CS 4920/5920 Applied Cryptography

Chapter 10 Other Public Key Cryptosystems

Diffie-Hellman Key Exchange

- the first published public-key algorithm
- by Diffie & Hellman in 1976 along with the exposition of public key concepts
  - note: now know that Williamson (UK CESG) secretly proposed the concept in 1970
- is a practical method for public exchange of a secret key
- used in a number of commercial products
- the invention of public-key cryptography: https://www.youtube.com/watch?v=ROCray7RTqM

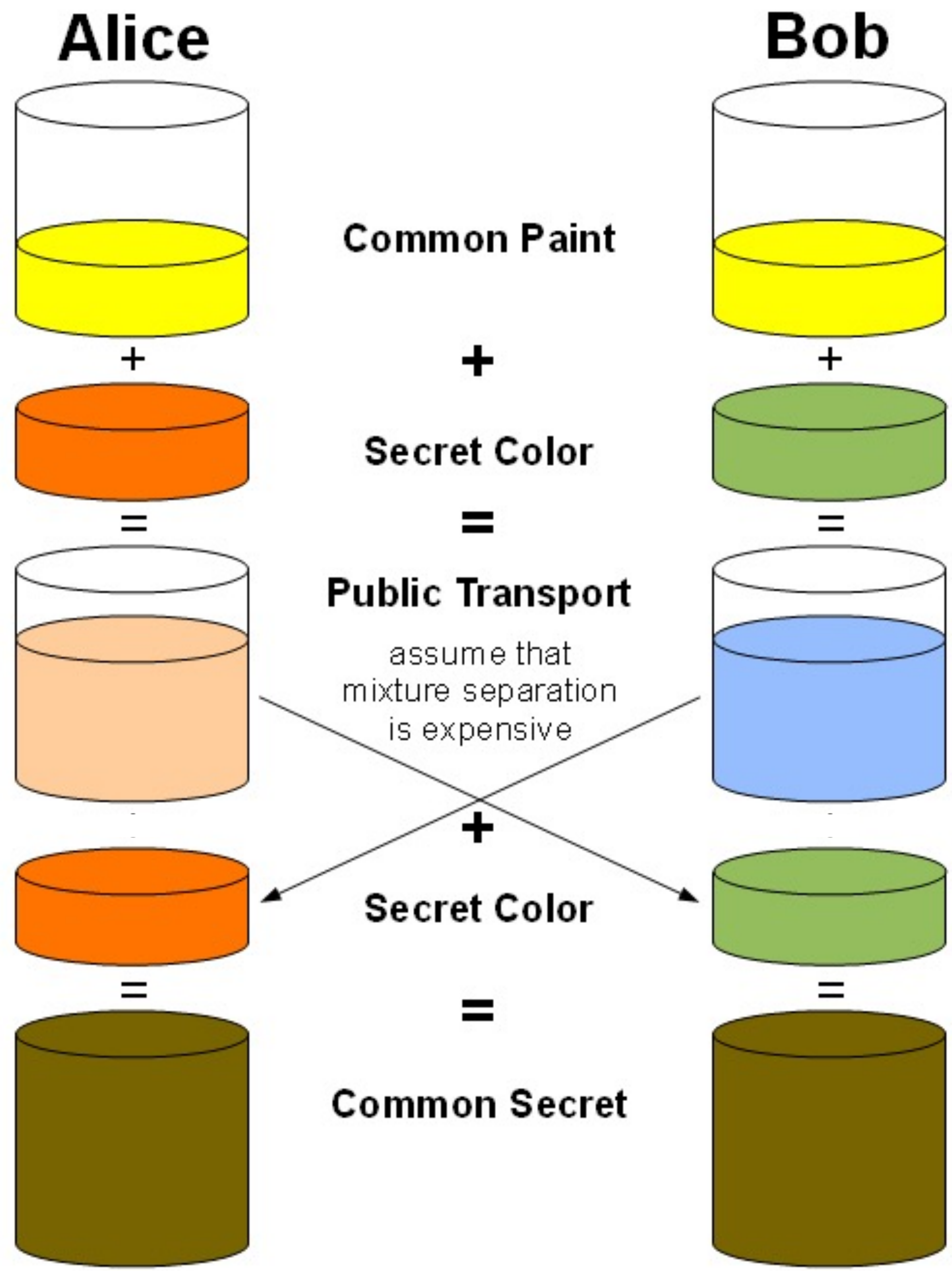
Diffie-Hellman Key Exchange

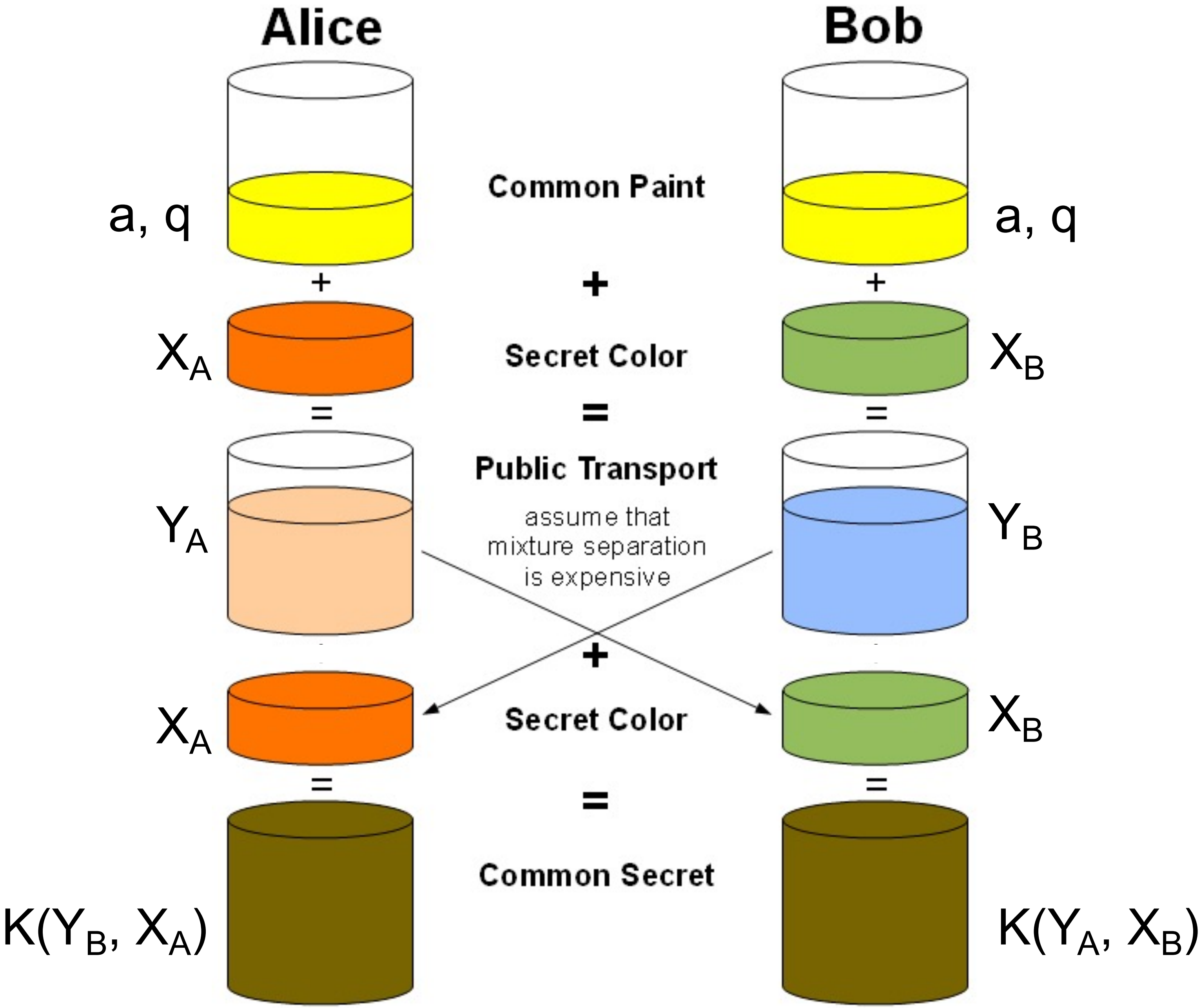
- enables two users to securely exchange a *secret key*, which can be used for subsequent encryption
- value of the *secret key* depends on the participants (and their private and public key information)
- security relies on
  - exponentiation in a finite (Galois) field (modulo a prime or a polynomial) is easy
  - computing discrete logarithms (similar to factoring) is hard

    recall: find i such that $b = a^i \pmod p$, written as $i = \text{dlog}_{a,p}\, b$

    if a is a primitive root of p then i always exists

    $a, a^2, a^3, \ldots, a^{p-1} \pmod p$ are distinct integers from 1 to p-1


Alice
Bob
Common Paint
+
+
+
Secret Color
=
=
=
Public Transport
assume that
mixture separation
is expensive
+
Secret Color
=
=
=
Common Secret



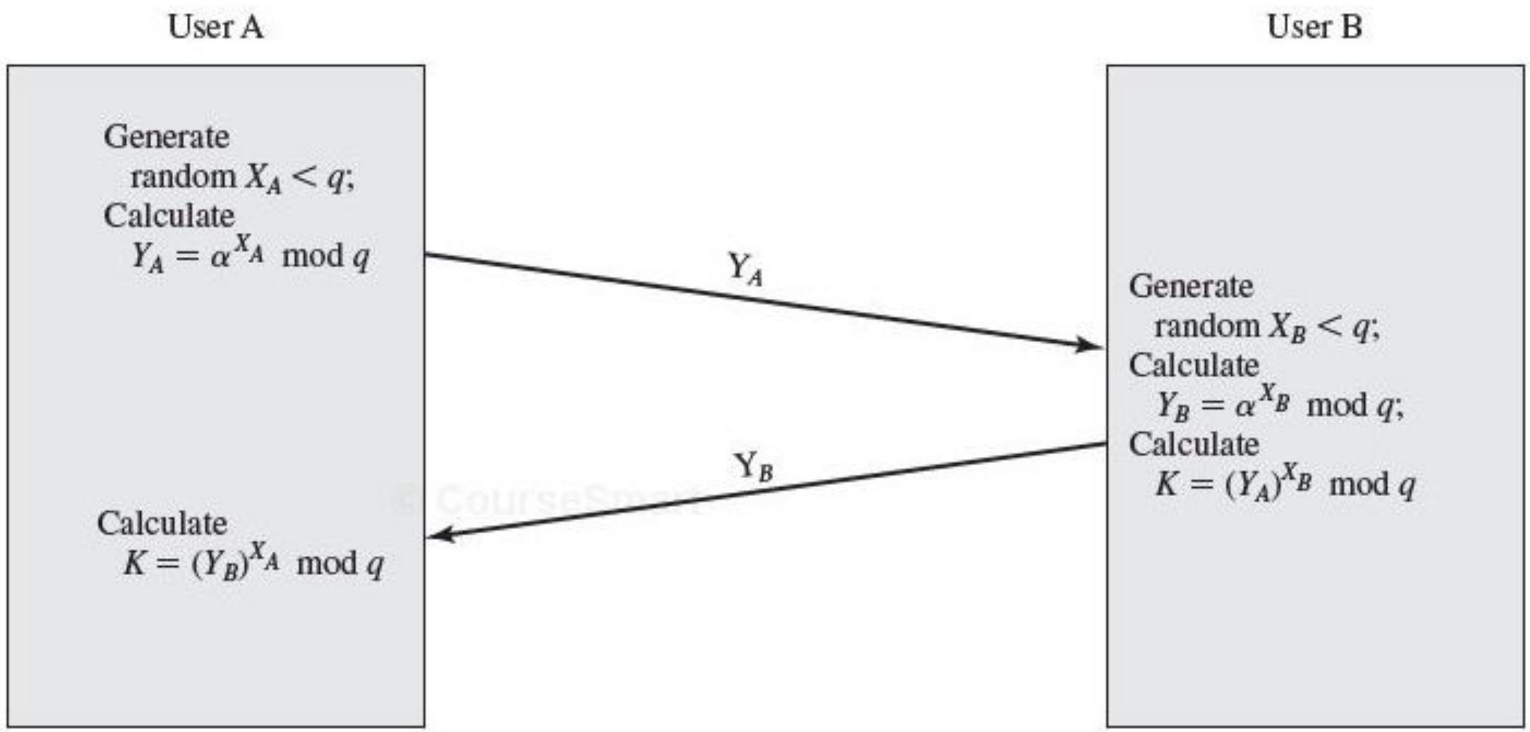
Alice
Bob
Common Paint
a, q
a, q
+
Secret Color
X_A
X_B
=
Public Transport
assume that mixture separation is expensive
Y_A
Y_B
+
Secret Color
X_A
X_B
=
Common Secret
$K(Y_B, X_A)$
$K(Y_A, X_B)$

Diffie-Hellman Setup

- all users agree on global parameters:
  - a large prime integer: q
  - a primitive root of q: a
- suppose users A and B wish to exchange a *secret key*
- user A
  - selects a random integer $X_A < q$, computes $Y_A = a^{X_A} \bmod q$
- user B independently
  - selects a random integer $X_B < q$, computes $Y_B = a^{X_B} \bmod q$
- each side keeps X as private key, makes Y as public key

Key Exchange Protocols

- could be between two users A and B
- could be between a group of users
- both are vulnerable to a Man-in-the-Middle Attack



Diffie-Hellman Key Exchange

- K_{AB} is the exchanged *secret key* for users A & B:

$$\begin{aligned} K_{AB} &= Y_B^{X_A} \bmod q && \text{//A can compute} \\ &= (a^{X_B} \bmod q)^{X_A} \bmod q \\ &= (a^{X_B})^{X_A} \bmod q \\ &= a^{X_B X_A} \bmod q \\ &= (a^{X_A})^{X_B} \bmod q \\ &= (a^{X_A} \bmod q)^{X_B} \bmod q \\ &= Y_A^{X_B} \bmod q && \text{//B can compute} \end{aligned}$$

- A and B subsequently use K_{AB} for symmetric encryption
- attacker knows q, a, Y_A, Y_B; needs to know X_A or X_B
  - $X_A = \mathrm{dlog}_{a,q}\, Y_A$ or $X_B = \mathrm{dlog}_{a,q}\, Y_B$ → hard for large numbers

Quiz 13a

Diffie-Hellman Example

- users Alice & Bob who wish to exchange a *secret key*:
- agree on prime $q=353$ and $a=3$ (is a primitive root)
- select random private keys:
  - A chooses $X_A=97$, B chooses $X_B=233$
- compute respective public keys:
  - $Y_A = a^{X_A} \bmod q$ (Alice)
  - $Y_B = a^{X_B} \bmod q$ (Bob)
- compute shared session key as:
  - $K_{AB} = Y_B^{X_A} \bmod q$ (Alice)
  - $K_{AB} = Y_A^{X_B} \bmod q$ (Bob)

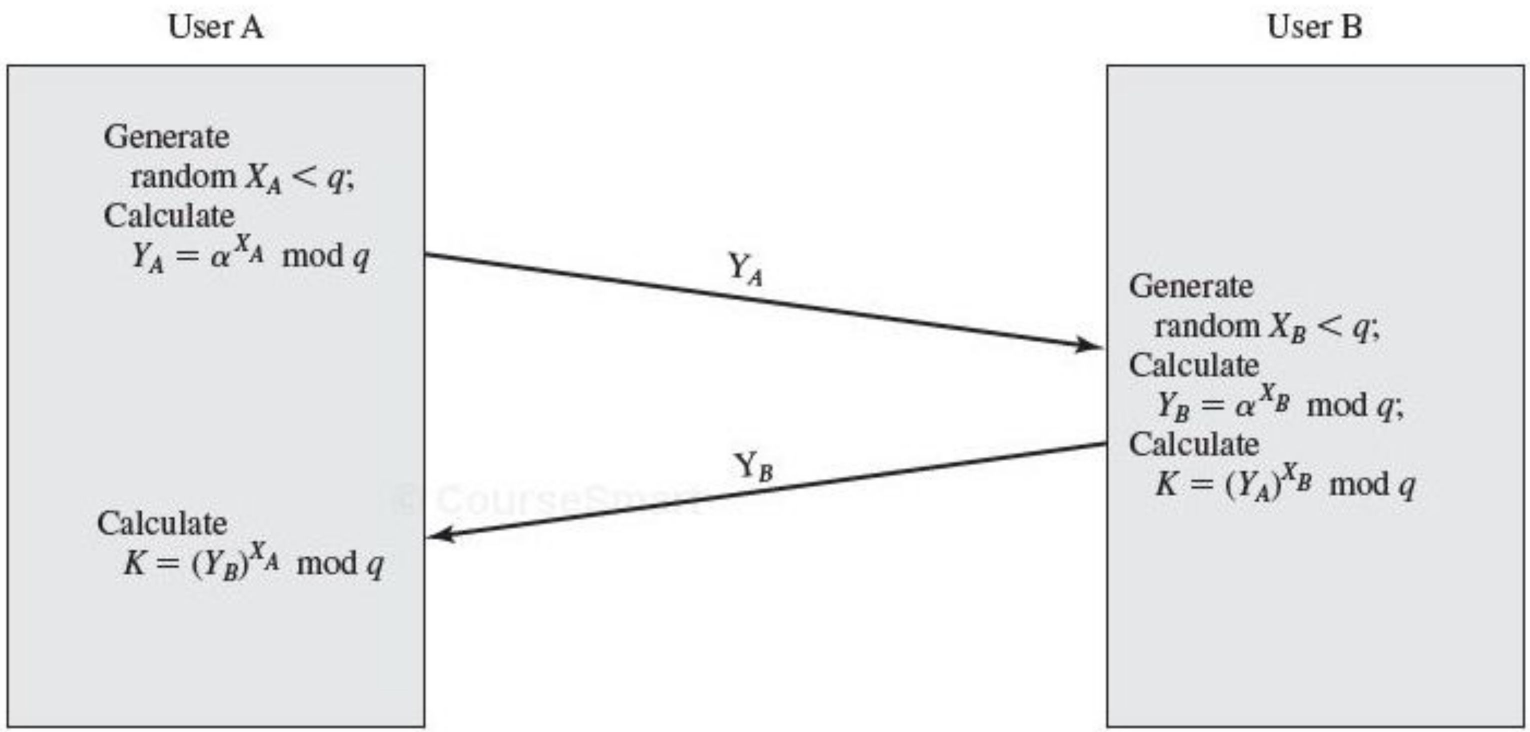
Quiz 13a

Diffie-Hellman Example

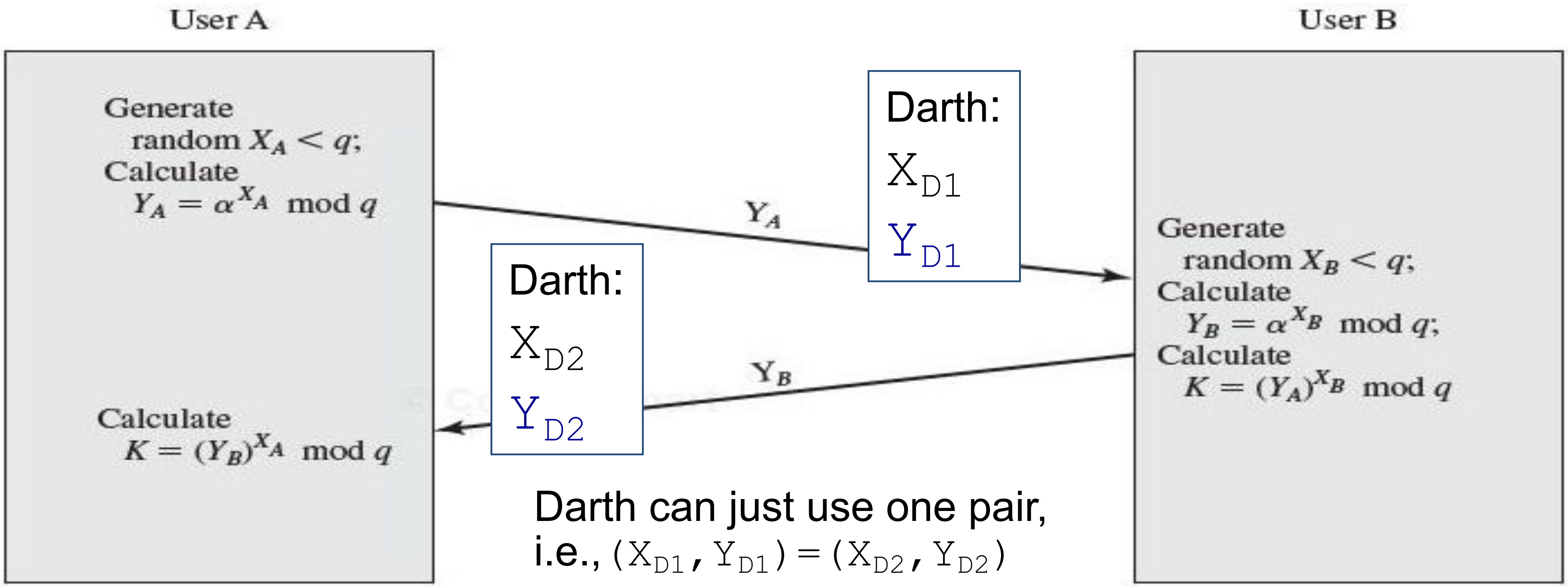
- users Alice & Bob who wish to exchange a *secret key*:
- agree on prime $q=353$ and $a=3$ (is a primitive root)
- select random private keys:
  - A chooses $X_A=97$, B chooses $X_B=233$
- compute respective public keys:
  - $Y_A=3^{97} \bmod 353 = 40$ (Alice)
  - $Y_B=3^{233} \bmod 353 = 248$ (Bob)
- compute shared session key as:
  - $K_{AB}= Y_B^{X_A} \bmod 353 = 248^{97} = 160$ (Alice)
  - $K_{AB}= Y_A^{X_B} \bmod 353 = 40^{233} = 160$ (Bob)

Key Exchange Protocols

- could be between two users A and B
- could be between a group of users
- both are vulnerable to a Man-in-the-Middle Attack

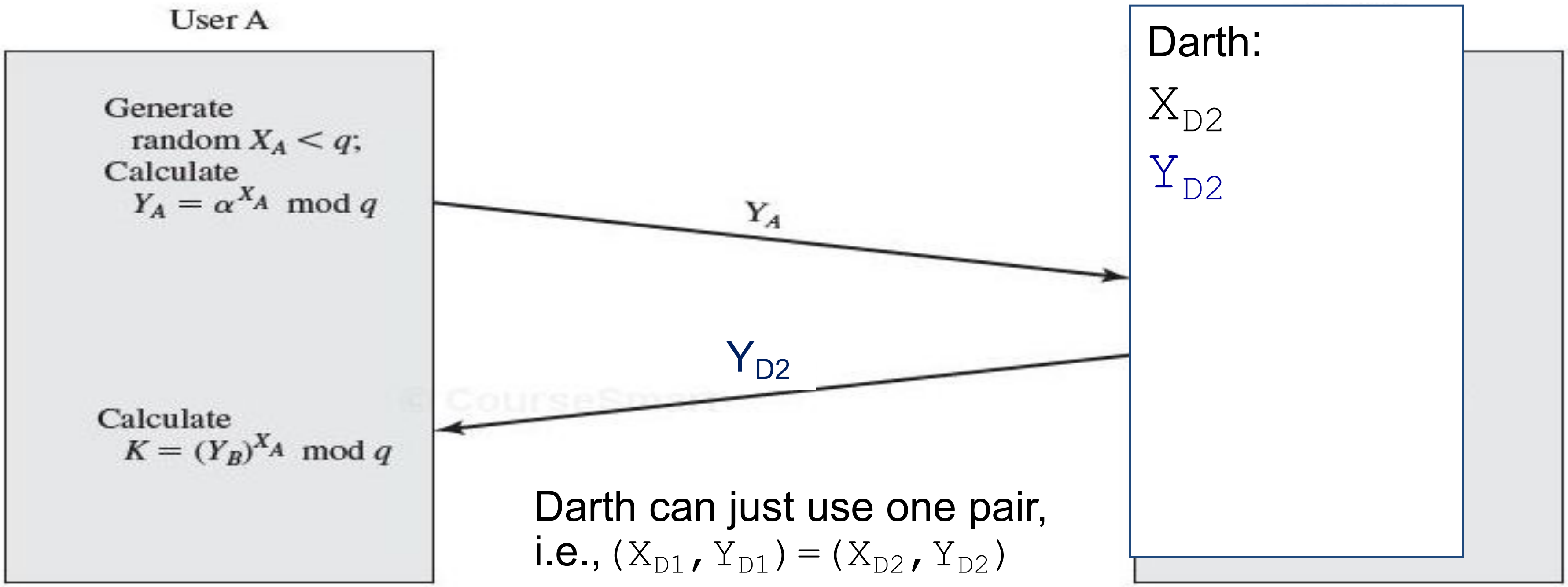


Man-in-the-Middle Attack



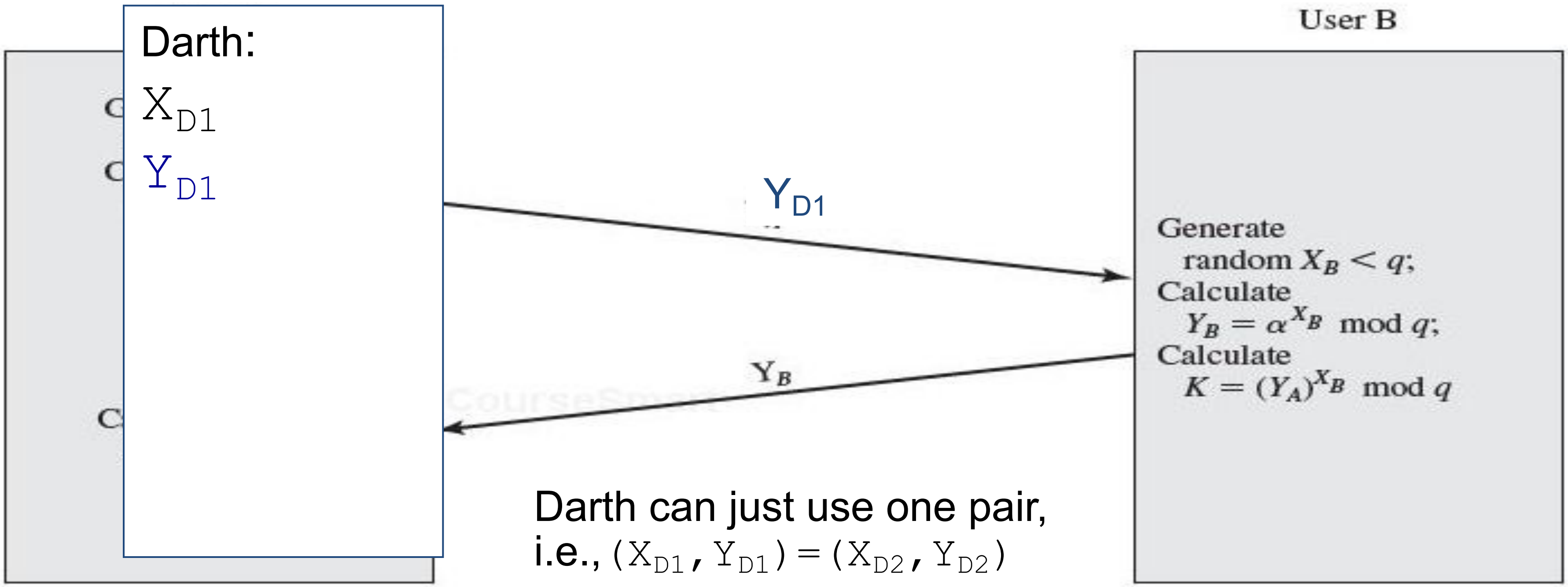
- $K_{AD} = {Y_{D2}}^{X_A} \bmod q = {Y_A}^{X_{D2}} \bmod q$
- $K_{BD} = {Y_B}^{X_{D1}} \bmod q = {Y_{D1}}^{X_B} \bmod q$
- Darth can eavesdrop or modify messages
- due to the authenticity of two parties are not established
- use public-key certificates and digital signatures to overcome

Man-in-the-Middle Attack



- $\boxed{K_{AD} = Y_{D2}^{X_A} \bmod q = Y_A^{X_{D2}} \bmod q}$
- $K_{BD} = Y_B^{X_{D1}} \bmod q = Y_{D1}^{X_B} \bmod q$
- Darth can eavesdrop or modify messages
- due to the authenticity of two parties are not established
- use public-key certificates and digital signatures to overcome

Man-in-the-Middle Attack



- $K_{AD} = {Y_{D2}}^{X_A} \bmod q = {Y_A}^{X_{D2}} \bmod q$
- $\boxed{K_{BD} = {Y_B}^{X_{D1}} \bmod q = {Y_{D1}}^{X_B} \bmod q}$
- Darth can eavesdrop or modify messages
- due to the authenticity of two parties are not established
- use public-key certificates and digital signatures to overcome

ElGamal Cryptography

- public-key cryptosystem related to Diffie-Hellman (DH)
- uses exponentiation in a finite (Galois) field
- security depends on difficulty of computing discrete logarithms, as in DH
- very closely related to the DH
- used in a number of standards
  - DSS (digital signature standard)
  - S/MIME email standard

ElGamal Setup

- all users agree on global parameters:
  - a large prime integer: q
  - a primitive root of q: a
- user B wants to securely send a message to user A
- user A
  - selects a random integer $X_A < q-1$
  - computes $Y_A = a^{X_A} \bmod q$
  - A's private key is X_A; A's public key is $\{q, a, Y_A\}$

ElGamal Message Exchange

- B encrypt a message to send to A computing
  - represent message M in range: $0 \leq M \leq q-1$
    - longer messages must be sent as blocks
  - choose a random integer k with $1 \leq k \leq q-1$
  - compute a one-time key $K = Y_A^k \bmod q$
  - encrypt and send M as a pair of integers (C_1, C_2) where
    - $C_1 = a^k \bmod q$; $C_2 = KM \bmod q$
- A then recovers message by
  - recovering key K as $K = C_1^{X_A} \bmod q$
  - computing M as $M = C_2 K^{-1} \bmod q$
  - Proof (K: same as in DH; K^{-1}: multiplicative inverse in GF(q))

ElGamal Message Exchange

- B encrypt a message to send to A computing
  - represent message M in range: $0 \le M \le q-1$
    - longer messages must be sent as blocks
  - choose a random integer k with $1 \le k \le q-1$
  - compute a one-time key $K = Y_A^{k} \bmod q$
  - encrypt and send M as a pair of integers (C_1, C_2) where
    - $C_1 = a^{k} \bmod q$; $C_2 = KM \bmod q$

k, K for one-time key

- A then recovers message by
  - recovering key K as $K = C_1^{X_A} \bmod q$
  - computing M as $M = C_2 K^{-1} \bmod q$
  - Proof (K: same as in DH; K^{-1}: multiplicative inverse in GF(q))

Use C_1 to retrieve K

Use C_2 to retrieve M

Quiz 13b

ElGamal Example

- use field GF(q) w/ $q=19$ and $a=10$ (primitive root)
- Alice computes her key:
  - A chooses $X_A=5$ & computes $Y_A = 10^5 \bmod 19 = 3$
- Bob send message $M=17$ as $(11,5)$ by
  - choosing a random $k=6$
  - computing $K = Y_A{}^k \bmod q$
  - computing $C_1 = a^k \bmod q$;
    $C_2 = KM \bmod q$
- Alice recovers original message by computing:
  - recover $K = C_1{}^{X_A} \bmod q$
  - Compute multiplicative inverse K^{-1} in $GF(q)$
  - recover $M = C_2K^{-1} \bmod q$

Quiz 13b

ElGamal Example

- use field GF(q) w/ $q=19$ and $a=10$ (primitive root)
- Alice computes her key:
  - A chooses $X_A=5$ & computes $Y_A = 10^5 \bmod 19 = 3$
- Bob send message $M=17$ as $(11,5)$ by
  - choosing a random $k=6$
  - computing $K = Y_A^k \bmod q = 3^6 \bmod 19 = 7$
  - computing $C_1 = a^k \bmod q = 10^6 \bmod 19 = 11$;
    $C_2 = KM \bmod q = 7*17 \bmod 19 = 5$
- Alice recovers original message by computing:
  - recover $K = C_1^{X_A} \bmod q = 11^5 \bmod 19 = 7$
  - Compute multiplicative inverse $K^{-1} = 7^{-1} = 11$
  - recover $M = C_2K^{-1} \bmod q = 5*11 \bmod 19 = 17$

ElGamal Long Message Exchange

- longer messages must be sent as blocks, and a **unique value of k** should be used for each block
  - otherwise, once one plaintext block, e.g, M_1, is known by attackers, others can be computed. Let

$$C_{1,1} = \alpha^k \bmod q; C_{2,1} = KM_1 \bmod q$$

$$C_{1,2} = \alpha^k \bmod q; C_{2,2} = KM_2 \bmod q$$

Then,

$$\frac{C_{2,1}}{C_{2,2}} = \frac{KM_1 \bmod q}{KM_2 \bmod q} = \frac{M_1 \bmod q}{M_2 \bmod q}$$

If M_1 is known, then M_2 is easily computed as

$$M_2 = (C_{2,1})^{-1} C_{2,2} M_1 \bmod q$$

Summary

- Based on discrete log problem:
  - Diffie-Hellman key exchange
  - ElGamal cryptography