Information Security

Diffie-Hellman Key Exchange

- First public-key type scheme proposed

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

- It is a practical method for public exchange of a secret key used in a number of commercial products

Diffie-Hellman Key Exchange

- A public-key distribution scheme
  - cannot be used to exchange an arbitrary message
  - rather it can establish a common key
  - known only to the two participants
- Value of key depends on the participants (and their private and public key information)
- Based on exponentiation in a finite (Galois) field (modulo a prime or a polynomial) - easy
- Security relies on the difficulty of computing discrete logarithms (similar to factoring) – hard

Diffie-Hellman Setup

- All users agree on global parameters:
  - large prime integer or polynomial q
  - a being a primitive root mod q
- Each user (eg. A) generates their key
  - chooses a secret key (number): $x_A < q$
  - compute their **public key**: $y_A = a^{x_A} \bmod q$
- Each user makes public that key y_A

Diffie-Hellman Key Exchange

- shared session key for users A & B is K_{AB}:

  $K_B = y_A^{x_B} \bmod q$ (which **B** can compute)

  $K_A = y_B^{x_A} \bmod q$ (which **A** can compute)
- K_{AB} is used as session key in private-key encryption scheme between Alice and Bob
- if Alice and Bob subsequently communicate, they will have the **same** key as before, unless they choose new public-keys
- attacker needs an x, must solve discrete log

Description of the algorithm

- **Suppose Alice and Bob want to agree upon a key to be used for encryption / decryption message that would be exchanged between them. Then diffie-hellman key exchange works as follows:**

- **1. Alice & bob agree on two large prime numbers: q and α**

  (note: these two integer need not be kept secret, insecure channel)

- **2. Alice choose another large random number Xa and calculate Ya such that,**

$$Ya = \alpha^{xa} \bmod q$$

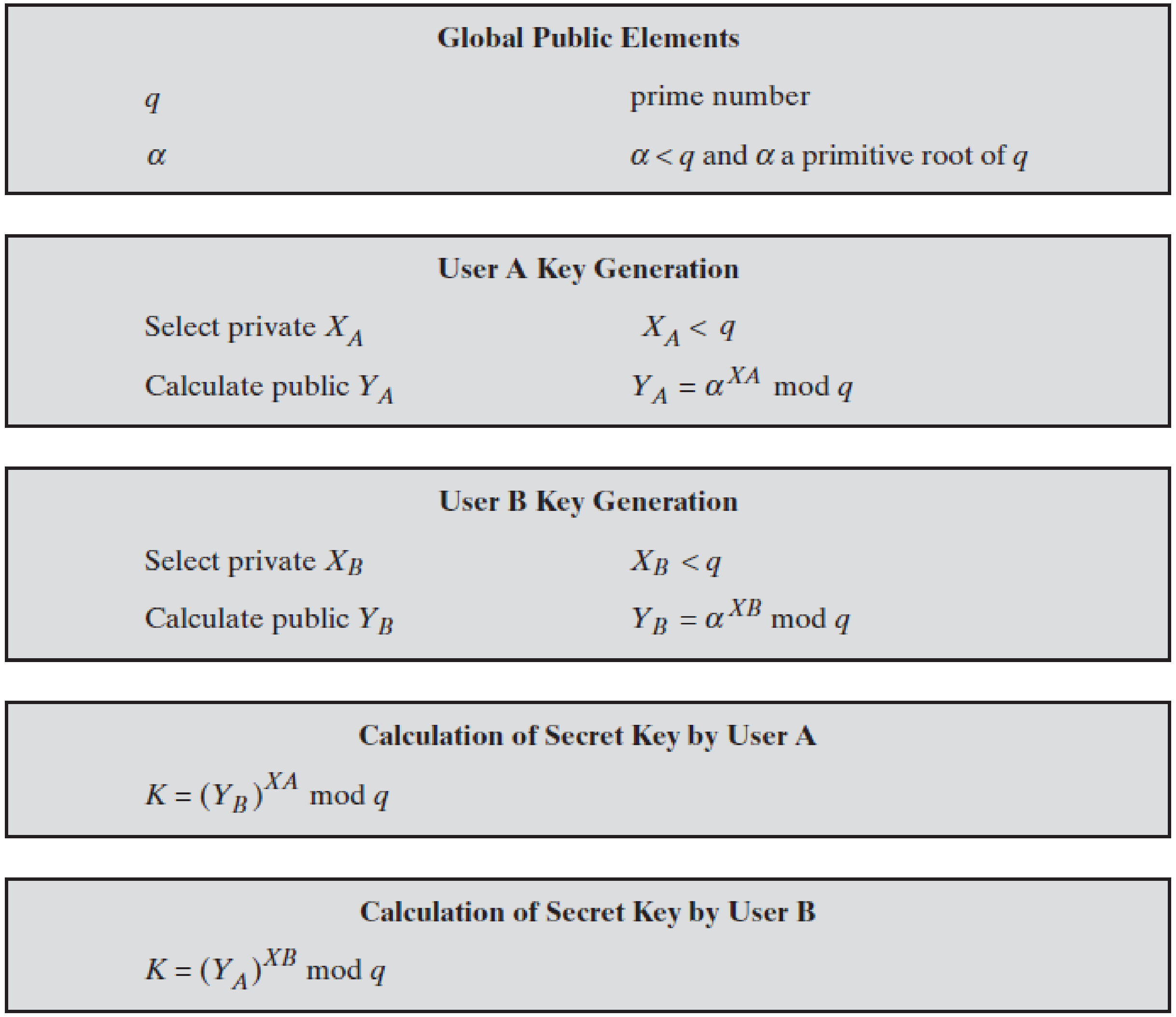
- **3. Alice send the number Xa to Bob.**
- **4. Bob independently choose another large random integer Xb and calculate Yb such that,**

$$Yb = \alpha^{Xb} \bmod q$$

- **5. Bob send the number B to Alice.**
- **6. Alice, now compute the secret key K1 as follows:**

$$K1 = Yb^{\,xa} \bmod q$$

Diffie-Hellman Key Exchange



Global Public Elements

q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation

Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

User B Key Generation

Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

Calculation of Secret Key by User A

$K = (Y_B)^{XA} \bmod q$

Calculation of Secret Key by User B

$K = (Y_A)^{XB} \bmod q$

Diffie-Hellman Key Exchange

It is a protocol that enables two users to establish a secret key using a public-key scheme based on discrete logarithms.

Primitive Root:

A primitive root of a prime number **p** as one whose powers modulo generate all the integers from **1** to **p-1**. That is, if **a** is a primitive root of the prime number **p**, then the numbers

$$a \bmod p, a^2 \bmod p, \ldots, a^{p-1} \bmod p$$

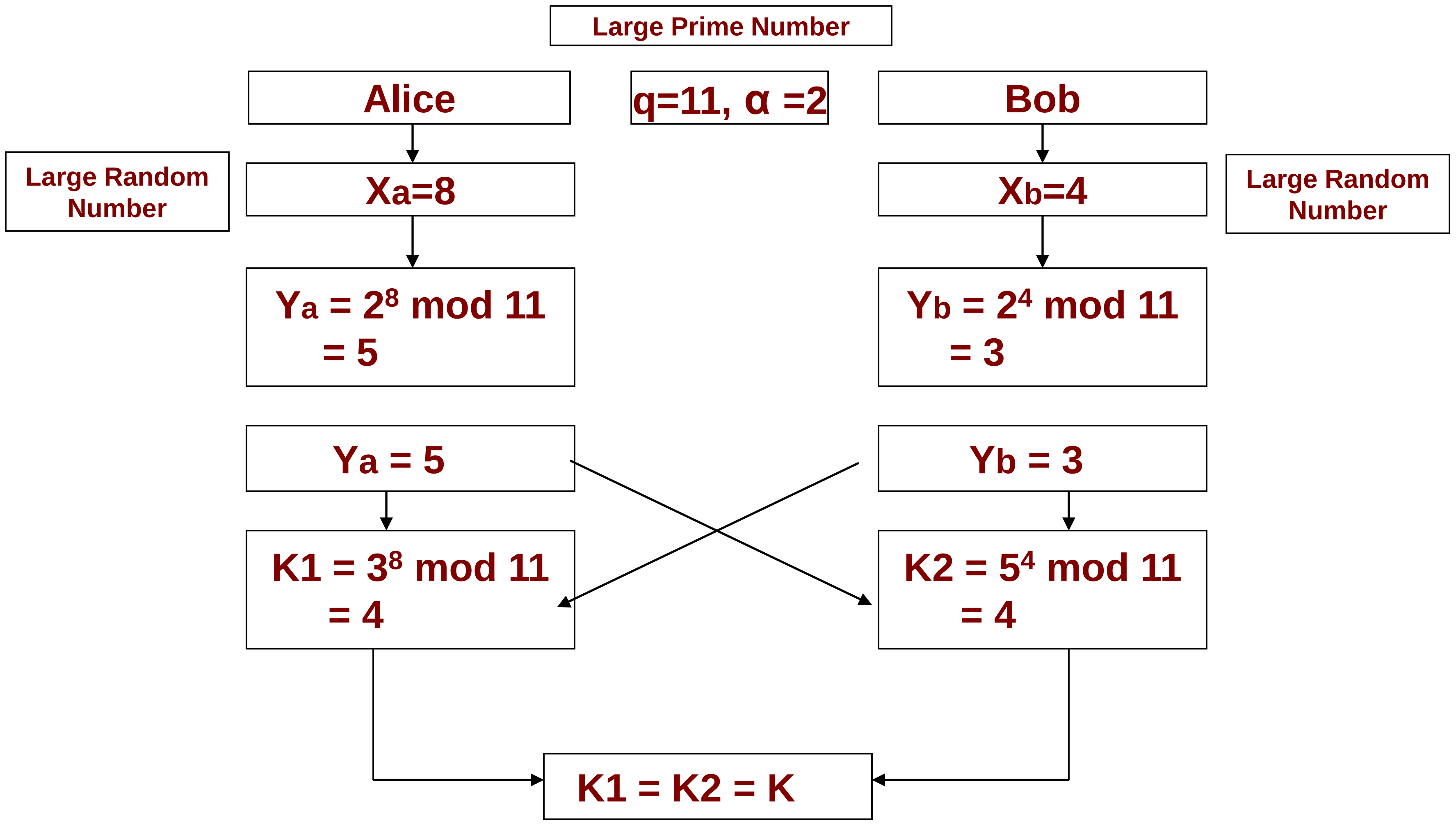
are distinct and consist of the integers from **1** through **p - 1** in some permutation.

For any integer **b** and a primitive root **a** of prime number **p**, we can find a unique exponent **i** such that

$$b \equiv a^i \pmod{p} \qquad \text{where } 0 \leq i \leq (p-1)$$

The exponent **i** is referred to as the **discrete logarithm** of **b** for the base **a**, mod **p**.

Example of the Diffie -Hellman algorithm



Note: A,B,K1,K2 are Private to others

Example of the Diffie -Hellman algorithm

- **7. Bob, now compute the secret key K2 as follows:**

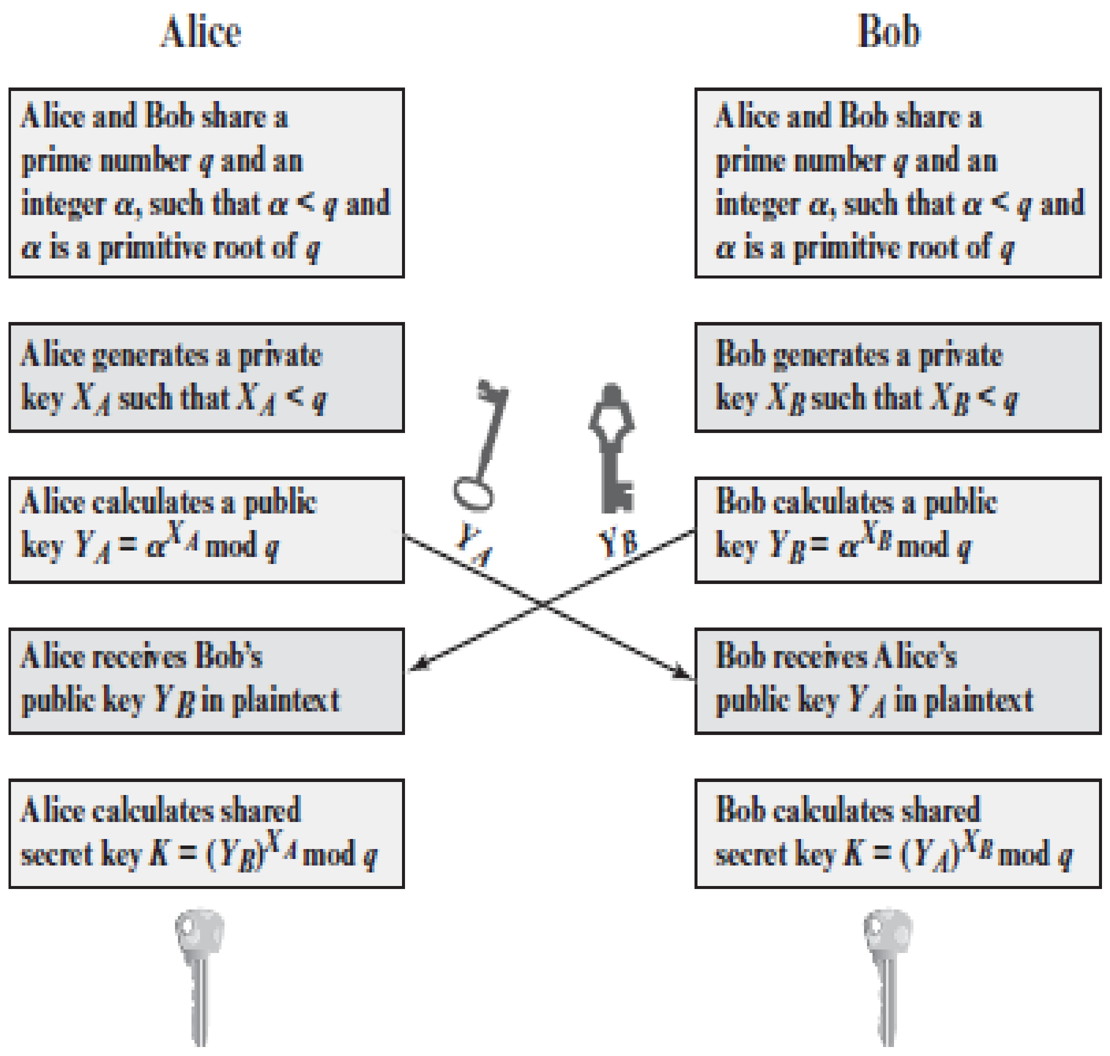
$$\boxed{K2 = Ya^{Xb} \bmod n}$$

- **8. Surprise, K1=K2=K (which is symmetric key)**

Key Exchange Protocols

- Users could create random private/public D-H keys each time they communicate
- Users could create a known private/public D-H key and publish in a directory, then consulted and used to securely communicate with them
- Both of these are vulnerable to a Man-in-the-Middle Attack
- Authentication of the keys is needed

Key Exchange Protocols



Alice
Bob
Alice and Bob share a prime number q and an integer α, such that $\alpha < q$ and α is a primitive root of q
Alice and Bob share a prime number q and an integer α, such that $\alpha < q$ and α is a primitive root of q
Alice generates a private key X_A such that $X_A < q$
Bob generates a private key X_B such that $X_B < q$
Alice calculates a public key $Y_A = \alpha^{X_A} \bmod q$
Bob calculates a public key $Y_B = \alpha^{X_B} \bmod q$
Y_A
Y_B
Alice receives Bob's public key Y_B in plaintext
Bob receives Alice's public key Y_A in plaintext
Alice calculates shared secret key $K = (Y_B)^{X_A} \bmod q$
Bob calculates shared secret key $K = (Y_A)^{X_B} \bmod q$

Man-in-the-Middle Attack

1. Darth prepares by creating two private / public keys
2. Alice transmits her public key to Bob
3. Darth intercepts this and transmits his first public key to Bob. Darth also calculates a shared key with Alice
4. Bob receives the public key and calculates the shared key (with Darth instead of Alice)
5. Bob transmits his public key to Alice
6. Darth intercepts this and transmits his second public key to Alice. Darth calculates a shared key with Bob
7. Alice receives the key and calculates the shared key (with Darth instead of Bob)
8. Darth can then intercept, decrypt, re-encrypt, forward all messages between Alice & Bob

Man-in-the-Middle Attack

- **Can diffie-hellman solve our problem associated with key exchange? Unfortunately, not quite!**
- **This algorithm fall into "man in the middle attack"**
- **This work as follows:**
- **Step-1:**
  - **Alice want to communicate to Bob so as usual they use diffie-hellman key exchange so Let q=11 & α =7. they find K1 and K2 .**
- **Step-2:**
  - **Alice does not realize that the attacker Darth is listening the conversation between her and Bob.**
  - **TOM simply pick up the value of n and α. (q=11, α =7)**
  - **Alice**

Alice	Darth	Bob
q=11, α =7	n=11, α =7	n=11, α =7

- **Step-3:**
  - **Now let us assume that Alice, Darth & Bob select random number Xa & Xb as bellow:**

  - **Alice**

Alice	Darth	Bob
Xa=3	Xa=8,Xb=6	Xb=9

Man-in-the-Middle Attack

- **Step-4:**
  - Now all these person calculate the value of Y_a and Y_b.

$Y_a = \alpha^{X_a} \bmod n$ $= 7^3 \bmod 11 = 2$

$Y_a = \alpha^{X_a} \bmod n$ $= 7^8 \bmod 11 = 9$

$Y_b = \alpha^{X_b} \bmod n$ $= 7^6 \bmod 11 = 4$

$Y_b = \alpha^{X_b} \bmod n$ $= 7^9 \bmod 11 = 8$

- **Step-5:**

Alice DARTH Bob

$Y_a=2 \longrightarrow$ Darth intercept The value of Y_a And send Bob his Own value of Y_a $\longrightarrow Y_a=9$

$Y_b=4 \longleftarrow$ Darth intercept The value of Y_b And send Alice his Own value of Y_b $\longleftarrow Y_b=8$

Man-in-the-Middle Attack

- **Step-6:**
  - So at last value of Ya,Yb to users are:

$\boxed{Y_a=2, Y_b=4^*}$ $\boxed{Y_a=2, Y_b=8}$ $\boxed{Y_a=9^*, Y_b=8}$

*** indicate value of A and B changed.**

- **Step-7:**
  - **Based on these values, all the three person now calculate their Keys.**

Alice	Darth	Bob
$K1=Yb^{Xa} \bmod n$ $=4^3 \bmod 11$ $=9$	$K1=Yb^{Xa} \bmod n$ $=8^8 \bmod 11$ $=5$ $K2=Ya^{Xb} \bmod n$ $=2^6 \bmod 11$ $=9$	$K2=Ya^{Xb} \bmod n$ $=9^9 \bmod 11$ $=5$

Solution: ? **(User authentication required)**

Diffie-Hellman Key Exchange

- The key exchange protocol is vulnerable to such an attack because it does not authenticate the participants. This vulnerability can be overcome with the use of digital signatures and public-key certificates.

Public Key Encryption Algorithms

- Almost all public-key encryption algorithms use either number theory and modular arithmetic, or elliptic curves
- RSA
  - based on the hardness of factoring large numbers
- El Gamal
  - Based on the hardness of solving discrete logarithm
  - Basic idea: public key g^x, private key x, to encrypt: $[g^y, g^{xy} M]$.

ElGamal Cryptography

- public-key cryptosystem related to D-H
- uses exponentiation in a finite field
- with security based difficulty of computing discrete logarithms, as in D-H
- each user (eg. A) generates their key
  - chooses a secret key (number): $1 < x_A < q-1$
  - compute their **public key**: $y_A = a^{x_A} \bmod q$

ElGamal Message Exchange

- Bob encrypts a message to send to A computing
  - represent message M in range $0 <= M <= q-1$
    - longer messages must be sent as blocks
  - chose random integer k with $1 <= k <= q-1$
  - compute one-time key $K = y_A^k \bmod q$

  Encryption: encrypt 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 $K = C_1^{xA} \bmod q$
  - computing M as $M = C_2 K^{-1} \bmod q$
- a unique k must be used each time
  - otherwise result is insecure

ElGamal Example1

- use field GF(19) q=19 and a=10 (primitive root of 19)
- Alice computes her key:
  - A chooses $x_A=5$ (private key) & computes $y_A=10^5 \bmod 19 = 3$ (public key)
- Encryption: Bob send message m=17 as (11,5) by
  - choosing random k=6
  - computing $K = y_A^{\,k} \bmod q = 3^6 \bmod 19 = 7$
  - computing $C_1 = a^k \bmod q = 10^6 \bmod 19 = 11$;

    $C_2 = KM \bmod q = 7.17 \bmod 19 = 5$
- Decryption: Alice recovers original message by computing:
  - recover $K = C_1^{\,xA} \bmod q = 11^5 \bmod 19 = 7$
  - compute inverse $K^{-1} = 7^{-1} = 11$
  - recover $M = C_2\,K^{-1} \bmod q = 5.11 \bmod 19 = 17$

RSA Algorithm

- Invented in **1978** by Ron **R**ivest, Adi **S**hamir and Leonard **A**dleman
  - Published as R L Rivest, A Shamir, L Adleman, "*On Digital Signatures and Public Key Cryptosystems*", Communications of the ACM, vol 21 no 2, pp120-126, Feb 1978
- Security relies on the difficulty of factoring large composite numbers
- Essentially the same algorithm was discovered in 1973 by Clifford Cocks, who works for the British intelligence.

RSA Algorithm

- It is a public key cryptography algorithm, which was proposed by Diffie and Hellman. RSA can be used for key exchange, digital signatures and the encryption of small blocks of data.

➢ RSA is primarily used to encrypt the session key used for secret key encryption (message integrity) or the message's hash value (digital signature).

➢ RSA's mathematical hardness comes from the ease in calculating large numbers and the difficulty in finding the prime factors of those large numbers.

RSA Key

Key generation:

1. Select 2 large prime numbers of about the same size, p and q but p and q has not equal values.

   Typically each p, q has between 512 and 2048 bits

2. Compute $n = pq$, and $\Phi(n) = (q-1)(p-1)$

3. Select e, s.t. $\gcd(e, \Phi(n)) = 1$

   Typically e=3 or e=65537

4. Compute d, s.t. $ed \equiv 1 \mod \Phi(n)$

   Knowing $\Phi(n)$, d easy to compute.

Public key: (e, n)

Private key: (d,n)

RSA: Choosing keys

1. Choose two large prime numbers $p, q.$ (e.g., 1024 bits each)

2. Compute $n = pq$, $\Phi = (p-1)(q-1)$

3. Choose e (with $e<n$) that has no common factors with Φ. (e, Φ are "relatively prime").

4. Choose d such that $ed-1$ is exactly divisible by Φ. (in other words: $ed \bmod \Phi = 1$).

5. *Public* key is $\underbrace{(n,e)}_{K_B^+}$. *Private* key is $\underbrace{(n,d)}_{K_B^-}$.

RSA Algorithm

Encryption

Given a message M, $0 < M < n$ $M \in Z_n - \{0\}$

use public key (e, n)

compute $C = M^e \bmod n$ $C \in Z_n - \{0\}$

Decryption

Given a ciphertext C, use private key (d)

Compute $C^d \bmod n = (M^e \bmod n)^d \bmod n = M^{ed} \bmod n = M$

RSA: Encryption, decryption

0. Given (n,e) and (n,d) as computed above

1. To encrypt bit pattern, m, compute

$$c = m^e \bmod n$$

(i.e., remainder when m^e is divided by n)

2. To decrypt received bit pattern, c, compute

$$m = c^d \bmod n$$

(i.e., remainder when c^d is divided by n)

$$\boxed{m = \underbrace{(m^e \bmod n)}_{c}{}^{d} \bmod n}$$

RSA Algorithm

$$\begin{tikzcd}
\text{Plaintext: } M \arrow[r, bend left, "C = M^e \bmod (n=pq)"] & \text{Ciphertext: } C \arrow[l, bend left, "C^d \bmod n"]
\end{tikzcd}$$

From n, difficult to figure out p,q

From (n,e), difficult to figure d.

From (n,e) and C, difficult to figure out M s.t. $C = M^e$

RSA Example

- P=5 & q=7(prime and p ≠q)
- n=5*7=35 n=p*q and Φ(n)=(4)*(6) = 24 Φ= *(p-1)(q-1)*
- e = 5 1<e< Φ(n), s.t. gcd(e, Φ(n)) = 1
- d = 29 , (29x5 –1) is exactly divisible by 24
- Keys generated are
  - Public key: (35,5) (n,e)
  - Private key is (35, 29) (n,d)
- Encrypt the word lo using ($c = m^e$ mod n)
  - Assume that the alphabets are between 1 & 26

Plain Text	**Numeric Representation**	**m^e**	**Cipher Text ($c = m^e$ mod n)**
l	12	248832	17
o	15	759375	15

- Decrypt the word love using (m = c^d mod n)
  - n = 35, c=29

Cipher Text	c^d	(m = m^e mod n)	Plain Text
17	481968572106750915091411825223072000	17	l
15	12783403948858939111232757568359400	15	o

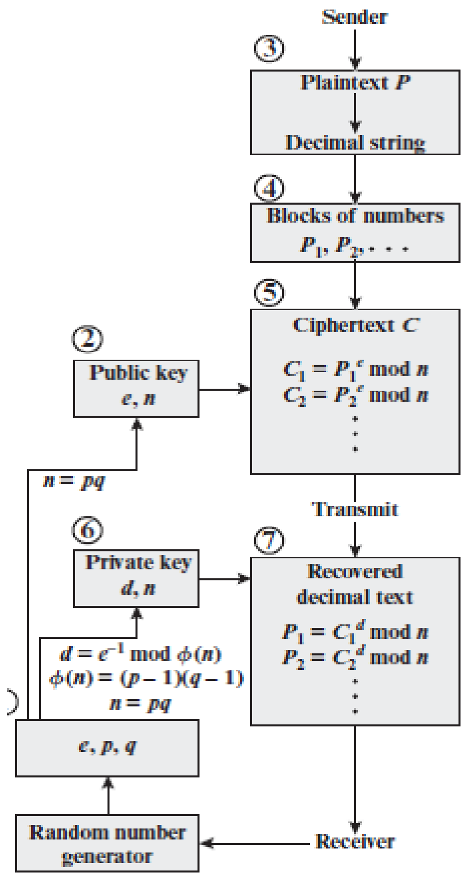
RSA Example

- $p = 11$, $q = 7$, $n = 77$, $\Phi(n) = 60$
- $d = 13$, $e = 37$ ($ed = 481$; $ed \bmod 60 = 1$)
- Let $M = 15$. Then $C \equiv M^e \bmod n$
  - $C \equiv 15^{37} \pmod{77} = 71$
- $M \equiv C^d \bmod n$
  - $M \equiv 71^{13} \pmod{77} = 15$

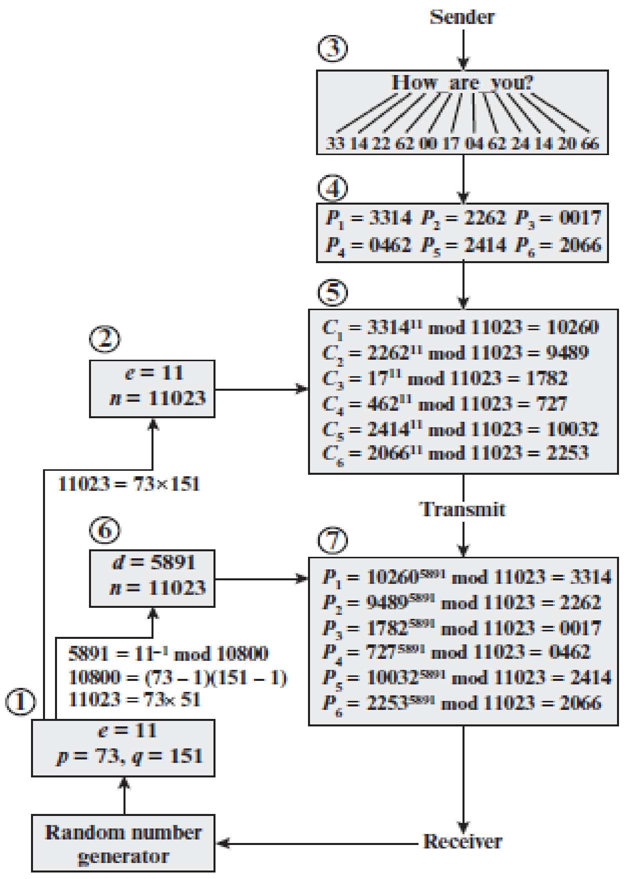
RSA Example 2

- Parameters:
  - $p = 3$, $q = 5$, $n = pq = 15$
  - $\Phi(n) = ?$
- Let $e = 3$, what is d?
- Given $M=2$, what is C?
- How to decrypt?

RSA Example



(a) General approach



Sender
③
How are you?
33 14 22 62 00 17 04 62 24 14 20 66
④
$P_1 = 3314$ $P_2 = 2262$ $P_3 = 0017$
$P_4 = 0462$ $P_5 = 2414$ $P_6 = 2066$
⑤
$C_1 = 3314^{11} \bmod 11023 = 10260$
$C_2 = 2262^{11} \bmod 11023 = 9489$
$C_3 = 17^{11} \bmod 11023 = 1782$
$C_4 = 462^{11} \bmod 11023 = 727$
$C_5 = 2414^{11} \bmod 11023 = 10032$
$C_6 = 2066^{11} \bmod 11023 = 2253$
②
$e = 11$
$n = 11023$
$11023 = 73 \times 151$
Transmit
⑥
$d = 5891$
$n = 11023$
⑦
$P_1 = 10260^{5891} \bmod 11023 = 3314$
$P_2 = 9489^{5891} \bmod 11023 = 2262$
$P_3 = 1782^{5891} \bmod 11023 = 0017$
$P_4 = 727^{5891} \bmod 11023 = 0462$
$P_5 = 10032^{5891} \bmod 11023 = 2414$
$P_6 = 2253^{5891} \bmod 11023 = 2066$
$5891 = 11^{-1} \bmod 10800$
$10800 = (73-1)(151-1)$
$11023 = 73 \times 51$
①
$e = 11$
$p = 73, q = 151$
Random number generator
Receiver

(b) Example

RSA Application

Encryption/decryption: The sender encrypts a message with the recipient’s public key, and the recipient decrypts the message with the recipient’s private key.

Digital signature: The sender “signs” a message with its private key.

Key exchange: Two sides cooperate to exchange a session key.