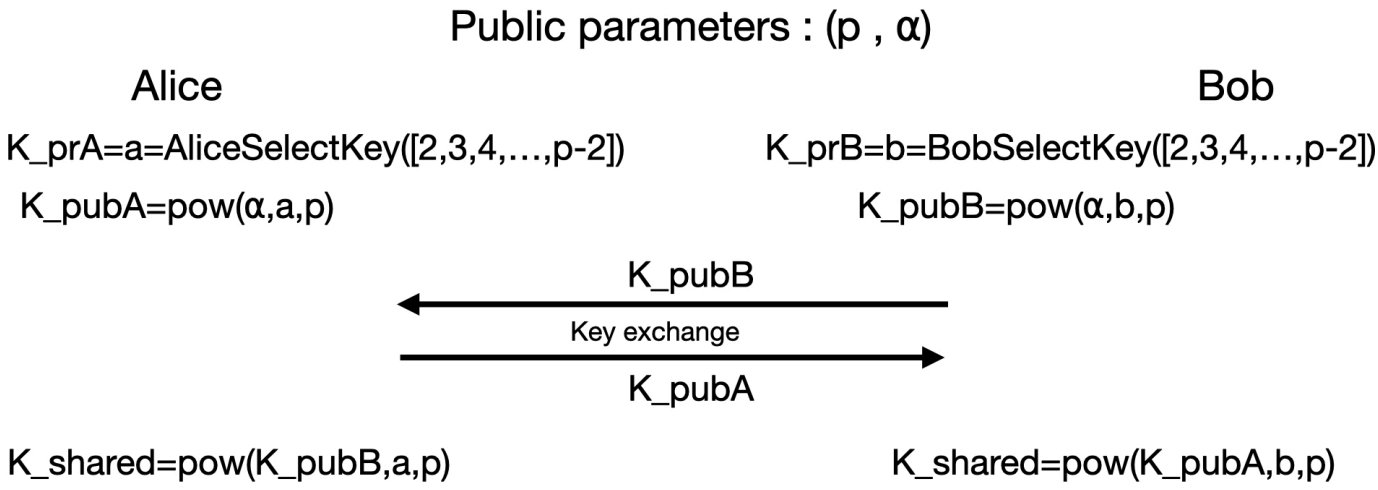
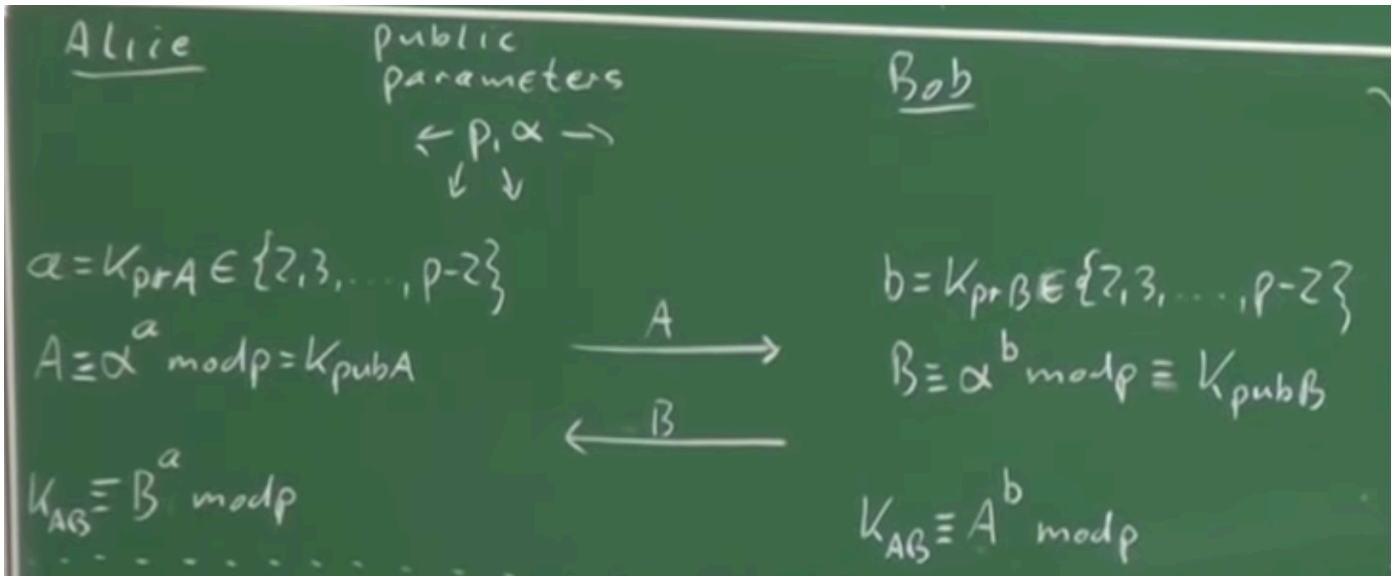
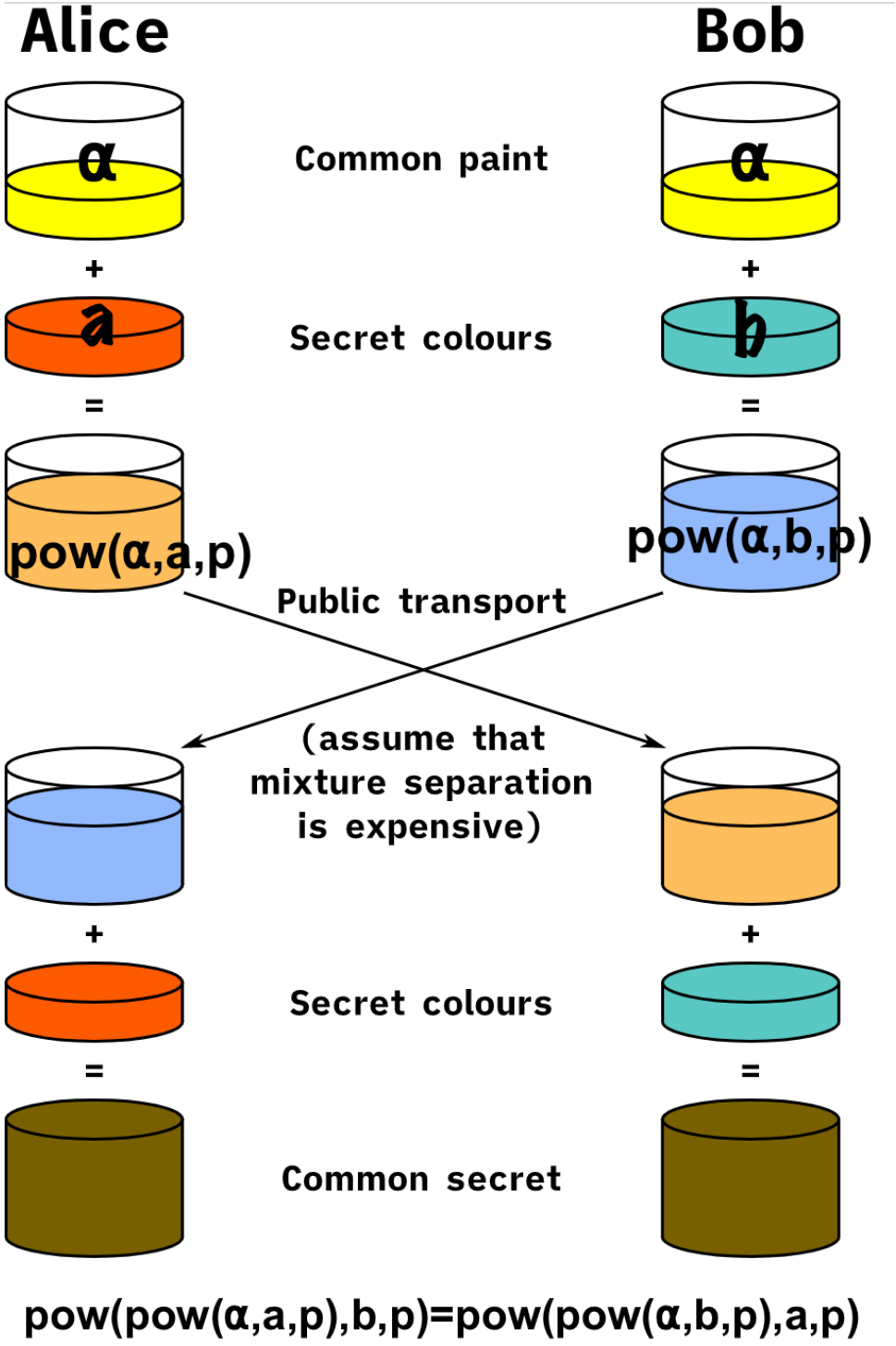
Discrete log

Diffie-Hellman key exchange

The protocol




Alice
public parameters
p, α
Bob
a = K_prA ∈ {2, 3, ..., p-2}
A ≡ α^a mod p = K_pubA
A
B
b = K_prB ∈ {2, 3, ..., p-2}
B ≡ α^b mod p ≡ K_pubB
K_AB ≡ B^a mod p
K_AB ≡ A^b mod p



Now Alice and Bob can use the same key to do all the symmetric cryptography

Proof of correctness

$$\text{Alice side:} K_{AB} = B^a \equiv (\alpha^b)^a \mod p \equiv (\alpha)^{ba} \mod p$$

$$\text{Bob side:} K_{AB} = A^b \equiv (\alpha^a)^b \mod p \equiv (\alpha)^{ab} \mod p$$

```
alpha = 5
p = 23

Alice_pr = 4
Bob_pr = 20

Alice_pub = pow(alpha, Alice_pr , p)
Bob_pub = pow(alpha, Bob_pr , p)

print(pow(Bob_pub, Alice_pr , p) == pow(Alice_pub ,
Bob_pr , p))
# True
```

Group

A Group is a collection of objects G, together with **ONE** operation, $\oplus$(the choosing of notation here is arbitrary, it doesn't that matters), which have the following properties :

- The group operation is closed:

$$\forall a, b \in G, a \oplus b = c, c \in G$$

- The group operation is associative:

$$\forall a, b, c \in G, (a \oplus b) \oplus c = a \oplus (b \oplus c)$$

- There is a neutral element(denoted by e):

$$\forall a \in G, a \oplus e = e \oplus a = a$$

- Every element has an inverse:

$$\forall a \in G, \exists a^{-1}, a \oplus a^{-1} = a^{-1} \oplus a = e$$

*If a Group is **commutative** , which means*
$\forall a, b \in G, a \oplus b = b \oplus a$ *, it is an **abelian group***

> **Theorem:**
>
> The set Z_n^* which consists of all integers $i = 0, 1, 2, 3, \ldots, n-1$ for which $\gcd(i, n) = 1$, forms an **abelian group under multiplication modulo n**. The nutural element $e = 1$.
>
> Deduction:
>
> set Z_p^*, where p is a prime, is $\{1, 2, 3, \ldots., p-1\}$. It is an abelian group under multiplication modulo p.

Cyclic Groups

Definition:

If a group has finite number of elements, it is a finite group.

The number of elements is call **Cardinality or order** of the group, denoted by $|G|$

Definition:

Order of an element : The order of an element x in Group G is the smallest positive integer k such that

$$x^k = \underbrace{x \oplus x \oplus \ldots \oplus x}_{\text{k times}} = e$$

where e is the identity element.

Example:

in Group Z_{11}^*:

$$
\begin{aligned}
2^1 &\equiv 2 \mod 11 \\
2^2 &\equiv 4 \mod 11 \\
2^3 &\equiv 8 \mod 11 \\
2^4 &\equiv 5 \mod 11 \\
2^5 &\equiv 10 \mod 11 \\
2^6 &\equiv 9 \mod 11 \\
2^7 &\equiv 7 \mod 11 \\
2^8 &\equiv 3 \mod 11 \\
2^9 &\equiv 6 \mod 11 \\
2^{10} &\equiv 1 \mod 11
\end{aligned}
$$

$$\therefore \text{order}(2) = 10$$

$$\text{for the same reason :order}(3) = 5$$

Definition:

a group which contains a element α with $\text{order}(\alpha) = |G|$,is said to be cyclic.

The elements α is called "primitive elements/generators" (Note that there may be more than one element fulfil the requirement)

Cyclic groups are the basis of discrete logarithm system!!!!

> **Theorem:**
>
> For every prime number p, Z_p^* is a finite abelian group

Important properties of cyclic group

- let a be an arbitrary element in cyclic group G
  - $$a^{|G|} = e$$
  - $$\text{order}(a) \text{ must devide } |G|$$
    $$order(a) \mid (|G|)$$
- Fermat's little theorem for Z_p^*

$$\because a^p \equiv a \mod p$$

$$\therefore a^{p-1} \equiv 1 \mod p$$

$$\therefore a^{|G|} \equiv 1 \mod p$$

$$\therefore a^{|G|} = e$$

How does this related to discrete logarithm?

given a generator α of cyclic group and an element Y in the group, it is difficult for us to find x makes that:

$$\alpha^x \equiv Y \mod p$$

Diffie-Hellman Problem

The security of Diffie-Hellman Key exchange

assumption : the hacker can only passively listen to the channel

> Diffie-Hellman problem : knowing $\alpha, p, K_{\text{pubA}}, K_{\text{pubB}}$
>
> How can we compute the K_{shared} ?

Attack : solve the discrete logarithm

$$\because K_{\text{shared}} \equiv \alpha^{ab} \mod p$$

$\therefore$ we need to know either a or b , take a as an example here

$$\because K_{\text{pubA}} \equiv \alpha^{a} \mod p$$

$$\therefore a \equiv \log_{\alpha}(K_{\text{pubA}}) \mod p$$

$$\therefore K_{\text{shared}} \equiv (K_{\text{pubB}})^{a} \mod p$$

Unfortunately, it is very hard to sovle the discrete logarithm when the p is really large(1024-2048 bits long)

Although it hasn't been prove that solving the logarithm is the only way of calculating K_{shared}, there aren't any better way appear now.(similiar to factoring the n in RSA)

The generalized discrete logarithm problem

discrete logarithm problem is not confined to Z_p^*, other cyclic groups can have the same.

> **Generalized discrete logarithm problem:**
>
> given a cyclic group (G,$\oplus$), the generator α, the value β
>
> $$\beta = \underbrace{\alpha \oplus \alpha \oplus \ldots \alpha}_{\text{k times}} = \begin{cases} \alpha^k, & \oplus \text{ is like mulitplication} \\ k \cdot \alpha, & \oplus \text{ is like addition} \end{cases}$$
>
> try to calculate the k

Some cyclic groups can be used to build crypto system:

- Z_p^*, with the operation is multiplication on modulo p
- $GF(Z^m)$, gauss field, with the operation is polynomial mulitplication on modulo Z
- Elliptic curve
- Hyper elliptic curve
- .....

Attacks against discrete logarithm problem

Attack one : brute force

$$\begin{aligned}\alpha^1 &== \beta? \\ \alpha^2 &== \beta? \\ &\cdots \\ \alpha^{|G|} &== \beta?\end{aligned}$$

$$O(|G|) \text{ complexity}$$

we will need more than 80 bits length to keep it secure against brute force.

Attack two : square-root-attack

baby-step giant-step algorithm and Pollard's-rho method

- Handbook of Applied Cryptography,P104,3.6.2 Baby-step giant-step algorithm
- Handbook of Applied Cryptography,P106,3.6.3 Pollard's rho algorithm for logarithms
- Squre-root-attack works for all cyclic group discrete logarithm
- The complexity of the attack is $O(\sqrt{|G|})$, so we need more than 160 bits to protect us

- this is the best known attack against elliptic curve, so the minimum key length for ECC is 160 bits

Attack three : index-calculate attack

- this kind of attack only exists for certain groups,like $Z_p^*, GF(Z^m)$.
- This attack is very powerful, it can break 700-800 bits long system, so we need at least 1024 to protect us.
- Handbook of Applied Cryptography, P109,3.6.5 Index-calculus algorithm.

Encrypt data with the discrete log problem

An overview of public key system

service	RSA	DL on Z_p^*	DL on Elliptic curve
Key exchange	RSA	D-H	ECDH(elliptic curve diffi-hellman)
Digital Signature	RSA	ElgamalDSA	ECDSA
Data Encryption	RSA	Elgamal	EC-elgamal

Elgamal encryption algorithm

The big idea behind the algorithm:

1. Do normal D-H key exchange
2. encrypt message m

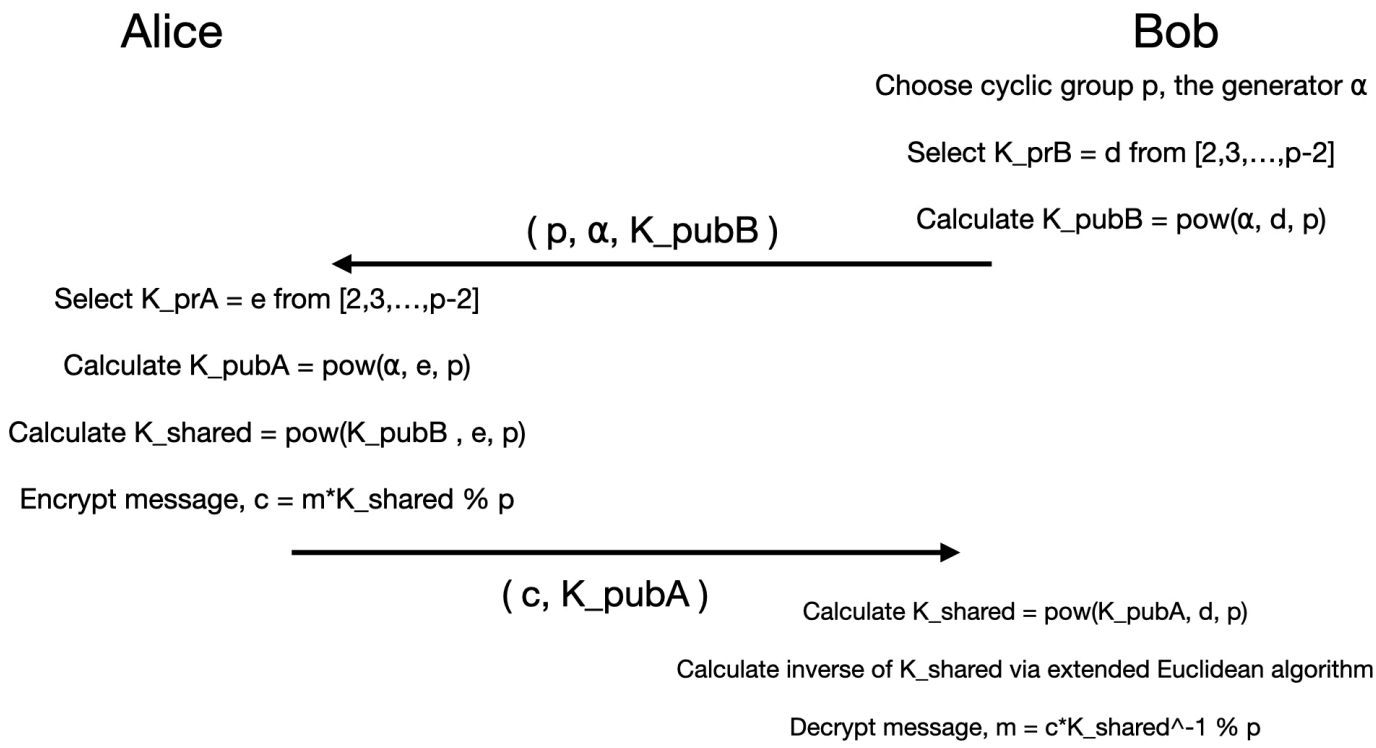
$$c \equiv m \cdot K_{\text{shared}} \mod p$$

3. Decryption

$$m \equiv c \cdot K_{\text{shared}}^{-1} \mod p$$

The Elgamal algorithm:

Just some reordering of the big idea



Advantages of Elgamal over direct Diffi-Hellman:

- Bob's public key is fixed and p,α are chosen by him.

  If some one else want to communicate with Bob, Bob do not have to do any extra work, he only generate his public key once and make it public known.

- the public key of Alice changes each time she communicate with Bob, that is to say, encrypt the same message will yield different ciphertext, making Elgamal a **probabilistic crypto system**.

Computation trick

we can merge the last two step in Bob side, which means merge fast-exponential and extended Euclidean into only one fast-exponential

we achieve this by using Fermat's little theorem

$$m \equiv K_{\text{shared}}^{-1} \cdot c \mod p$$

$$\equiv ((K_{\text{pubA}})^d)^{-1} \cdot c \mod p$$

$$\equiv 1 \cdot ((K_{\text{pubA}})^d)^{-1} \cdot c \mod p$$

$$\because K_{\text{pubA}} \in Z_p^*$$

$$\therefore (K_{\text{pubA}})^{p-1} \equiv 1 \mod p \quad (\text{Fermat's littel theorem})$$

$$\therefore m \equiv (K_{\text{pubA}})^{(p-1-d)} \cdot c \mod p$$

Attack

- try to solve the discrete logarithm problem(🤷)
- if Alice forget to change her private key, the attacker will knows that(her public key will stay same, too.), so he only need to kown one pair of (message,cipher), then he can calculate the K_{shared} and break the rest of the ciphers.