# Software Security and Dependability

ENGR5560G

## Lecture 08

# Public Key Cryptography

Dr. Khalid A. Hafeez

Spring, 25

# Public Key Cryptography

- **Asymmetric Keys**
  - Two related keys, a public key and a private key.
  - 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.

- **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.

# Misconceptions Concerning Public-Key Encryption

- Public-key encryption is more secure from cryptanalysis than symmetric encryption.
  - Actually, the security of any encryption scheme depends on:
    - The length of the key
    - The computational work involved in breaking a cipher.

- Public-key encryption is a general- purpose technique that has made symmetric encryption obsolete.
  - On the contrary, because of the computational overhead of current public-key encryption schemes, there seems no foreseeable likelihood that symmetric encryption will be abandoned.
- There is a feeling that key distribution is trivial when using public-key encryption, compared to the rather 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 difficulties 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**

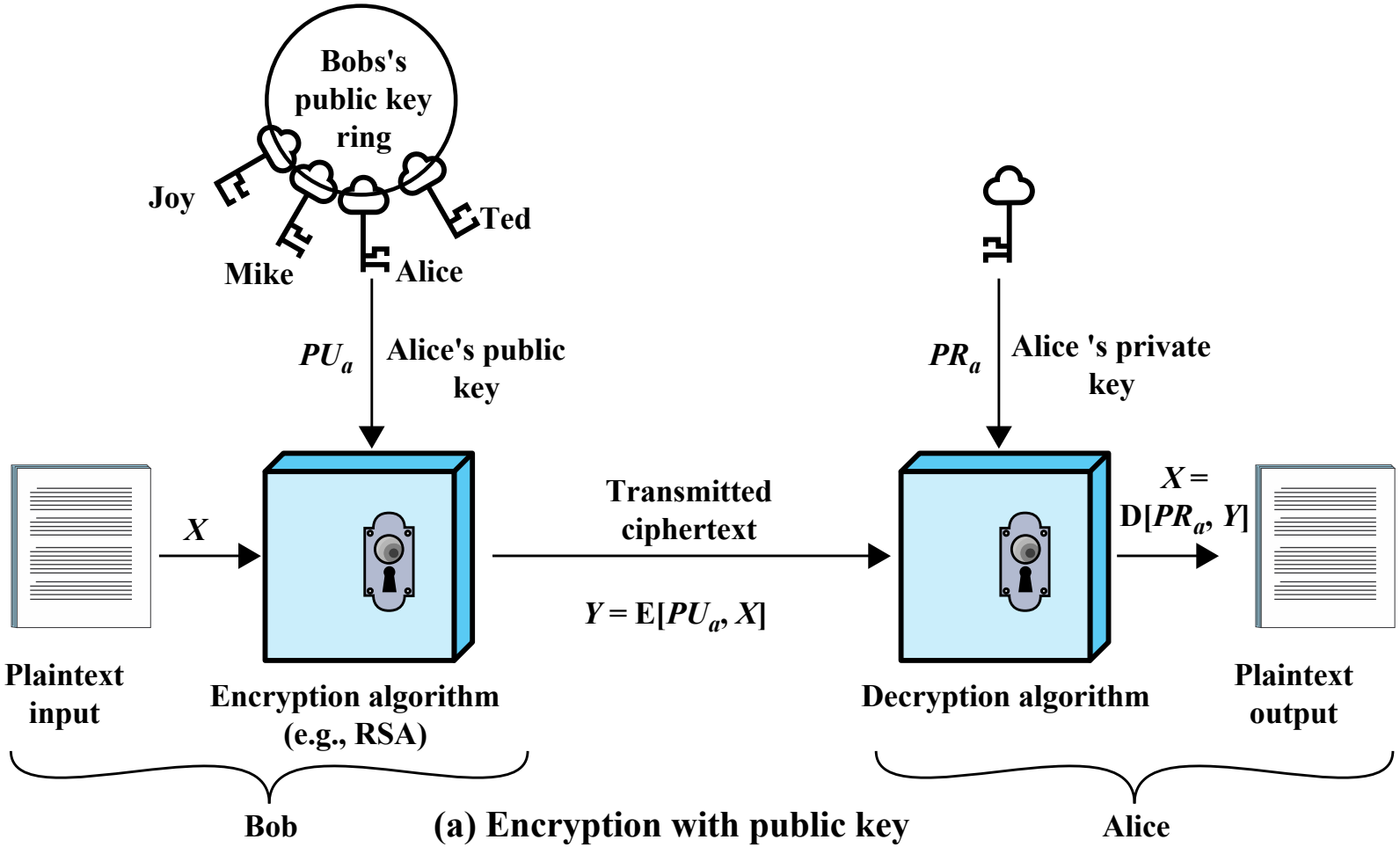
- Whitfield Diffie and Martin Hellman from Stanford University achieved a breakthrough in 1976 by producing a method that addressed both problems and was radically different from all previous approaches to cryptography

# Public-Key Cryptosystems

- A public-key encryption scheme has six ingredients:
  - Plaintext: This is the readable message or data that is fed into the algorithm as input.
  - Encryption algorithm: The encryption algorithm performs various transformations on the plaintext.
  - Public and private key: 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 encryption 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 uses the matching key to produce the original plaintext.

# Encryption with public key

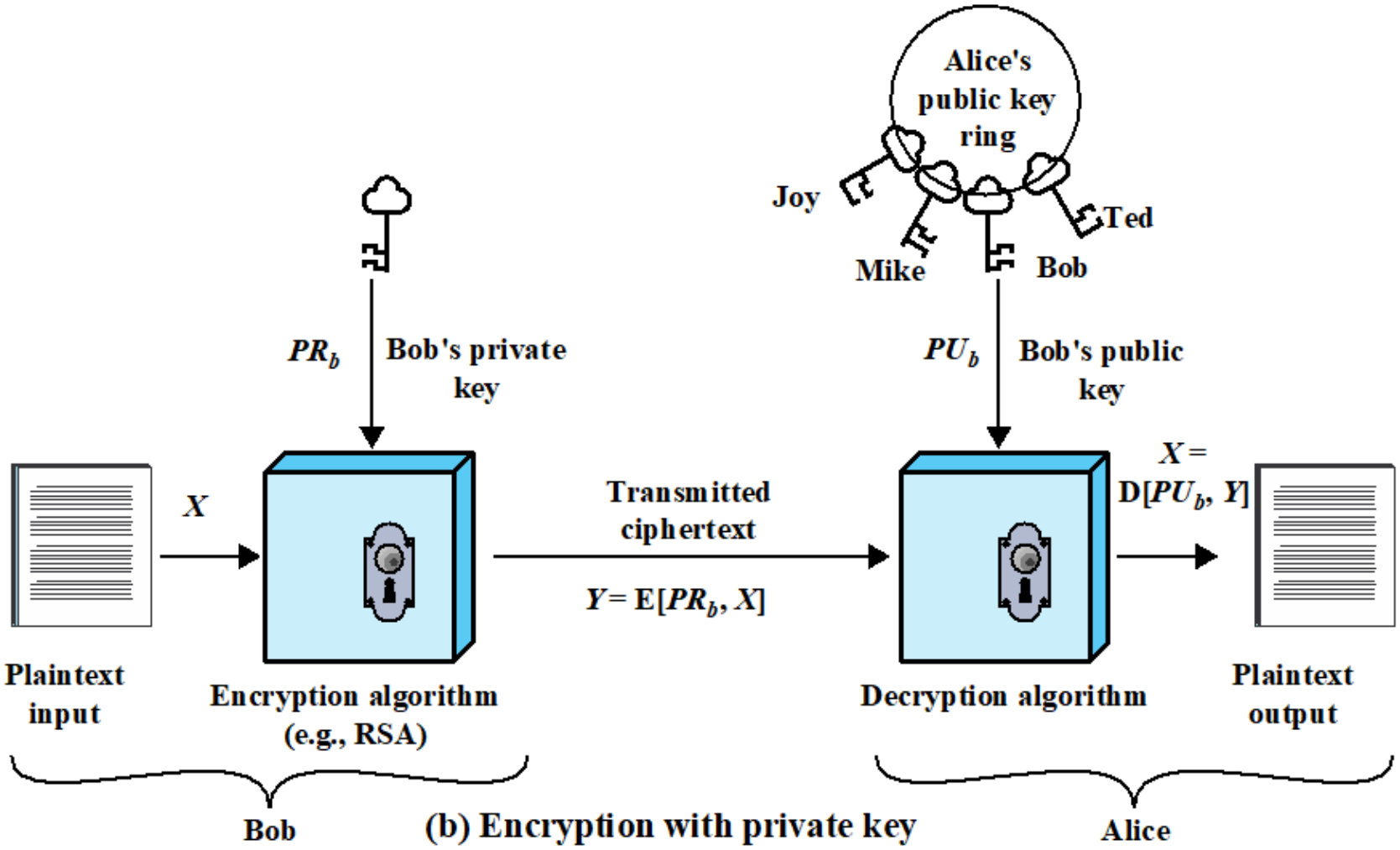
- Does this encryption provide:
  - Confidentiality?
  - Integrity?
  - Authenticity?




(a) Encryption with public key

# Encryption with private key

- Does this encryption provide:
  - Confidentiality?
  - Integrity?
  - Authenticity?

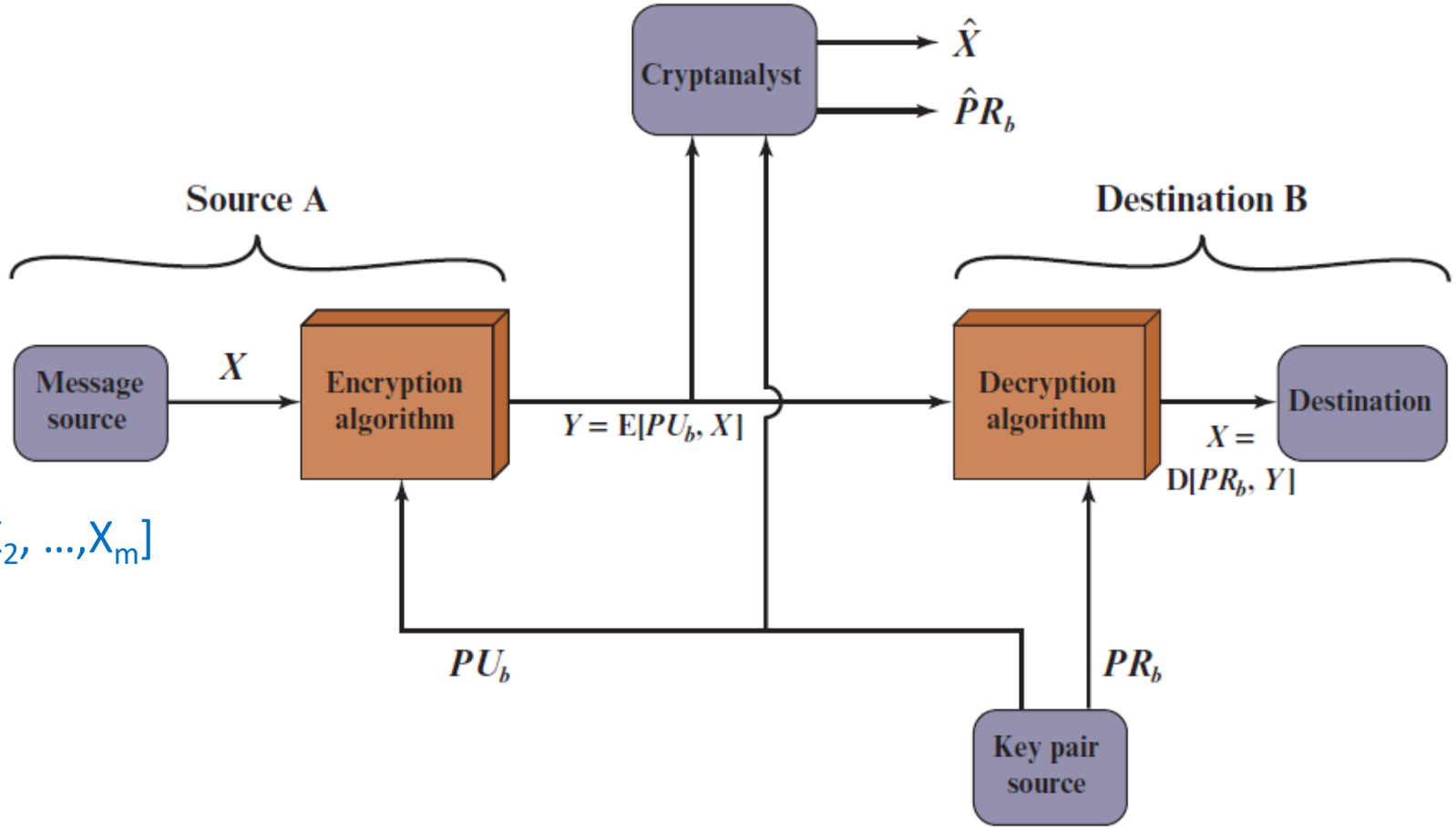



(b) Encryption with private key

# Conventional and Public-key Encryption

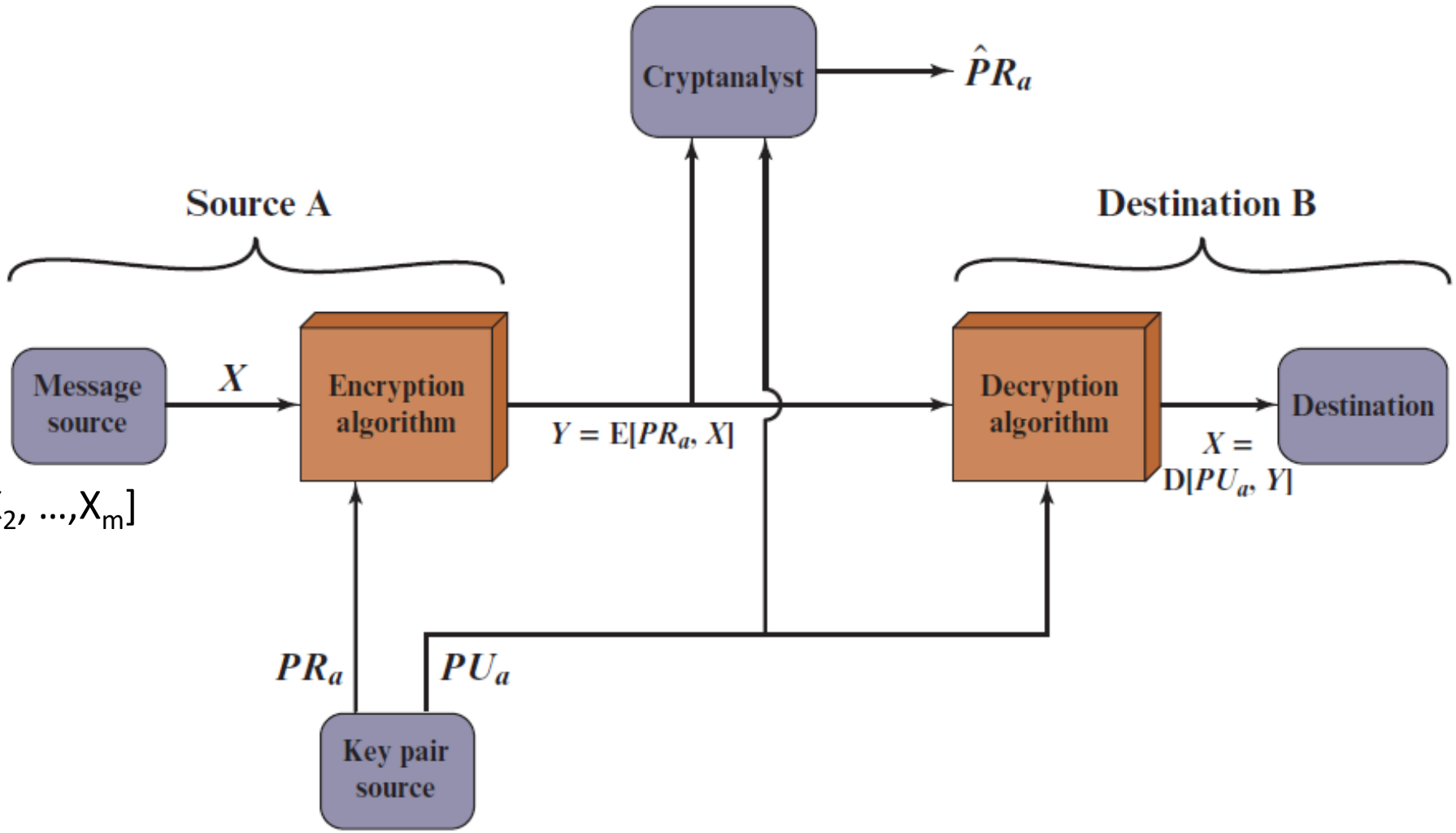
| Conventional Encryption | Public-Key Encryption |
|---|---|
| *Needed to Work:*<br>1. The same algorithm with the same key is used for encryption and decryption.<br>2. The sender and receiver must share the algorithm and the key. | *Needed to Work:*<br>1. One algorithm is used for encryption and a related algorithm for decryption with a pair of keys, one for encryption and one for decryption.<br>2. The sender and receiver must each have one of the matched pair of keys (not the same one). |
| *Needed for Security:*<br>1. The key must be kept secret.<br>2. It must be impossible or at least impractical to decipher a message if the key is kept secret.<br>3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key. | *Needed for Security:*<br>1. One of the two keys must be kept secret.<br>2. It must be impossible or at least impractical to decipher a message if one of the keys is kept secret.<br>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




$X = [X_1, X_2, \ldots, X_m]$

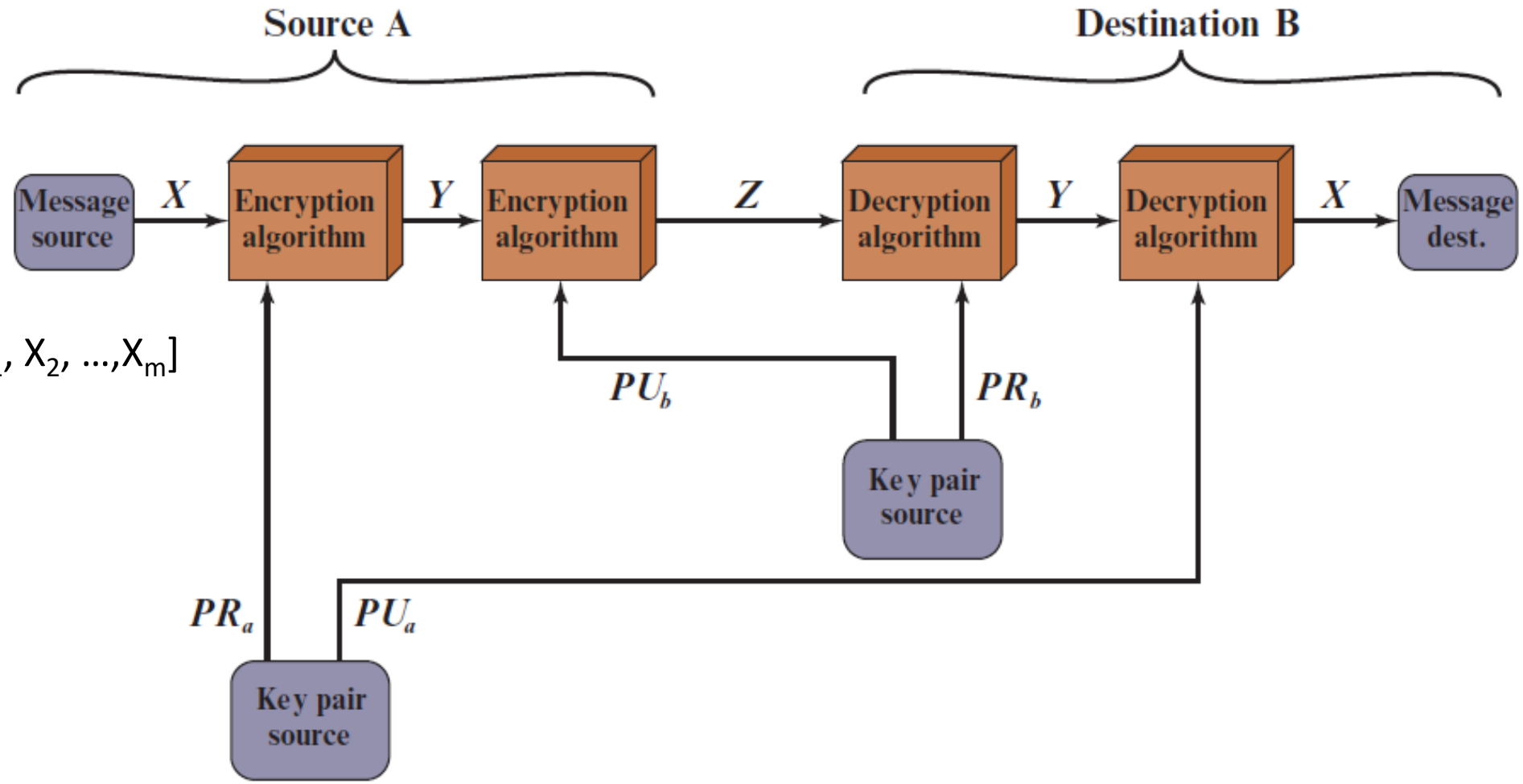
# Public-Key Cryptosystem: Authentication



# Public-Key Cryptosystem: Authentication and Confidentiality

$Z = E(PU_b, E(PR_a, X))$

$X = D(PU_a, D(PR_b, Z))$




$X = [X_1, X_2, \ldots, X_m]$

# Applications for Public-Key Cryptosystems

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

| | |
|---|---|
| **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 |

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

| 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 |

# 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

    $$C = E(PU_b, M)$$

  - It is computationally easy for the receiver B to decrypt the resulting ciphertext using the private key to recover the original message

    $$M = D(PR_b, C) = D[PR_b, E(PU_b, M)]$$

  - 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:

    $$M = D[PU_b, E(PR_b, M)] = D[PR_b, E(PU_b, M)]$$

# Public-Key Cryptanalysis

- A public-key encryption scheme is vulnerable to a brute-force attack
  - **Countermeasure**: use large keys
    - But 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
    - Hence, 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 (so all algorithms could be a suspect)
- Finally, there is a probable-message attack (message is small size)
  - This attack can be thwarted by appending some random bits to simple messages

# 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
- Plaintext is encrypted in blocks
- Block size are integers between 0 and $n - 1$ for some $n$
  - A typical size for $n$ is $n <= 2^{1024}$ or 309 decimal digits
- 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 $d$
  - Public key is $PU=\{e, n\}$
  - Private key is $PR=\{d, n\}$

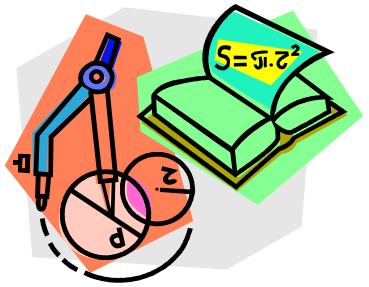
# RSA Algorithm Requirements

- **Main requirements:**
  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 $n$

- *The relationship $M^{ed} \bmod n = M$ holds if $e$ and $d$ are multiplicative inverses modulo $\phi(n)$, where $\phi(n)$ is the Euler totient function.*
  - *if $p$, $q$ are prime, $\phi(pq) = (p - 1)(q - 1) \rightarrow ed \bmod \phi(n) = 1$* (they are mutiplicative inverses *mod* $\phi(n)$)

- **The ingredients of the RSA scheme are:**

| | |
|---|---|
| $p$, $q$, two prime numbers | (private, chosen) |
| $n = pq$ | (public, calculated) |
| $e$, with $\gcd(\phi(n), e) = 1$; $1 < e < \phi(n)$ | (public, chosen) |
| $d \equiv e^{-1} \pmod{\phi(n)}$ | (private, calculated) |



# The RSA Algorithm – example 1

1. Select two prime numbers, $p = 17$ and $q = 11$
2. Calculate $n = pq = 17 * 11 = 187$.
3. Calculate $\phi(n) = (p - 1)(q - 1) = 16 * 10 = 160$
4. Select e such that e is relatively prime to $\phi(n) = 160$ and less than $\phi(n)$; we choose $e = 7$.
5. Determine $d$ such that $de \bmod 160 = 1$ and $d < 160$. The correct value is $d = 23$, because $23 \times 7 = 161 = (1 \times 160) + 1$. (use extended Euclid's algorithm)
6. The resulting keys are
   - public key PU = {7, 187}
   - private key PR = {23, 187}.

**Let M = 88**

Encryption: calculate $C = 88^7 \bmod 187$.

$C = [(88^3 \bmod 187) * (88^3 \bmod 187) * (88^1 \bmod 187)] \bmod 187$

$= [44*44*88] \bmod 187$

$= 11$

Decryption: $M = 11^{23} \bmod 187$:

$M = [(11^7 \bmod 187) * (11^8 \bmod 187) * (11^8 \bmod 187)] \bmod 187$

$= (88 * 33 * 33) \bmod 187$

$= 88$

**Key Generation**

| | |
|---|---|
| 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 $d$ | $de \bmod \phi(n) = 1$ |
| Public key | $KU = \{e, n\}$ |
| Private key | $KR = \{d, n\}$ |

**Encryption**

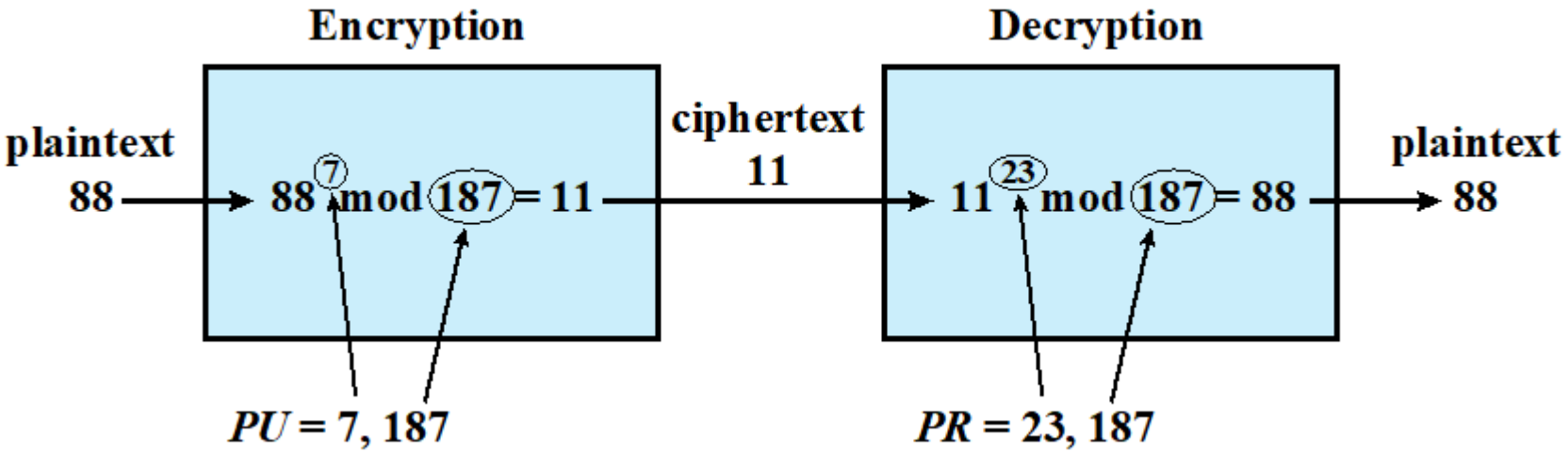
| | |
|---|---|
| Plaintext: | $M < n$ |
| Ciphertext: | $C = M^e \pmod n$ |

**Decryption**

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

# The RSA Algorithm – example 1

Two prime numbers are **p=17** and **q=11**



# The RSA Algorithm – example 2

- If P = 3 and q = 11, calculate public key and private key.
- Consider plaintext M= 5. Do the encryption and decryption using your public key and private key.
- Solution:

1. Select two prime numbers, $p = 3$ and $q = 11$
2. Calculate $\boldsymbol{n} = pq = 33$
3. Calculate $\phi(n) = (p - 1)(q - 1) = 20$
4. Select e <20 and relatively prime to 20 → e =
5. Select $d$ <20 such that $de \bmod 20 = 1$ → d =
6. The resulting keys are
   - public key PU = { , 33}
   - private key PR = { , 33}.

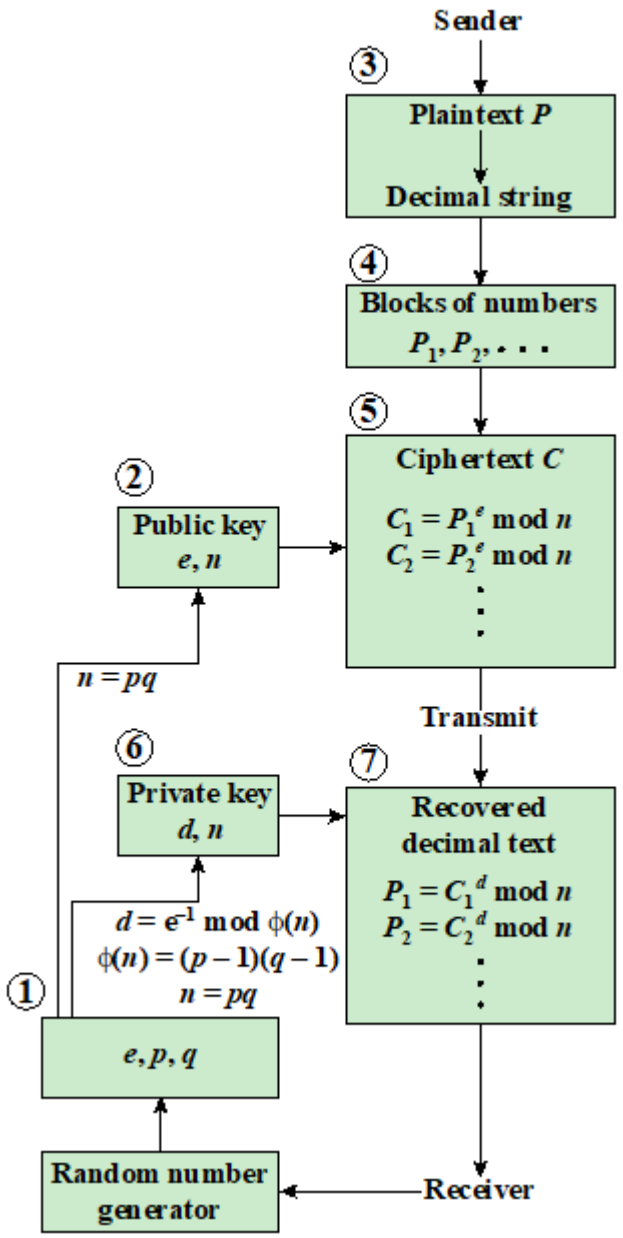
Let M = 5.
Encryption: calculate C

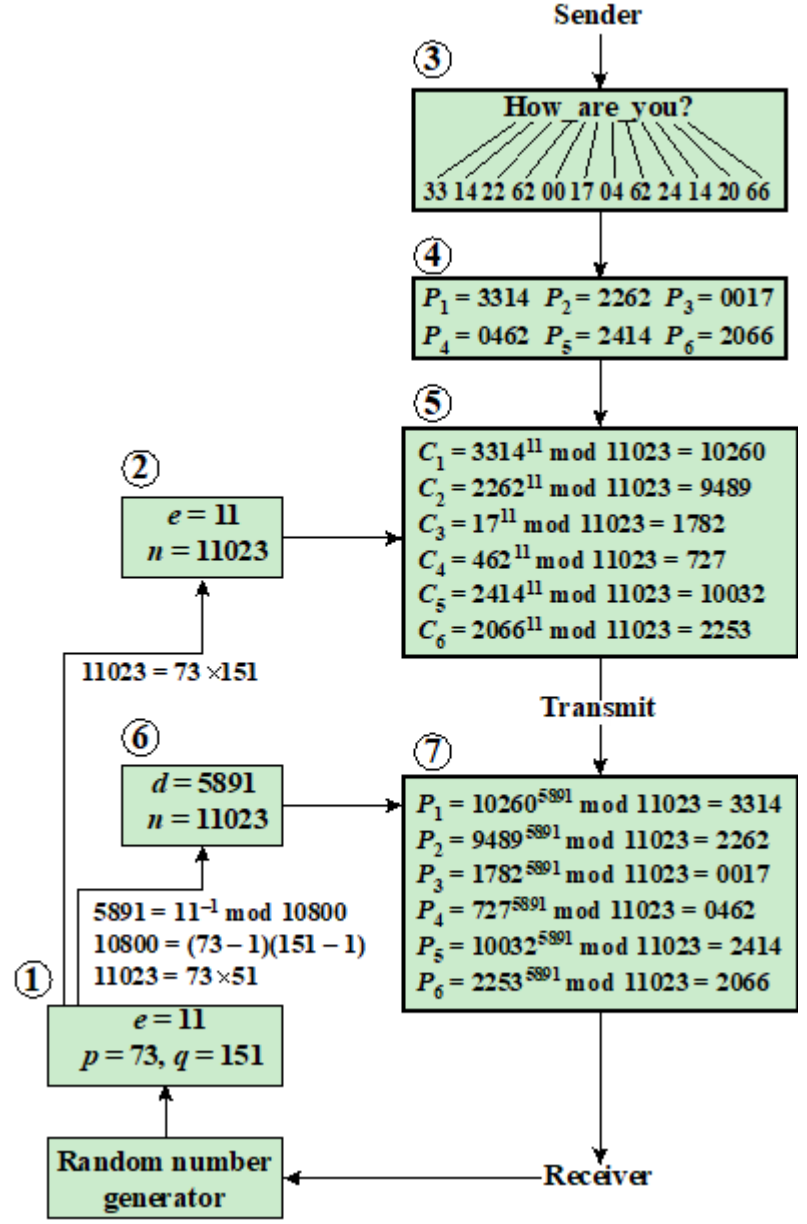
Decryption: M =

# RSA -Processing of Multiple Blocks example

- Here the plaintext is an alphanumeric string.
  Each plaintext symbol is assigned a unique code of two decimal digits (e.g., a = 00, A = 26)

  A plaintext block consists of four decimal digits, or two alphanumeric characters



# 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

Example:

$x^4$

$x^{16}$

$x^{11}$

# Algorithm for Computing $a^b \bmod n$

- **Computing $a^b \bmod n$**
  - *Note:* The integer $b$ is expressed as a binary number $b_k b_{k-1} \ldots b_0$
  - *Note:*

```
f ← 1
For i ← k to 0 do
    f ← (f × f) mod n
    if b_i = 1 then
        f ← (f × a) mod n
return f
```

# Fast Modular Exponentiation Algorithm – Example 1

Calculate: $a^b \bmod n$, where a = 7, b = 560 = 1000110000, and n = 561

| *i* | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|
| $b_i$ | 1 | 0 | 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| *f* | 7 | 49 | 157 | 526 | 160 | 241 | 298 | 166 | 67 | 1 |

$f \leftarrow 1$
For $i \leftarrow k$ to $0$ do
    $f \leftarrow (f \times f) \bmod n$
    if $b_i = 1$ then
        $f \leftarrow (f \times a) \bmod n$
return $f$

# Fast Modular Exponentiation Algorithm – Example 2

Find $11^{23} \bmod 187$

a=11
b=23 = 10111
n=187

```
f ← 1
For i ← k to 0 do
    f ← (f × f) mod n
    if b_i = 1 then
        f ← (f × a) mod n
return f
```

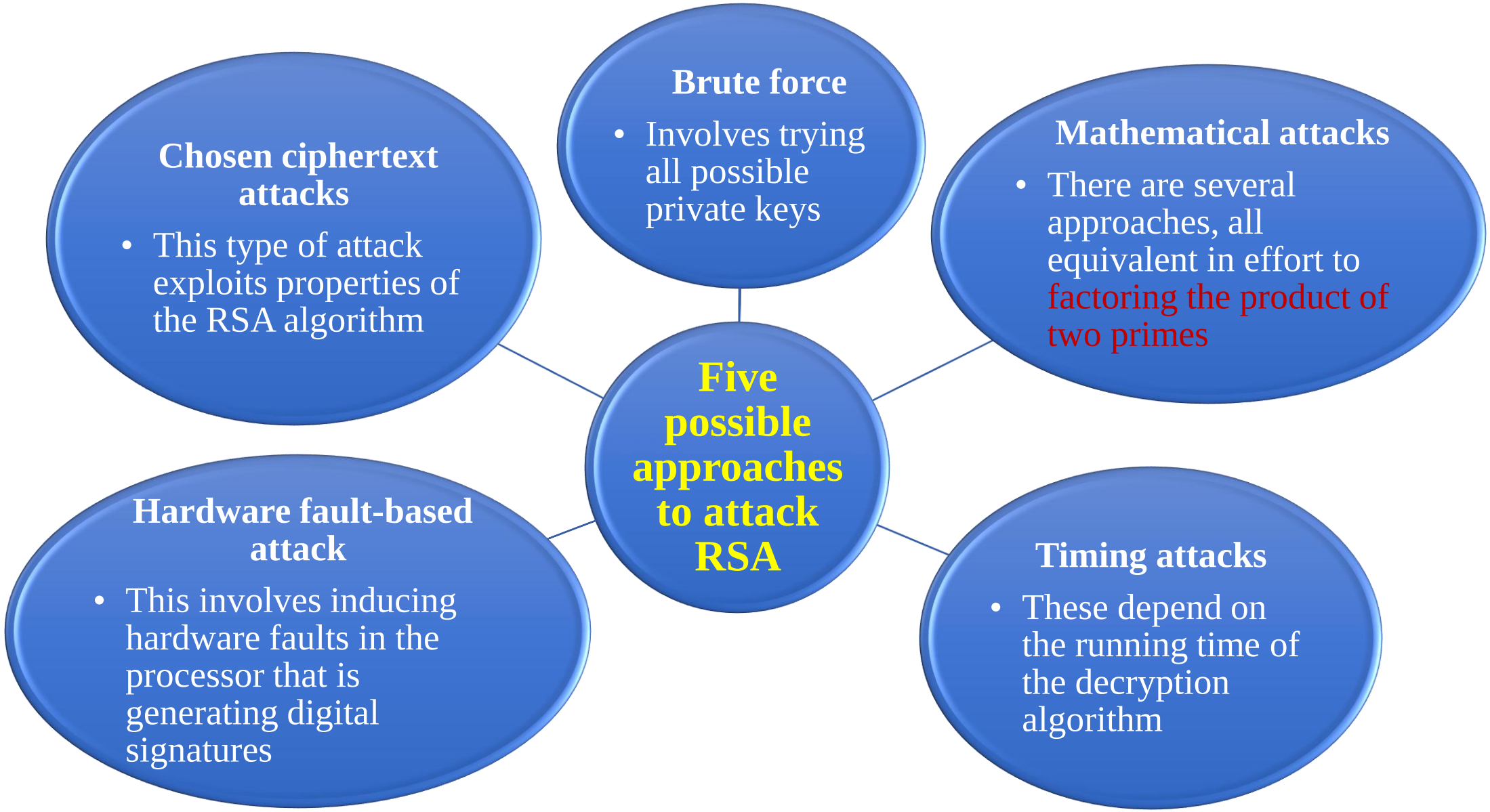
| $i$ | | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|
| $b_i$ | | 1 | 0 | 1 | 1 | 1 |
| $f$ | 1 | 11 | 121 | 44 | 165 | 88 |

# 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

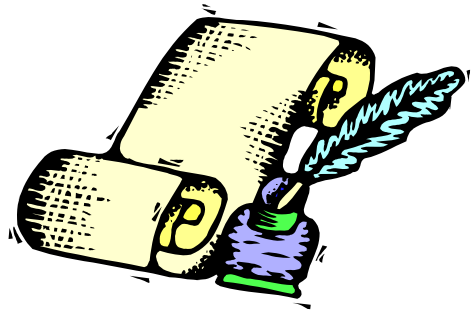
- The most common choice is 65537 ($2^{16} + 1$)
  - Two other popular choices are $e$=3 and $e$=17
  - Each of these choices has only two 1 bits, so the number of multiplications required to perform exponentiation is minimized
  - With a very small public key, such as $e$ = 3, RSA becomes vulnerable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

# The Security of RSA



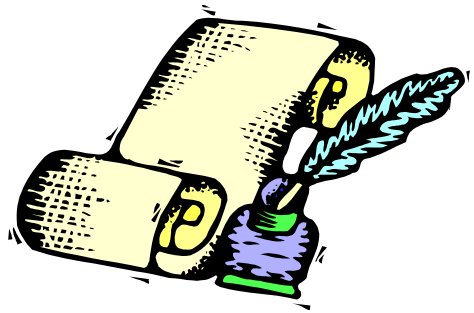
# Digital Signature and Management

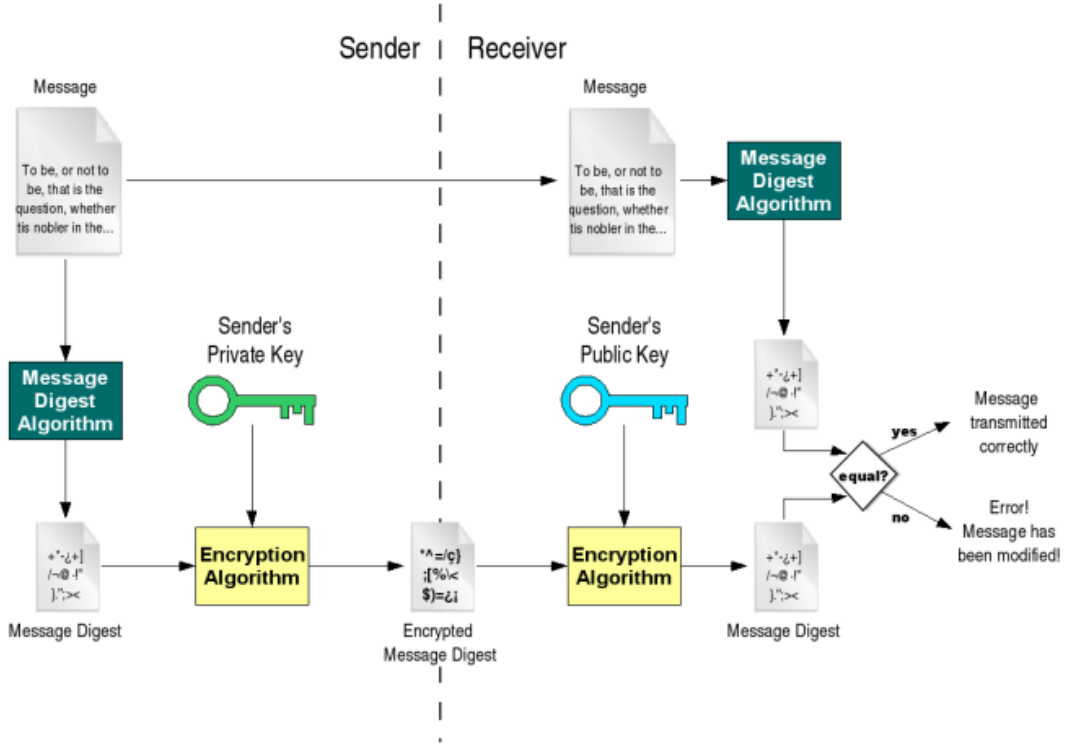
- Public-key algorithms used in two categories of applications:
  - Digital signatures
  - Key management and distribution:



# Digital Signature and Management

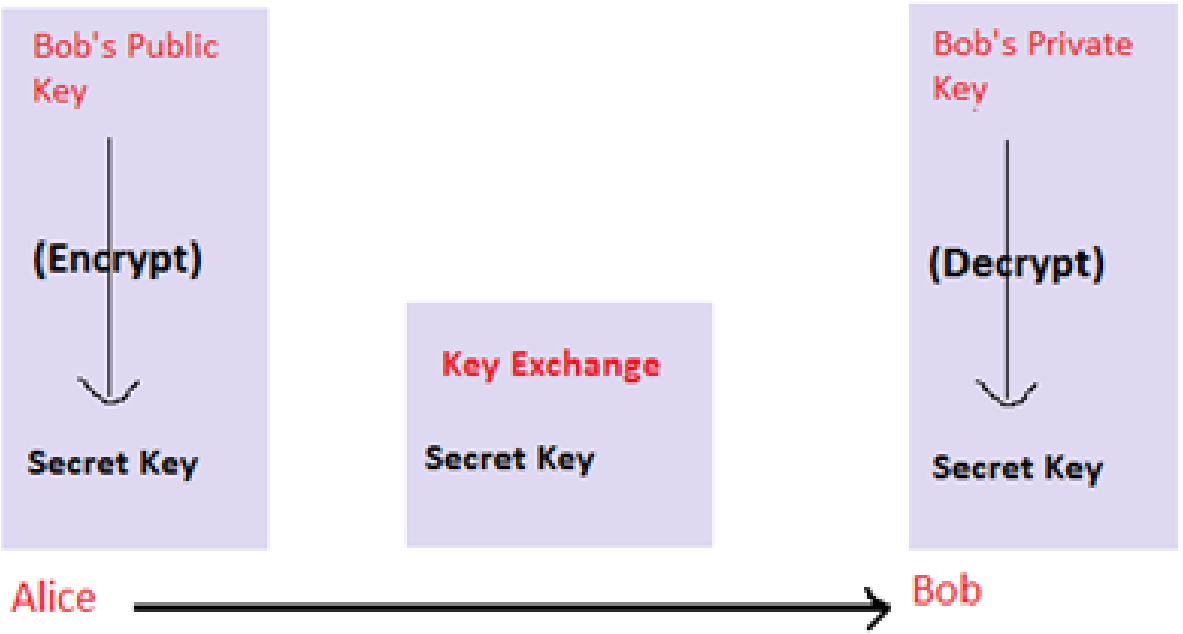
- Digital Signatures
  - Used for authenticating both source and data integrity
  - Created by encrypting hash code with private key
  - Does not provide confidentiality
  - Even in the case of complete encryption
    - Message is safe from alteration but not eavesdropping





# Digital Signature and Management

- Symmetric Key Exchange using Public-key Encryption



# Digital Signature and Management

- Digital Envelopes
  - Protects a message without needing to first arrange for sender and receiver to have the same secret key
  - Equates to the same thing as a sealed envelope containing an unsigned letter