

Asymmetric cryptography; public key cryptography

Prepared by Dr. Ayman Khallel

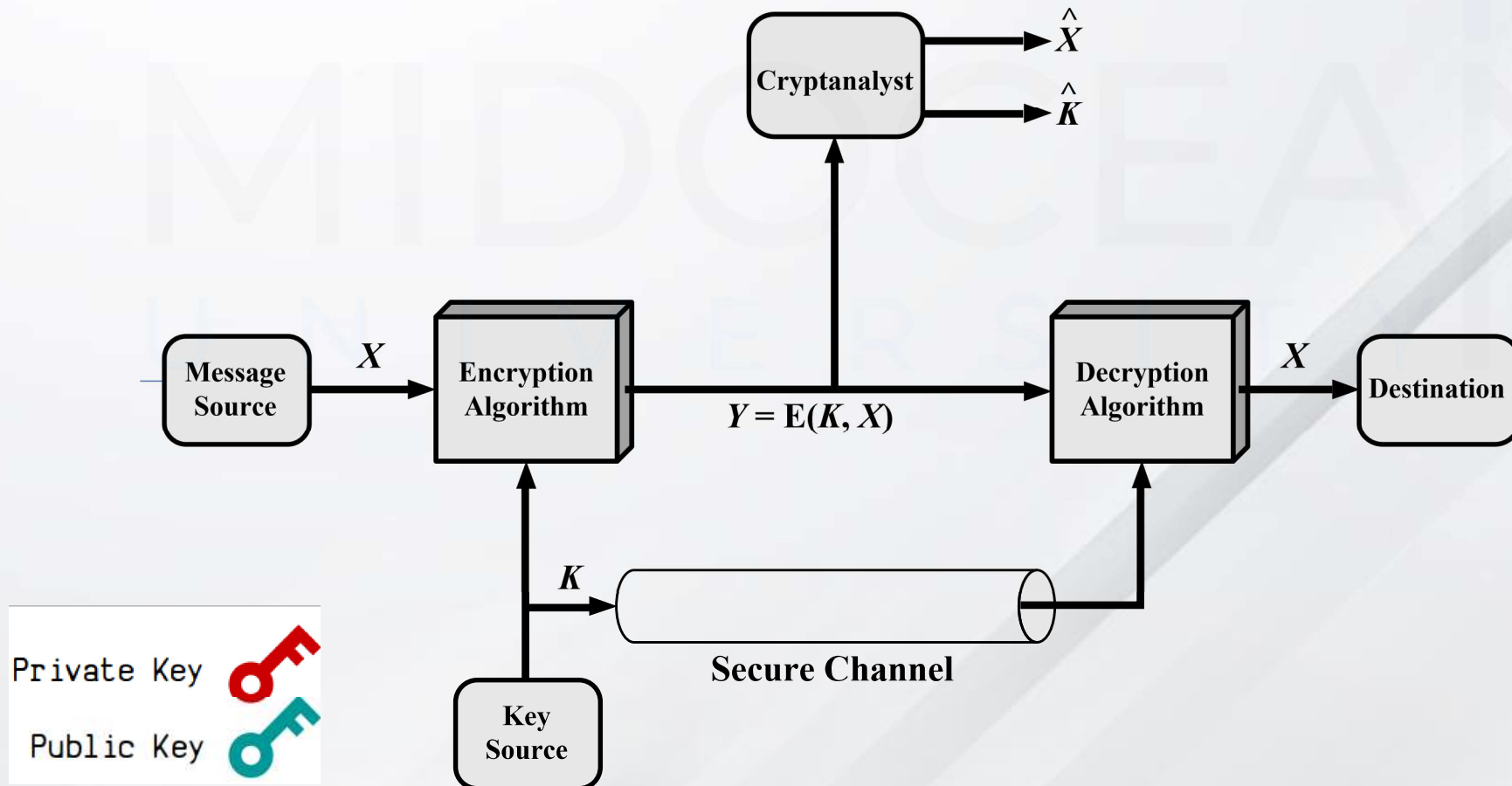
IT373
Information Security

Lecture 6



Course content

1. Overview of security concepts and definitions
2. Symmetric cryptography; modern encryption algorithms (DES, AES)
3. Modern data encryption standards
4. Block cipher cryptanalysis and usage
5. Modern stream ciphers
6. Asymmetric cryptography; public key cryptography, key certificates and management
7. Message authentication and hash functions
8. Hash algorithms
9. Digital signature and authentication protocols
10. Electronic mail security, IP security, web security.

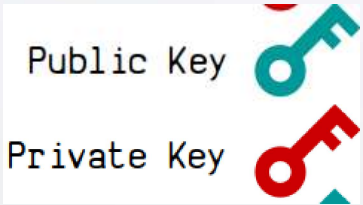


Terminology Related to Asymmetric Encryption



Asymmetric Keys

Two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.



Public Key Certificate

A digital document issued and digitally signed by the private key of a Certification Authority that binds the name of a subscriber to a public key. The certificate indicates that the subscriber identified in the certificate has sole control and access to the corresponding private key.

Public Key (Asymmetric) Cryptographic Algorithm

A cryptographic algorithm that uses two related keys, a public key and a private key. The two keys have the property that deriving the private key from the public key is computationally infeasible.

Public Key Infrastructure (PKI)

A set of policies, processes, server platforms, software and workstations used for the purpose of administering certificates and public-private key pairs, including the ability to issue, maintain, and revoke public key certificates.

Source: *Glossary of Key Information Security Terms*, NIST IR 7298

Misconceptions Concerning Public-Key Encryption

- Public-key encryption is more secure from cryptanalysis than symmetric encryption
- Public-key encryption is a general-purpose technique that has made symmetric encryption obsolete
- There is a feeling that key distribution is trivial when using public-key encryption, compared to the cumbersome handshaking involved with key distribution centers for symmetric encryption



Principles of Public-Key Cryptosystems

- The concept of public-key cryptography evolved from an attempt to attack two of the most difficult problems associated with symmetric encryption:

Key distribution

- How to have secure communications in general without having to trust a KDC with your key

Digital signatures

- How to verify that a message comes intact from the claimed sender

- Whitfield Diffie and Martin Hellman from Stanford University achieved a breakthrough in 1976 by coming up with a method that addressed both problems and was radically different from all previous approaches to cryptography



Public-Key Cryptosystems

A public-key encryption scheme has six ingredients:

Plaintext

The readable message or data that is fed into the algorithm as input

Encryption algorithm

Performs various transformations on the plaintext

Public key

Used for encryption or decryption

Private key

Used for encryption or decryption

Ciphertext

The scrambled message produced as output

Decryption algorithm

Accepts the ciphertext and the matching key and produces the original plaintext



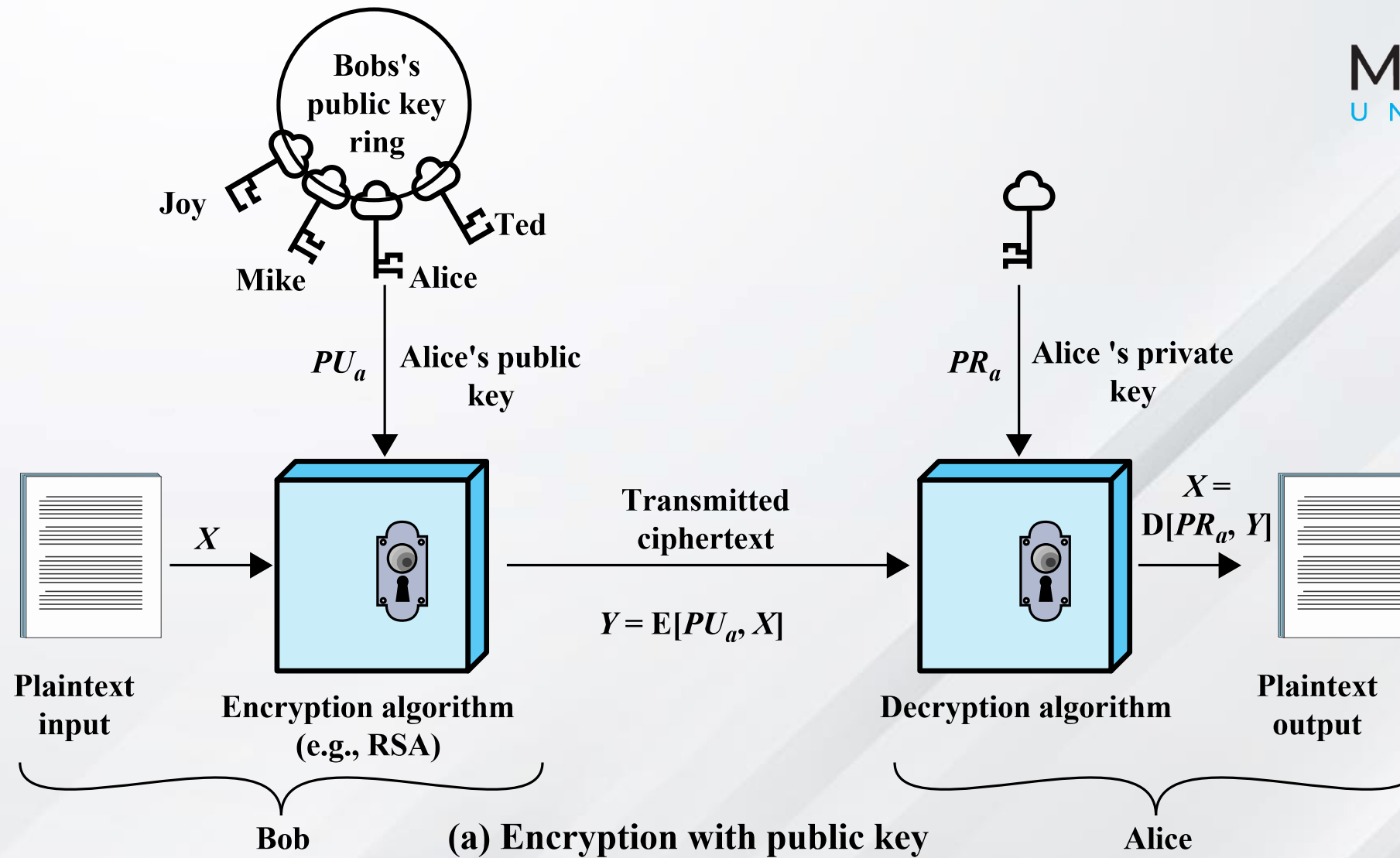


Figure 9.1 Public-Key Cryptography



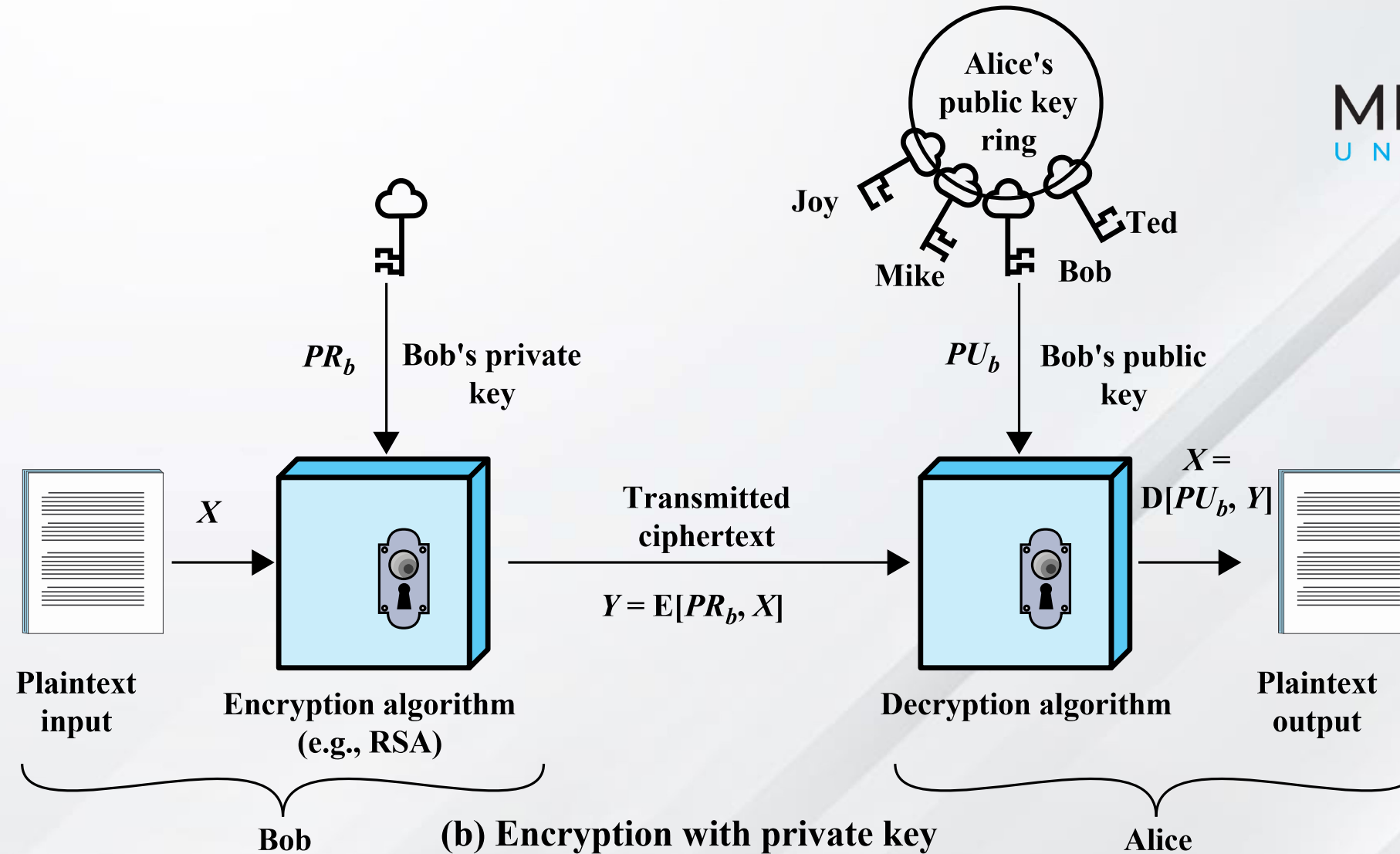


Figure 9.1 Public-Key Cryptography



Table 9.2 CONVENTIONAL AND PUBLIC-KEY ENCRYPTION

Conventional Encryption/symmetric Encryption

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.

Needed for Security:

1. The key must be kept secret.
2. It must be impossible or at least impractical to decipher a message if the key is kept secret.
3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.

Public-Key Encryption/asymmetric Encryption

Needed to Work:

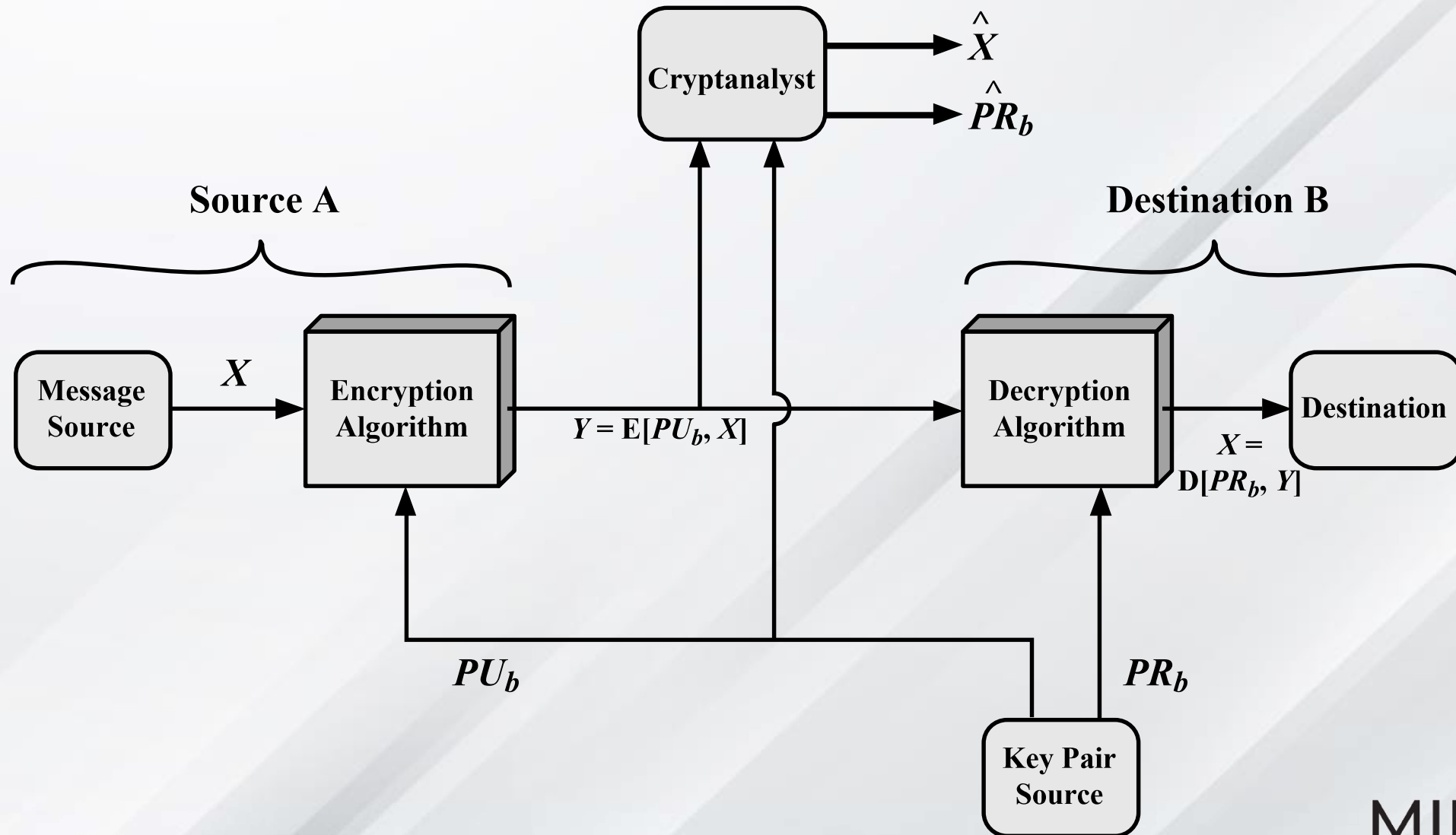
1. One algorithm is used for encryption and a related algorithm for decryption with a pair of keys, one for encryption and one for decryption.
2. The sender and receiver must each have one of the matched pair of keys (not the same one).

Needed for Security:

1. One of the two keys must be kept secret.
2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret.
3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.



Public-Key Cryptosystem: Confidentiality



Public-Key Cryptosystem: Authentication

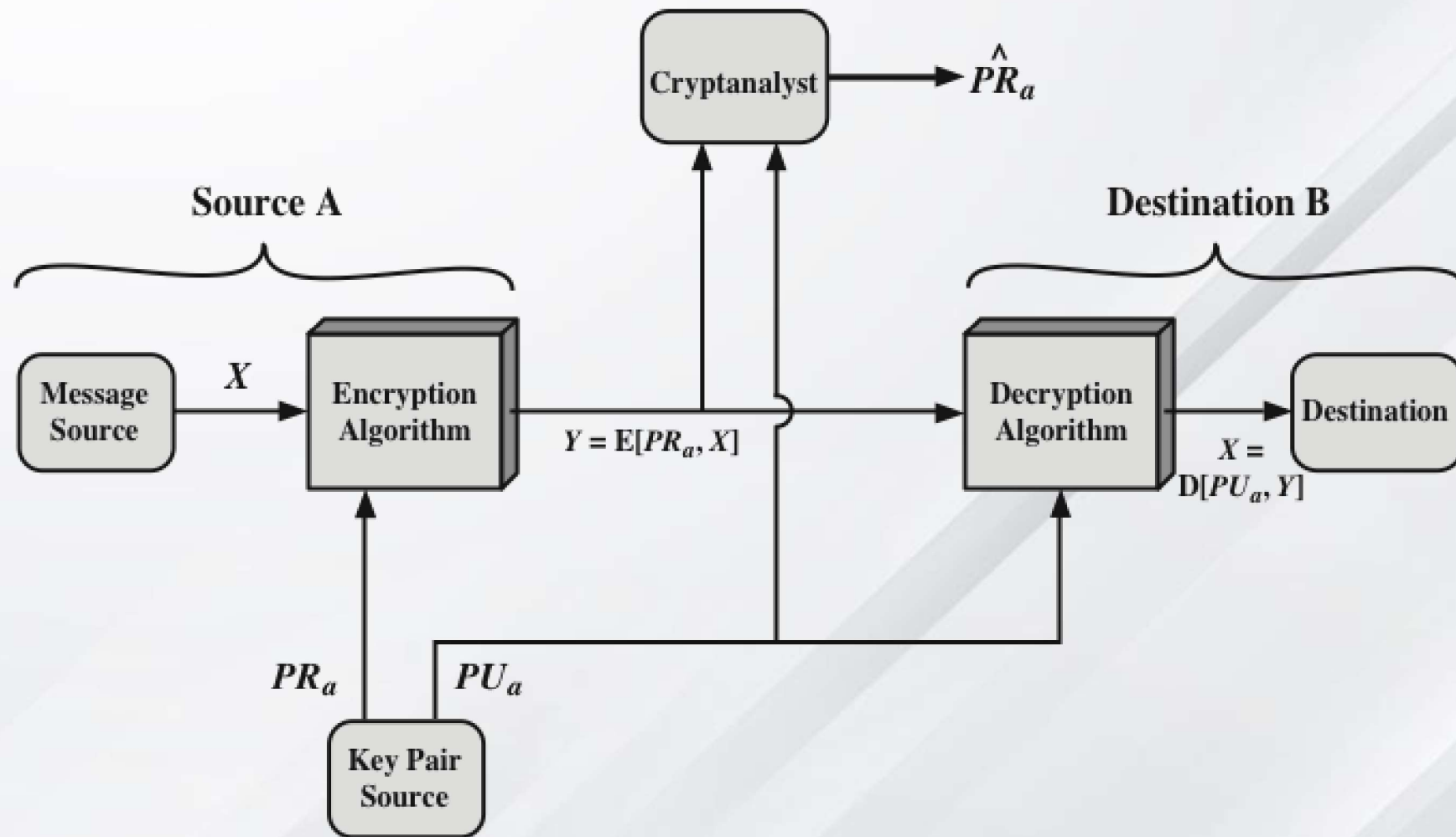


Figure 9.3 Public-Key Cryptosystem: Authentication

Dr. Ayman Khallel Ibrahim.



Public-Key Cryptosystem: Authentication and Secrecy

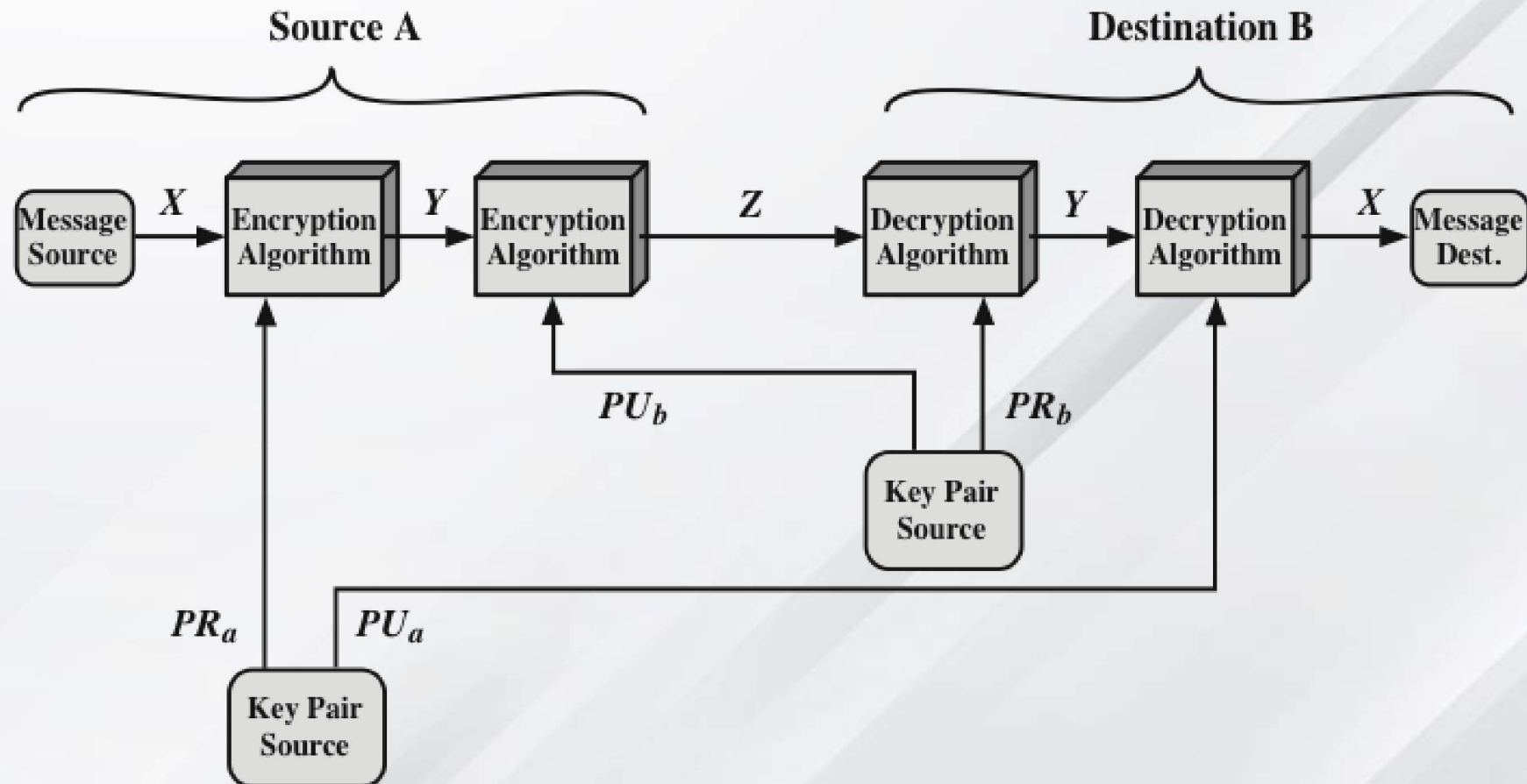
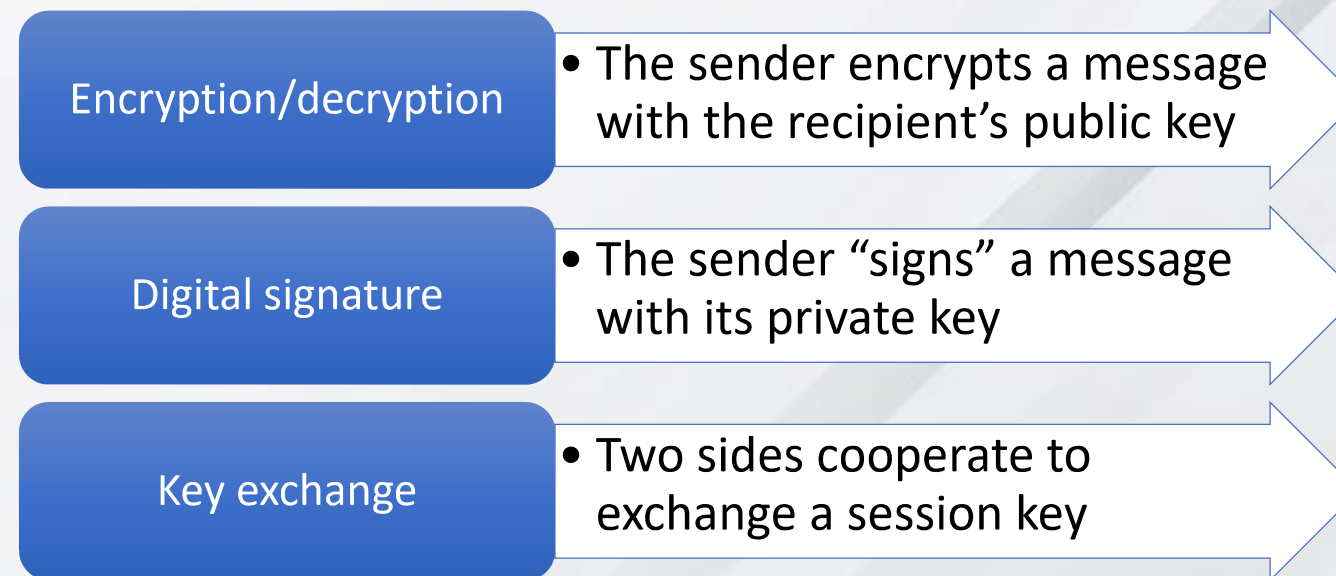


Figure 9.4 Public-Key Cryptosystem: Authentication and Secrecy

Dr. Ayman Khallel Ibrahim.

Applications for Public-Key Cryptosystems

- Public-key cryptosystems can be classified into three categories:



- Some algorithms are suitable for all three applications, whereas others can be used only for one or two



Table 9.3

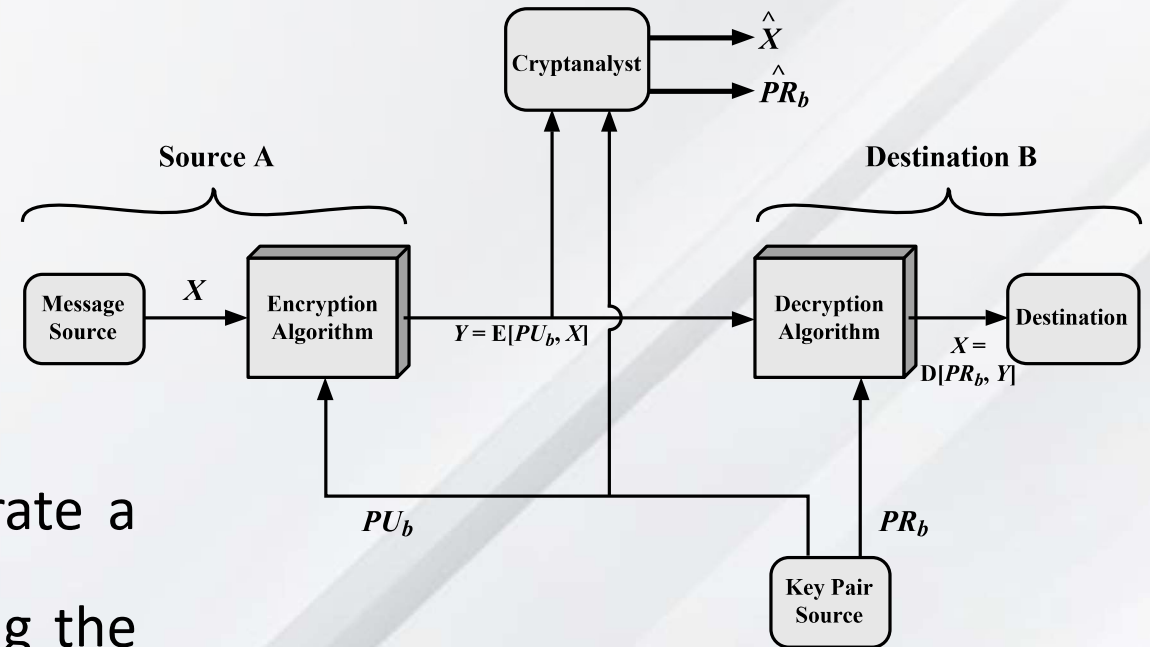
Applications for Public-Key Cryptosystems

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

Table 9.3 Applications for Public-Key Cryptosystems

Public-Key Requirements

- Conditions that these algorithms must fulfill:
 - It is computationally easy for a party B to generate a pair (public-key PU_b , private key PR_b)
 - It is computationally easy for a sender A, knowing the public key and the message to be encrypted, to generate the corresponding ciphertext
 - It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message
 - It is computationally infeasible for an adversary, knowing the public key, to determine the private key
 - It is computationally infeasible for an adversary, knowing the public key and a ciphertext, to recover the original message
 - The two keys can be applied in either order



Public-Key Requirements

- Need a trap-door one-way function
 - A one-way function is one that maps a domain into a range such that every function value has a unique inverse, with the condition that the calculation of the function is easy, whereas the calculation of the inverse is infeasible
 - $Y = f(X)$ easy
 - $X = f^{-1}(Y)$ infeasible
- A trap-door one-way function is a family of invertible functions f_k , such that
 - $Y = f_k(X)$ 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 known but k not known
- A practical public-key scheme depends on a suitable trap-door one-way function

Public-Key Cryptanalysis



- A public-key encryption scheme is vulnerable to a brute-force attack
 - Countermeasure: use large keys
 - Key size must be small enough for practical encryption and decryption
 - Key sizes that have been proposed result in encryption/decryption speeds that are too slow for general-purpose use
 - Public-key encryption is currently confined to key management and signature applications
- Another form of attack is to find some way to compute the private key given the public key
 - To date it has not been mathematically proven that this form of attack is infeasible for a particular public-key algorithm
- Finally, there is a probable-message attack
 - This attack can be thwarted by appending some random to simple messages

bits



Rivest-Shamir-Adleman (RSA) Algorithm

- Developed in 1977 at MIT by Ron Rivest, Adi Shamir & Len Adleman
- Most widely used general-purpose approach to public-key encryption
- Is a cipher in which the plaintext and ciphertext are integers between 0 and $n - 1$ for some n
 - A typical size for n is 1024 bits, or 309 decimal digits



RSA Algorithm

- RSA makes use of an expression with exponentials
- Plaintext is encrypted in blocks with each block having a binary value less than some number n
- Encryption and decryption are of the following form, for some plaintext block M and ciphertext block C

$$C = M^e \bmod n$$

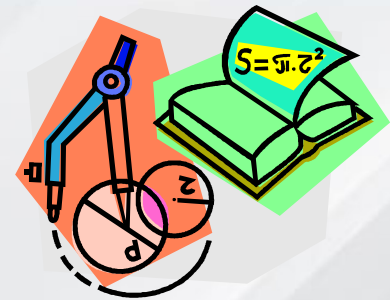
$$M = C^d \bmod n = (M^e)^d \bmod n = M^{ed} \bmod n$$

- Both sender and receiver must know the value of n
- The sender knows the value of e , and only the receiver knows the value of d
- This is a public-key encryption algorithm with a public key of $PU=\{e,n\}$ and a private key of $PR=\{d,n\}$



Algorithm Requirements

- For this algorithm to be satisfactory for public-key encryption, the following requirements must be met:
 1. It is possible to find values of e, d, n such that $M^{ed} \bmod n = M$ for all $M < n$
 2. It is relatively easy to calculate $M^e \bmod n$ and $C^d \bmod n$ for all values of $M < n$
 3. It is infeasible to determine d given e and n



Key Generation by Alice

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

$d \equiv e^{-1} \pmod{\phi(n)}$

Public key

$PU = \{e, n\}$

Private key

$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

Plaintext:

$M < n$

Ciphertext:

$C = M^e \pmod n$

Decryption by Alice with Alice's Private Key

Ciphertext:

C

Plaintext:

$M = C^d \pmod n$

Step 1: Key Generation

- 1. Choose two prime numbers: $p = 3$ and $q = 11$.
- 2. Calculate n :

$$n = p * q = 3 * 11 = 33$$

- 3. Calculate Euler's Totient Function (ϕ):

$$\phi(n) = (p - 1) * (q - 1) = 2 * 10 = 20$$

- 4. Choose an integer e (**public exponent**) such that $1 < e < \phi(n)$ and e is coprime to $\phi(n)$. Choose $e = 3$.

- 5. Calculate d (private key exponent):

$$d * e \equiv 1 \pmod{\phi(n)}, \text{ so } d = 7$$

Public Key: $(e, n) = (3, 33)$

Private Key: $(d, n) = (7, 33)$

20	3	4	7
1	1	1	1
2	3	2	7
4		4	
5			
10			
20			

Figure 9.5 The RSA Algorithm

Key Generation by Alice

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	$d \equiv e^{-1} \pmod{\phi(n)}$
Public key	$PU = \{e, n\}$
Private key	$PR = \{d, n\}$

Encryption by Bob with Alice's Public Key

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

Decryption by Alice with Alice's Private Key

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

Step 2: Encryption

Encrypt the message $m = 7$ using the public key $(3, 33)$.

Calculate ciphertext c :

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

$$c = 7^3 \pmod{33} = 343 \pmod{33} = 13$$

Encrypted message: $c = 13$

Step 3: Decryption

Decrypt the ciphertext $c = 13$ using the private key $(7, 33)$.

Calculate original message m :

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

$$m = 13^7 \pmod{33} = 7$$

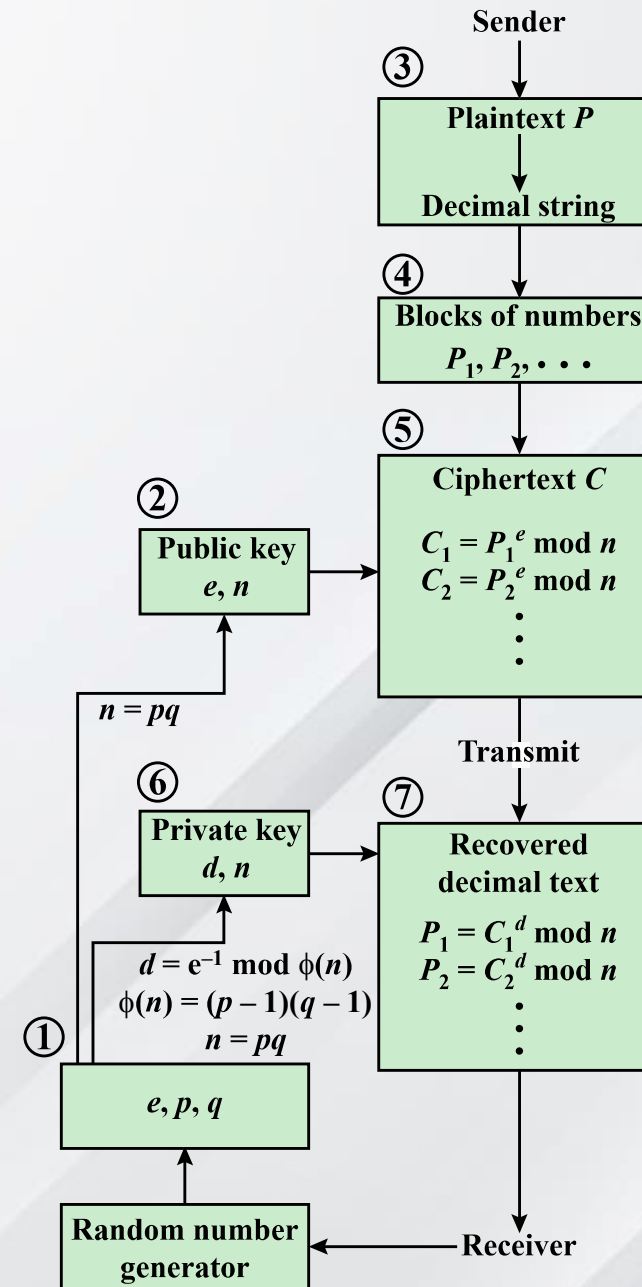
Decrypted message: $m = 7$, which matches the original message.



Figure 9.5 The RSA Algorithm

Figure 9.7

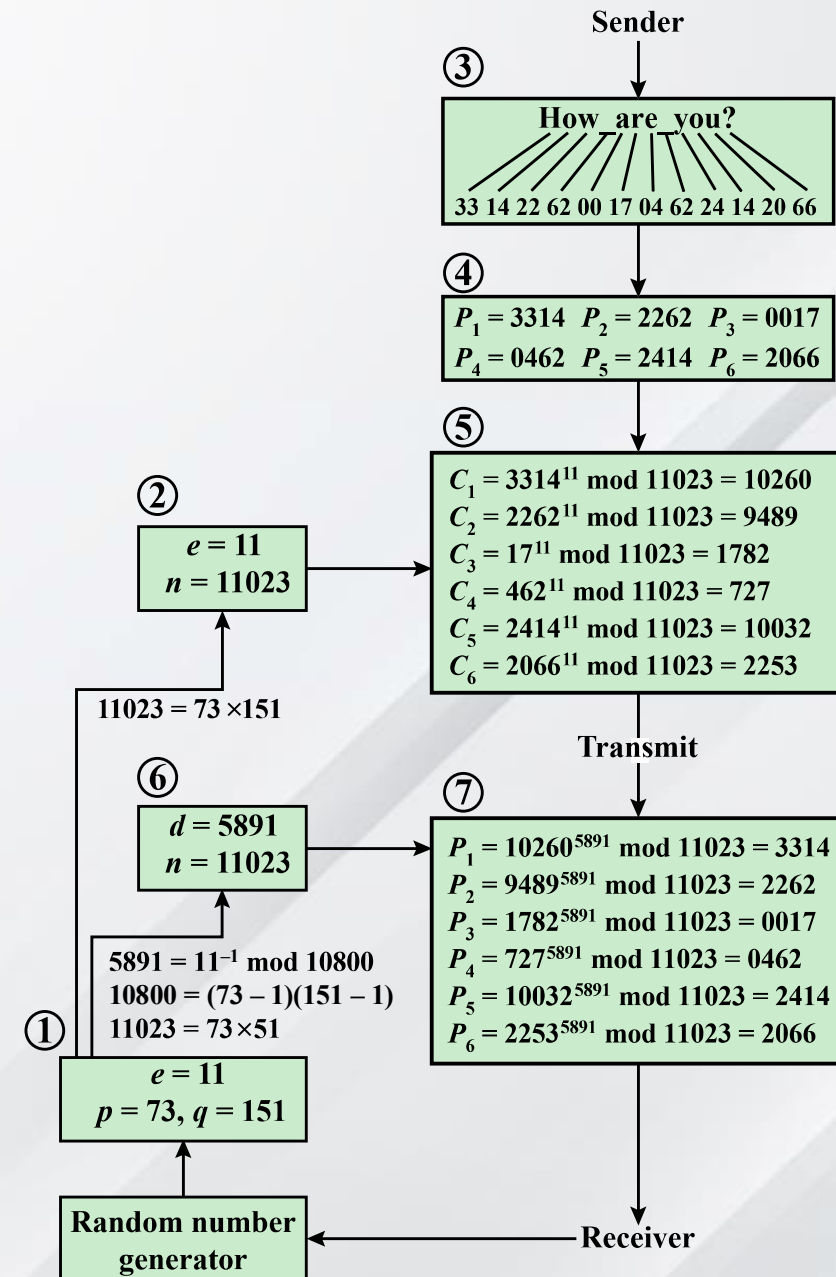
RSA Processing
of Multiple Blocks



(a) General approach

Figure 9.7

RSA Processing
of Multiple
Blocks



Exponentiation in Modular Arithmetic

- Both encryption and decryption in RSA involve raising an integer to an integer power, mod n
- Can make use of a property of modular arithmetic:
$$[(a \bmod n) \times (b \bmod n)] \bmod n = (a \times b) \bmod n$$
- With RSA you are dealing with potentially large exponents so efficiency of exponentiation is a consideration
- Ex:
 - $x^{16} = x * x * x * x * x * x * x * x * x * x * x * x * x * x * x * x$
 - $x^{16} = x^2, x^4, x^8, x^{16}$



Example: Calculating $x^{11} \bmod n$

- **Goal:** Compute $x^{11} \bmod n$ for integers x and n .
- **Breakdown:**
 - $X^{11} = x^{1+2+8} = (X)(X^2)(X^8)$
 - We decompose X^{11} using powers of 2 to simplify the calculation.
- **Step-by-Step Calculation:**
- Calculate the following modulo n :
 - $X \bmod n$
 - $X^2 \bmod n$
 - $X^4 \bmod n$
 - $X^8 \bmod n$
- Finally, combine results:

$$x^{11} \bmod n = [(x \bmod n) \cdot (x^2 \bmod n) \cdot (x^8 \bmod n)] \bmod n$$



Efficient Operation Using the Public Key

- To speed up the operation of the RSA algorithm using the public key, a specific choice of e is usually made
- The most common choice is 65537 ($2^{16} + 1$)
 - Two other popular choices are $e=3$ and $e=17$
 - Each of these choices has only two 1 bits, so the number of multiplications required to perform exponentiation is minimized
 - With a very small public key, such as $e = 3$, RSA becomes vulnerable to a simple attack



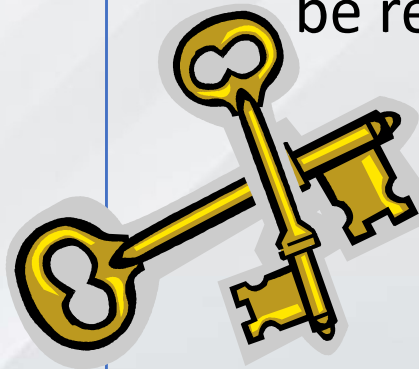
Efficient Operation Using the Private Key

- Decryption uses exponentiation to power d
 - A small value of d is vulnerable to a brute-force attack and to other forms of cryptanalysis
- Can use the Chinese Remainder Theorem (CRT) to speed up computation
 - The quantities $d \bmod (p - 1)$ and $d \bmod (q - 1)$ can be precalculated
 - End result is that the calculation is approximately four times as fast as evaluating $M = C^d \bmod n$ directly



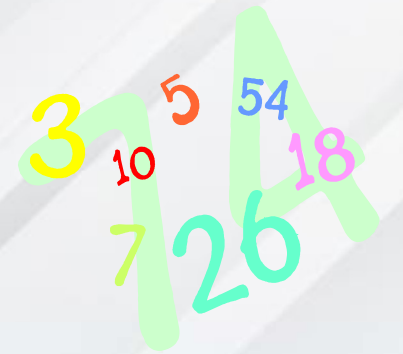
Key Generation

- Before the application of the public-key cryptosystem each participant must generate a pair of keys:
 - Determine two prime numbers p and q
 - Select either e or d and calculate the other
- Because the value of $n = pq$ will be known to any potential adversary, primes must be chosen from a sufficiently large set
 - The method used for finding large primes must be reasonably efficient



Procedure for Picking a Prime Number

1. Pick an odd integer n at random
2. Pick an integer $a < n$ at random
3. Perform the probabilistic primality test with a as a parameter. If n fails the test, reject the value n and go to step 1
4. If n has passed a sufficient number of tests, accept n ; otherwise, go to step 2



The Security of RSA



RSA Key Size

- **Definition:** Key size = length of the modulus n (product of two large primes p and q).
- **Common Sizes:**
 - 1024 bits (Weak, avoid using).
 - 2048 bits (Secure, widely used).
 - 3072 or 4096 bits (Higher security for sensitive applications).
- **Prime Size:** p and q must be large enough to ensure n is the desired key size (e.g., for a 2048-bit RSA key, p and q should each be around 1024 bits).
- **Security Impact:** Larger key sizes offer higher security but are slower to process.

RSA Key Size (bits)	Security Level (approx.)	Common Use Cases
1024	Weak	Deprecated
2048	Good	General applications
3072	Strong	Sensitive applications
4096	Very Strong	High-security requirements



Padding in RSA Encryption

- **What is Padding?**

- Padding adds extra data to a message before RSA encryption.
- Ensures message length compatibility with key size.
- Enhances security against attacks.

- **Common Padding Schemes**

- **1. PKCS#1 v1.5 Padding:**

- Simple padding method.
- Fills the remaining space with specific bytes.
- Vulnerable to padding oracle attacks.

- **2. OAEP (Optimal Asymmetric Encryption Padding):**

- Uses hash functions and masks for randomness.
- Provides semantic security, making ciphertext indistinguishable.
- Recommended for modern RSA encryption due to enhanced security.



pythonProject8

RSA.py

Project

pythonProject8

venv library root

AEC CBC D.py

AEC CBC E.py

AEC CFB Dec.py

AEC CFB Enc.py

AES EBC.py

DEC EBC.py

main.py

RSA.py

External Libraries

Scratches and Consoles

1

2

3

4

5

6

7

8

9

10

11

12

13

#pip install cryptography

from cryptography.hazmat.primitives.asymmetric import rsa, padding

from cryptography.hazmat.primitives import serialization

Generate RSA private and public keys

private_key = rsa.generate_private_key(

public_exponent=65537,

key_size=2048

)

public_key = private_key.public_key()

PEM (Privacy-Enhanced Mail) format is a widely used format for encoding and storing cryptographic

Run: RSA

IPZK+yECgYEALECSHOWT3KH4REtHXWLS/yanX/daR01SLyKHFjM2OWBQmFtVZ013

9Yi0INh2nRM/HslbXKc70iAIxIQqtnw+8V6R0jr6cowrs5X22+6yoKatk1loypHi

eNC/rcVLNw0yqk+EWgyLEHkWYV746uqMQCAcvwoy1sTPpypJDuDnasE=

-----END RSA PRIVATE KEY-----

Encrypted Message: b"n\x05\x07GR\x0f5#\x95-^\x87\xb54.\xf9\x92\xff/\x8c\xdb\xceX&j\xefv\\ \x12\x05m7~\xe9\xdfw\x04\xfc\x0e

Run

TODO

Problems

Terminal

ExcelReader

Python Console

45:65 (46 chars)

0% / 7%

Python 3.9 (pythonProject8)

1006 of 1979M

PyCharm 2020.3.5 available // Update... (today 12:18 AM)

Event Log

ExcelEditor

Search

ENG UKX

9:45 AM 11/5/2024

Summary

- Present an overview of the basic principles of public-key cryptosystems
- Explain the two distinct uses of public-key cryptosystems
- List and explain the requirements for a public-key cryptosystem
- Present an overview of the RSA algorithm
- Summarize the relevant issues related to the complexity of algorithms





Thank You

