澳門理工學院
Instituto Politécnico de Macau
Macao Polytechnic Institute

COMP412: Computer Security

Public Key Cryptography

Dr. Kim, Song-Kyoo (Amang)
Associate Professor,

Computer Science Program
MACAO POLYTECHNIC INSTITUTE
Macau, SAR

Agenda

- Modulo arithmetic
- Fermat’s and Euler’s theorems
- Primality test
- Public key theorem
- RSA and its security
- Diffie-Hellmen key exchange

澳門理工學院
Instituto Politécnico de Macau
Macao Polytechnic Institute

Number Theory

An Introduction to Number Theory

- Overview
  - You need to be able to do modulo arithmetic computations to understand public key crypto algorithms
  - You need to know how to compute
    - Exponentiation
    - Greatest common divisor
    - Inverses

Modulo Arithmetic

- Divisors
  - b != 0 divides number a if for some m have m = a / b (a, b, m all integers) that is b divides a with no remainder
  - “b is a divisor of a” Eg. All of 1,2,3,4,6,8,12,24 divide 24
- To factor a number n, it’s written as a product of other numbers, eg n = a × b × c ; 36 = 22 × 32
- Note that factoring a number is relatively hard compared to multiplying the factors together to generate the number
- The prime factorization of a number n is when it’s written as a product of prime numbers
  - Eg. 91 = 7 × 13; 3600 = 24 × 32 × 52

Modulo Arithmetic

- Modulo arithmetic is 'clock arithmetic'
- If any number mod 7 and we have a finite number of values (eg 0..6) and loop back from either end
- a = b mod n is congruence When divided by n that a and b have the same remainder
  (eg. 100 mod 11= 34 mod 11 = 1)
- Arithmetic with integers modulo *n* with all results between 0 and *n* – 1
- Using modulo arithmetic is how we keep the size of problems fixed, since only compute with a fixed number of numbers.

Modulo Arithmetic

...						
-14	-13	-12	-11	-10	-9	-8
-7	-6	-5	-4	-3	-2	-1
0	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
...						

- Modulo Arithmetic Example
  - Eg. when working modulo 7
    - -12 = -2 x 7 + 2
    - -5 = -1 x 7 + 2
    - 2 = 0 x 7 + 2
    - 9 = 1 x 7 + 2
- All numbers in a column are equivalent (have same remainder)

Modulo Arithmetic

- Addition: a + b mod n
- Subtraction: a – b mod n
- Multiplication: a × b mod n
  - Derived from repeated addition
  - Can get a × b = 0 where neither a, b = 0
    - Eg, 2 × 5 = 0 mod 10
- Division: a / b mod n = a × b^{-1} mod n
  - If b^{-1} is relatively prime to n, b × b^{-1} = 1 mod n
    - Eg, 2 × 3 = 1 mod 5, hence 4 / 2 = 4 × 3 = 2 mod 5
- If a + b = 0 mod n, b is additive inverse of a
- If a × b = 1 mod n, b is multiplicative inverse of a

Modulo Arithmetic

- Addition: x + y = 0 mod 7
- Multiplication: x × y = 1 mod 7
(y is the inverse of x)

modulo 7

+	0	1	2	3	4	5	6
0	0	1	2	3	4	5	6
1	1	**2**	**3**	**4**	**5**	**6**	**0**
2	2	**3**	**4**	**5**	**6**	**0**	**1**
3	3	**4**	**5**	**6**	**0**	**1**	**2**
4	4	**5**	**6**	**0**	**1**	**2**	**3**
5	5	**6**	**0**	**1**	**2**	**3**	**4**
6	6	**0**	**1**	**2**	**3**	**4**	**5**

×	0	1	2	3	4	5	6
0	0	0	0	0	0	0	0
1	0	1	2	3	4	5	6
2	0	2	**4**	**6**	**1**	**3**	**5**
3	0	3	**6**	**2**	**5**	**1**	**4**
4	0	4	**1**	**5**	**2**	**6**	**3**
5	0	5	**3**	**1**	**6**	**4**	**2**
6	0	6	**5**	**4**	**3**	**2**	**1**

Relatively Prime Numbers

- Two numbers a, b are relatively prime if they have no common divisors apart from 1

- Two numbers a, b are relatively prime if gcd(a, b) = 1

- Eg. 8 & 15 are relatively prime since factors of 8 are 1, 2, 4, 8 and of 15 are 1, 3, 5, 15 and 1 is the only common factor

- Two numbers may be relatively prime if one of them is prime and one is not a multiple of the other

Greatest Common Divisor

- A common problem in number theory is to determine the Greatest Common Divisor (GCD)
- The GCD(a, b) of a and b is the largest number that divides evenly into both a and b
- Often want this to prove that there are no common factors (except 1) and that the numbers are relatively prime

Modulo Inverses

- Another common problem is to find the *inverse* of a number
- Unlike normal integer arithmetic, sometimes an integer in modular arithmetic has a unique inverse
- a^{-1} is inverse of a mod n if $(a \times a^{-1})$ mod n = 1, where a, a^{-1} in {0, n – 1}
  - Eg., (3 × 7) mod 10 = 1
- If GCD(a, n) = 1 then the inverse always exists

Modulo Inverses

- If GCD(d, f) = 1, then d has a multiplicative inverse module f. That is, for positive integer d < f, there exists a $d^{-1} < f$ such that d × d^{-1} = 1 mod f.
- Extended Euclid's algorithm

```
  x1 = 1, x2 = 0, x3 = f
  y1 = 0, y2 = 1, y3 = d
➔ if y3 = 0 return x3 = GCD(d, f) (no inverse)
  if y3 = 1 return y3 = GCD(d, f); y2 = d-1 mod f
  Q = x3 / y3
  t1 = x1 – Q × y1;
  t2 = x2 – Q × y2;
  t3 = x3 – Q × y3
  x1 = y1; x2 = y2; x3 = y3
  y1 = t1; y2 = t2; y3 = t3
```

back to ➔

Modulo Inverses

- Example:
  - As we evaluate a / b mod n, we do $a \times b^{-1}$ mod n
  - Finding the inverse of b is needed.
- Let’s evaluate 3 / 8 mod 11.
  - We then do it as 3×8^{-1} mod 11
  - We got to find the inverse of 8 mod 11
- y3 = GCD(8, 11) = 1 and we stop here
- y2 = 8^{-1} = -4 = 7 as we don’t take negative

Q	x1	x2	x3	y1	y2	y3
	1	0	11	0	1	8
1	0	1	8	1	-1	3
2	1	-1	3	-2	3	2
1	-2	3	2	3	-4	1

Modulo Inverses

- Example: Find inverse of 540 mod 1769
- y2 = 540^{-1} = -95 = (1760-95) = 1674
- y3 = GCD(540, 1769) = 1

Q	x1	x2	x3	y1	y2	y3
	1	0	1769	0	1	540
3	0	1	540	1	-3	149
3	1	-3	149	-3	10	93
1	-3	10	93	4	-13	56
1	4	-13	56	-7	23	37
1	-7	23	37	11	-36	19
1	11	-36	19	-18	59	18
1	-18	59	18	29	-95	1

Fermat’s Little Theorem

- Fermat’s little theorem and Euler’s theorem play an important role in public-key cryptography
- Fermat’s theorem states that if p is prime and a is a positive integer not divisible by p, then

$$a^{p-1} \equiv 1 \bmod p$$

Euler's Theorem

- Euler's theorem holds for every a and n that are relatively prime
- RSA public key algorithm employs this theorem

$$a^{\phi(n)} \equiv 1 \bmod n$$

$$a^{\phi(n)+1} \equiv a \bmod n$$

a = 3; n = 10; ø(10) = 4; 3^4 = 81 = 1 mod 10
a = 2; n = 11; ø(11) = 10; 2^{10} = 1024 = 1 mod 11

Euler Totient Function Ø(n)

- Euler’s totient function, written Ø(n), where Ø(n) is the number of positive integers less than n and relatively prime to n

$$\varphi(n) = n \prod_{p|n} \left(1 - \frac{1}{p}\right)$$

- Euler’s totient is essential to the RSA

- Determine Ø(35) and Ø(37)
  - Ø(37) = 37 · 1 = 36 since 37 is prime
  - Ø(35) = all positive integers less than 35 and relatively prime to 35: 1, 2, 3, 4, 6, 8, 9, 11, 12, 13, 16, 17, 18, 19, 22, 23, 24, 26, 27, 29, 31, 32, 33, 34

Euler Totient Function Ø(n)

- Now we have two prime numbers, p and q, and
- n = p × q then
- Ø(n) = Ø(p × q) = Ø(p) × Ø(q) = (p – 1) × (q – 1)
- Example:
- Ø(21) = Ø(3) × Ø(7) = (3 · 1) × (7 · 1) = 12
- Ø(21) = 12

  = Ø(3) × Ø(7)

  = 2 × 6

  = (3 – 1) × (7 – 1)

  where the 12 integers are {1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20}

Prime numbers

- A number is prime if its only divisors are 1 and itself that is it cannot be written as a product of other numbers
- 1 is prime, but is generally not of interest. 2,3,5,7 are also prime
- The list of prime number less than 200 is: 2 3 5 7 11 13 17 19 23 29 31 37 41 43 47 53 59 61 67 71 73 79 83 89 97 101 103 107 109 113 127 131 137 139 149 151 157 163 167 173 179 181 191 193 197 199

Primality Testing

- Classic way of finding primes is to sieve for them by trial division, ie. Divide by all numbers (primes) in turn less than the square of the number, but only works for small numbers

- Hence because of the size of numbers used, must find primes by trial and error

- These primality tests use properties of primes:
  - $a^{n-1} = 1$ mod n where GCD(a, n)=1 (Fermat theorem)
  - All prime numbers 'n' will satisfy this equation
  - Some composite numbers will also satisfy the equation
  - Such composites are called pseudo-primes

- These tests:
  - Guess at a prime number '*n*'
  - Then take a large number (eg 100) of numbers '*a*'
  - Apply this test to each
  - If it fails the number is composite, otherwise it is probably prime

Primality Testing

- A primality test called Jacobi function:

$$a^{n-1} \bmod n = a^{\frac{n-1}{2}} \bmod n$$

- will give a result of 1 or –1
- Example
- $5^{37-1} = 5^{36} = 5^{3\times 3\times 4}$ (mod 37)
- $= 125^{3\times 4} = 14^{3\times 4}$ (mod 37)
- $= 6^{4} = 1296 = 1$ (mod 37)

Primitive Roots

- Recall from Euler's theorem that, every *a* and *n* that are relatively prime,

$$a^{\phi(n)} \equiv 1 \bmod n$$

- If the length of the period generated by *a* is *n - 1*, *a* is then the primitive roots of the modulus *n*.

- If *a* is a primitive root, all possible outputs except *n – 1* will be generated.

Primitive Roots

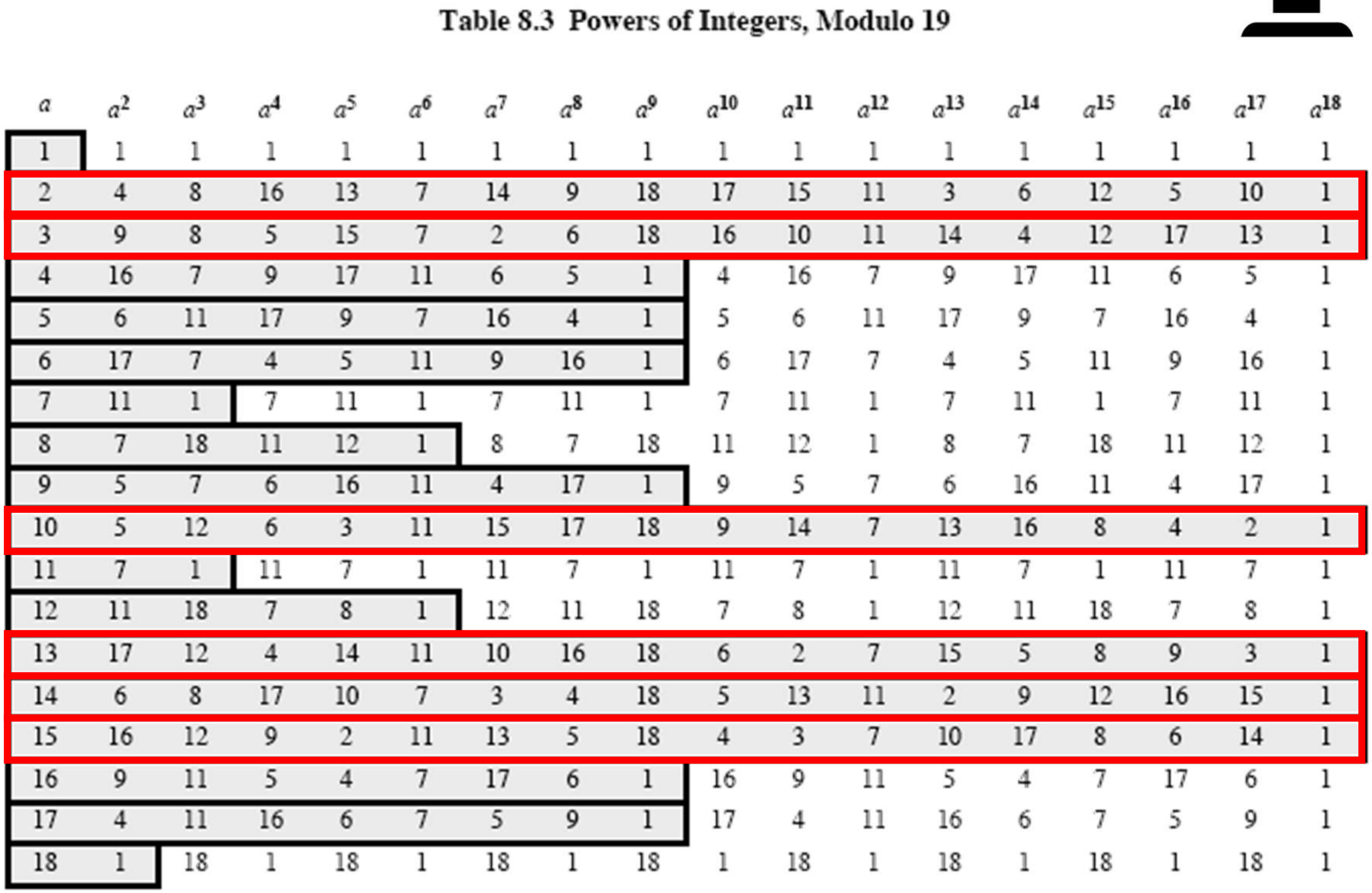


Table 8.3 Powers of Integers, Modulo 19

a	a^{2}	a^{3}	a^{4}	a^{5}	a^{6}	a^{7}	a^{8}	a^{9}	a^{10}	a^{11}	a^{12}	a^{13}	a^{14}	a^{15}	a^{16}	a^{17}	a^{18}
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
3	9	8	5	15	7	2	6	18	16	10	11	14	4	12	17	13	1
4	16	7	9	17	11	6	5	1	4	16	7	9	17	11	6	5	1
5	6	11	17	9	7	16	4	1	5	6	11	17	9	7	16	4	1
6	17	7	4	5	11	9	16	1	6	17	7	4	5	11	9	16	1
7	11	1	7	11	1	7	11	1	7	11	1	7	11	1	7	11	1
8	7	18	11	12	1	8	7	18	11	12	1	8	7	18	11	12	1
9	5	7	6	16	11	4	17	1	9	5	7	6	16	11	4	17	1
10	5	12	6	3	11	15	17	18	9	14	7	13	16	8	4	2	1
11	7	1	11	7	1	11	7	1	11	7	1	11	7	1	11	7	1
12	11	18	7	8	1	12	11	18	7	8	1	12	11	18	7	8	1
13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
15	16	12	9	2	11	13	5	18	4	3	7	10	17	8	6	14	1
16	9	11	5	4	7	17	6	1	16	9	11	5	4	7	17	6	1
17	4	11	16	6	7	5	9	1	17	4	11	16	6	7	5	9	1
18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1	18	1

Discrete Logarithms

- Consider the equation

$$y = g^x \bmod p$$

- Given g, x, and p, it is straightforward matter to calculate y

- Given g, y, and p, it is in general very difficult to calculate x

- The asymptotically fastest algorithm, which is not feasible for large primes, is on the order of:

$$e^{((\ln p)^{1/3} (\ln(\ln p))^{2/3})}$$

Modulo Arithmetic in Computers

- Square and multiply algorithm
  - A fast, efficient algorithm for doing exponentiation
  - Idea is to repeatedly square the base and multiply in the ones that are needed to compute the result
  - Find this by looking at binary representation of exponent

```
let base = a, result = 1
for each bit $e_i$ (LSB to MSB) of exponent
   if $e_i=0$ then
     square base mod p
   else if $e_i=1$ then
     multiply result by base mod p
   square base mod p
```

  - at the end, the required answer is result

Modulo Arithmetic in Computers

- Example: evaluate 7^5 mod 11
- $7^5 = 7^1 \times 1 \times 7^2 \times 0 \times 7^4 \times 1$ mod 11
  - $= 7^1 \times 1 \times 7^4$ mod 11
  - $= 7 \times 3$ mod 11
  - $= 10$ mod 11

Base	Result	Exp (5 = 1012)
	1	Init
7	1 × 7 = 7	1 (result = result × base; square base)
$7^2 = 49 = 5$	7	0 (square base)
$5^2 = 25 = 3$	7 × 3 = 10	1 (result = result × base; square base)
$3^2 = 9$		

澳門理工學院

Instituto Politécnico de Macau

Macao Polytechnic Institute

Public Key Cryptography

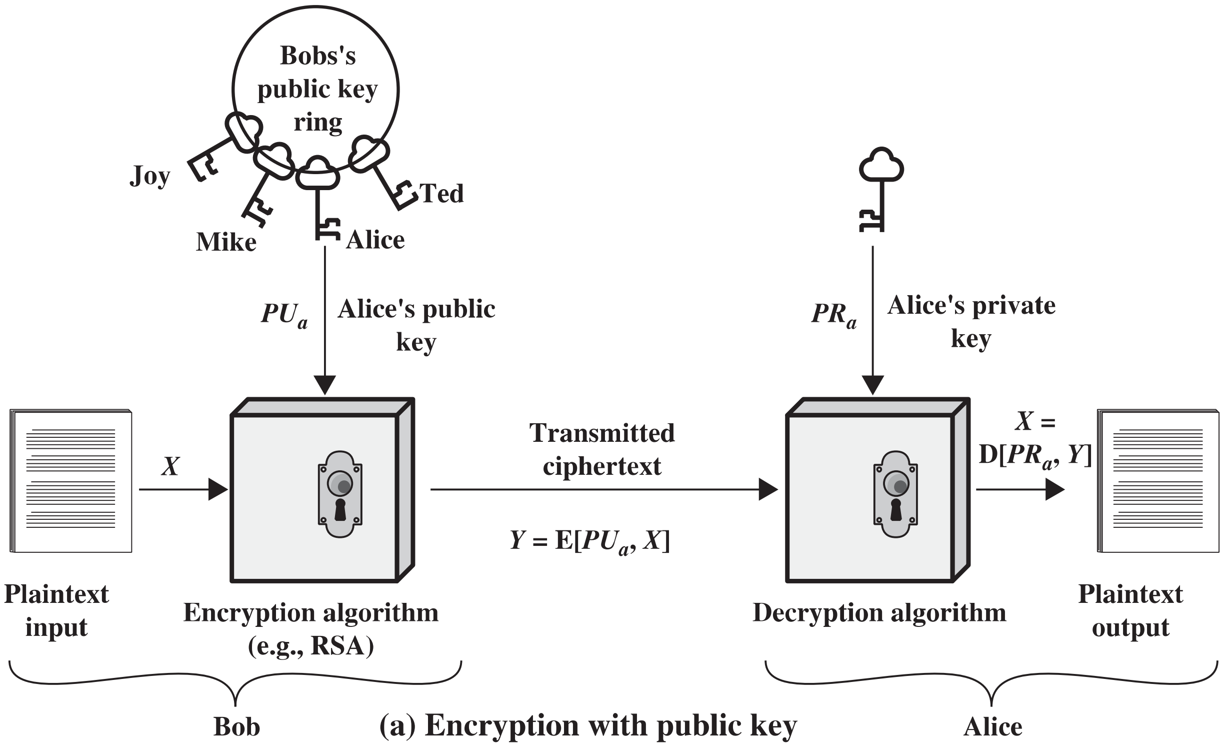
Private Key Cryptography

- Traditional private/secret/single key cryptography uses one key shared by both sender and receiver
- If this key is disclosed, communications are compromised
- It is symmetric, because parties are equal
- Hence it does not protect sender from receiver forging a message & claiming is sent by sender

Public Key Cryptography

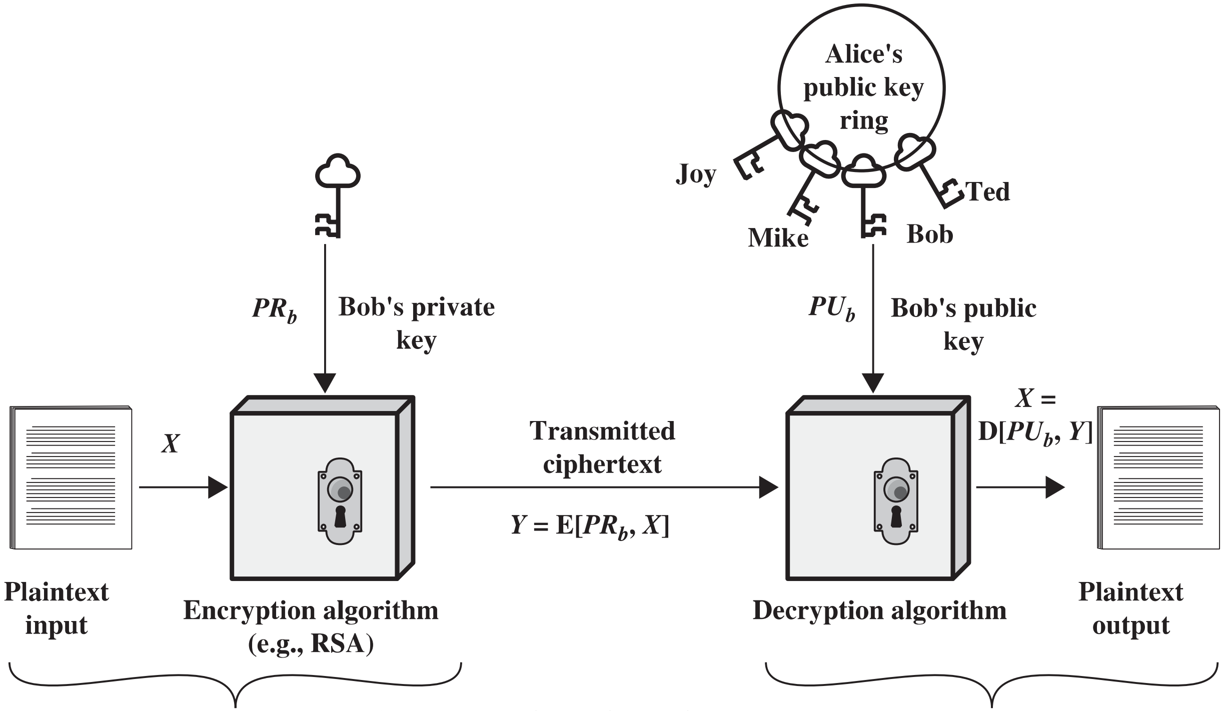
- Public key/two-key/asymmetric cryptography involves the use of two keys:
  - A public key, which may be known by anybody, and can be used to encrypt messages, and verify signatures
  - A private-key, known only to the recipient, used to decrypt messages, and sign (create) signatures
- Those who encrypt messages or verify signatures cannot decrypt messages or create signatures
- It is asymmetric because parties are not equal
- It is achieved by number theoretic concepts

Public Key Encryption



(a) Encryption with public key

Public Key Authentication



(b) Encryption with private key

Public Key Cryptography

- There are six principle elements:
  - *Plaintext*: This is the readable message or data that is fed into the algorithmas input.
  - *Encryption algorithm*: The encryption algorithm performs various transformations on the plaintext.
  - *Public and private keys*: This is a pair of keys that have been selected so that if one is used for encryption, the other is used for decryption. The exact transformations performed by the algorithm depend on the public or private key that is provided as input.
  - *Ciphertext*: This is the scrambled message produced as output. It depends on the plaintext and the key. For a given message, two different keys will produce two different ciphertexts.
  - *Decryption algorithm*: This algorithm accepts the ciphertext and the matching key and produces the original plaintext.

Public Key Cryptography

- The essential steps are the following:
  - Each user generates a key pair.
  - Each user publishes its public key.
  - Each user keeps its private key and maintains .a collection of public keys obtained from others.
  - If Bob wants to send a confidential message to Alice, Bob encrypts the message using Alice’s public key.
  - When Alice receives the encrypted message, she decrypts it using her private key.

Theory of Public Key

- The public key is easily computed from the private key and other information about the cipher

- Knowing the public key and public description of the cipher, it is still computationally infeasible to compute the private key

- The public key may be distributed to anyone wishing to communicate securely with its owner, although the secure distribution of the public key is a non-trivial problem – the key distribution problem

Conventional and Public Key Encryption

Conventional Encryption	**Public Key Encryption**
Needed to Work	*Needed to Work*
1. The same algorithm with the *same key* is used for encryption and decryption 2. The sender and receiver must *share* the algorithm and the key	1. One algorithm is used for encryption and decryption with *a pair of keys*, one for encryption and one for decryption 2. The sender and receiver must have their *own* matched pair of keys (not the same one)
Needed for Security	*Needed for Security*
1. The key must be kept *secret* 2. It must be impossible or at least *impractical to decipher* a message if no other information is available 3. Knowledge of the algorithm plus samples of ciphertext must be *insufficient* to determine the key	1. One of the two keys must be kept *secret* 2. It must be impossible or at least *impractical to decipher* a message if no other information is available 3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be *insufficient* to determine the other key

Public Key Cryptography

- Party A computes public key KU_a and private key KR_a
- Party B computes public key KU_b and private key KR_b
- A sends message M to B
  - $C = E_{KUb}(M)$
- B recovers the message
  - $M = D_{KRb}(C) = D_{KRb}[E_{KUb}(M)]$
- Private key is computationally infeasible with the knowledge of public key
- Message is unrecoverable with the knowledge of public key and ciphertext
- Encryption and decryption is reversible

Public Key Cryptography

- Facts
  - As the keys are a pair, no other key can replace one of them and form a pair. The key pair must work together
  - Private key is computationally infeasible with the knowledge of public key
  - Message is unrecoverable with the knowledge of public key and ciphertext
  - Encryption and decryption is reversible
  - The secure distribution of the public key is a non-trivial problem

Trapdoor One-way Function

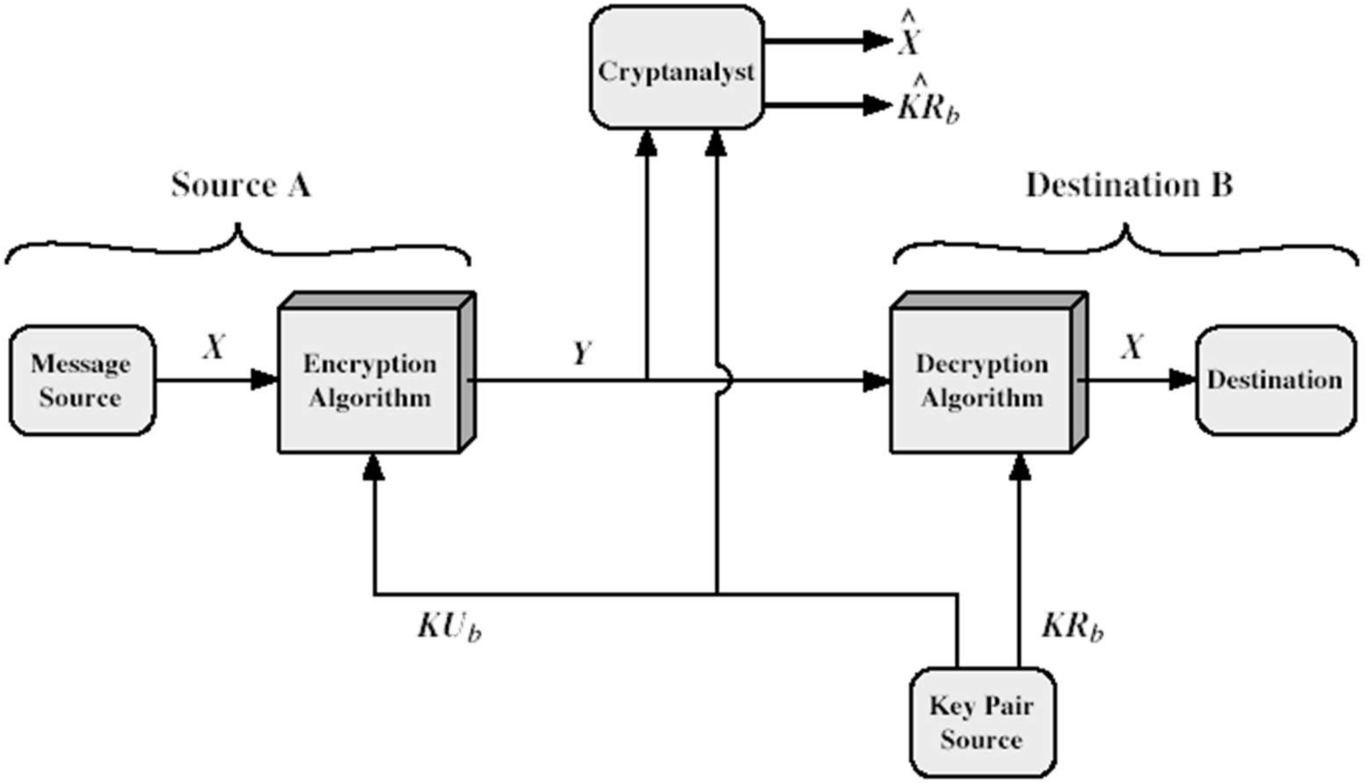
- A trapdoor one-way function is easy to calculate in one direction and infeasible to calculate in the other direction unless certain additional information is known
  - $Y = f_k(X)$ easy if k and X are known
  - $X = f_k^{-1}(Y)$ easy if k and Y are known
  - $X = f_k^{-1}(Y)$ infeasible if Y is known but k is not known
- The development of a practical public key scheme depends on discovery of a suitable trapdoor one-way function

Public Key Cryptanalysis

- Public key encryption scheme is vulnerable to a brute-force attack. The countermeasure is to use large keys at a acceptable encryption/decryption speeds for general-purpose use
- Another form of attack is to find some way to compute the private key given the public key
  - *No mathematical proof* shows that it is infeasible

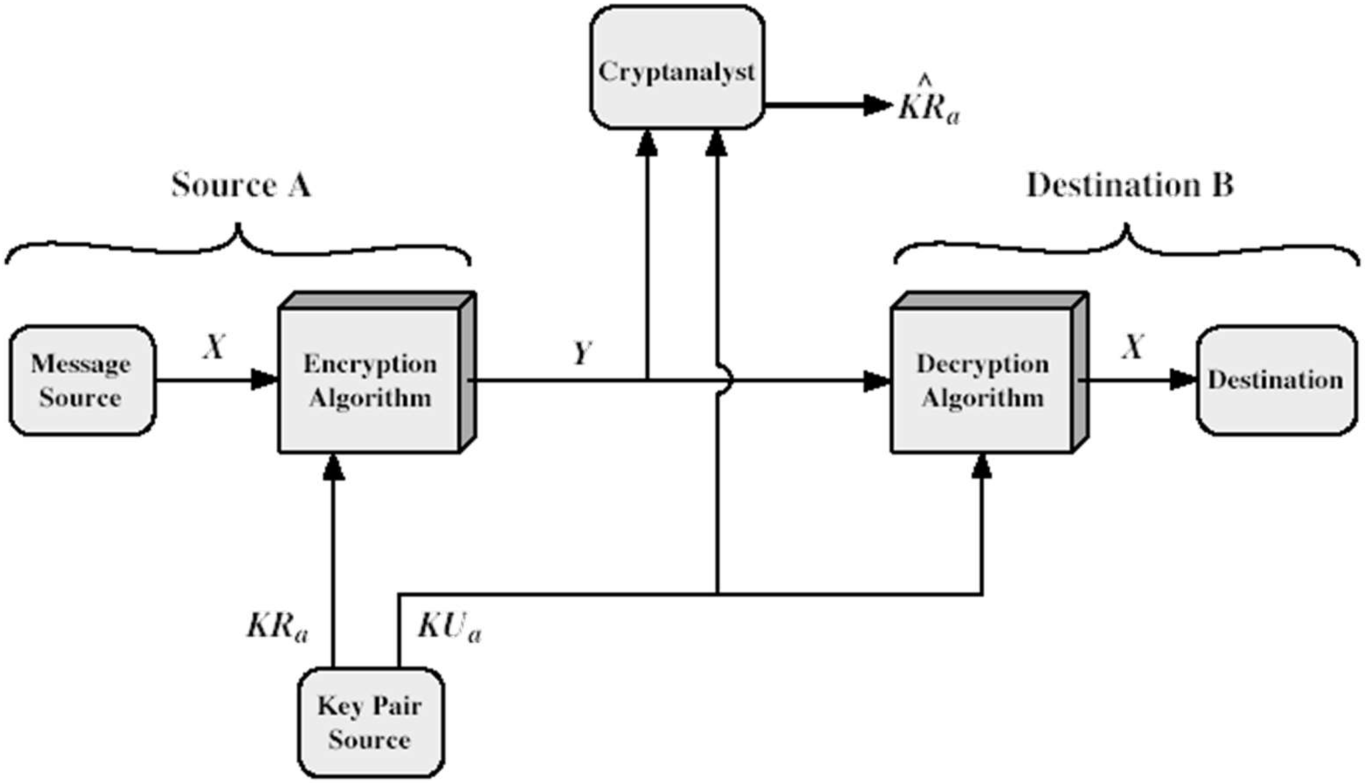
Public Key Cryptosystem

- Encryption



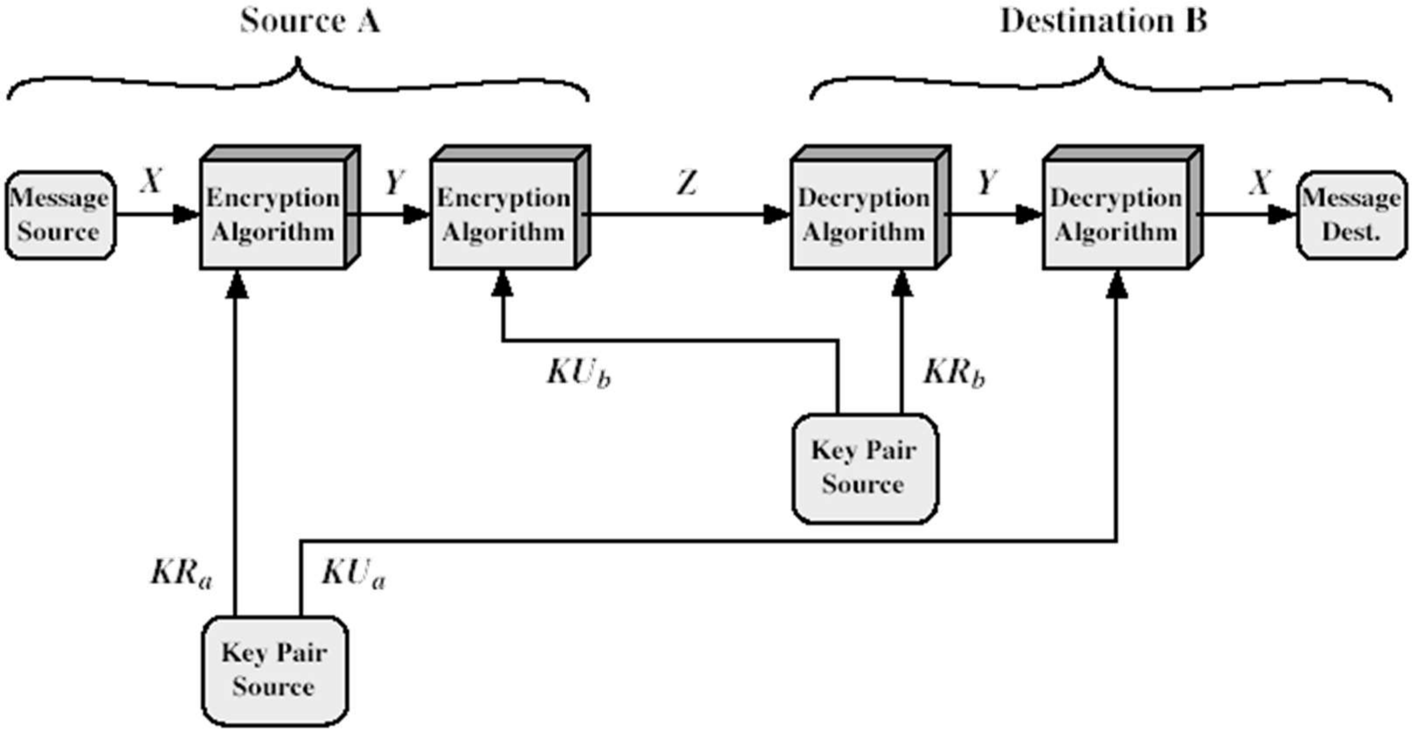
Public Key Cryptosystem

- Authentication



Public Key Cryptosystem

- Secrecy and Authentication



Classes of Public Key Algorithms

- Public key encryption (PKE)
  - Used to encrypt any arbitrary message
  - Anyone can use the public key to encrypt a message
  - Owner uses the private key to decrypt the messages
  - Any public key encryption scheme can be used as a PKDS by using the session key as the message
  - Many public key encryption schemes are also signature schemes (provided encryption & decryption can be done in either order)

- Public key distribution schemes (PKDS)
  - Used to securely exchange a single piece of information
  - Value depends on the two parties, but cannot be set
  - Value is normally used as a session key for a private-key scheme

- Signature schemes
  - Used to create a digital signature for some message
  - Owner uses private-key to signs (create) the signature
  - Anyone can use the public key to verify the signature

Public Key Encryption

- Public key encryption schemes can be used:
  - To encrypt arbitrary messages
  - As signature schemes
  - To exchange keys
- Common public key encryption schemes are RSA and ElGamal.
- These are based on exponentiation in various finite fields
- Other problems have been proposed for use (knapsacks, error correcting codes), but all these have been broken

澳門理工學院
Instituto Politécnico de Macau
Macao Polytechnic Institute

RSA

RSA (Rivest, Shamir, Adleman)

- Best known and widely regarded as most practical public key scheme
- First proposed by Rivest, Shamir & Adleman (RSA) in 1977
- Based on exponentiation over integers modulo a prime
- Security relies on the difficulty of finding discrete logarithms
  - Find x where $a^x = b \bmod p$
    - Eg. $x = \log_3 4 \bmod 13$ (ie $3^x = 4 \bmod 13$) has NO answer
    - Eg. $x = \log_2 3 \bmod 13 = 4$ by trying successive powers

RSA Algorithm

- To encrypt message M with public key (KU) = {e, n}
    - C = M^e mod n
- To decrypt ciphertext C with private key (KR) = {d, n}
    - M = C^d mod n
        - = $(M^e)^d$ mod n
        - = M^{ed} mod n,
    - where d is e^{-1} and e × d = 1 mod Ø(n)
- Euler's Theorem holds for the above

$$a^{\phi(n)+1} \equiv a \bmod n$$

RSA Setup

- Initially each user generates their public/private key pair by:
  - Select two large primes at random (~100 digit), p and q
  - Compute their system modulus $n = p \times q$
  - Compute $\emptyset(n) = (p - 1) \times (q - 1)$
  - Selecting at random the encryption key e, where $e < \emptyset(n)$, $\gcd(e, \emptyset(n)) = 1$
  - Compute the decryption key d where $d = e^{-1} \bmod \emptyset(n)$ and $0 <= d <= \emptyset(n)$
- Make the public key public : $KU = \{e, n\}$
- Keep the private key secret : $KR = \{d, n\}$

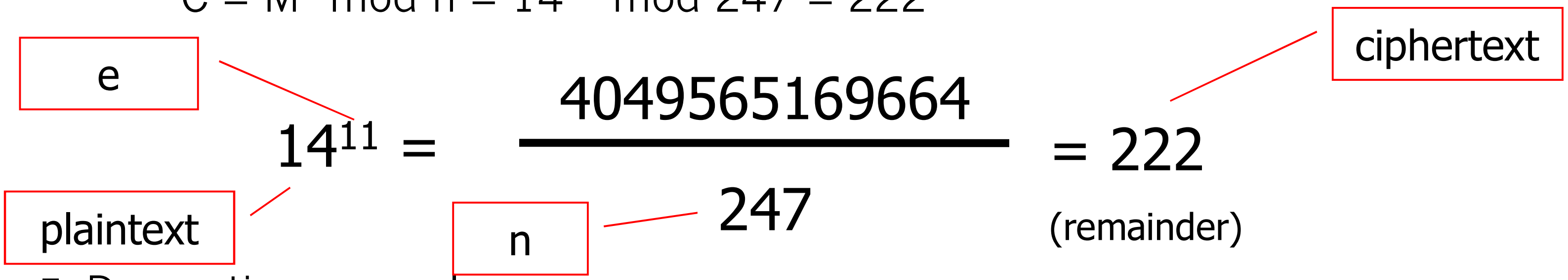
RSA Setup

- Key generation example:
  - $p = 13, q = 19$
  - $n = p \times q = 13 \times 19 = 247$
  - $\varnothing(n) = (p - 1)(q - 1) = (13 - 1)(19 - 1) = 216$
  - $e = 11 \bmod \varnothing(n)$
  - $d = e^{-1} = 59 \bmod \varnothing(n)$

  - The public key KU is {11, 247}
  - The private key KR is {59, 247}

RSA Setup

- Encryption example:
  - To encrypt the message 14 with public key, we do
    - $C = M^e \bmod n = 14^{11} \bmod 247 = 222$



e

$$14^{11} = \frac{4049565169664}{247} = 222$$

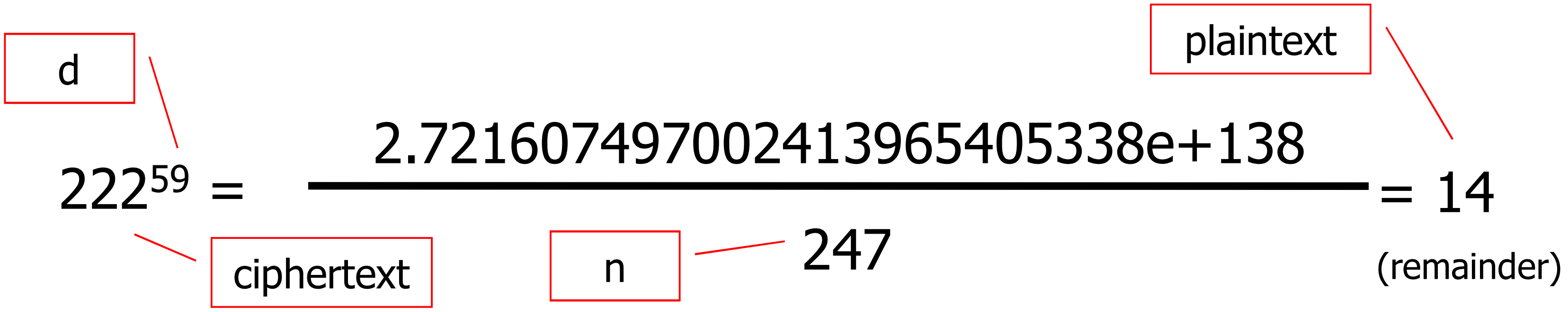
(remainder)

plaintext

n

ciphertext

- Decryption example:
  - To decrypt the ciphertext above, we do
    - $M = C^d \bmod n = 222^{59} \bmod 247 = 14$



d

$$222^{59} = \frac{2.7216074970024139654053 38e+138}{247} = 14$$

(remainder)

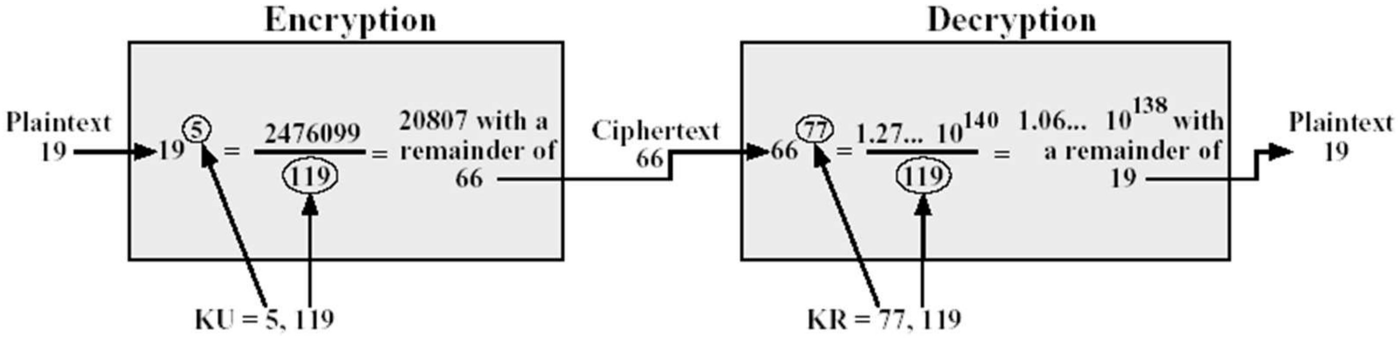
ciphertext

n

plaintext

Simple RSA Example

- Select p = 7, q = 17 $\boxed{\leftarrow}$ p and q are prime
  - n = p × q = 119
  - ∅(n) = (p – 1) × (q – 1) = 96
- Select a random number e = 5, which is gcd(5, 96) = 1
- The public key KU = {5, 119}
- Determine d such that d x e = 1 mod 96 and d < 96
  - d = 77 (As 77 × 5 = 385 = 4 * 96 + 1)
- The private key KR = {77, 119}


Encryption
Plaintext 19
19^5 = 2476099 / 119 = 20807 with a remainder of 66
KU = 5, 119
Ciphertext 66
Decryption
66^77 = 1.27... 10^140 / 119 = 1.06... 10^138 with a remainder of 19
KR = 77, 119
Plaintext 19

RSA Parameter Selection

- Need to choose suitably large p and q, n = p × q is then the size of key

- Usually choose the encryption exponent e to be a small number which must be relatively prime to Ø(n)

- Typically e may be the same for all users (provided certain precautions are taken). Originally a value of 3 was suggested. Now regarded as too small. So 2^{16-1} = 65535 is often used

- Note that the decryption exponent d will then be large if e is small

Chinese Remainder Theorem (CRT)

- Note that encryption using small exponents will be fast
- But then decryption exponent will be large and hence decryption is slower
- A significant improvement in decryption speed for RSA can be obtained by using the Chinese remainder theorem to work modulo p and q respectively
- Since p and q are only half the size of $n = p \times q$ and thus the arithmetic is much faster

Chinese Remainder Theorem (CRT)

- CRT is used in RSA by creating two equations from the decryption calculation: $M = C^{d} \bmod n$
  - $M_1 = C^{d \bmod (p-1)} \bmod p$
  - $M_2 = C^{d \bmod (q-1)} \bmod q$

- We now apply the CRT to reconstruct our message:

- $M = M_1(q^{-1} \bmod p)q + M_2(p^{-1} \bmod q)\ p(\bmod n)$

Or

- $M = M_1 + [(M_2 - M_1) \times (p^{-1} \bmod q) \bmod q] \times p$

Example

- Using the previous example, we have C = 66, d = 77, n = 119, p = 7 and q = 17
  - $M_1 = 66^{77 \bmod (7 - 1)} \pmod{7} = 5$
  - $M_2 = 66^{77 \bmod (17 - 1)} \pmod{17} = 2$
  - $M = 5 \times (17^{-1} \bmod 7) \times 17 + 2 \times (7^{-1} \bmod 17) \times 7 \pmod{119}$

    $= 68 + 70 \pmod{119}$

    $= 19 \pmod{119}$
  - $M = 5 + [(2 - 5) \times 7^{-1} \pmod{17}] \pmod{17} \times 7$

    $= 5 + [-3 \times 5] \pmod{17} \times 7$

    $= 5 + 2 \times 7$

    $= 19$

Computational Aspects

- Encryption and decryption involve raising an integer to an integer power, mod n. It is inefficient at all!
- Make use of modular arithmetic property, reduce intermediate result modulo n. Practical!
- Solving X_{16} equals to repeatedly take the square of X_2, X_4, X_8, X_{16}
- Given X_5, and $5 = 101_2$

$$X^5 = X^{2^2 \times 1} \times X^{2^2 \times 0} \times X^{2^2 \times 1}$$

Computational Aspects

- To prevent p and q to be discovered by exhaustive search, they have to be large enough

- So far, no useful technique that yield arbitrarily large primes

- We pick a large huge number and prove that it is probable prime.

- Probabilistic tests allow an educated guess as to whether a candidate number is prime or not. This means that the probability of the guess being wrong can be made arbitrarily small

Security of RSA

- Three possible approaches to attacking the RSA
  - Brute force
    - Involves trying all possible private keys
  - Mathematical attacks
    - Several approaches, all equivalent in effect to factoring the product of two primes
  - Timing attacks
    - Depends on the running time of the decryption algorithm

Mathematical Attacks

- The factoring problem
  - Factor n into its two prime factors
  - Determine Ø(n) directly, without first determining p and q
  - Determine d directly without first determining Ø(n)
- Factoring is a hard problem (Exponential)
- Note that determining Ø(n) given n is same
- Table 9.3 shows the progress of factorization
- The continuing increase in computing power and refinement of factoring algorithms make a big threat to larger key sizes. A key size in the range of 1024 to 2048 bits seems reasonable.

Mathematical Attacks

- Other than specifying the key size, a number of other constraints have been suggested by researchers
  - p and q should be different in length by only a few digits, on the order of magnitude 10^{75} to 10^{100} for a 1024-bit key (309 decimal digits)
  - Both $(p - 1)$ and $(q - 1)$ should contain a large prime factor
  - $\gcd(p - 1, q - 1)$ should be small
- It is demonstrated that if $e < n$ and $d < n^{1/4}$, then d can be easily determined

Timing Attacks

- It can determine a private key by keeping track of how long a computer takes to decipher messages. It applies to any public key algorithms
- It is alarming because
    - It comes from a completely unexpected direction
    - It is a cipher text only attack
- It is analogous to a burglar guessing the combination of a safe by observing how long it takes for someone to turn the dial from number to number
- It is assumed that in modular multiplication, the execution is particular slow in a iteration, the bit is 1

Timing Attacks

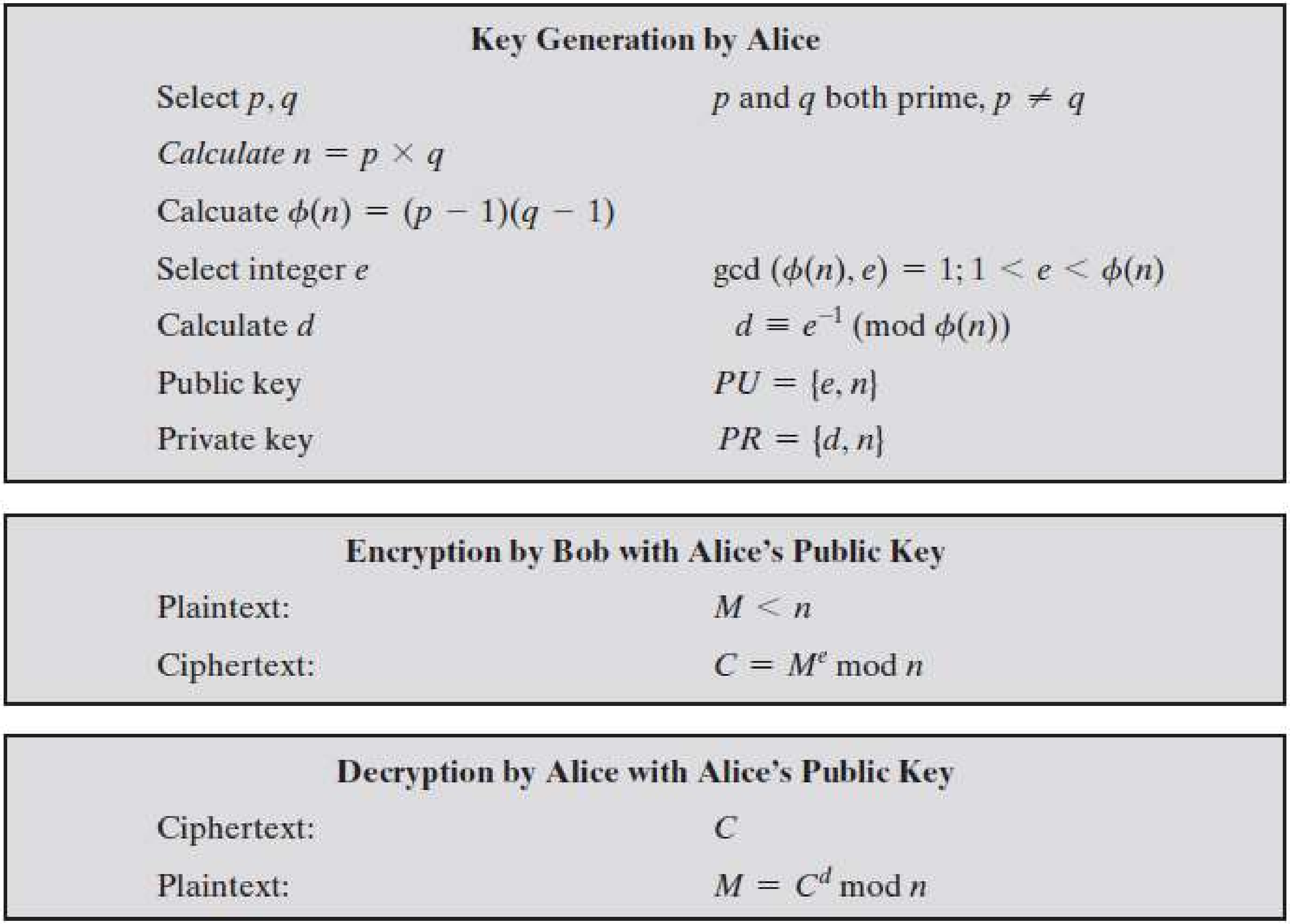
- Some simple countermeasures
  - Constant exponentiation time
    - Degrade performance
  - Random delay
    - Confuse the attack
  - Blinding
    - Multiply the ciphertext by a random number before performing exponentiation
    - It has 2 to 10% performance penalty
    - Algorithm
      - Generate a secret random *r* between 0 and n – 1
      - Compute
        - $C' = C(r^{e}) \bmod n$
        - $M' = (C')^{d} \bmod n$
        - $M = M'r^{-1} \bmod n$

Practical Use of Public Key Schemes

- Given the issue of speed, public key schemes cannot replace the traditional encryption ciphers. (Eg, DES)
- Generally use public key to exchange a single message containing a session key for use in a block cipher. Use the fast private key schemes to handle the bulk of the data
- And another with a signature to verify message content (next topic)

RSA Key Generation Summary

- RSA Algorithm



Key Generation by Alice

Select p, q	p and q both prime, $p \neq q$
Calculate $n = p \times q$	
Calcuate $\phi(n) = (p - 1)(q - 1)$	
Select integer e	$\gcd(\phi(n), e) = 1; 1 < e < \phi(n)$
Calculate d	$d \equiv e^{-1} (\bmod \phi(n))$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Public Key

Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

澳 門 理 工 學 院

Instituto Politécnico de Macau

Macao Polytechnic Institute

Diffie-Hellman Key Exchange

Key Exchange

- A situation that two parties are going to use DES encryption algorithm for the exchange of highly confidential information over the Internet.
- They are now thinking to setup their DES secret…How do they share the secret keys?

Diffie-Hellman Key Exchange

- The purpose of the algorithm is for key exchange for subsequent encryption of messages

- 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 whose value depends on the participants (and their private and public key information)

- Its effectiveness is depended on the difficulty of computing discrete logarithms

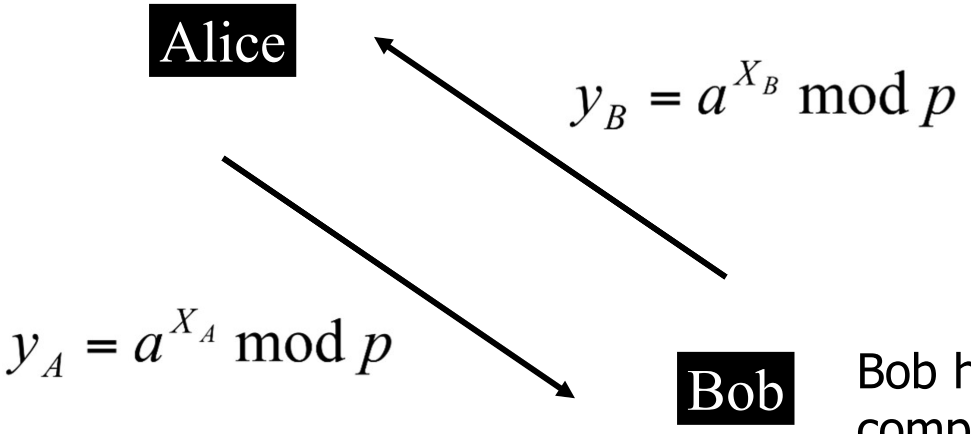
Diffie-Hellman Key Exchange

Alice has a secret component: X_A

Public component: a mod p

The session key K_{AB} is

$$y_B^{X_A} \bmod p$$



Alice

$$y_B = a^{X_B} \bmod p$$

$$y_A = a^{X_A} \bmod p$$

Bob

Bob has a secret component: X_B

The session key K_{AB} is

$$y_A^{X_B} \bmod p$$

Diffie-Hellman Setup

- Have two people Alice & Bob who wish to exchange some key over an insecure communications channel
- Initially they (and all who participate in key exchanges):
  - Select a large prime integer or polynomial p (~200 digits) and a primitive root a (mod p)
  - Alice chooses a secret key $x^A < p$
  - Bob chooses a secret key $x^B < p$
  - Alice and Bob compute their public keys:
  - $y^A = a^{XA}$ mod p and $y^B = a^{XB}$ mod p
  - Alice and Bob make public y^A and y^B respectively
- The public components are a, p, y^A, and y^B
- The private components are X^A and X^B

Diffie-Hellman Key Exchange

- The key is then calculated as: $K_{AB} = a^{XA \times XB} \bmod p = y_A^{XB} \bmod p$ (which B can compute) $= y_B^{XA} \bmod p$ (which A can compute)

- K_{AB} may then be used as a session key in a private-key cipher to secure communications between Alice and Bob

- Note if Alice and Bob subsequently wish to communicate, they will have the same key as before, unless they choose new public keys

Diffie-Hellman Example

- Given prime p = 97 with primitive root a = 5
- Alice chooses secret x_A = 36 & computes public key $y_A = 5^{36} = 50 \bmod 97$
- Bob chooses secret x_B = 58 & computes public key $y_B = 5^{58} = 44 \bmod 97$
- Alice and Bob exchange their public keys (50 & 44 respectively)
- Alice computes the shared secret $K = 44^{36} = 75 \bmod 97$
- Bob computes the shared secret $K = 50^{58} = 75 \bmod 97$
- From {50, 44}, an attacker cannot easily compute 75
- An attacker Charlie would need to first crack one of the secrets knowing only the public information, eg Alice's by solving $x_A = \log_5 50 = 36 \bmod 97$ (hard), and then doing Alice's key computation $K = 44^{36} = 75 \bmod 97$

Diffie-Hellman in Practice

- Each time any 2 parties want to communicate they could choose new, random secret keys, and compute and exchange public keys
- Safe against passive eavesdropping, but not active attacks
- Does give new session keys each time though to secure against active attack additional protocols are needed
- Alternatively they can generate "long-term" public keys and have them placed in a secure central directory, assumed that the directory is trusted

Exercises

1. Perform encryption and decryption using the RSA algorithm for the following
   - p = 3; q = 11, d = 7; M = 5
   - p = 17; q = 31; e = 7; M = 2

2. Illustrate the operation of RSA, given the following parameters:
   - n=119 (7 x 17)
   - e=11

3. Determine the decryption exponent d, and hence details the public and private keys for this user. Then show how a message M=20 would be encrypted and decrypted

Exercises

4. Generate a RSA key pair and perform encryption and decryption using the following information
    - p = 7, q = 11
    - p = 11, q = 19
    - p = 13, q = 19

5. What are the three possible attacks to RSA algorithm?

6. How do you secure the RSA algorithm so that it does not suffer from the mathematical attack or timing attack?

Exercises

7. Determine a suitable private and public key, and then show the exchange of a message M = 4

8. Describe and illustrate how Diffie-Hellman algorithm is used for key exchange.

9. Consider a Diffie-Hellman scheme with a common prime p = 11 and a primitive root a = 2
   - If user A has a private key $X_A = 9$, what is A's public key Y_A
   - If user B has a public key $Y_B = 3$, what is the shared secret key K?

Exercises

10. Set up a session key with your friend by Diffie-Hellman key exchange protocol using $a = 11$ and $p = 79$.

11. Use Chinese Remainder Theory to decrypt the encrypted message M in question one.

澳門理工學院
INSTITUTO POLITÉCNICO DE MACAU
MACAO POLYTECHNIC INSTITUTE