



# Chapter 9

---

Public Key Cryptography and RSA

# 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



# Table 9.1

## 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.



# 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

# Public-Key Cryptography

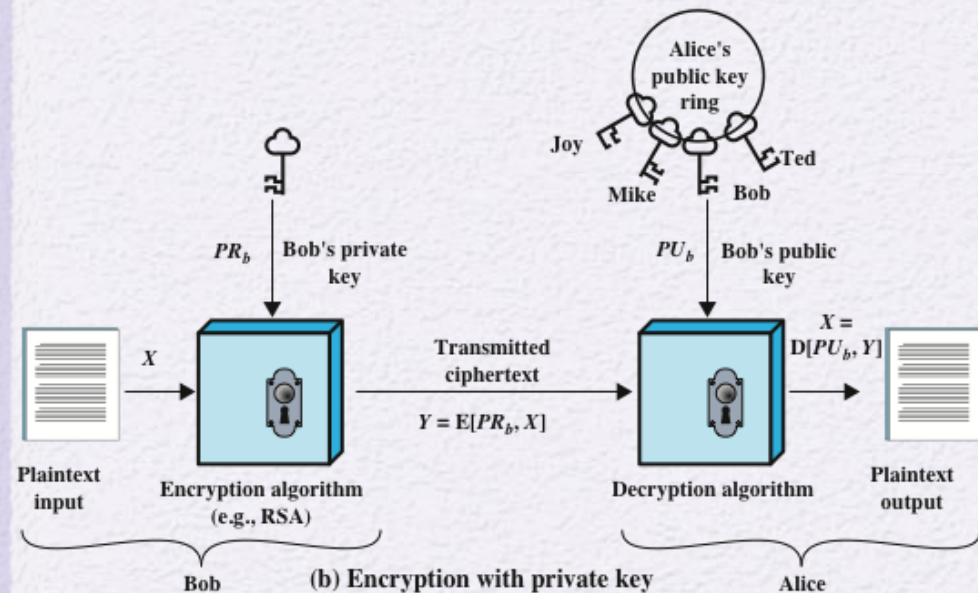
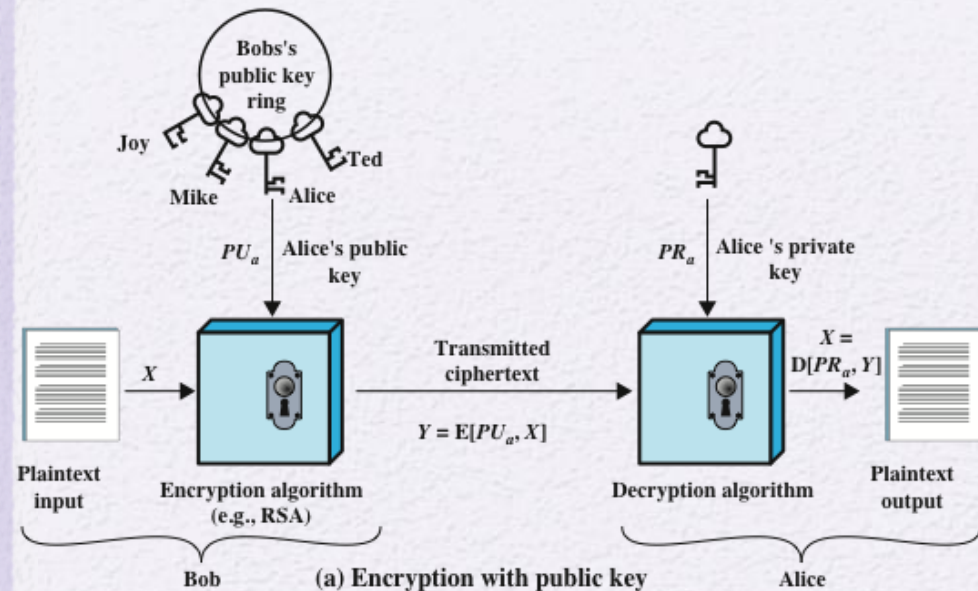
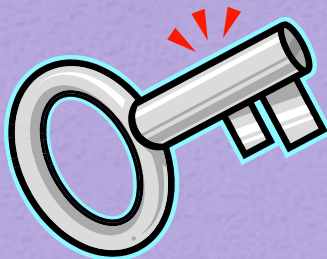


Figure 9.1 Public-Key Cryptography



# Table 9.2

## Conventional and Public-Key Encryption

Conventional Encryption	Public-Key Encryption
<p><i>Needed to Work:</i></p> <ol style="list-style-type: none"><li>1. The same algorithm with the same key is used for encryption and decryption.</li><li>2. The sender and receiver must share the algorithm and the key.</li></ol> <p><i>Needed for Security:</i></p> <ol style="list-style-type: none"><li>1. The key must be kept secret.</li><li>2. It must be impossible or at least impractical to decipher a message if the key is kept secret.</li><li>3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.</li></ol>	<p><i>Needed to Work:</i></p> <ol style="list-style-type: none"><li>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.</li><li>2. The sender and receiver must each have one of the matched pair of keys (not the same one).</li></ol> <p><i>Needed for Security:</i></p> <ol style="list-style-type: none"><li>1. One of the two keys must be kept secret.</li><li>2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret.</li><li>3. Knowledge of the algorithm plus one of the keys plus samples of ciphertext must be insufficient to determine the other key.</li></ol>

# Public-Key Cryptosystem: Secrecy

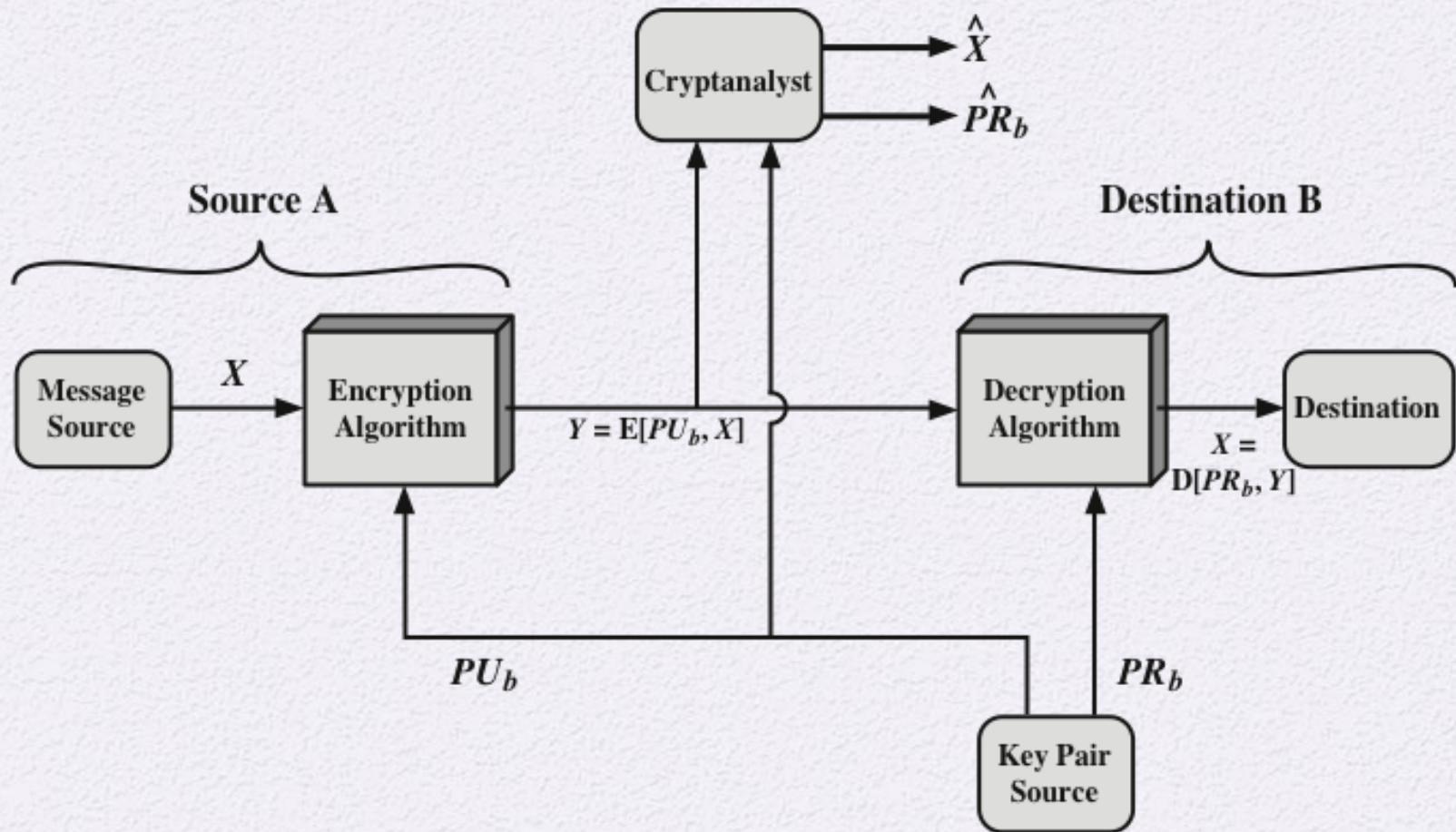


Figure 9.2 Public-Key Cryptosystem: Secrecy



# Public-Key Cryptosystem: Authentication

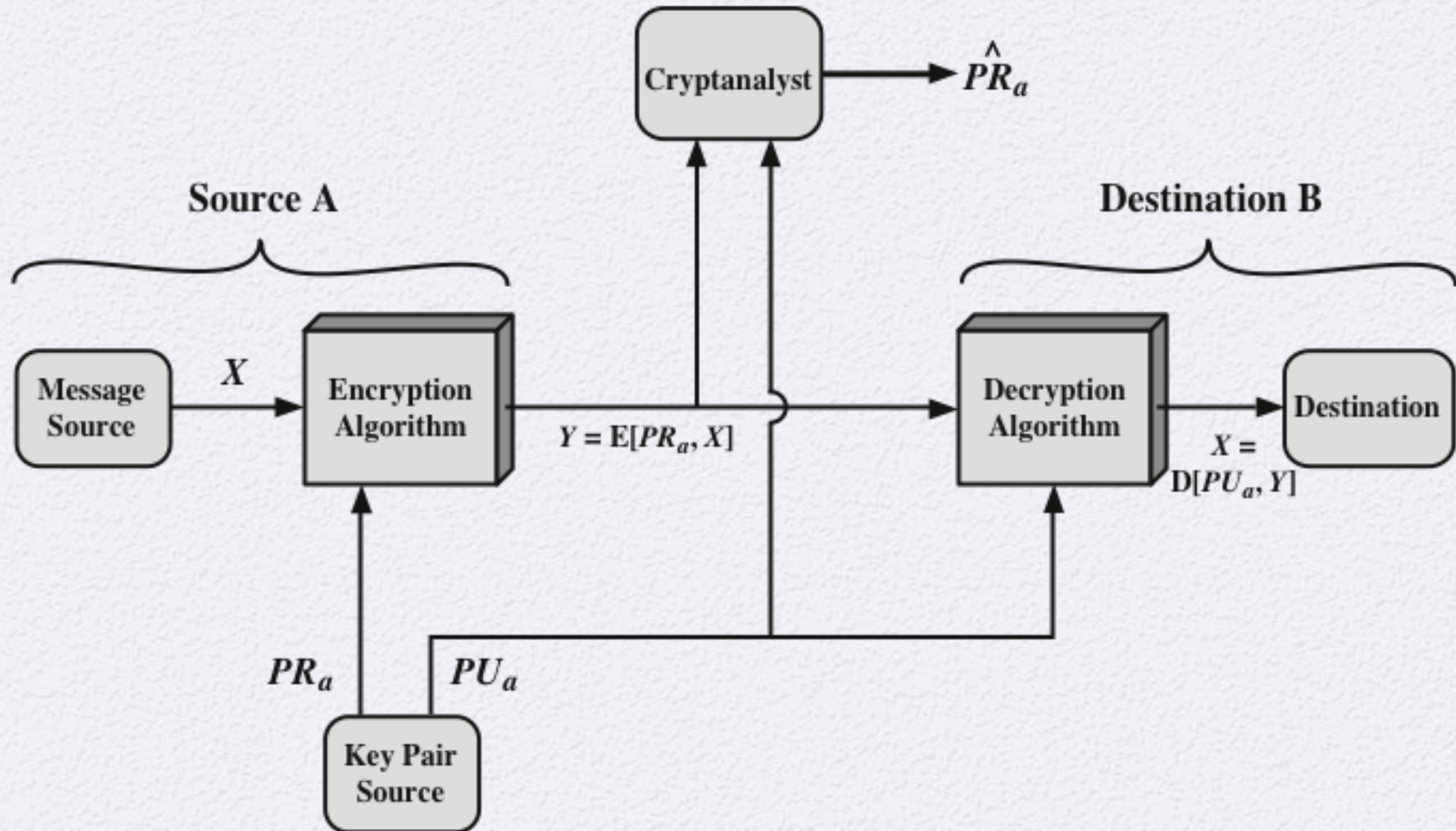


Figure 9.3 Public-Key Cryptosystem: Authentication

# Public-Key Cryptosystem: Authentication and Secrecy

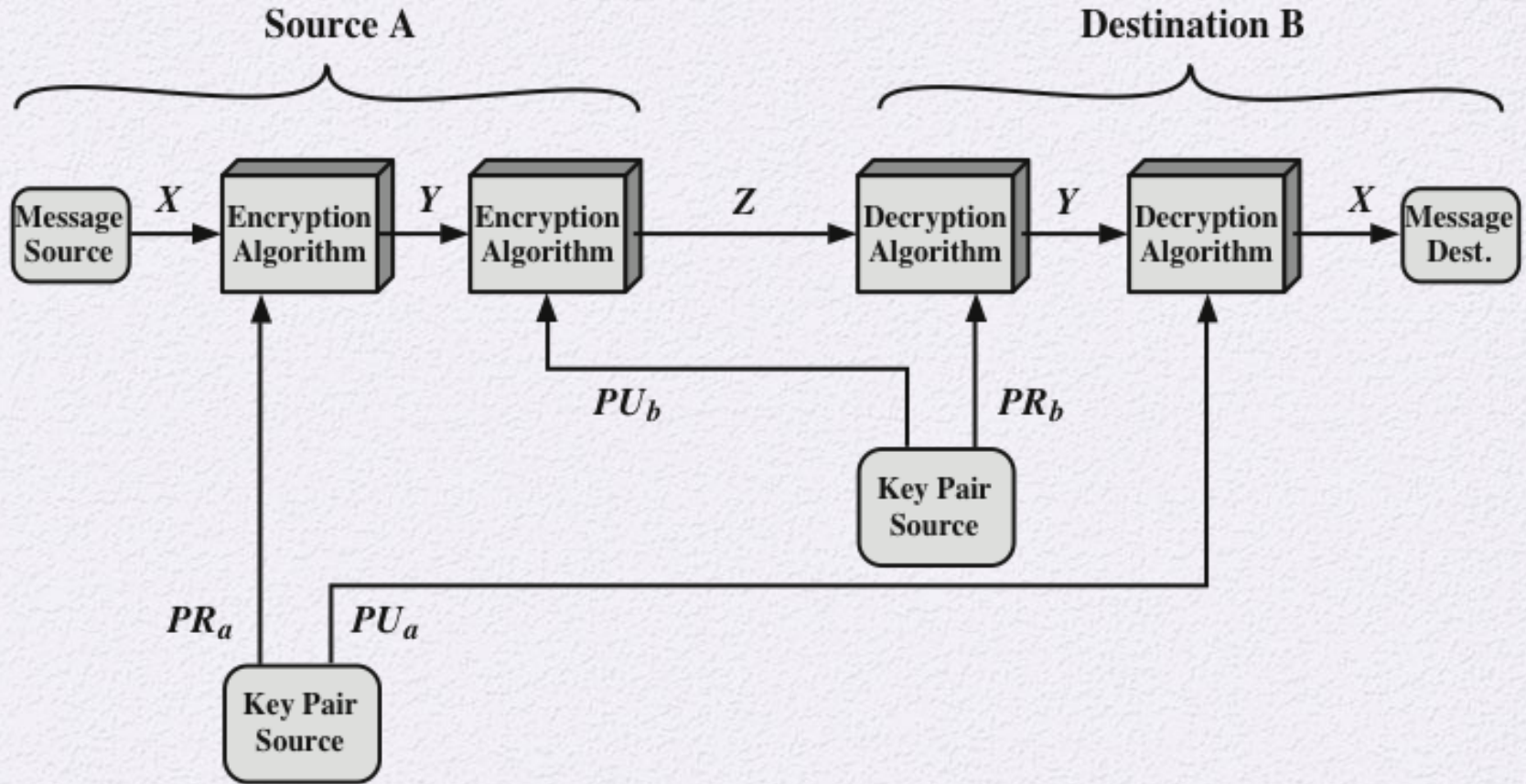
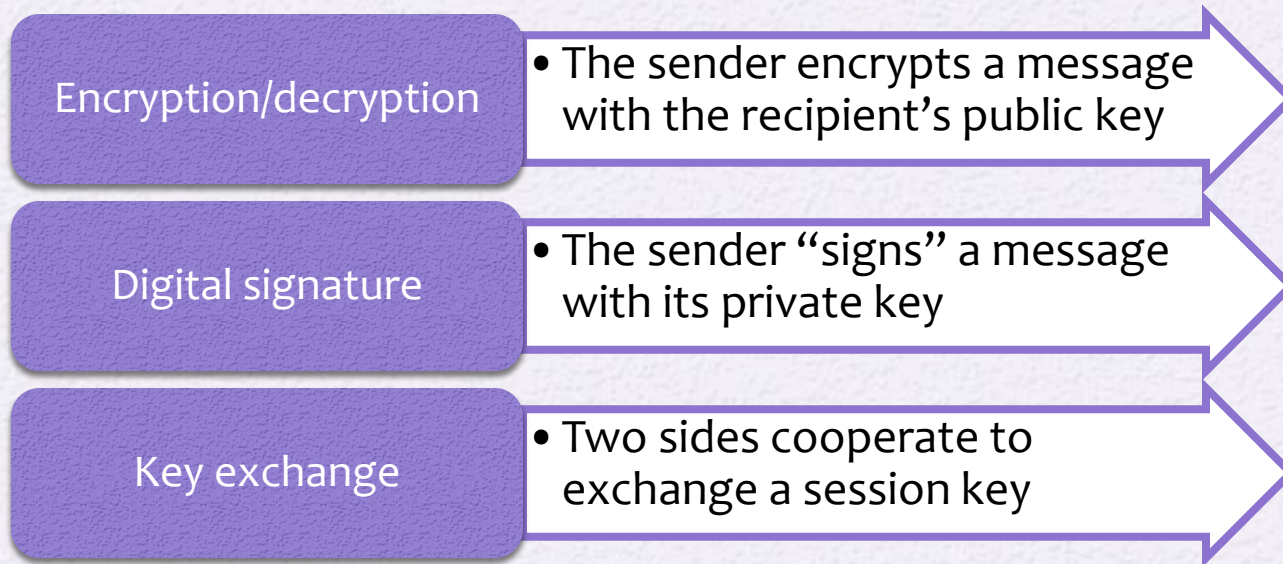


Figure 9.4 Public-Key Cryptosystem: Authentication and Secrecy



# 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:
  1. It is computationally easy for a party B to generate a pair (public-key  $PU_b$ , private key  $PR_b$ )
  2. It is computationally easy for a sender A, knowing the public key and the message to be encrypted, to generate the corresponding ciphertext
  3. It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message
  4. It is computationally infeasible for an adversary, knowing the public key, to determine the private key
  5. It is computationally infeasible for an adversary, knowing the public key and a ciphertext, to recover the original message
  6. The two keys can be applied in either order

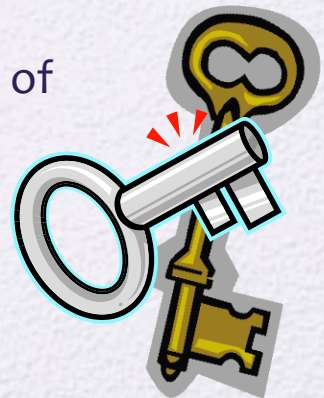
# Public-Key Requirements

- 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 bits to simple messages



# Rivest-Shamir-Adleman (RSA) Scheme

- 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



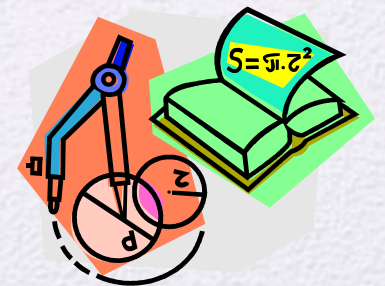
# RSA Algorithm

- 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  $n$



# RSA Key Setup

- each user generates a public/private key pair by:
- selecting two large primes at random -  $p, q$
- computing their system modulus  $n=p \cdot q$ 
  - note  $\phi(n) = (p-1)(q-1)$
- selecting at random the encryption key  $e$ 
  - where  $1 < e < \phi(n)$ ,  $\gcd(e, \phi(n)) = 1$
- solve following equation to find decryption key  $d$ 
  - $e \cdot d = 1 \pmod{\phi(n)}$  and  $0 \leq d \leq n$
- publish their public encryption key:  $PU = \{e, n\}$
- keep secret private decryption key:  $PR = \{d, n\}$

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 $d$	$d = e^{-1} \pmod{\phi(n)}$
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:	$M = C^d \pmod n$

**Figure 9.5 The RSA Algorithm**



# RSA Example - Key Setup

1. Select primes:  $p=17$  &  $q=11$
2. Compute  $n = pq = 17 \times 11 = 187$
3. Compute  $\phi(n) = (p-1)(q-1) = 16 \times 10 = 160$
4. Select  $e$ :  $\gcd(e, 160) = 1$ ; choose  $e=7$
5. Determine  $d$ :  $de \equiv 1 \pmod{160}$  and  $d < 160$  Value is  $d=23$   
since  $23 \times 7 = 161 = 10 \times 160 + 1$
6. Publish public key  $PU = \{7, 187\}$
7. Keep secret private key  $PR = \{23, 187\}$

# RSA Example - En/Decryption

- sample RSA encryption/decryption is:

- given message  $M = 88$

- encryption:

$$C = 88^7 \bmod 187 = 11$$

- decryption:

$$M = 11^{23} \bmod 187 = 88$$

# Example of RSA Algorithm

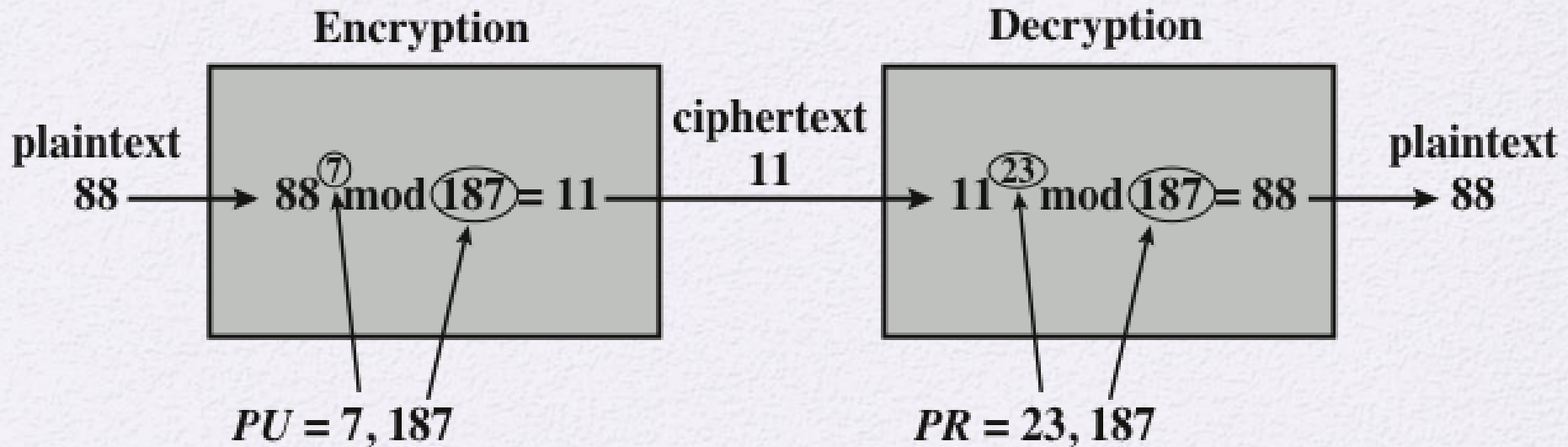


Figure 9.6 Example of RSA Algorithm



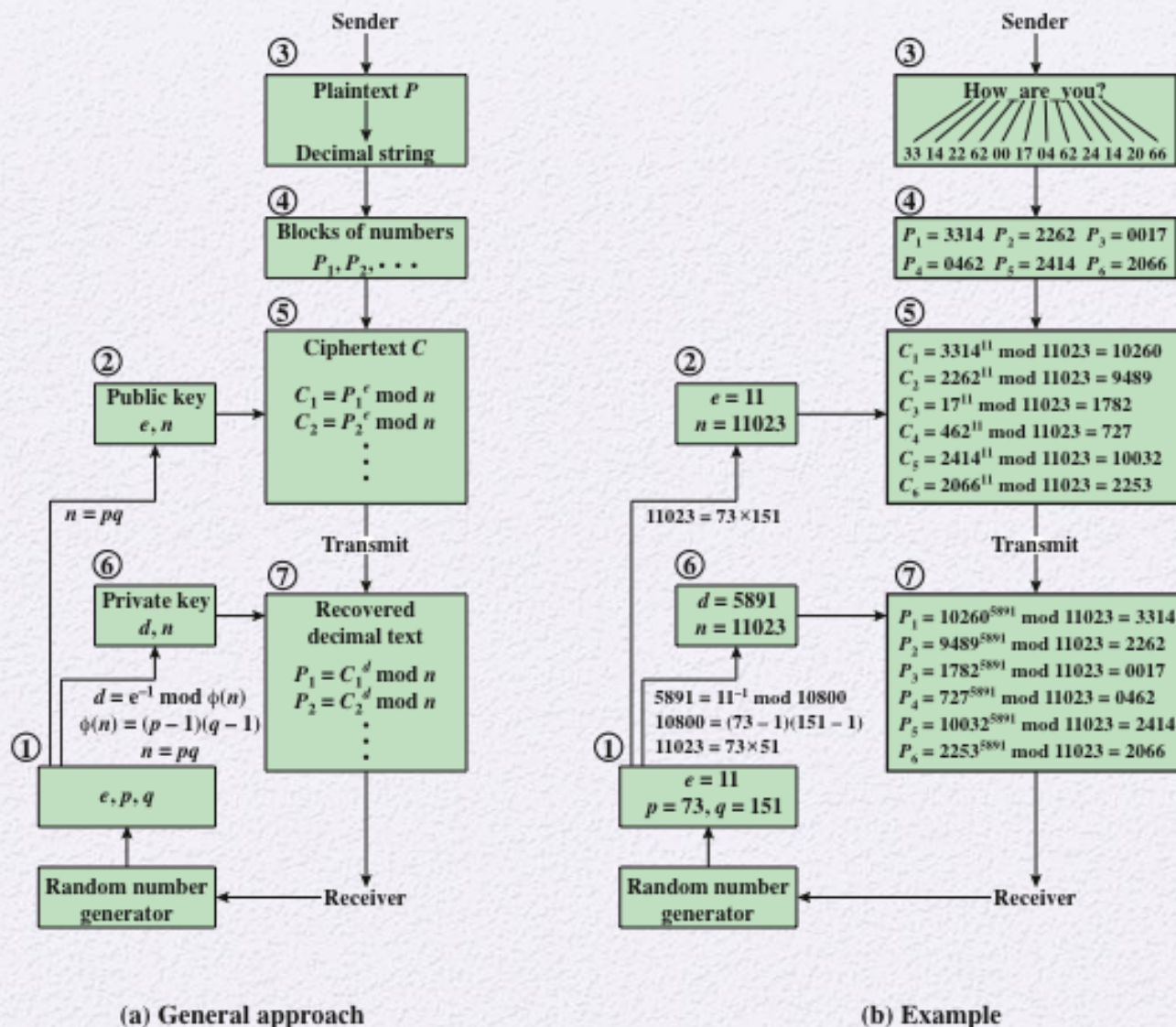


Figure 9.7 RSA Processing of Multiple Blocks

# 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

```

c ← 0; f ← 1
for i ← k downto 0
    do    c ← 2 × c
          f ← (f × f) mod n
    if    bi = 1
        then c ← c + 1
            f ← (f × a) mod n
return f

```

Note: The integer  $b$  is expressed as a binary number  $b_k b_{k-1} \dots b_0$

**Figure 9.8** Algorithm for Computing  $a^b \bmod n$



# Table 9.4

$i$	9	8	7	6	5	4	3	2	1	0
$b_i$	1	0	0	0	1	1	0	0	0	0
$c$	1	2	4	8	17	35	70	140	280	560
$f$	7	49	157	526	160	241	298	166	67	1

**Table 9.4** Result of the Fast Modular Exponentiation Algorithm for  $a^b \bmod n$ , where  $a = 7$ ,  $b = 560 = 1000110000$ , and  $n = 561$

# 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

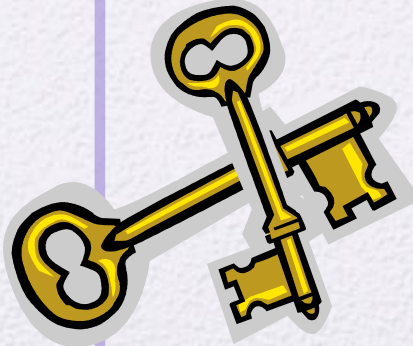
# Efficient Operation Using the Private Key

- 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





# Factoring Problem

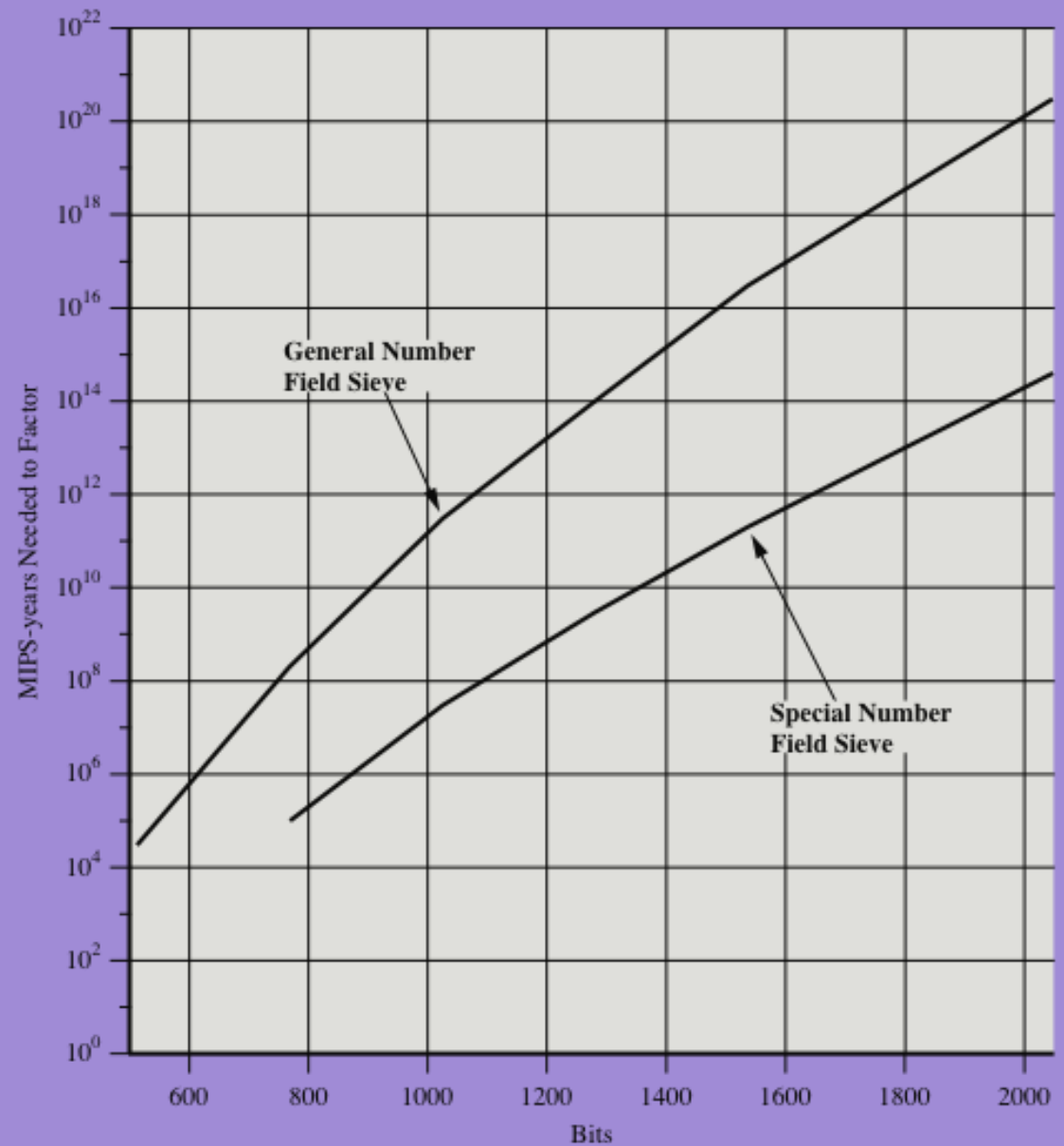
- We can identify three approaches to attacking RSA mathematically:
  1. Factor  $n$  into its two prime factors. This enables calculation of  $\phi(n) = (p - 1) \times (q - 1)$ , which in turn enables determination of  $d = e^{-1} \pmod{\phi(n)}$
  2. Determine  $\phi(n)$  directly without first determining  $p$  and  $q$ . Again this enables determination of  $d = e^{-1} \pmod{\phi(n)}$
  3. Determine  $d$  directly without first determining  $\phi(n)$

# Table 9.5

Number of Decimal Digits	Number of Bits	Date Achieved
100	332	April 1991
110	365	April 1992
120	398	June 1993
129	428	April 1994
130	431	April 1996
140	465	February 1999
155	512	August 1999
160	530	April 2003
174	576	December 2003
200	663	May 2005
193	640	November 2005
232	768	December 2009

Table 9.5 Progress in RSA Factorization

# MIPS-Years Needed to Factor



**Figure 9.9 MIPS-years Needed to Factor**



# Timing Attacks

- Paul Kocher, a cryptographic consultant, demonstrated that a snooper can determine a private key by keeping track of how long a computer takes to decipher messages
- Are applicable not just to RSA but to other public-key cryptography systems
- Are alarming for two reasons:
  - It comes from a completely unexpected direction
  - It is a ciphertext-only attack



# Countermeasures

## Constant exponentiation time

- Ensure that all exponentiations take the same amount of time before returning a result; this is a simple fix but does degrade performance

## Random delay

- Better performance could be achieved by adding a random delay to the exponentiation algorithm to confuse the timing attack

## Blinding

- Multiply the ciphertext by a random number before performing exponentiation; this process prevents the attacker from knowing what ciphertext bits are being processed inside the computer and therefore prevents the bit-by-bit analysis essential to the timing attack

# Fault-Based Attack

- An attack on a processor that is generating RSA digital signatures
  - Induces faults in the signature computation by reducing the power to the processor
  - The faults cause the software to produce invalid signatures which can then be analyzed by the attacker to recover the private key
- The attack algorithm involves inducing single-bit errors and observing the results
- While worthy of consideration, this attack does not appear to be a serious threat to RSA
  - It requires that the attacker have physical access to the target machine and is able to directly control the input power to the processor



# Chosen Ciphertext Attack (CCA)

- The adversary chooses a number of ciphertexts and is then given the corresponding plaintexts, decrypted with the target's private key
  - Thus the adversary could select a plaintext, encrypt it with the target's public key, and then be able to get the plaintext back by having it decrypted with the private key
  - The adversary exploits properties of RSA and selects blocks of data that, when processed using the target's private key, yield information needed for cryptanalysis
- To counter such attacks, RSA Security Inc. recommends modifying the plaintext using a procedure known as optimal asymmetric encryption padding (OAEP)

# Optimal Asymmetric Encryption Padding (OAEP)

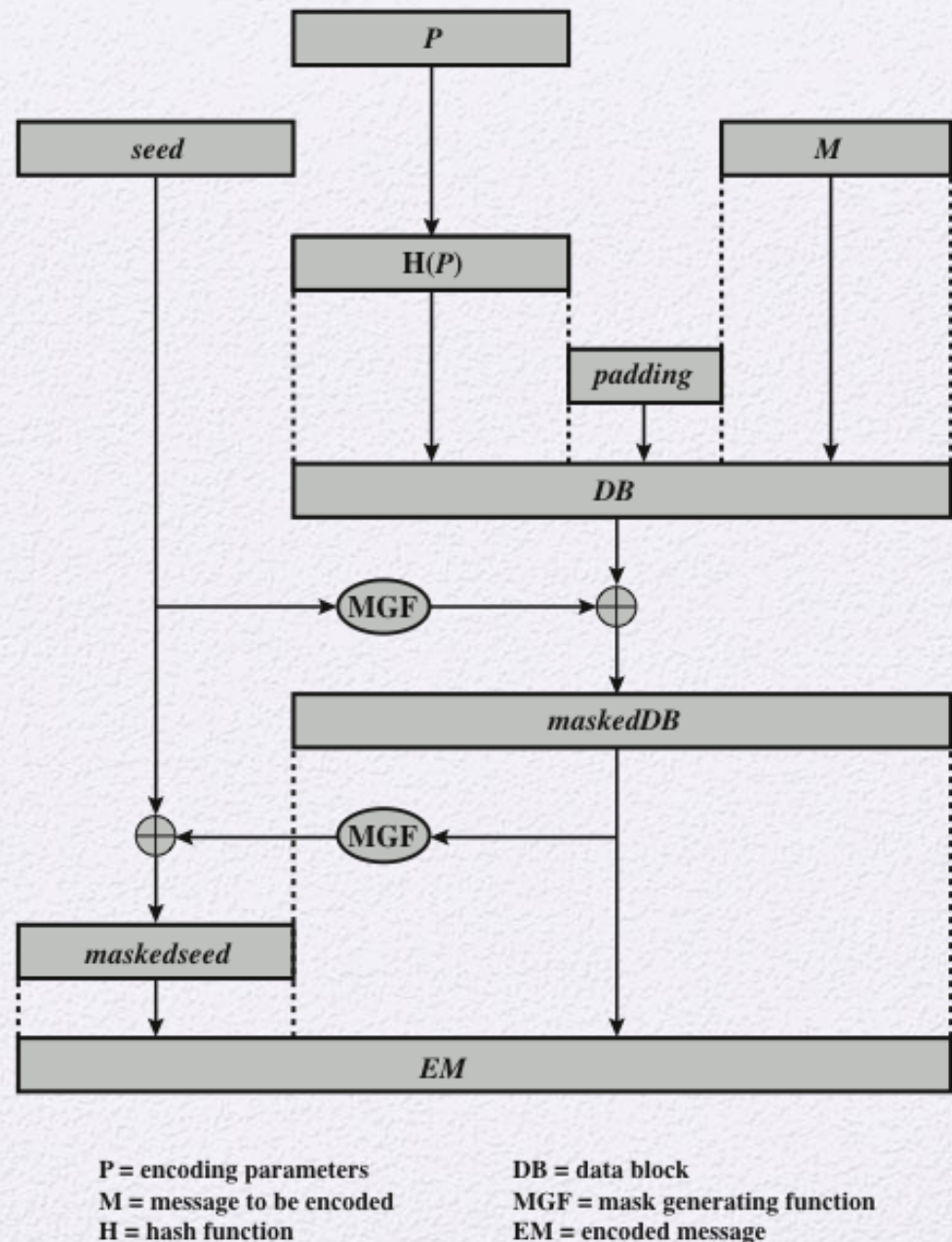


Figure 9.10 Encryption Using Optimal Asymmetric Encryption Padding (OAEP)



# Summary

- Public-key cryptosystems
- Applications for public-key cryptosystems
- Requirements for public-key cryptography
- Public-key cryptanalysis



- The RSA algorithm
  - Description of the algorithm
  - Computational aspects
  - Security of RSA