Introduction to Information Security

—— Symmetric/Secret Key Cryptography & Asymmetric/Public Key Cryptography

Dr. Tianlei HU

Associate Professor

College of Computer Science, Zhejiang Univ.

htl@zju.edu.cn

Outlines

- **Symmetric/Secret Key Cryptography**
  - Model of Symmetric key Cryptography
  - Feistel cipher structure, DES, and other modern cryptography
  - Key Distribution problem and the solution
- **Asymmetric/Public Key Cryptography**
  - Fundamentals and model of Asymmetric/Public Key Cryptography
  - Diffie-Hellman algorithm and attack
  - RSA algorithm

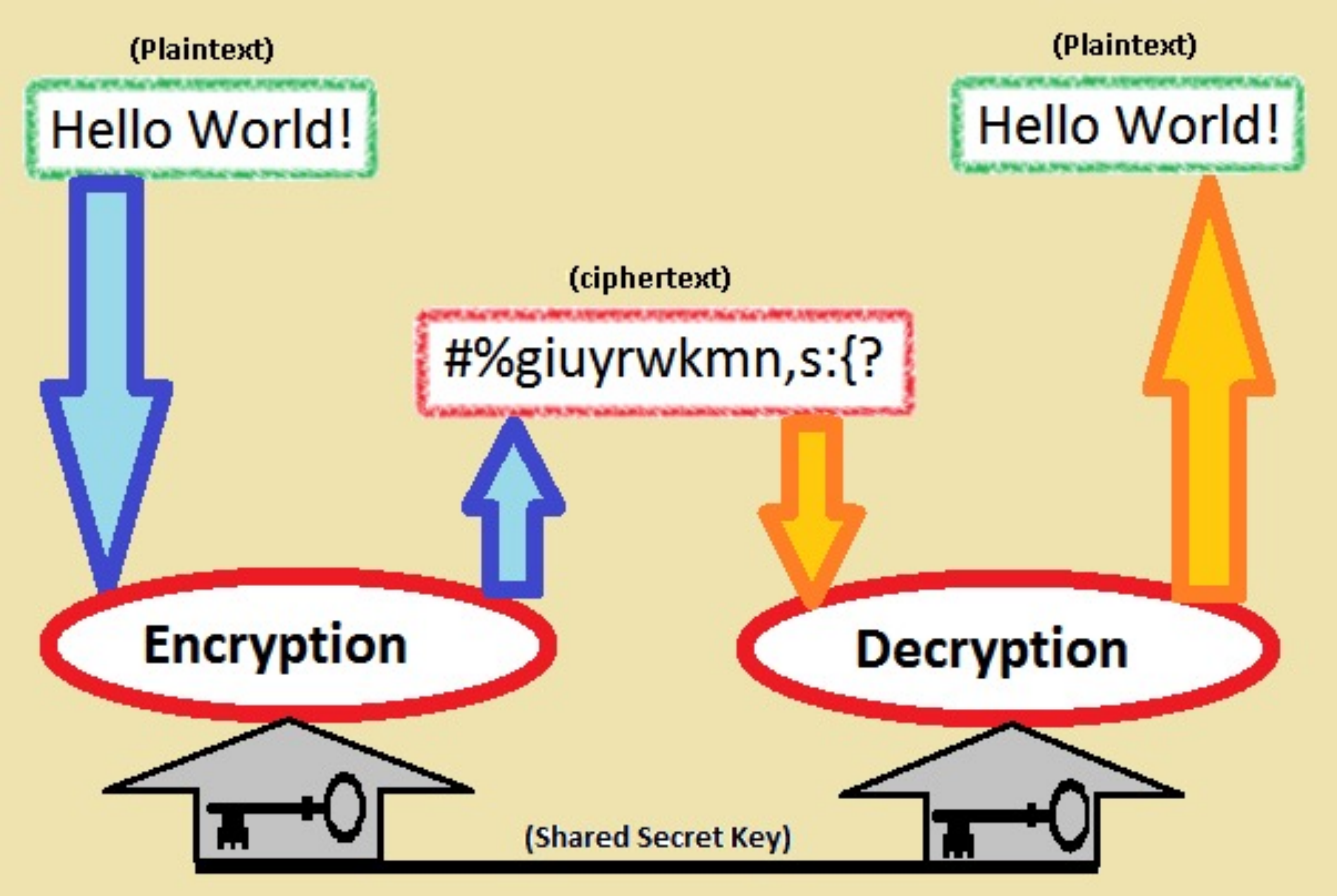
Symmetric/Secret Key Cryptography

Direct impact by computer in the field of cryptography ...

What’s Symmetric Key Cryptography ?

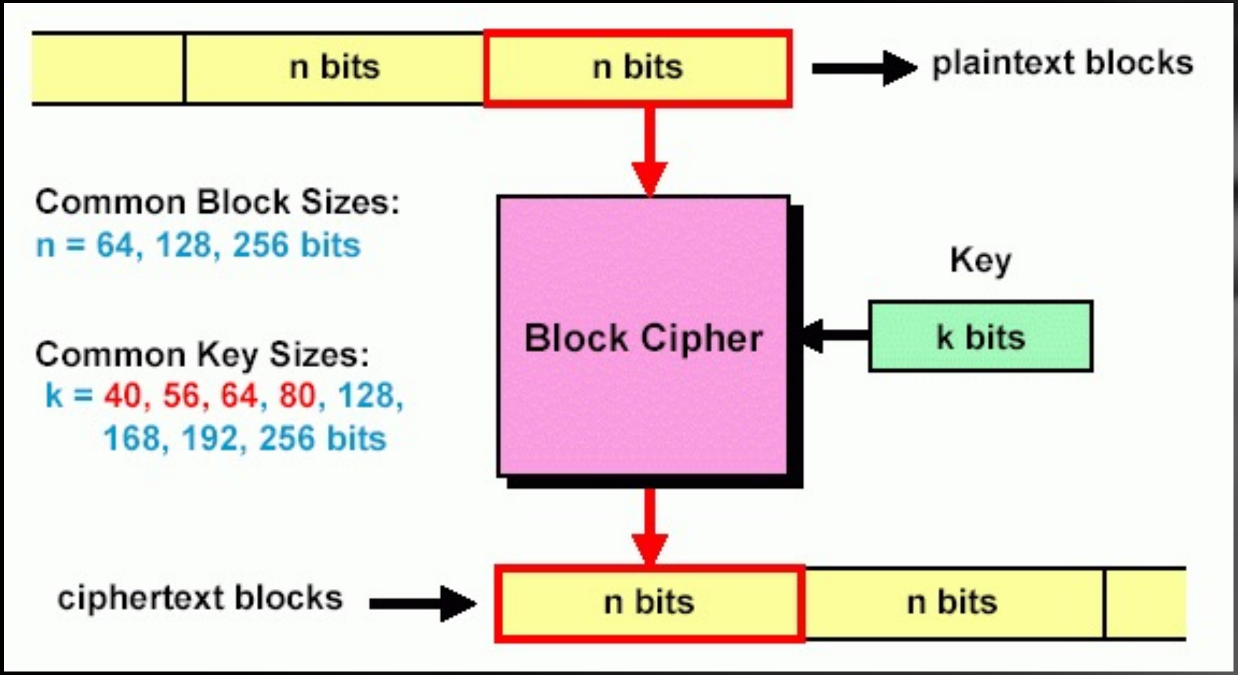
- Symmetric Key Ciphers（对称密钥加密算法）, also called
  - Shared Key Ciphers（共享密钥加密算法）
  - Secure Key Ciphers（保密密钥加密算法）
- Symmetric Key Algorithm is one of the encryption algorithms in Cryptography.
  - Its encryption algorithm is an Antagonistic function, so **decryption algorithm is the same as encryption algorithm**, that is, with the same encryption algorithm, we can get the plaintext.
  - In another words, with the proper key, two encryption can get the original message.(It is called **involution** in mathematic)
  - In application, it means encryption and decryption use the same key, or can calculate the other key easily.
- Complies with the Kerckhoffs’s principle

Model of Secret Key Cryptography



Block Cipher

Divide input bit stream into n-bit sections, encrypt only that section, no dependency/history between sections



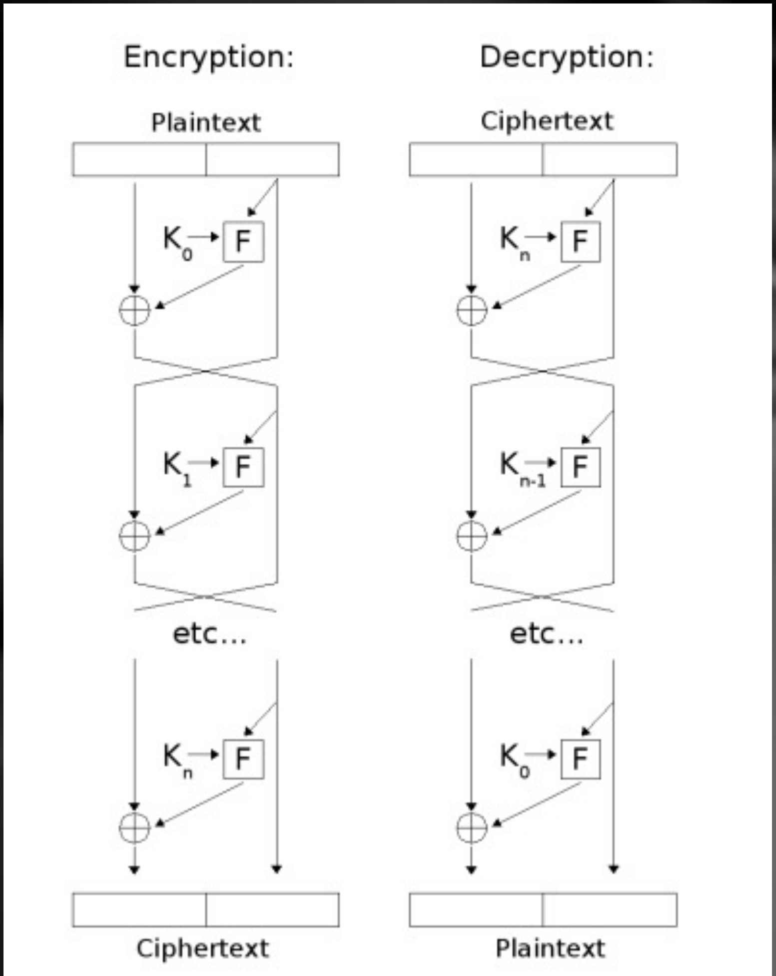
In a good block cipher, each output bit is a function of all n input bits and all k key bits

Feistel Cipher Structure

- Proposed by IBM Feistel in 1973
  - Almost all modern symmetric encryption algorithms are based on this structure
- Use block cipher, and increase block size
  - If adopting the ideal block cipher (Completely random mapping), may cause length of key too long ($n*2^n$, n=64, needs 2^{70}bits length of key
  - Thus, needs an approximation to the ideal block cipher
- Design：
  - Feistel utilized the concept of a product cipher to solve this problem
  - With two approaches to cause avalanche effect：
    - Diffusion 扩散 ——使得密文的统计特性与明文之间的关系尽量复杂
    - Confusion 扰乱 ——使得密文的统计特性与加密密钥之间的关系尽量复杂

Feistel cipher encryption & decryption process

- **Diffusion** - iteratively interchange left-right half
  Confusion - round function F
  - **Block Size**： Larger block size means greater security, typical size is 64bits or 128bits
  - **Key Length**： Larger key size means greater security, typical size is 128bits
  - **Number of rounds:** more number of rounds means greater security, typical size is 16
  - **Sub-key generation algorithm** ： Greater complexity in this algorithm should lead to greater difficult of cryptanalysis
  - **Round function F:** Greater complexity generally means greater resistance of cryptanalysis



Feistel Cipher

DES Algorithm —— Progress

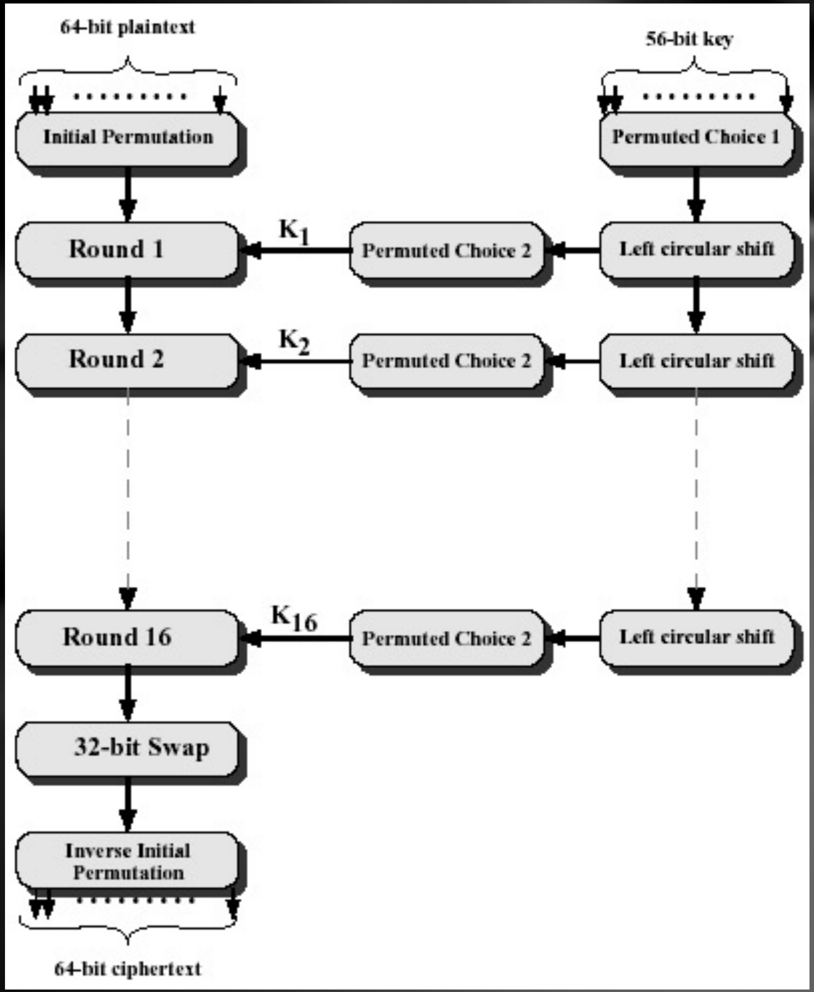
Data Encryption Standard, DES

- Adopted in 1977 by the National Bureau of Standards
- Widespread use at present
- Encrypted in 64-bits blocks
- Using a 56-bits key, based on Feistel structure

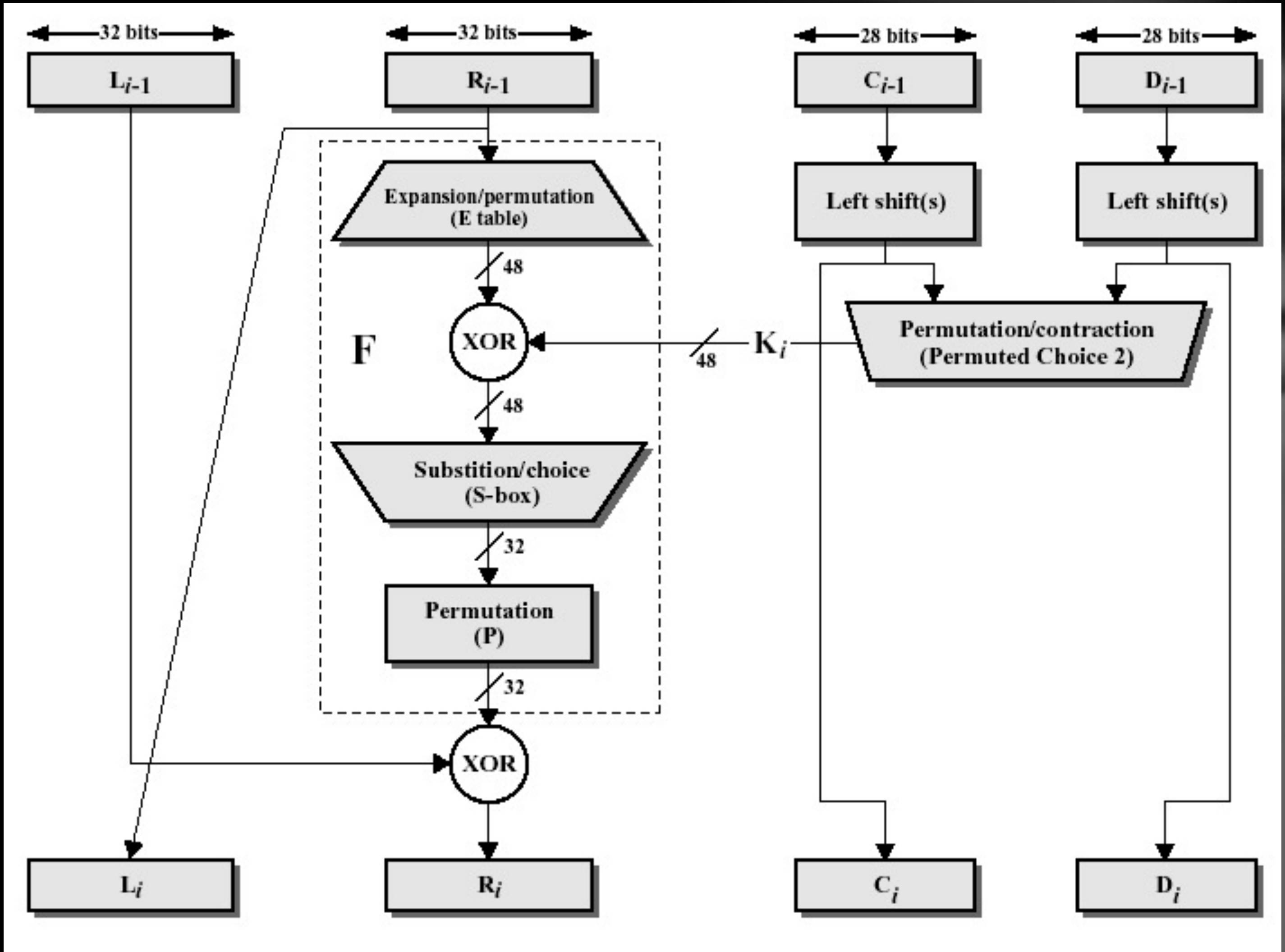
Characteristics of DES：

- Strong avalanche effect
- Has a strong anti-crack strength, only can be attacked with brute-force method
- In Internet age, it is not safe enough by only 56bits-key.

http://en.wikipedia.org/wiki/Data_Encryption_Standard



Single Round of DES Algorithm



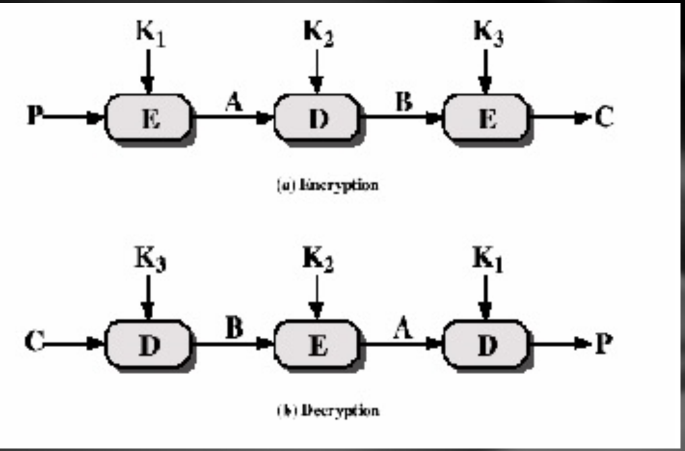
Cracking DES

- DES has excellent anti-crack performance. Secure for about 25 years.
  - estimated 2283 years
- Cracked in 1997
  - Key is only 56bits, 2 exp 56: 72,057,584,037,927,936
  - Computing capability is increasing exponentially
  - Parallel attack – exhaustively search key space
- 1997: Team leaded by Roche Verse using 70000 PCs connected with Internet, 96 days
- 1998: EFF (Electronic Frontier Foundation) using a specially designed machine ($250,000), 3 days
- 1999: Using supercomputer, only 22 hours.

Triple DES

- Utilize encryption－decryption－encryption
- to complete data encryption

$$C = E_{K3}[D_{K2}[E_{K1}[P]]]$$



- C: Ciphertext; P: Plaintext

- Key size is up to 56 x 3＝168
- Utilize K3＝K2 or K1＝K2 to provide backward compatibility for the DES algorithm
- Adopted for Internet applications, e.g., PGP and S/MIME
- http://en.wikipedia.org/wiki/Triple_DES

Other Symmetric Key Cryptography

- **International Data Encryption Algorithm(IDEA)**
  - Designed by Sweden Royal Institute of Technology (KTH), James Massey and Lai Xuejia
  - Based on Feistel cipher structure， 64bits block, 128bits key
  - Adopted by PGP (Pretty Good Privacy)
  - http://en.wikipedia.org/wiki/International_Data_Encryption_Algorithm
- **Blowfish Algorithm**
  - Invented by American cryptologist Bruce Schneier in 1993;
  - Based on Feistel cipher structure, encrypted both two parts of data in each roundS box depends on the key and harder to decipher， Key size from 32bit to 448bit
  - Easy to implementation, fast to encryption, can run blow 5k memory!
  - http://en.wikipedia.org/wiki/Blowfish_(cipher)
- **RC5**
  - Invented by MIT Prof. Ronald L. Rivest in 1994
  - Only use common preliminary computing operations, satisfied for both hardware and software implementation
  - Easy and fast to implementation
  - RC5-w/r/b, w/r/b are all parameters
    - w: word size 16/32/64, satisfied for different CPU
    - r: number of rounds (0 to 255)
    - b: key size (0 to 2040)
  - Cost low memory, great security
  - http://en.wikipedia.org/wiki/RC5

New International Encryption Standard —— AES

- **Advanced Encryption Standard, promulgation of the new US Encryption Standard in 2001**
  - Adopted Rijndael Algorithm proposed by Belgian scientist Joan Daemen and Vincent Rijmen
  - Replace DES and 3DES to overcome the following disadvantages of 3DES:
    - 3DES is slow implemented by software method
    - Block size is only 64bits
- **Characteristics of AES：**
  - Block size: 128bits
  - Key size: 128/192/256 bits
  - Immune to all known attacks
  - Execution fast and code compactness on every platform
  - Simple design

http://en.wikipedia.org/wiki/Advanced_Encryption_Standard

Mode of Operation

A block cipher by itself is only suitable for the secure cryptographic transformation (encryption or decryption) of one fixed-length group of bits called a "block".

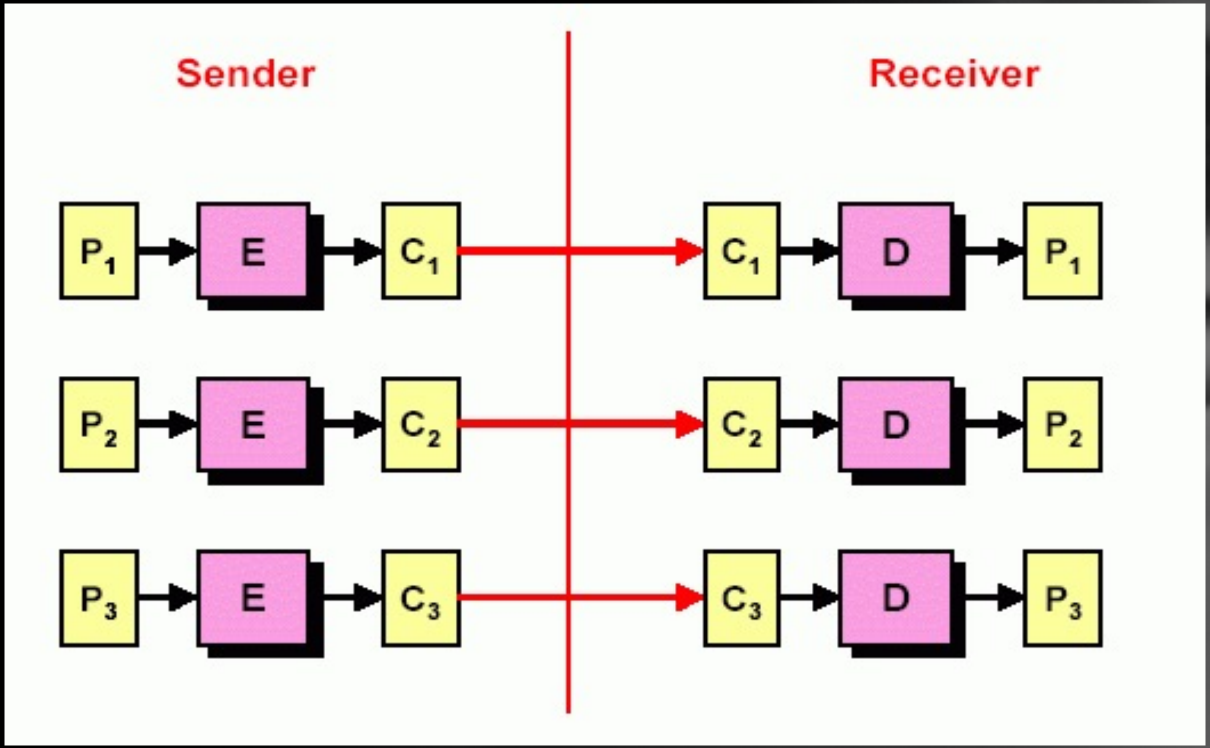
A mode of operation describes how to repeatedly apply a cipher's single-block operation to securely transform amounts of data larger than a block.

- Electronic Codebook (ECB)
- Cipher-block chaining (CBC)
- Propagating cipher-block chaining (PCBC)
- Cipher feedback (CFB)
- Output feedback (OFB)
- Counter (CTR)

We will address ECB and CBC. For more information, please refer to: http://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

ECB

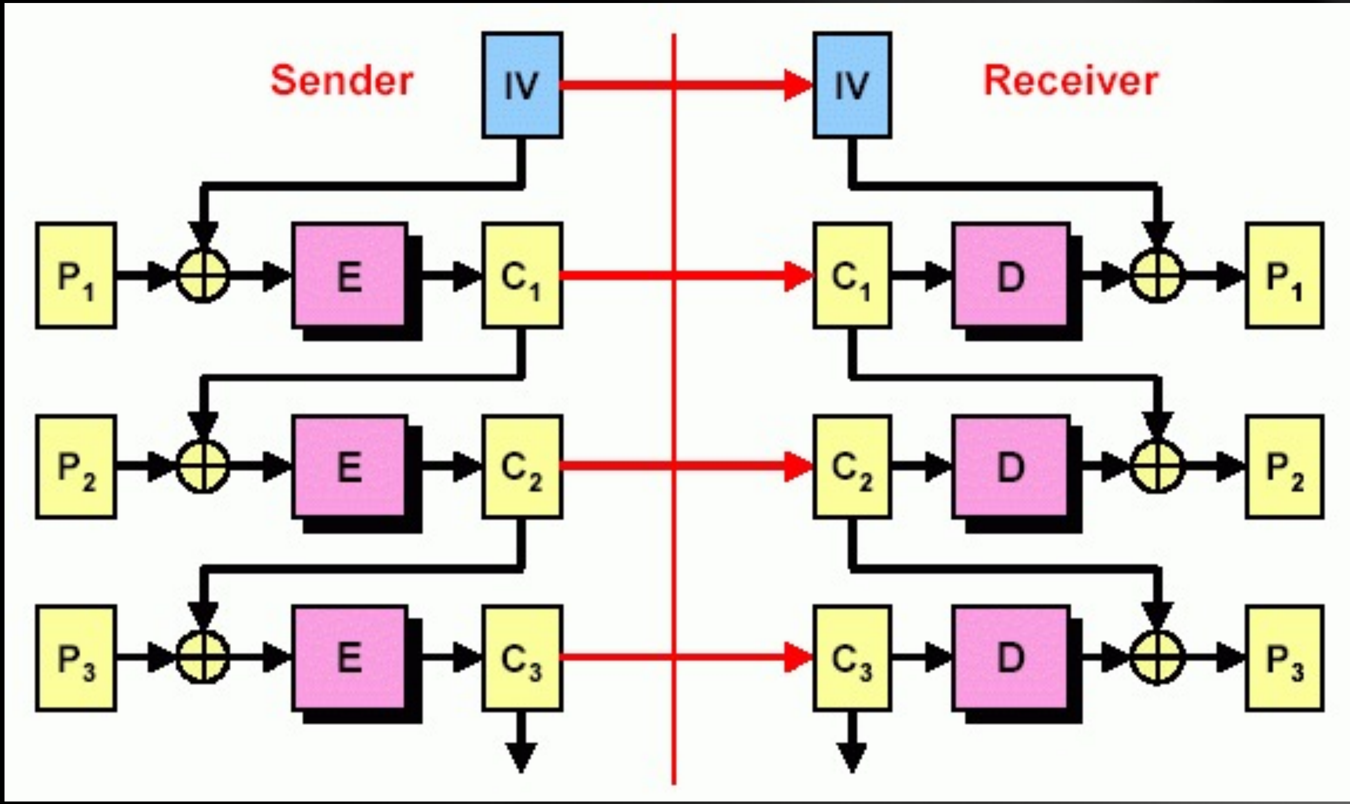
for block ciphers of a long digital sequence



If an attacker thinks block C_2 corresponds to $ amount, then substitute another C_k (ciphertext only attacks)

Attacker can also build a codebook of <C_k, guessed P_k> pairs (chosen plaintext attacks). Replay Attacks?

CBC



Inhibits replay attacks and codebook building: identical input plaintext $P_i = P_k$ won't result in same output code due to memory-based chaining

IV = Initialization Vector – use only once

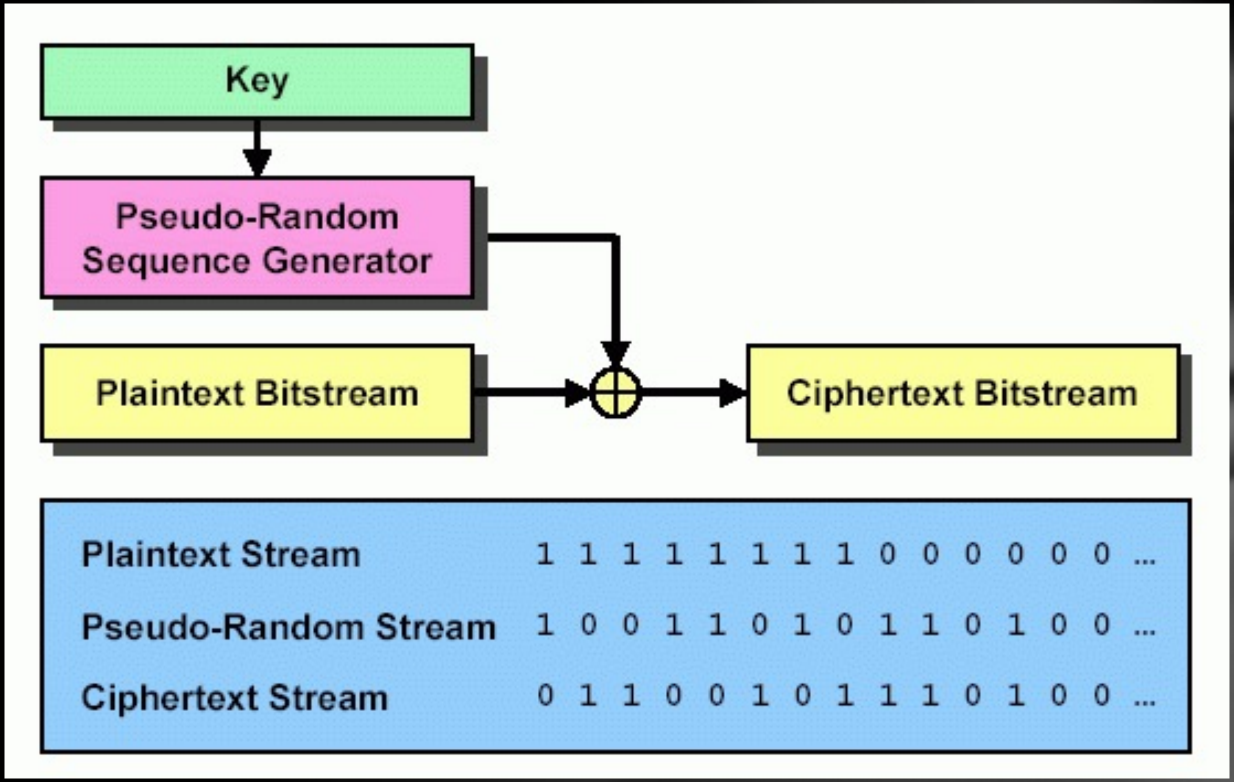
Different Mode of Operations

Original Image

Encrypting using ECB

Encrypting using other modes

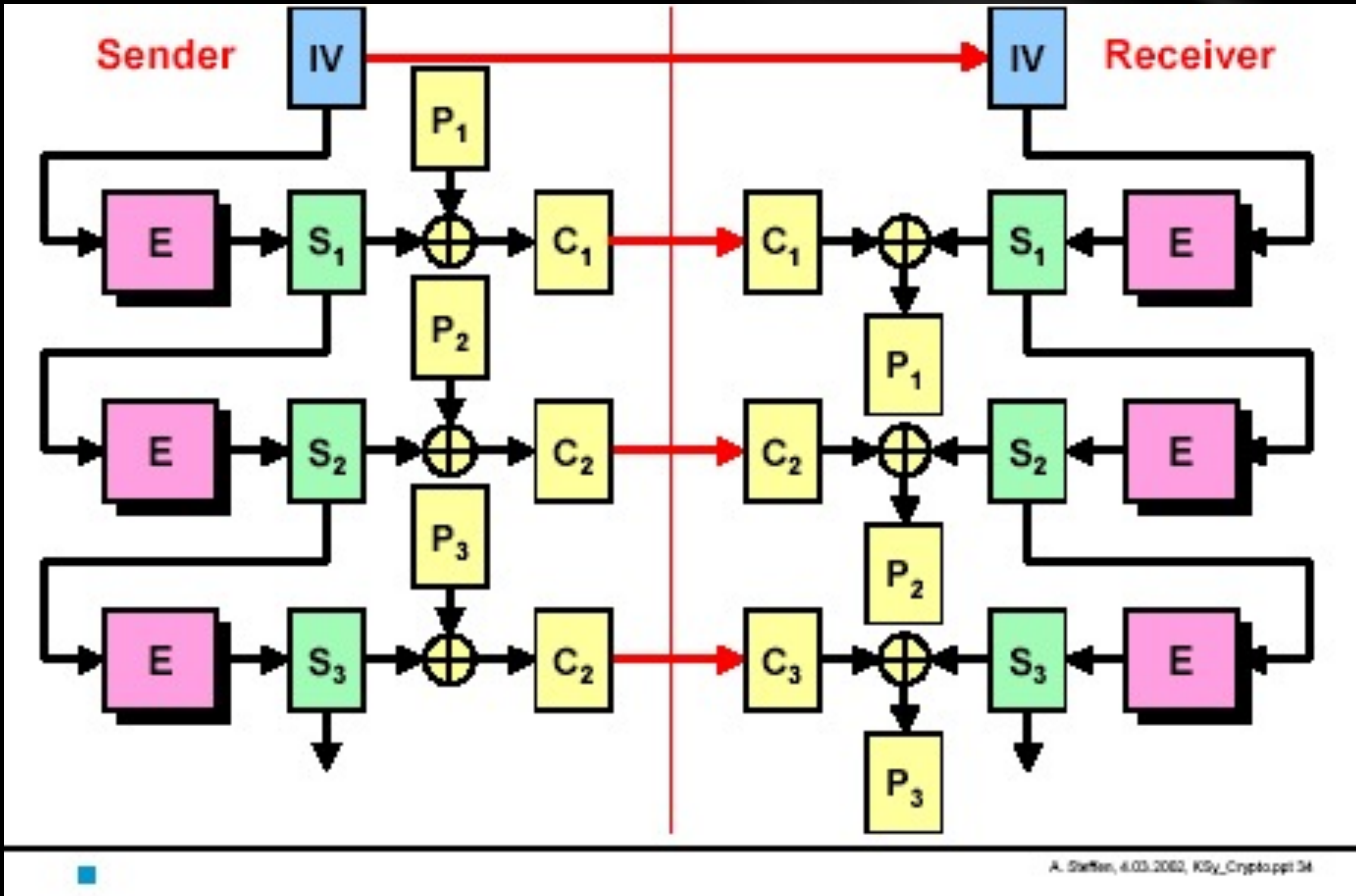
Beyond Block Ciphers, Stream Cipher



Rather than divide bit stream into discrete blocks, as block ciphers do, XOR each bit of your plaintext continuous stream with a bit from a pseudo-random sequence

At receiver, use same symmetric key, XOR again to extract plaintext

Beyond Block Ciphers, Stream Cipher



The Key Distribution Problem

- **According to Kerckhoffs's principle, key is most important!**
  - For symmetric encryption, the key should be shared, and how to share the key?
- **For symmetric encryption, key distribution as follows：**
  - A can select a key and physically deliver it to B
  - A third party can select the key and physically deliver it to A and B
  - If A and B have previously and recently used a key, one party can transmit the new key to the other, encrypted using the old key.
  - If A and B each has an encrypted connection to a third party C, C can deliver a key on the encrypted links to A and B
- Typical solution—— Key Distribution Center(KDC)
- Can this ensure the safety？

Public Key Cryptography

The greatest revolution in the history of cryptography ……

Existing problem of Secret Key Cryptography

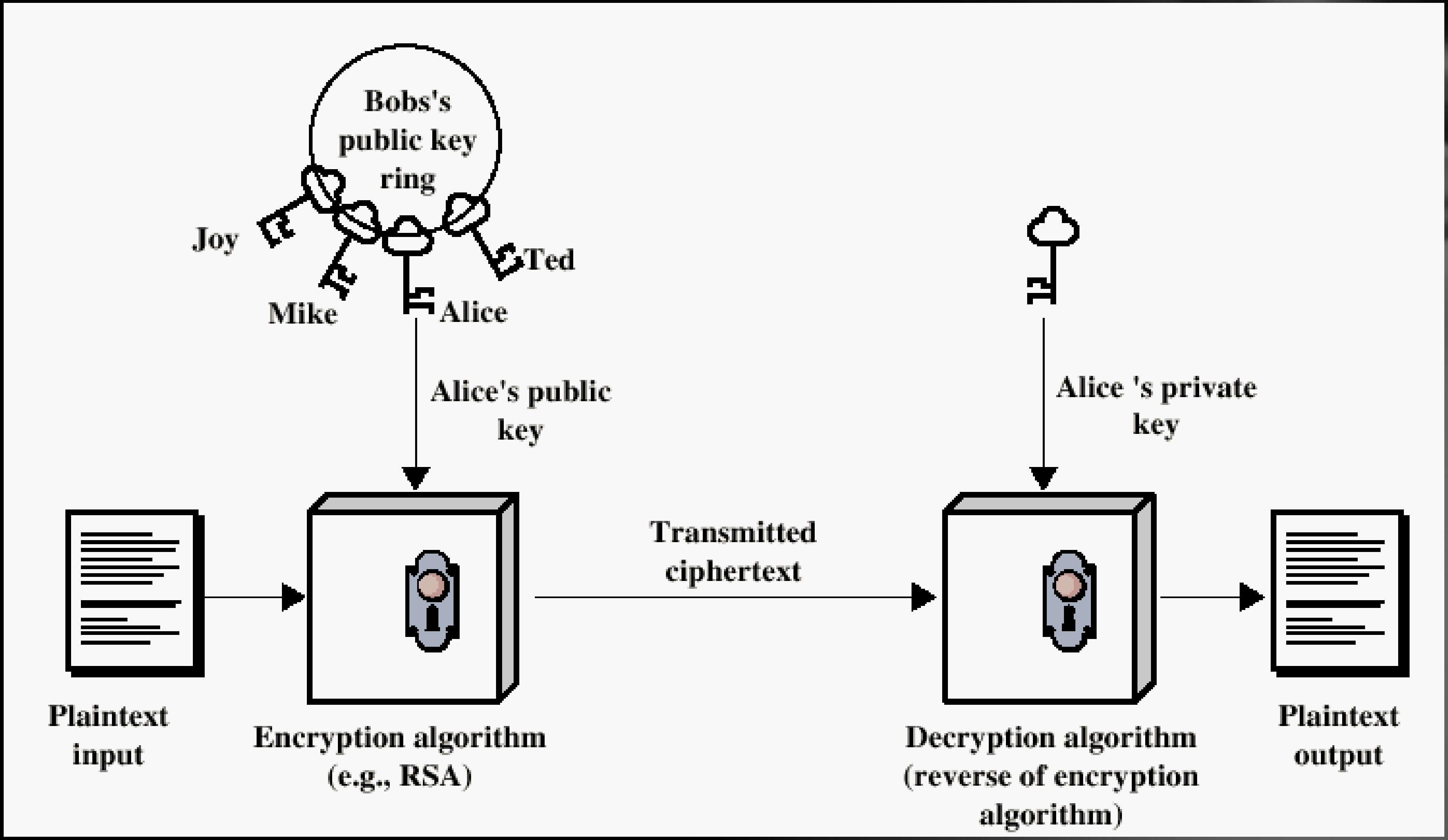
Alice send one of her unpublished paper to Bob by Email:

- Problem 1: Alice do not want others except Bob to read her paper
  - Alice need to encrypt her paper, but how can she tell the password to Bob?
  - If she send the password by email, any one who capture all the emails between Alice and Bob can read the unpublished paper.
- Problem 2: If Bob plagiarize and publish the paper, Alice should be able to prove Bob's plagiarism

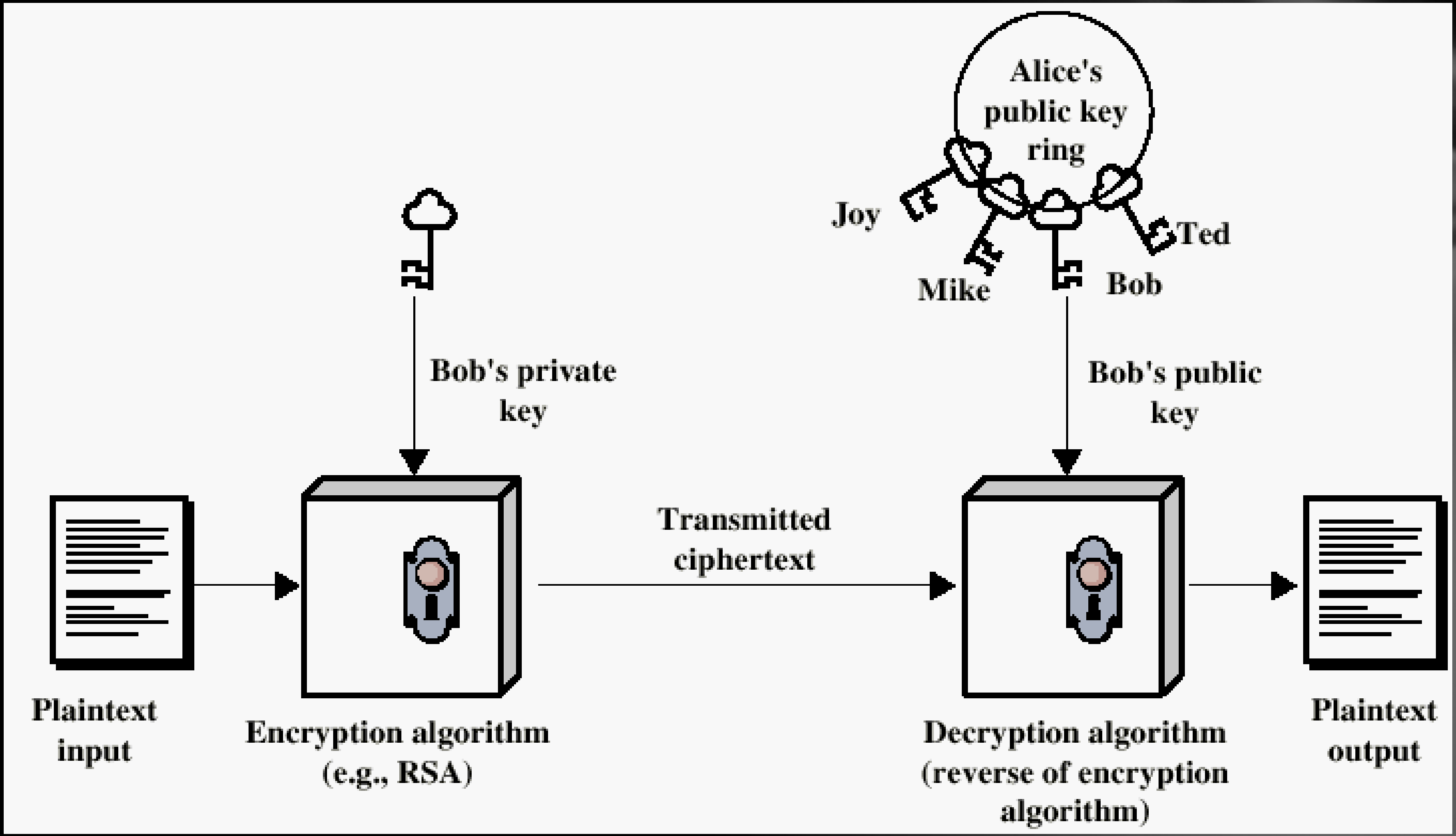
Public Key Cryptography

- Public key cryptography is the greatest revolution in the history of cryptography, It may be said to be the **ONLY** revolution.
  - Public key encryption algorithms rely on mathematical functions, instead of substitution or transposition
  - Public key cryptography is asymmetric, using two separated key, also known as "asymmetric cryptography".
- Public key ciphers enable the exchange of secret information without sharing any secret message between the sender and the receiver.
  - Solve the key distribution problem (independent of KDC), correspond to previous problem 1
- Public key ciphers enable keeping secrecy of the sender and the receiver when they secretly communicate with each other!
  - Solve digital signature problem, correspond to problem 2

Model of Public Key Cryptosystem —— for Secrecy



Model of Public Key Cryptosystem —— for Authentication



Major Differences with Secret Key Ciphers

- The public encryption key is different from the private decryption key.
  - Infeasible for an attacker to find out the private decryption key from the public encryption key.
  - no need for Alice & Bob to distribute a shared secret key beforehand !
  - only one pair of public and private keys is required for each user ! No matter how many communication counterparties
- It is also called ”asymmetric key” algorithm, instead of “symmetric key” algorithm.

Principle of Public Key

- **A public key model consists of six elements：**
  - Plaintext
  - Public key KU
  - Private key KR
  - Encryption Algorithm
  - Ciphertext
  - Decryption Algorithm
- **The key point is to find a one-way function (Calculating the result of the function is easy, but the inverse is infeasible)**
- **Public Key Cryptosystems Used in Three Domains：**
  - Encryption/Decryption: the sender encrypts a message with the recipient’s public key.
  - Digital signature: The sender “signs” a message with its private key.
  - Key exchange: two sides cooperate to exchange a session key.

Requirements of Public Key Cryptography

- It is computationally easy for party B to generate a pair (public key KU_b, private key KR_b).
  - Ensure: key generation is easy!
- It is computationally easy for a sender A to encrypt.
  - Ensure: Encryption is acceptable in time!
- It is computationally easy for the receiver B to decrypt
  - Ensure: Decryption is acceptable in time!
- It is computationally infeasible for an attacker, knowing the public key, KU_b, to determine the private key $KR_{b.}$
- It is computationally infeasible for an attacker, knowing the public key KU_b and a ciphertext C, to recover the original message M .
- Cipher pair can be exchanged.
  - Ensure: can be used either in encryption, or in signature.

Development of Public Key Cryptography

- Diffie & Hellman proposed thought of public key cryptography in "New Directions in Cryptography" for the first time in 1976.
  - IEEE TRANSACTIONS ON INFORMATION THEORY,. 22(6), NOVEMBER. 1976
  - http://www-ee.stanford.edu/~hellman/publications/24.pdf
- Rivest, Shamir & Adleman proposed the RSA algorithm in 1977.
  - "A METHOD FOR OBTAINING DIGITAL SIGNATURES AND PUBLIC-KEY CRYPTOSYSTEMS"
  - COMMUNICATION OF THE ACM, 21 (2): 120–126, 1978
  - http://people.csail.mit.edu/rivest/Rsapaper.pdf
- Other public key cryptography appeared.
  - ElGamal Algorithm (By Taher ElGamal, 1985)
    - "A Public-Key Cryptosystem and a Signature Scheme Based on Discrete Logarithms". IEEE Transactions on Information Theory 31 (4): 469–472, 1985
    - http://caislab.kaist.ac.kr/lecture/2010/spring/cs548/basic/B02.pdf
  - Elliptic Curves Algorithm （ By Neal Koblitz and Victor S. Miller, 1985）
    - "Elliptic curve cryptosystems". Mathematics of Computation 48 (177): 203–209.
    - "Use of elliptic curves in cryptography". CRYPTO 85: 417–426.

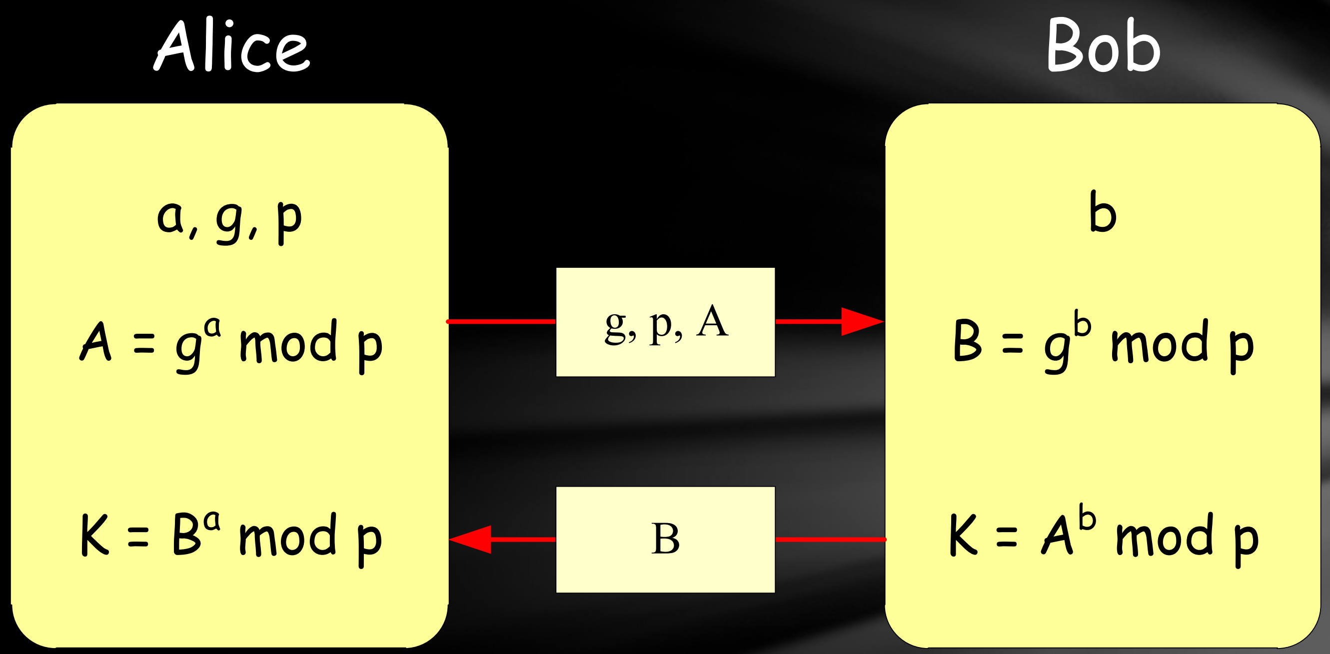
Diffie-Hellman Algorithm

- For a prime number *p* and an integer *g* :
  - g is a primitive root （原根） of p, if g mod p, g^2 mod p, ……, g^{p-1} mod p are different integers, and include all integers from 1 to p-1.
  - For any **integer A** (0 ≤ A ≤ p-1), we can find an only **exponent a** let
    - A = g^a mod p （0≤a ≤p-1， 0≤A ≤p-1）
    - We call exponent a as A's discrete logarithm (离散对数) with base g and mod p
- **Calculating the remainder of the power of an integer dividing a prime is relatively easy, but calculating the discrete logarithm is very hard:**
  - When **p, g** are fixed and p is big enough
    - given **a** to calculate **A** is easy
    - However, given **A** to calculate **a** is difficult

Diffie-Hellman Algorithm

- First, let's prove a mathematical formula :
  - **$g^{ab} \bmod p = (g^a \bmod p)^b \bmod p = (g^b \bmod p)^a \bmod p$**
  - Prove :
    - Let $g^a = n*p + i$， then : $g^a \bmod p = i$
    - $g^{ab} = (n*p + i)^b$ -> $g^{ab} \bmod p = (n*p + i)^b \bmod p = i^b \bmod p$
    - So， $g^{ab} \bmod p = (g^a \bmod p)^b \bmod p$
    - Also， $g^{ab} \bmod p = (g^b \bmod p)^a \bmod p$

Principle of Diffie-Hellman



$$K = A^b \bmod p = (g^a \bmod p)^b \bmod p = g^{ab} \bmod p = (g^b \bmod p)^a \bmod p = B^a \bmod p$$

A's private key： a， B's private key： b

A's public key： A， B's public key： B

Shared message： g, p

Session key： K

Example of Diffie-Hellman

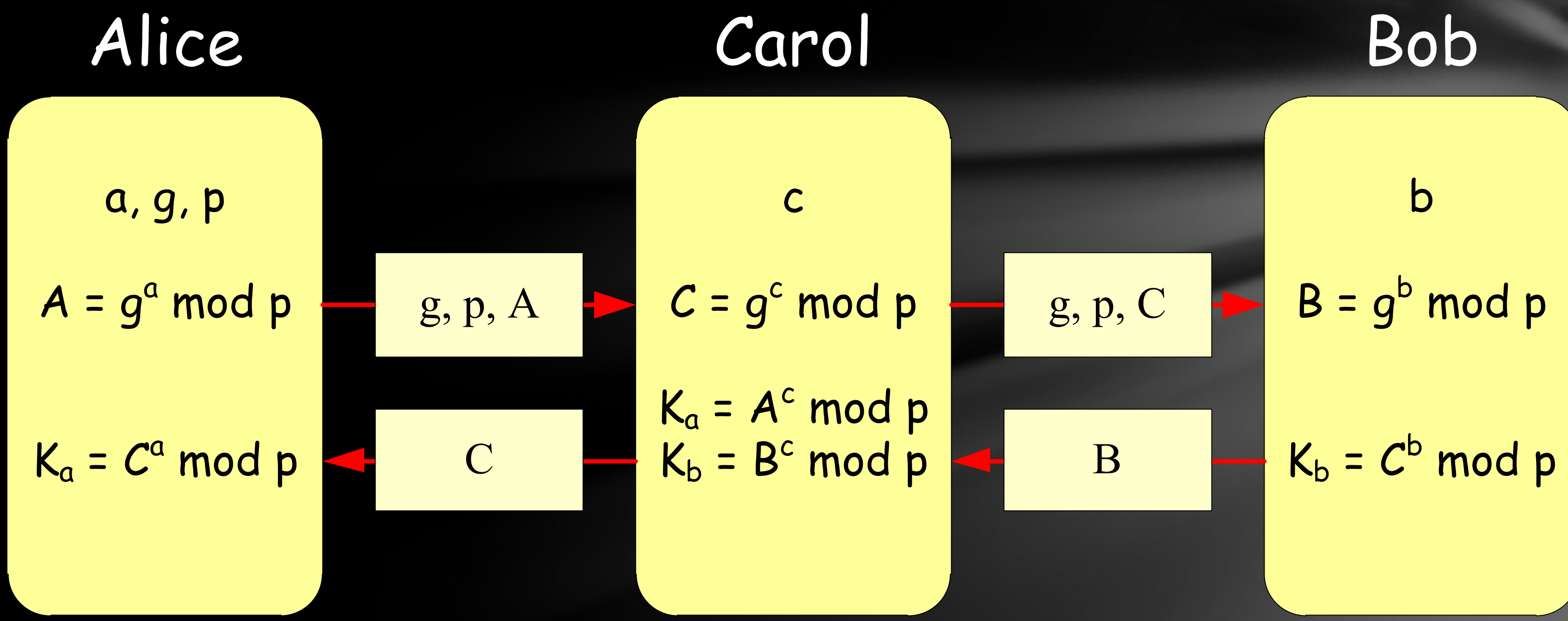
- Choose a prime number p＝353， primitive root g＝3
- Choose a private key a＝97， b＝233
- Computes public key in each：
  - A: $A = 3^{97} \bmod 353 = 40$
  - B: $B = 3^{233} \bmod 353 = 248$
- Computers key of exchanging in each：
  - A： $K = B^{a} \bmod 353 = (248)^{97} \bmod 353 = 160$
  - B： $K = A^{b} \bmod 353 = (40)^{233} \bmod 353 = 160$

Example of Diffie-Hellman

- Attention： Selection of **a, b, p** largely affect the security of the algorithm
  - If p is only a small number, result can be find easily by simple brute-force.
  - Generally, when using DH , p is at least a 300-digit prime number, a and b are at least 100 digits. On this condition, even today, the best algorithms and the best computer can not break this encryption algorithm in significant time.
  - In the algorithm, g need not to be very large. We commonly choose 2, 3, 5 in practice.

Drawback of Diffie-Hellman

- can be used only in key exchange
- Man-in-the-Middle Attack



RSA Algorithm

RSA is proposed by Ron Rivest, Adi Shamir and Leonard Adleman in MIT in 1977

One way function: **large primes multiplication** . Multiplication is easy, but factorization is very difficult

Reference : http://en.wikipedia.org/wiki/RSA_(algorithm)

Mathematical foundation of RSA

Euler's totient function ϕ(n) is defined to be the number of positive integers less than n that are coprime to n

- If n is prime， ϕ(n)=n-1
- If n is composite number, it can be factorized as $n = \prod p_i^{a_i}$, ai>0, p_i is different, then： $\phi(n)= n(1-1/p_1)(1-1/p_2)\ldots(1-1/p_k)$
- For example： 20 = 2*2*5, then:
  - ϕ(20)=20*(1-1/2)*(1-1/5)=8
  - integers from 1-19 which are coprime to 20 are:
    - 1,3,7,9,11,13,17,19， totally 8

- If p and q are coprime, then ϕ(pq)=ϕ(p)ϕ(q)

  In particular, if p≠q, and both are prime, then ϕ(pq)=(p-1)(q-1)

Mathematical foundation of RSA

Euler's theorem (also known as the **Fermat–Euler theorem** or **Euler's totient theorem):** if n and a are coprime positive integers, then : $a^{\phi(n)} \equiv 1 \pmod n$

- Prove:
  - Consider the set of all numbers less than n and coprime to it. Let $\{a_1, a_2, \ldots, a_{\phi(n)}\}$ be this set.
  - Consider a number c<n and coprime to it i.e. $c \in \{a_1, a_2, \ldots, a_{\phi(n)}\}$.
  - First observe that for any a_i, $c*a_i \equiv a_j \pmod n$ for some j. (True since c and a_i are themselves coprime to n, their product has to be coprime to n).
  - And if $c*a_i \equiv c*a_j \pmod n$ then $a_i=a_j$. (True as cancellation can be done since c is coprime to n).
  - **Hence, if we now consider the set $\{c*a_1, c*a_2, \ldots, c*a_{\phi(n)}\}$, this is just a (mod n) permutation of the set $\{a_1, a_2, \ldots, a_{\phi(n)}\}$.**
  - Thereby, we have: $\prod_{k=1..\phi(n)} c*a_k \equiv \prod_{k=1..\phi(n)} a_k \pmod n$
  - Hence, we get: $c^{\phi(n)} * \prod_{k=1..\phi(n)} a_k \equiv \prod_{k=1..\phi(n)} a_k \pmod n$
  - Since $\prod_{k=1..\phi(n)} a_k$ is coprime to n and hence you can cancel them on both sides to get: $c^{\phi(n)} \equiv 1 \pmod n$, whenever c coprime to n.

Mathematical foundation of RSA

- **Fermat Little Theorem**

  - If p is prime, for any integer a： $a^p \equiv a \pmod{p}$
    - Example：
      $5^2 \bmod 2 = 25 \bmod 2 = 1 = 5 \bmod 2$； $5^3 \bmod 3 = 125 \bmod 3 = 2 = 5 \bmod 3$

  - If a is a positive integer not divisible by p, then： $a^{p-1} \equiv 1 \pmod{p}$
    - Prove:
      - It is a special case of the Euler Theorem $a^{\phi(n)} \equiv 1 \pmod{n}$
      - If p is prime, $\phi(p) = p-1$.
      - So $a^{\phi(p)} = a^{p-1} \equiv 1 \pmod{p}$

RSA – Key Generation & Encryption/Decryption

Bob generates key pair, keeps his private key and sends public key to Alice

- Choose two prime p and q (at least 100 digits), Multiplies p and q： $n = p * q$
- Finds out two numbers e & d such that：
  - e and (p-1)(q-1) are co-prime, and $1 < e$ & $d < (p-1)(q-1)$
  - $e * d \equiv 1 \pmod{(p-1)(q-1)}$
- Publish (e, n) as public key on Public key directory, and keep d as private key.

Alice have to encrypt plaintext m （m must smaller than n） to c, and send it to Bob：

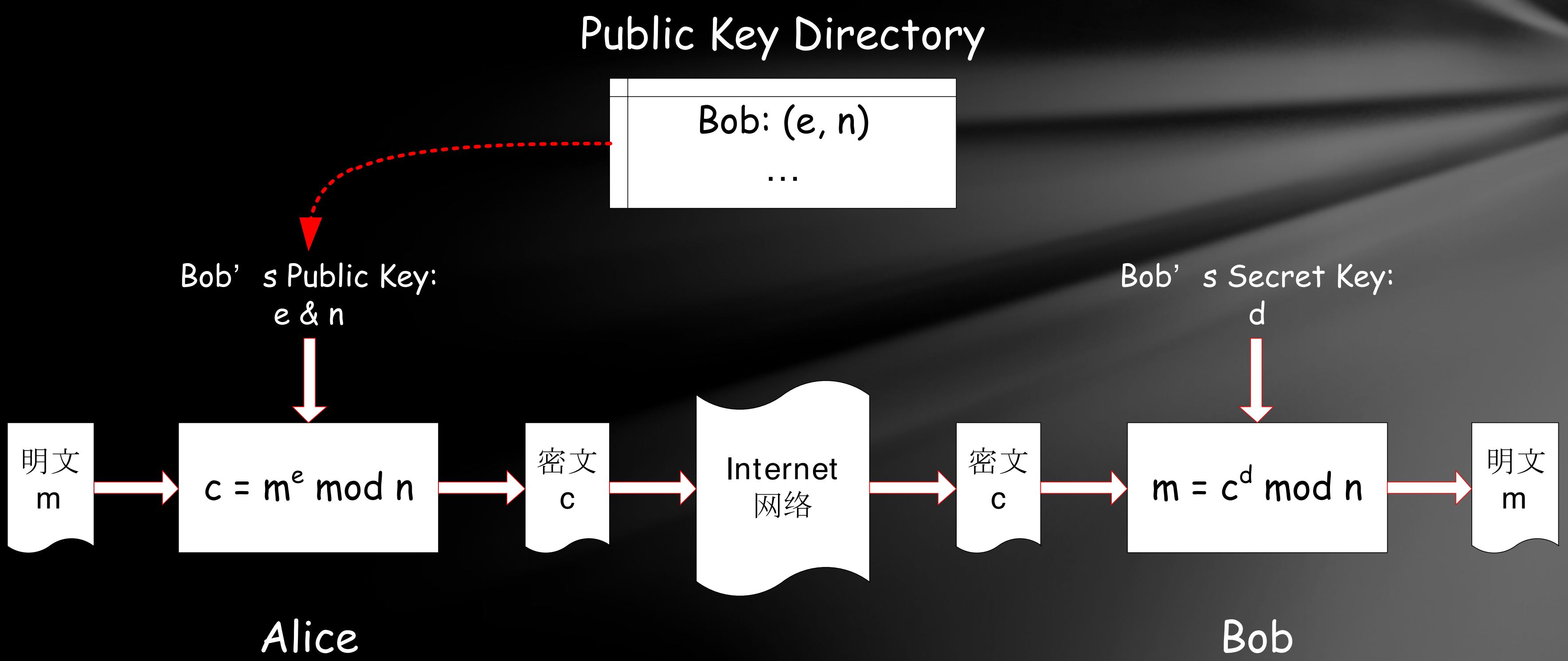
- First find Bob's public key (e, n), and calculate： $c = m^e \bmod n$
- Sends cipher c to Bob

Bob receives cipher c, decrypts and gets plaintext m：

- Use private key d to calculate： $m = c^d \bmod n$

RSA – Key Generation & Encryption/Decryption

Alice want to send message m to Bob:



Principle of RSA —— Why RSA is Correct？

- 1. $c^d = (m^e \bmod n)^d \equiv m^{ed} \pmod n$
- 2. If m & n coprime:
  - From Euler Thereom: $m^{\phi(n)} \equiv 1 \pmod n$
  - $c^d \equiv m^{ed} \pmod n = m^{h\phi(n)+1} \pmod n \equiv m \pmod n$
- 3. Otherwise:
  - From Requirement of key generation：
    - $ed \equiv 1 \pmod{(p-1)(q-1)}$ ->
    - $ed \equiv 1 \pmod{(p-1)}$ and $ed \equiv 1 \pmod{(q-1)}$
  - That is： $ed = k(p-1) + 1$ and $ed = h(q-1) + 1$
- 4. If m is not a multiple of p, then m and p are co-prime,
  - According to Fermat Little Theorem： $m^{p-1} \equiv 1 \pmod p$
  - $m^{ed} = m^{k(p-1)+1} = (m^{p-1})^k m \equiv 1^k m \pmod p = m \pmod p$
- 5. If m is a multiple of p, then： $m^{ed} \bmod p = 0 \equiv m \pmod p$
- 6. Synthesize 4 and 5： $m^{ed} \equiv m \pmod p$, and： $m^{ed} \equiv m \pmod q$
- 7. Since p and q are prime, $m^{ed} - m$ divisible by pq, we have： $m^{ed} \equiv m \pmod{pq}$
- 8. Since n=pq, according to 1, we get： $c^d \equiv m^{ed} \pmod n \equiv m \pmod n$

Requirement of key generation：
- $n = pq$, $\phi(n) = (p-1)(q-1)$
- $ed \equiv 1 \pmod{(p-1)(q-1)}$

Encryption：
- $c = m^e \bmod n$

Decryption：
- $m = c^d \bmod n$

RSA Example (1)

- Bob choose two prime p=5, q=11, then n=p*q=55
  - (p-1)(q-1) = 4*10 = 40
  - Find two numbers: e=3, d=27 and： $3 * 27 \equiv 1 \pmod{40}$
  - So： Bob's public key is : (3, 55), private key is： 27
- Alice sends message m=13 to Bob：
  - Receive Bob's public key(3,55), and calculates： $c = m^e \bmod n = 13^3 \bmod 55 = 2197 \bmod 55 = 52$
  - Send cipher c=52 to Bob。
- Bob receives message c=52：
  - With private key 27, calculate： $m = c^d \bmod n = 52^{27} \bmod 55 = 13$

RSA Example (2)

- Bob choose two prime p=101, q=113, then n=p*q=11413
  - (p-1)(q-1) = 100*112 = 11200
  - Find two numbers: e=3533, d=6597 which： $3533 * 6597 \equiv 1 \pmod{11200}$
  - So： Bob's pubic key: (3533, 11413)， Bob's private key： 6597
- **Alice sends message m=9726 to Bob**：
  - Receive Bob's private key (3533,11413), and calculate： $c = m^e \bmod n = 9726^{3533} \bmod 11413 = 5761$
  - Send ciphertext c=5761 to Bob。
- Bob receives message c=5761：
  - With private key 6597， calculate： $m = c^d \bmod n = 5761^{6597} \bmod 11413 = 9726$

Is RSA Secure？

- **Attack scenarios ：**
  - Marvin wants to get the information m from Alice to Bob, and which are supposed to be seen by Bob only;
  - However, Alice uses RSA with Bob's public key (e,n) and encrypts plaintext m to ciphertext $c = m^e \bmod n$
  - Marvin is a determined attacker and managed to intercept the ciphertext c on its way from Alice's to Bob's computer
  - Marvin also looked up Bob's public key (e, n) to help him in his attack.
  - Now, Begin……
- Marvin now has (c, e, n), and wants to find out m!

Is RSA Secure？

- Marvin now has (c, e, n), and wants to find out m!
  - **Approach 1： If Marvin could also find out Bob's private key d……He knows All~**
    - Suppose Bob guards his private key d very well, what can Marvin do then?
  - **Approach 2： Marvin knows m is a number between 1 and n, so he could search bruteforcely**
    - But if n is large (as mentioned before, p and q are commonly 100 digits primes)
  - **Approach3： Marvin can try to compute Bob's private key d from (e, n), and then use Approach 1.**
    - $ed \equiv 1 \pmod{(p-1)(q-1)}$， Marvin found a very fast algorithm called "extended EUCLID algorithm" in a "Number Theory" book to solve the following problem: given two numbers (r, s), computes x such that
      - $r * x \equiv 1 \pmod{s}$.
    - Once n can be factorized to p and q, then d can be easily found out!
- **Approach 3 is the most efficient known method to attack RSA!**

Is RSA Secure？ —— factorization problem

The time taken for Marvin to attack in Approach 3 is essentially the time to factorize：

- Therefore, we say that RSA is based on the factorization problem: While it is easy to multiply large primes together, **it is computationally infeasible to factorize or split a large composite into its prime factors!**
- It is the “one-way function” of RSA

Is RSA Secure？ —— factorization problem

Research of prime factorization algorithm：

- the largest factored RSA number is a 250-digit prime, RSA-250, in 2020.02
  - RSA-250 (250 digits, 829 bits) =
  - 2140324650240744961264423072839333563008614715144755017797754920881418023447
    140136643345519095804679610992851872470914587687396261921557363047454770520805119056493106687691590019759405693457452230589325976697471681738069364894699871578494975937497937
    =
    64135289477071580278790190170577389084825014742943447208116859632024532344630238623598752668347708737661925585694639798853367 × 3337202759497815655622601060535511422794076034476755466678452098702384172921
    0037080257448673296881877565718986258036932062711

- The factorization of RSA-250 utilized approximately 2700 CPU core-years, using a 2.1Ghz Intel Xeon Gold 6130 CPU as a reference.

Is RSA Secure？ —— factorization problem

Research of prime factorization algorithm：

- RSA-1024 (309 digits, 1024 bits) =
  135066410865995223349603216278805969938881475605667027524485143851526510604859533833940287150571909441798207282164471551373680419703964191743046496589274256239341020864383202110372958725762358509643110564073501508187510676594629205563685529475213500852879416377328533906109750544334999811150056977236890927563
- Successful factorization of RSA-1024 will have important security implications for many users of the RSA algorithm, NIST has "deprecated" 1024 bit RSA since 2011, with the recommendation switching to "disallowed" starting 2014.
- However, RSA-2048, the largest RSA number, may not be factorizable for many years to come, unless considerable advances are made in integer factorization or computational power in the near future.
- RSA-2048 (617 digits, 2048 bits) =
  25195908475657893494027183240048398571429282126204032027777137836043662020707595556264018525880784406918290641249515082189298559149176184502808489120072844992687392807287776735971418347270261896375014971824691165077613379859095700097330459748808428401797429100642458691817195118746121515172654632282216869987549182422433637259085141865462043576798423387184774447920739934236584823824281198163815010674810451660377306056201619676256133844143603833904414952634432190114657544454178424020924616515723350778707749817125772467962926386356373289912154831438167899885040445364023527381951378636564391212010397122822120720357

Is RSA Secure？

- Marvin never gives up! He cannot find out the key passively, but he can use active attack! (active attack vs. passive attack ）
- Approach 4 ：
  - Marvin generates a RSA key pair of his own
    - Public key： Kpub_* = (n_*, e_*)， Private key： Ksec_* = d_*
    - Marvin sends a mail to Alice in the name "Bob"：
    - Dear Alice ，
    - Please send mail to me with my new public key Kpub_*
    - Yours sincerely, Bob
    - Alice follows, sending mail to Bob encrypted by Kpub_*, Marvin can decrypt by Ksec_*!

Is RSA Secure?

Why Approach 4 can be successful?

- Naïve Alice is cheated by wicked Marvin!
- Alice uses fake Bob's public key (in fact it's Marvin's)!

How to counter attack?

- Before Alice sending ciphertext to Bob, she must make sure that Bob's public key is correct.
- Alice needs to verify correctness of all the information which inform Bob's key.
- Except Bob, no one can create such a message which can pass verification of Alice!
- This leads Message Integrity problem:

Alice and Bob need to avoid "Bob's key" being forged or distorted by attackers.

The tool of cryptography to solve this problem is "Digital Signatures"

Symmetric vs. Asymmetric ciphers

- **Symmetric ciphers:**
  - Good：cheap and fast；Low cost VLSI chips available.
  - Bad：key distribution is a problem！
- **Asymmetric ciphers:**
  - Good：Key distribution is NOT a problem!
  - Bad：Relatively expensive and slow; VLSI chips not available or relatively high cost.
- **In practice:**
  - Use a public key cipher (such as RSA) to distribute key
  - Use a private key cipher (such as AES) to encrypt and decrypt messages
- 另外，需要澄清的两个常见误解：
  - **公开密钥加密在防范密码攻击上比常规加密更安全。**
    - 实际上，两者都依赖于密钥长度和解密的计算工作量，从抗密码分析的角度分析，互相之间都不比对方优越
  - **公开密钥加密使得常规加密过时。**
    - 实际上，公开密钥加密在计算上相对的巨大开销，使得公开密钥加密更多地用于密钥管理和数字签名应用

Review

- **Symmetric/Secret Key Cryptography**
  - Fundamentals and model
  - Feistel cipher structure and DES cryptography
  - Existing problems of Symmetric/Secret Key Cryptography
- **Asymmetric/Public Key Cryptography**
  - Fundamentals and model
  - DH Algorithm： Methods and issues
  - RSA Algorithm： Methods
- **Compare and combination of Symmetric and Asymmetric cryptography**