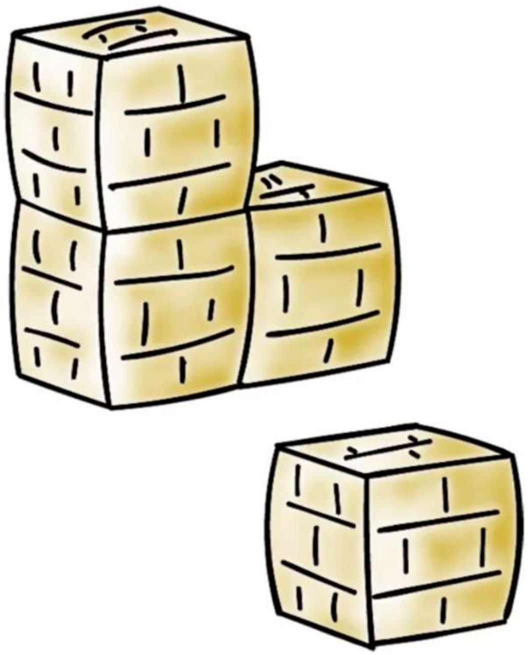
Public Key Algorithms
Lesson Introduction

- Modular arithmetic
- RSA
- Diffie-Hellman

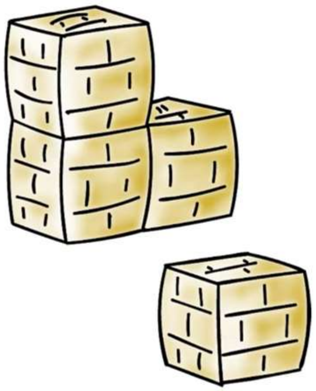
Modular Arithmetic



- **Public key algorithms are based on modular arithmetic**

- Modular addition
- Modular multiplication
- Modular exponentiation

Modular Arithmetic



- **Addition modulo (MOD) *M***
- **Additive inverse:** addition MOD *M* yields 0
  - E.g., M=10, for k=2, its inverse is k^{-1}=8 because 2+8 MOD 10 = 0
- **Reversible:** by adding the inverse
  - Convenient for decryption
  - E.g., for c = 3, p = c+k = 3+2 MOD 10 = 5;

    $p+k^{-1}$ = 5+8 MOD 10 = 3 = c

Modular Multiplication

- Multiplication modulo M
- **Multiplicative inverse**: multiplication MOD M yields 1
  - E.g., M=10, 3 and 7 are inverse of each other because 3×7 MOD 10 = 1
- **Only some numbers have inverse**
  - But 2, 5, 6, 8 do not have inverse when M=10

与M互质的数才对M有multiplicative inverse

Modular Multiplication

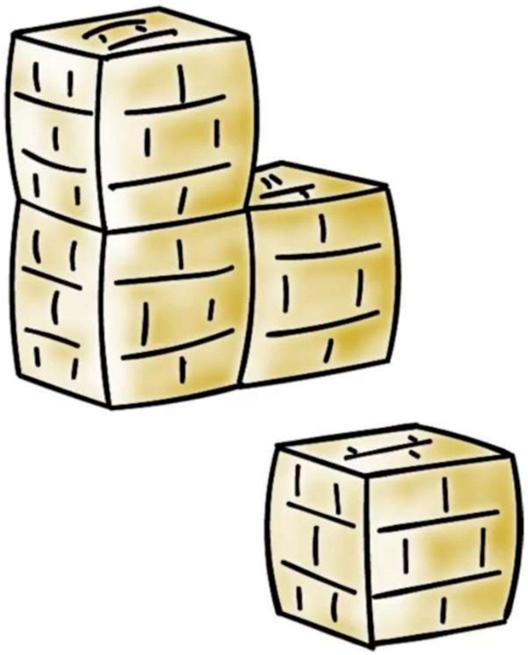
- **Use Euclid's algorithm to find inverse**
  - Given x, n, it finds y such that $x \times y \bmod n = 1$
- **Only the numbers relatively prime to n** has MOD n multiplicative inverse

Totient Function

$$\varphi(n)$$

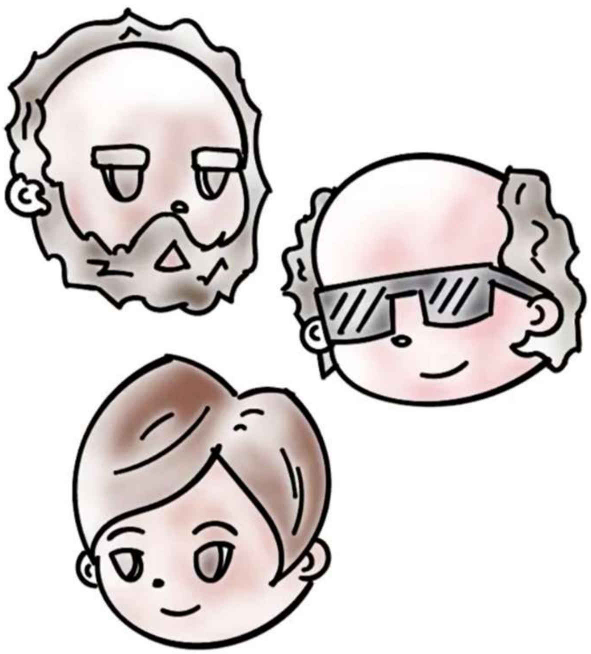
- **x is relatively prime to n**: no common factor other than 1
- **Totient function ø(n):** number of integers smaller than n and relatively prime to n
  - if n is prime, ø(n)=n-1
  - if n=p×q, and p, q are primes, ø(n)=(p-1)(q-1)
  - if n=p×q, and p, q are relative prime to each other, ø(n)=ø(p)ø(q)

Modular Exponentiation



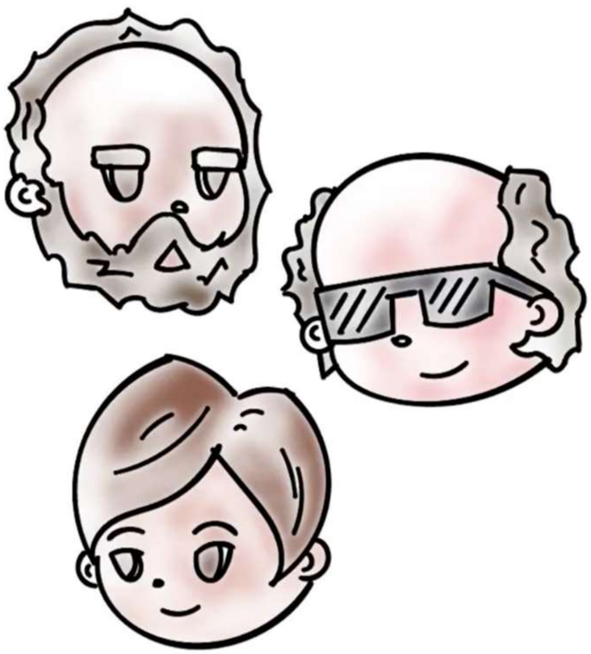
- $x^y \bmod n = x^{y \bmod \phi(n)} \bmod n$
- if $y = 1 \bmod \phi(n)$ then $x^y \bmod n = x \bmod n$

RSA (Rivest, Shamir, Adleman)



- **Widely used**, and one of the first (1977)
- Support both **public key encryption and digital signature**
- **Assumption/theoretical basis:**
  - Factoring a very large integer is hard

RSA Characteristics



- **Variable key length**
- **Variable plaintext block size**
  - Plaintext treated as an integer, and must be “smaller” than the key
  - Ciphertext block size is the same as the key length

RSA Algorithm

Key Generation

Select p, q	p and q both prime; $p \neq q$
Calculate $n = p \times q$	
Calculate $\phi(n) = (p-1)(q-1)$	
Select integer e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calculate d	$de \bmod \phi(n) = 1$
Public key	$KU = \{e, n\}$
Private key	$KR = \{d, n\}$

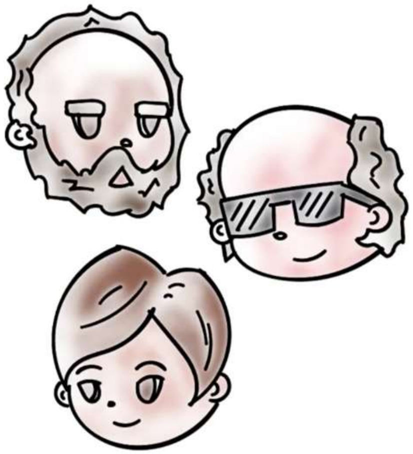
Encryption

Plaintext:	$M < n$
Ciphertext:	$C = M^e \pmod{n}$

Decryption

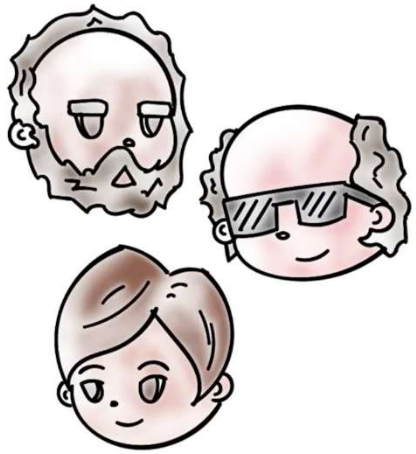
Plaintext:	C
Ciphertext:	$M = C^d \pmod{n}$

How Does RSA Work?



- **Given KU = <e, n> and KR = <d, n>**
  - **encryption:** $c = m^e \bmod n$, $m < n$
  - **decryption:** $m = c^d \bmod n$
  - **signature:** $s = m^d \bmod n$, $m < n$
  - **verification:** $m = s^e \bmod n$

Why does RSA Work?



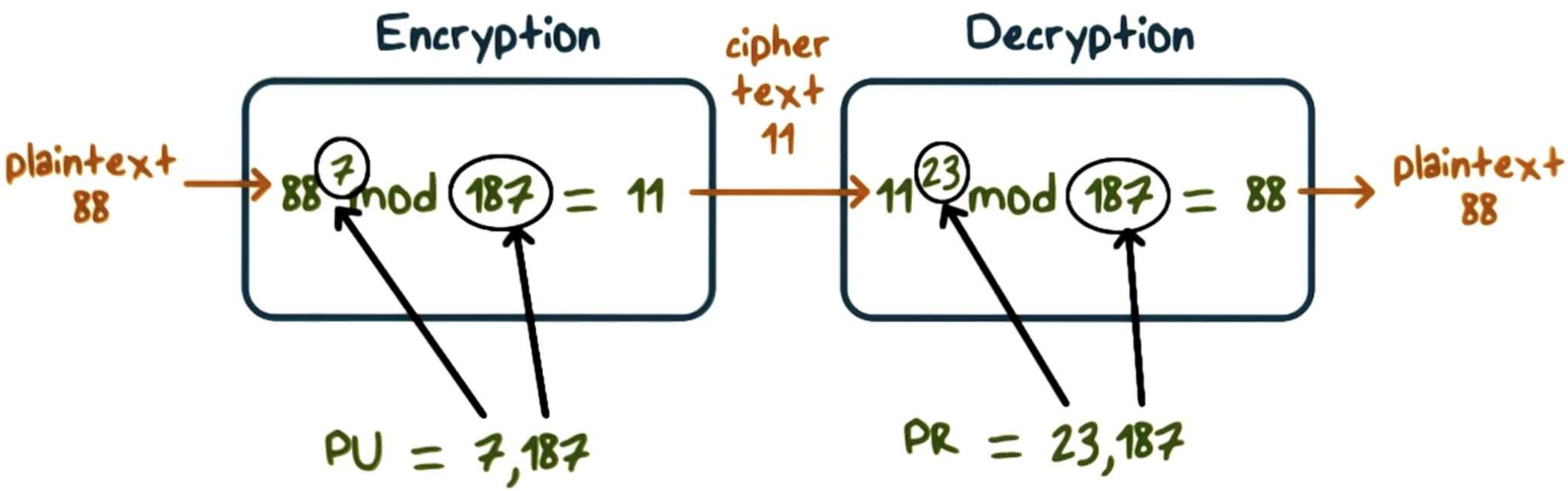
Given pub = <e, n> and priv = <d, n>

- $n = p \times q$, $\phi(n) = (p-1)(q-1)$
- $e \times d = 1 \bmod \phi(n)$
- $x^{e \times d} = x \bmod n$
- **encryption:** $c = m^e \bmod n$
- **decryption:** $m = c^d \bmod n =$

  $m^{e \times d} \bmod n =$

  $m \bmod n = m$ (since $m < n$)
- digital signature (similar)

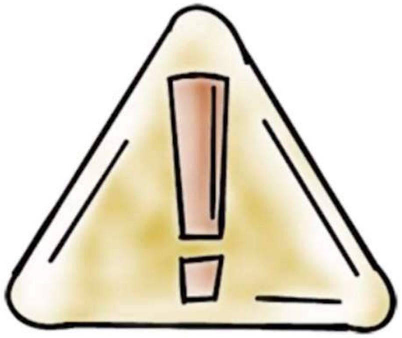
Why does RSA Work?



Why RSA is Secure?

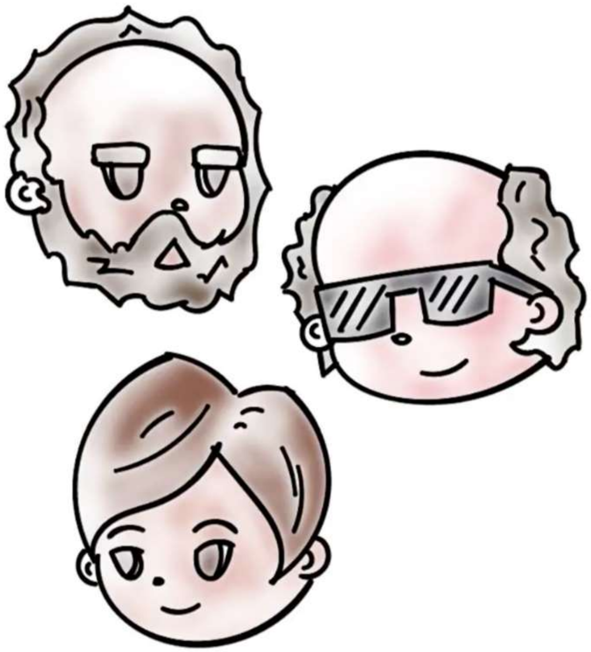
- Factoring an integer with at least 512-bit is **very hard**!
- **But if you can factor big number n then given public key <e,n>, you can find d, and hence the private key by:**
    - Knowing factors p, q, such that, $n = p \times q$
    - Then compute $\phi(n) = (p-1)(q-1)$
    - Then find d such that $e \times d = 1 \bmod \phi(n)$

RSA in Practice



- **Issues with schoolbook RSA**
  - **Deterministic:**
    - For the same key, a particular plaintext is always mapped to a particular ciphertext
  - Special-case plaintexts 0, 1, or -1 produce ciphertexts 0, 1, or -1 regardless of keys
  - **Malleable:**
    - Transforming a ciphertext into another leads to predictable transformation to plaintext
      - For $c = m^e \bmod n$, attacker change c to $s^e \times c$
      - Receiver gets $s \times m$ instead of m

RSA in Practice

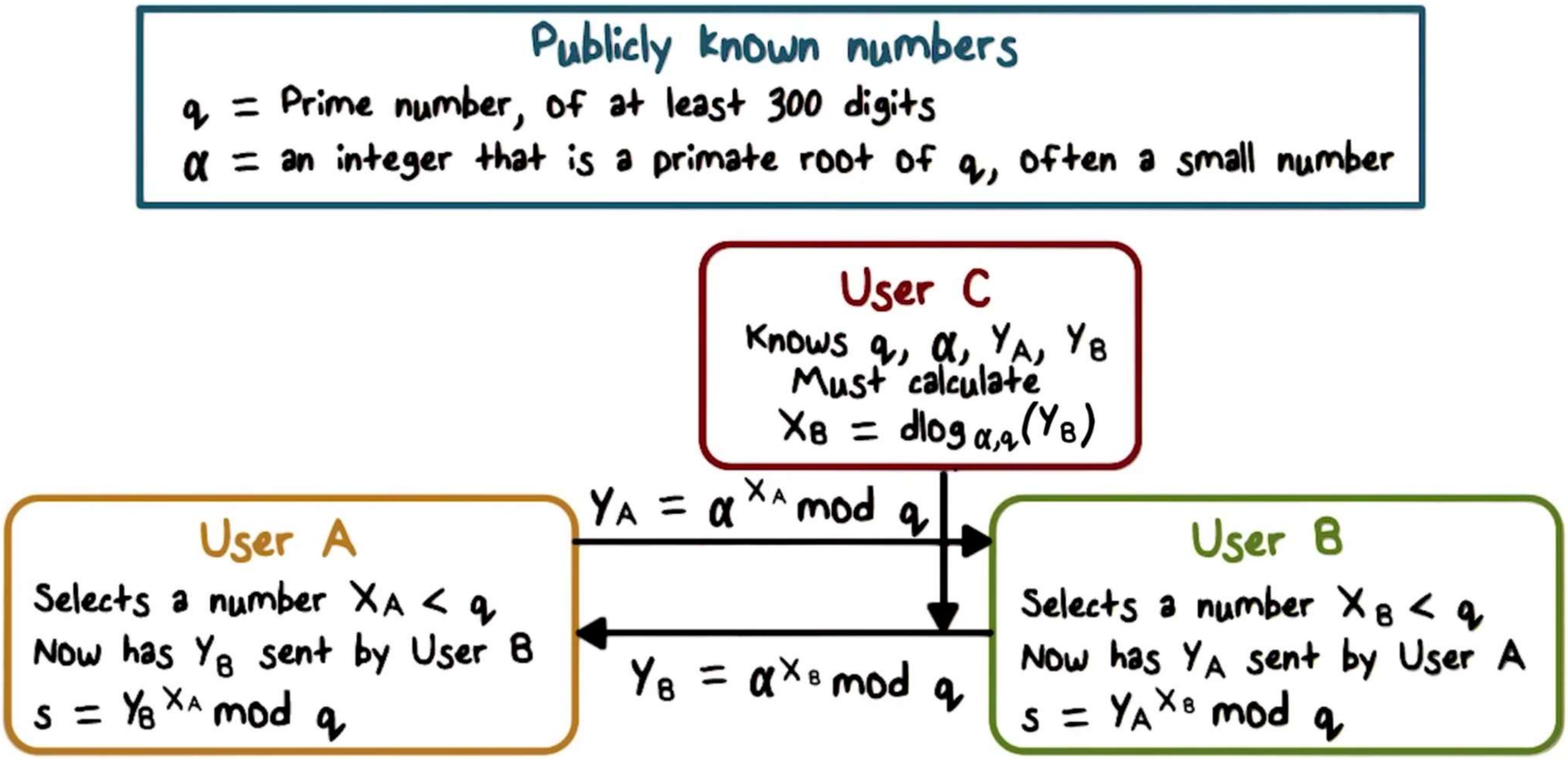


- PKCS (public key cryptography standard) uses OAEP (optimal asymmetric encryption padding)
    - Append padding (seeded from random byte) as prefix to m

Diffie and Hellman Key Exchange

- **First published** public-key algorithm
- By Diffie and Hellman in 1976 along with the **exposition of public key concepts**
- Used in a number of commercial products
- **Practical method to exchange a secret key** securely that can then be used for subsequent encryption of messages
- Security **relies on difficulty of computing discrete logarithms**

Diffie and Hellman Key Exchange



Diffie-Hellman Example

Have

- Prime number $q = 353$
- Prime number $\alpha = 3$

A and B each computer their public keys

- A computes $Y_A = 3^{97} \bmod 353 = 40$
- B computes $Y_B = 3^{233} \bmod 353 = 248$

Then exchange and computer secret key:

- For A: $K = (Y_B)^{XA} \bmod 353 = 248^{97} \bmod 353 = 160$
- For B: $K = (Y_A)^{XB} \bmod 353 = 40^{233} \bmod 353 = 160$

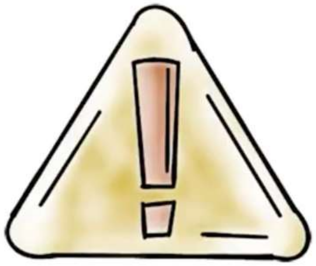
Attacker must solve:

- $3^{\alpha} \bmod 353 = 40$ which is hard
- Desired answer is 97, then computer key as B does

Diffie-Hellman Security

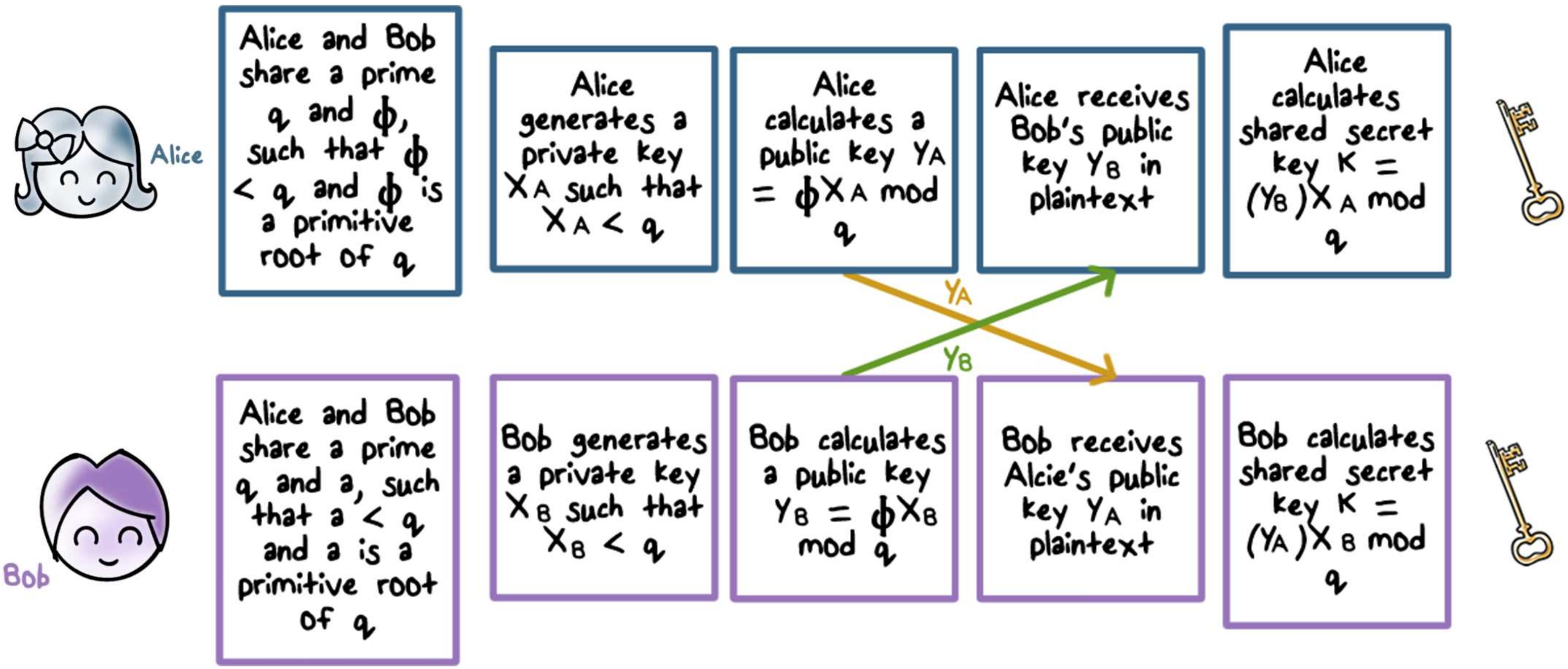
- **Shared key (the secret) itself never transmitted**
- Discrete logarithm is very hard
- $Y = \alpha^X \bmod q$
- **Conjecture:** given Y, α, and q, it is extremely hard to compute the value of X because q is a very large prime (discrete logarithm)

Diffie-Hellman Limitations

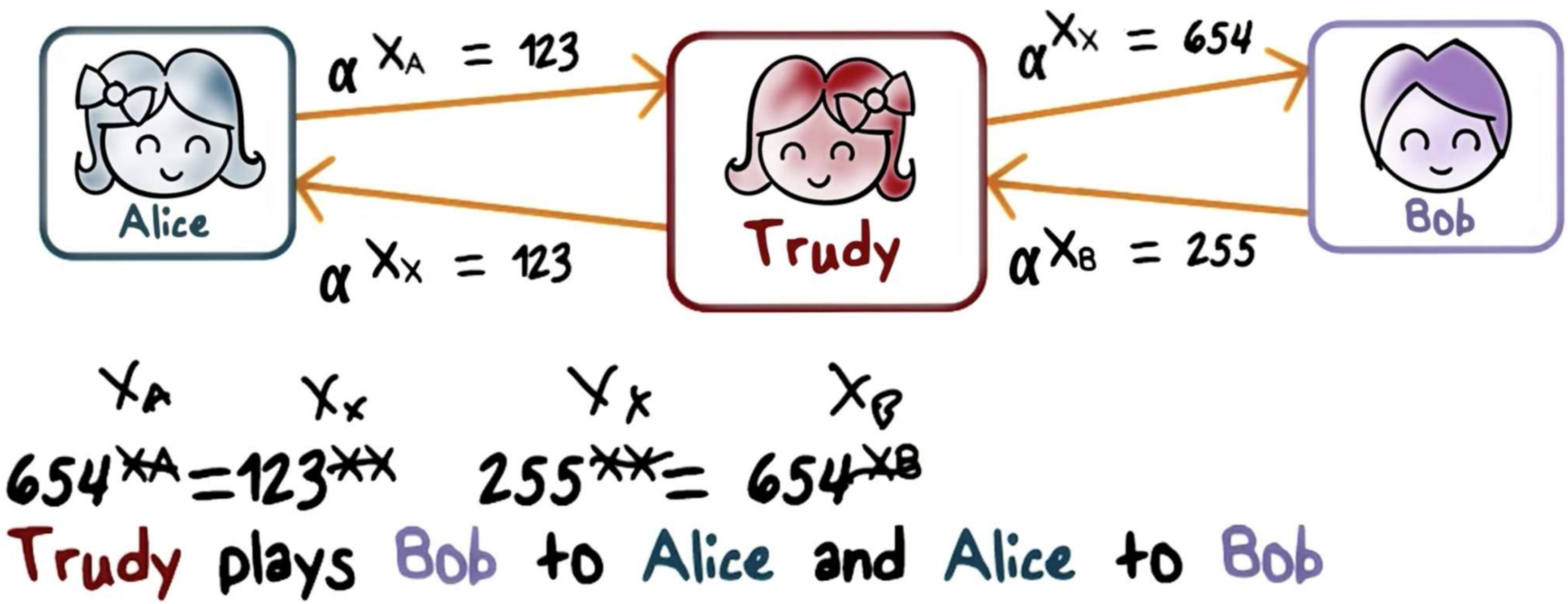


- Expensive exponential operation
  - DoS possible
- The scheme itself **cannot be used to encrypt anything –** it is for secret key establishment
- **No authentication**, so you cannot sign anything

Implementing the Diffie-Hellman Key Exchange



Bucket Brigade Attack, Man-in-the-Middle(MIM)



Other Public-Key Algorithms

Digital Signal Standard:

- Makes use of SHA-1 and the Digital Signature Algorithm (DSA)
- Originally proposed in 1991, revised in 1993 due to security concerns, and another minor revision in 1996
- **Cannot be used for encryption or key exchange**
- Uses an algorithm that is designed to provide only the digital signature function

Public Key Algorithms

Lesson Summary

- Modular arithmetic the foundations of several public key algorithms
- RSA can be used for encryption and signature, its security is based on assumption that factoring a larger integer into two primes is hard
- Diffie-Hellman is used for key exchange, its security is based on the assumption that discrete logarithm on large numbers is hard
  - No authentication means man-in-the-middle attack possible