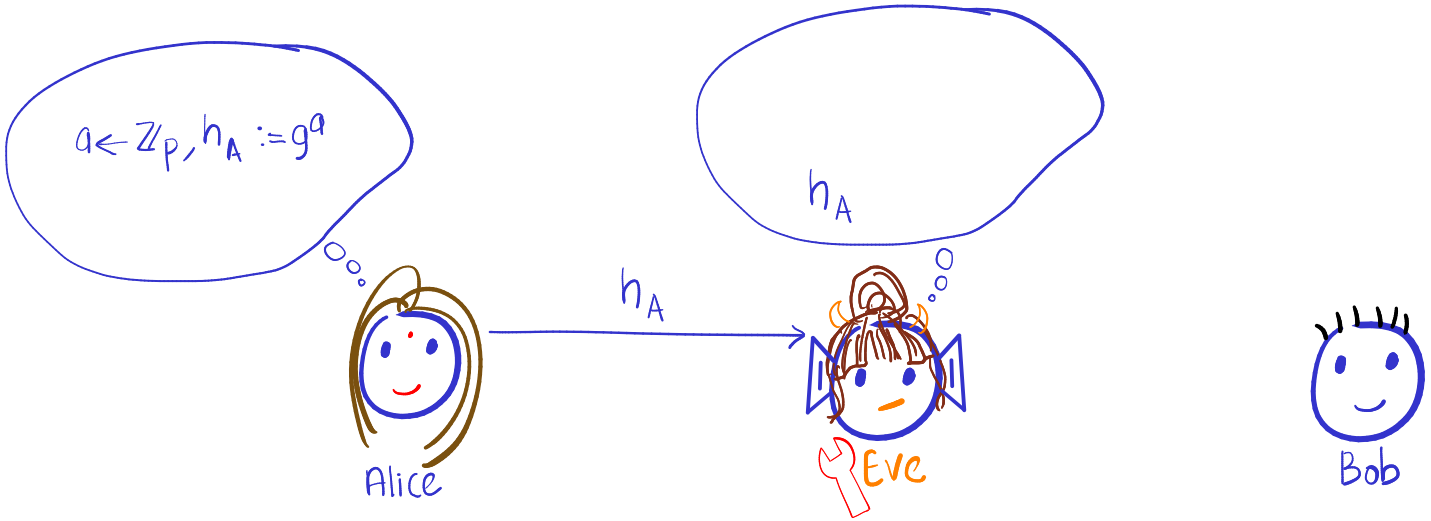
- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

What About Secrecy Against Active Eve?



- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

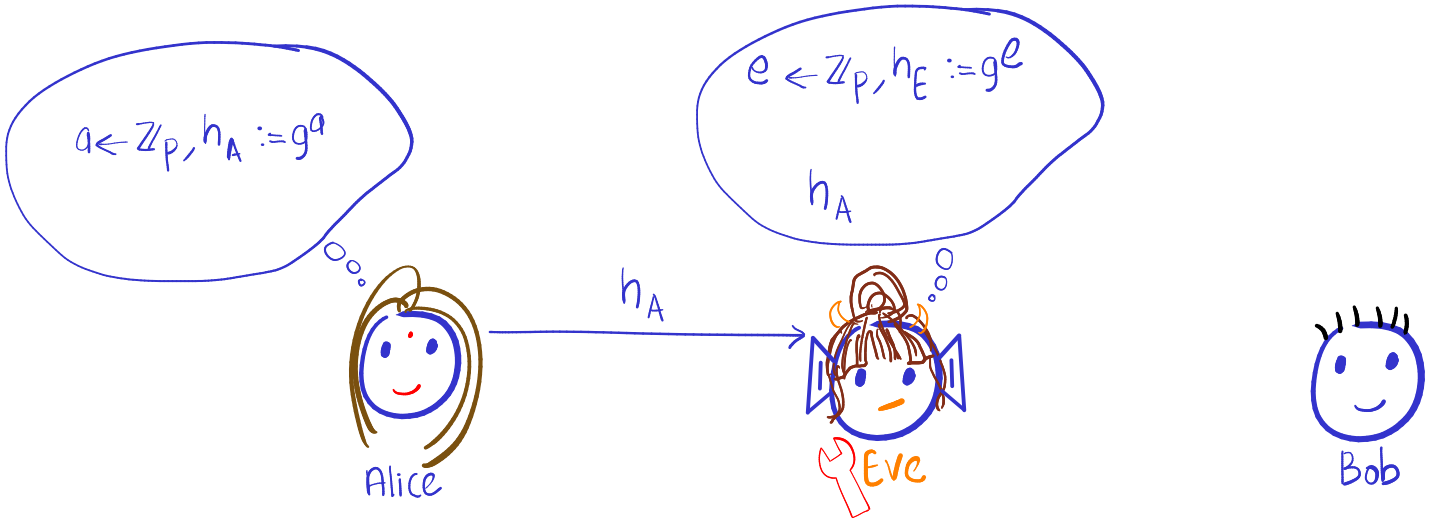
What About Secrecy Against Active Eve?



- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!
  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

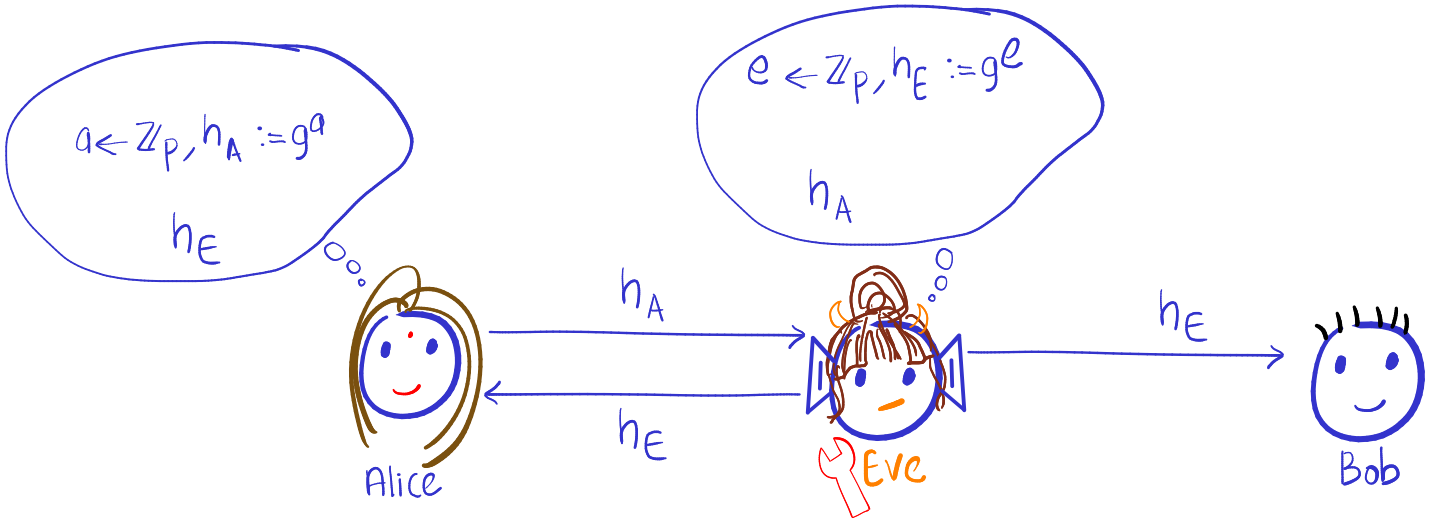
What About Secrecy Against Active Eve?



- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!
  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

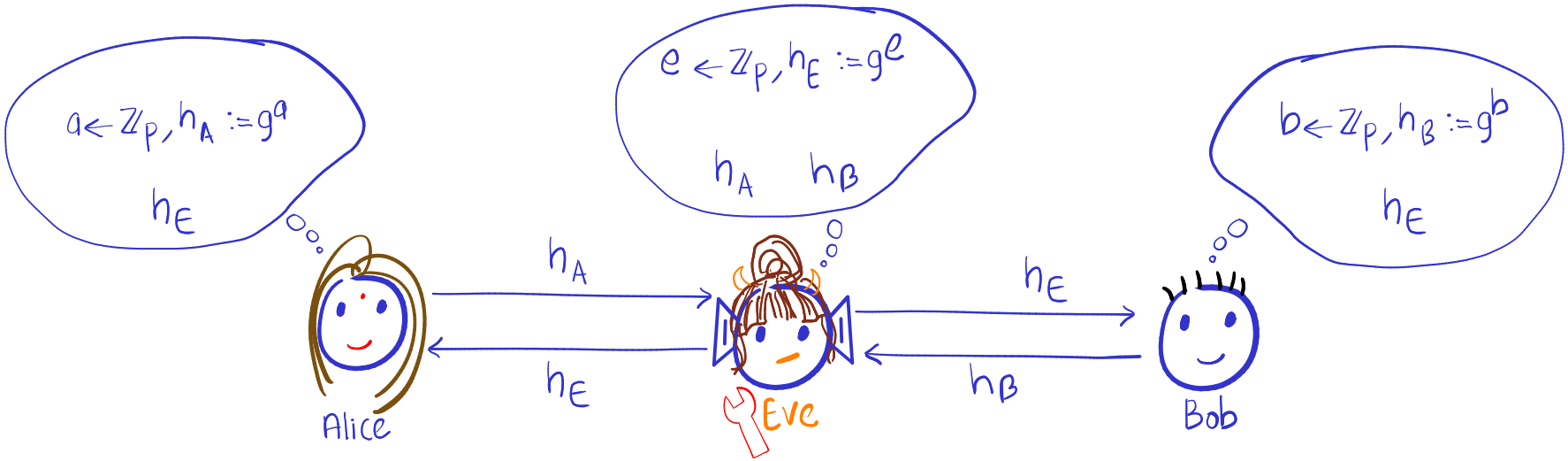
What About Secrecy Against Active Eve?



- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!
  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

What About Secrecy Against Active Eve?

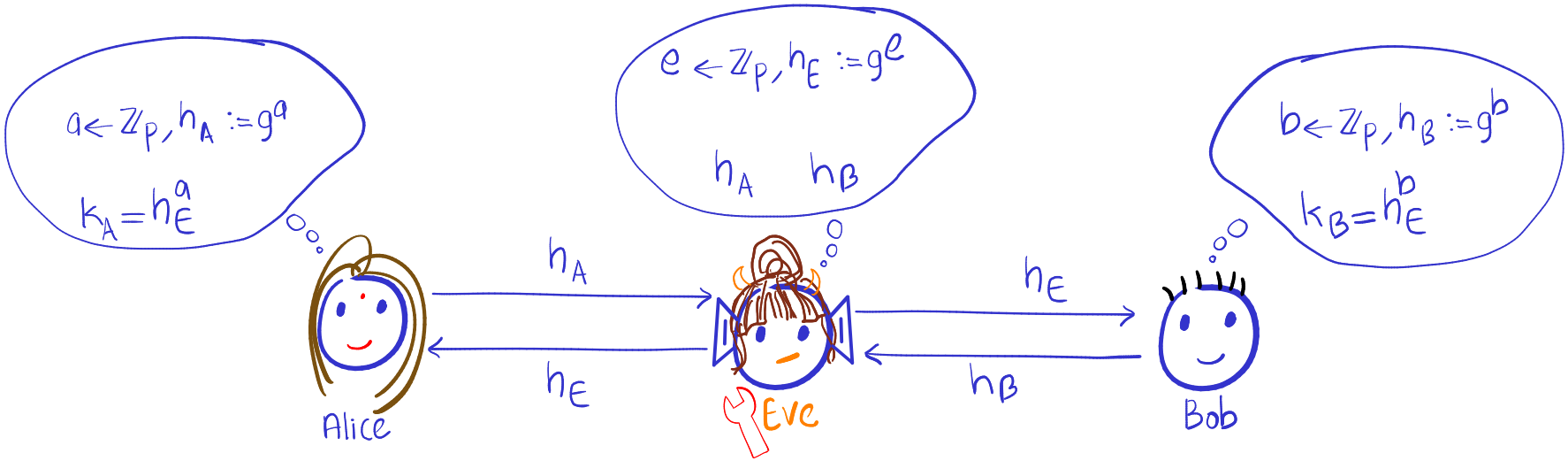


- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!

  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

What About Secrecy Against Active Eve?

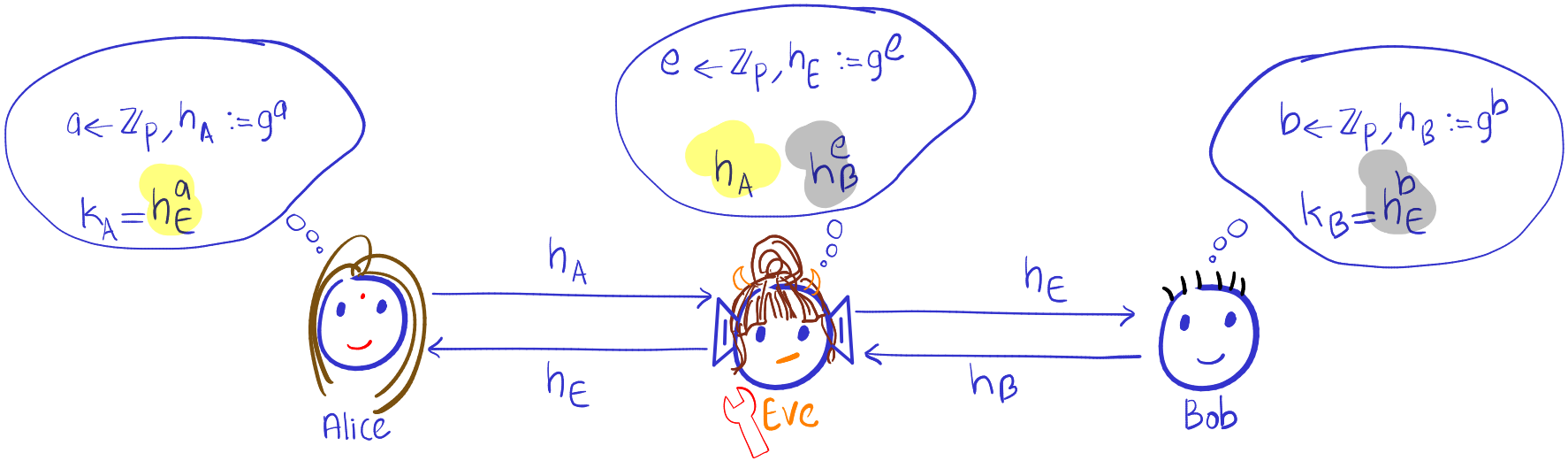


- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!

  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

What About Secrecy Against Active Eve?

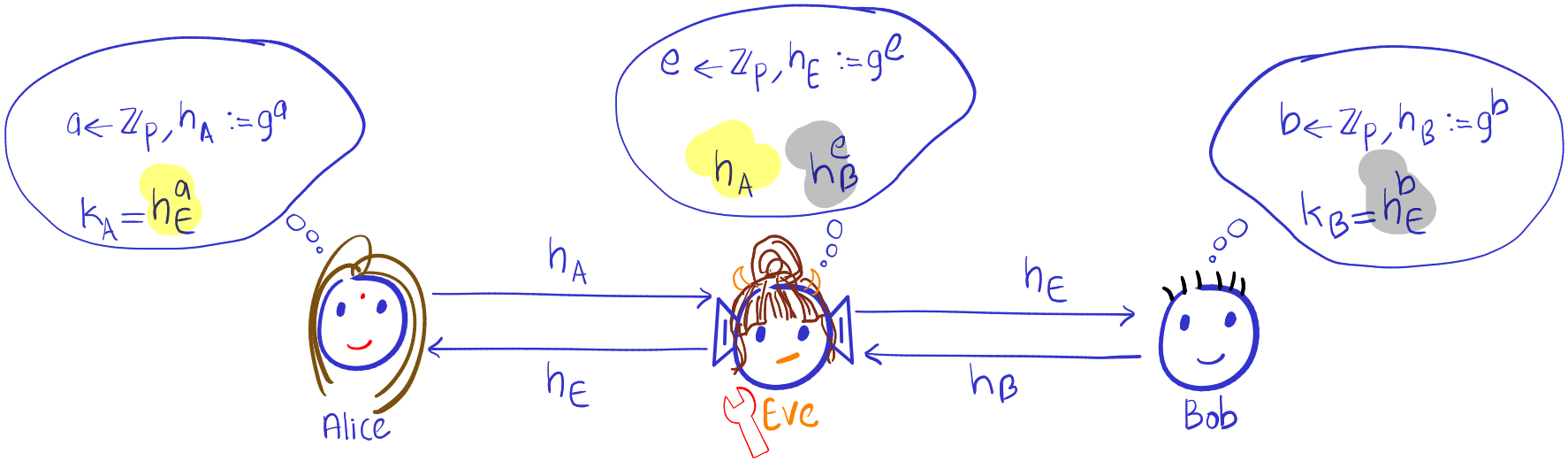


- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

⚠ There is a person-in-the-middle attack!

  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

What About Secrecy Against Active Eve?



- What if Eve is an *active* adversary?
  - Recall that active Eve can intercept/tamper messages

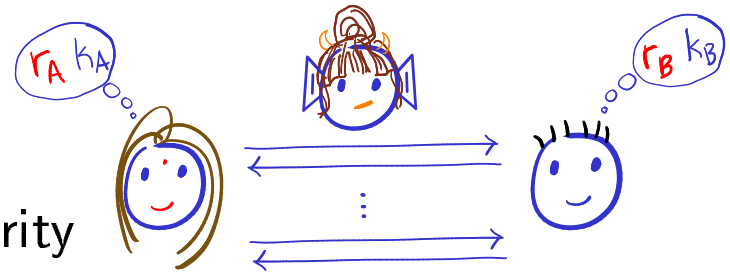
⚠ There is a person-in-the-middle attack!

  - Pretends to be Alice to Bob and Bob to Alice
  - Eve sets up two separate key exchanges with Alice and Bob

⚠ Insecure against active adversary

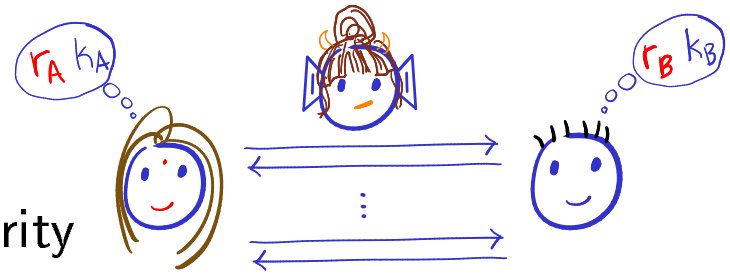
Recap/Next Lecture

- Key exchange against eavesdroppers
  - Modelled key exchange setting and security

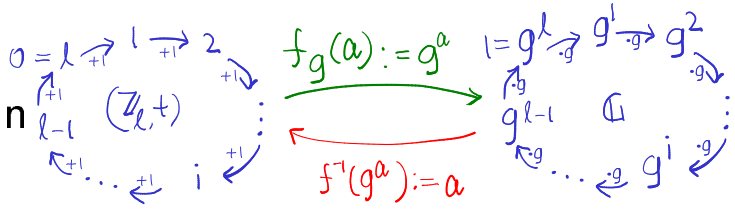


Recap/Next Lecture

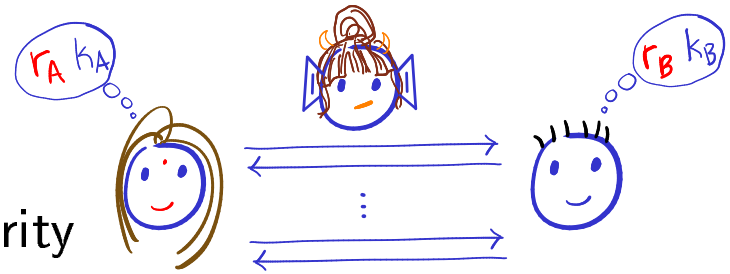
- Key exchange against eavesdroppers
  - Modelled key exchange setting and security



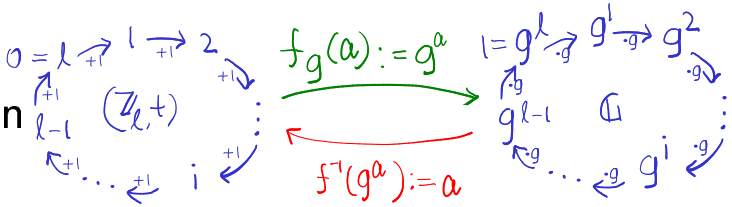
  - Diffie-Hellman key exchange protocol
    - Based security on the DDH assumption



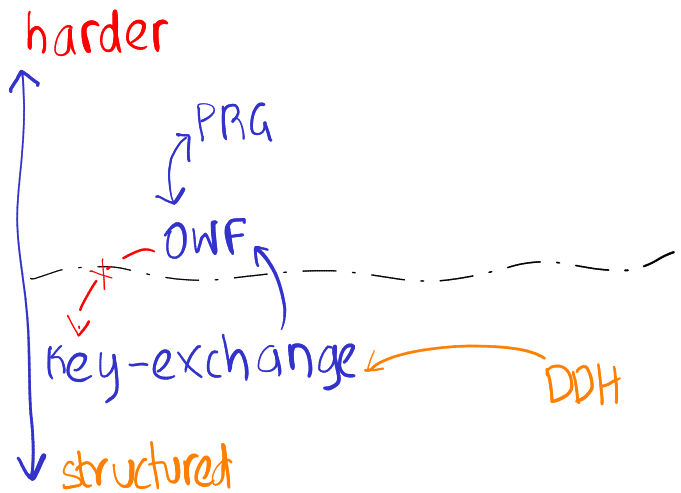
Recap/Next Lecture



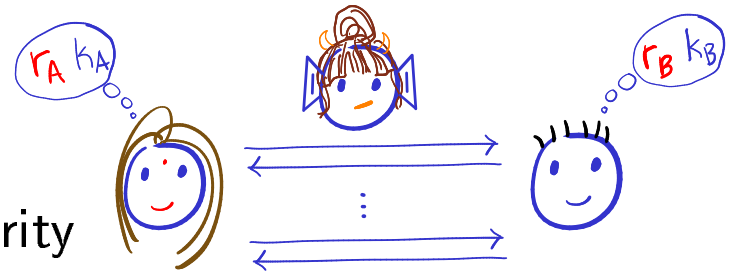
- Key exchange against eavesdroppers
  - Modelled key exchange setting and security
  - Diffie-Hellman key exchange protocol
    - Based security on the DDH assumption



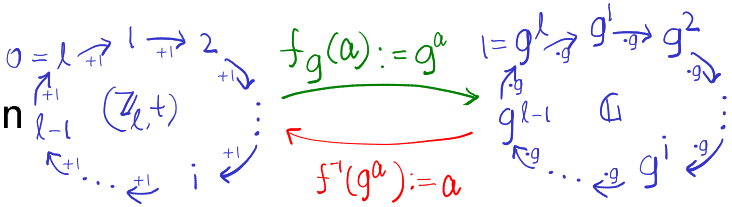
- Takeaway: structure vs hardness
  1. Structure is useful for protocol design and proofs
  2. Also makes it susceptible to algorithms



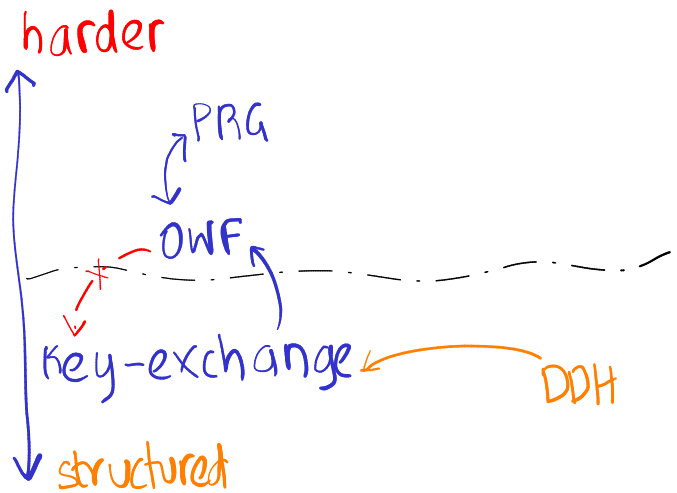
Recap/Next Lecture



- Key exchange against eavesdroppers
  - Modelled key exchange setting and security
  - Diffie-Hellman key exchange protocol
    - Based security on the DDH assumption



- Takeaway: structure vs hardness
  1. Structure is useful for protocol design and proofs
  2. Also makes it susceptible to algorithms



- Next lecture: public-key encryption (PKE)
  - Syntax and security
  - Relationship with key-exchange
  - Elgamal PKE

References

1. Basic group theory and algorithmic number theory can be found in [KL14, Appendix B]. MIT 6875 handout is also an excellent resource.
2. More motivation about groups can be found in Keith Conrad's expository paper on the topic
3. [KL14, Chapter 11] for more details on key exchange
4. Read the seminal paper by Diffie and Hellman [DH76] for a description of the namesake key-exchange. In general this paper is a very insightful read.
5. Boneh's survey [Bon98] is an excellent source on the DDH problem.


Dan Boneh.

The decision diffie-hellman problem.

In *ANTS*, volume 1423 of *Lecture Notes in Computer Science*, pages 48–63. Springer, 1998.


Whitfield Diffie and Martin E. Hellman.

New directions in cryptography.

IEEE Trans. Inf. Theory, 22(6):644–654, 1976.


Jonathan Katz and Yehuda Lindell.

Introduction to Modern Cryptography (3rd ed.).

Chapman and Hall/CRC, 2014.