Public key cryptography / Asymmetric key cryptography

Public key encryption structure

- First proposed by Diffie & Hellman – 1976
- Algorithms are based on mathematical functions & not on bit patterns
- Uses 2 separate keys
- Plain text, EA , public & private key, cipher text, DA

Table 9.1 Terminology Related to Asymmetric Encryption

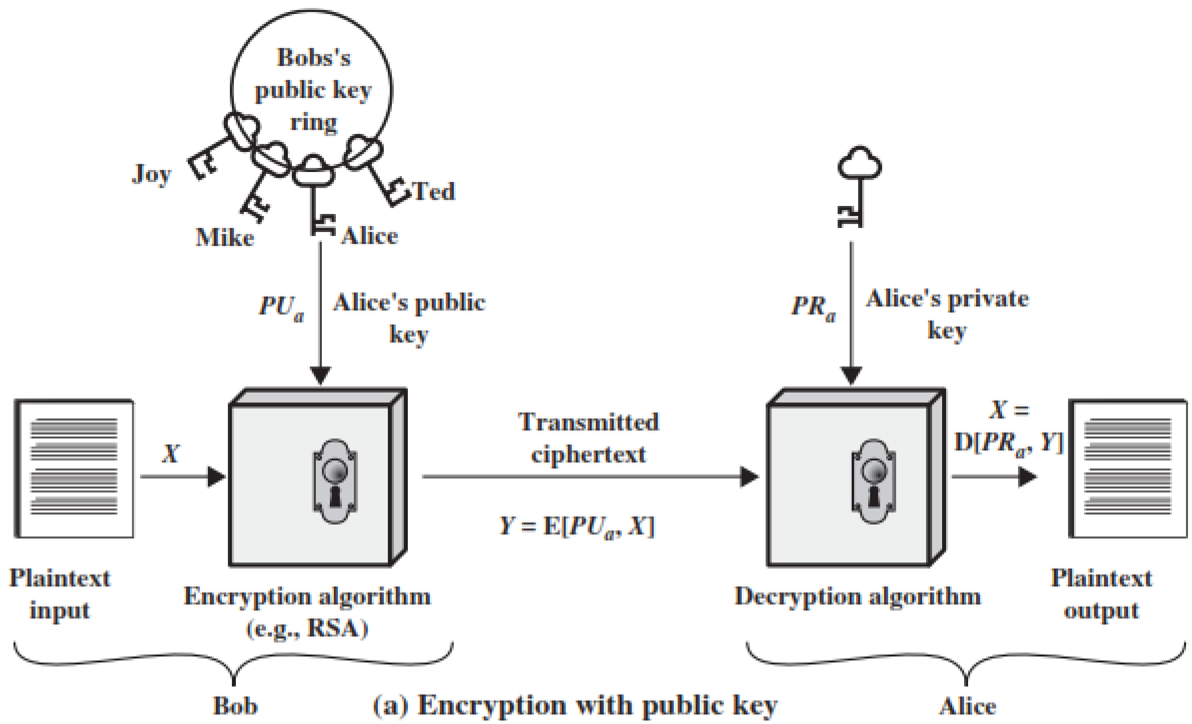
Asymmetric Keys Two related keys, a public key and a private key, that are used to perform complementary operations, such as encryption and decryption or signature generation and signature verification.
Public Key Certificate A digital document issued and digitally signed by the private key of a Certification 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.

characteristics

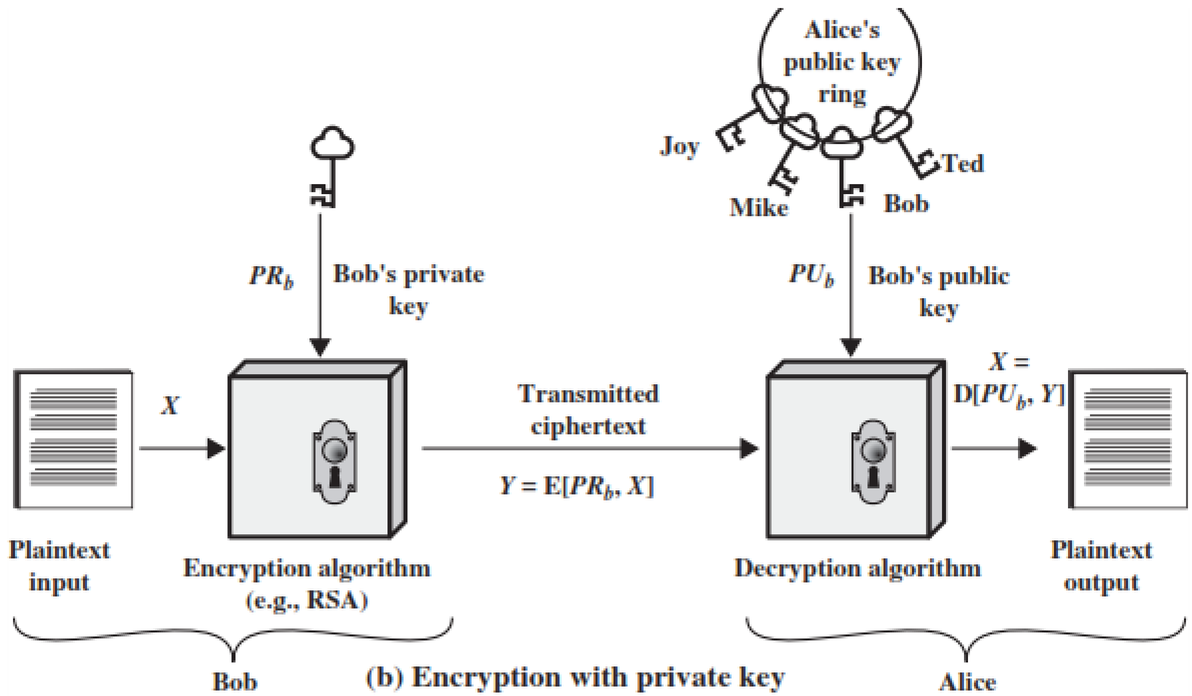
- It is computationally infeasible to determine the decryption key given only knowledge of the cryptographic algorithm and the encryption key.

- Either of the two related keys can be used for encryption, with the other used for decryption.

A **public-key encryption** scheme has six ingredients (Figure 9.1a; compare with Figure 2.1).



(a) Encryption with public key



(b) Encryption with private key

6 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 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.

Steps

1. Each user generates a pair of keys to be used for the encryption and decryption of messages.
2. Each user places one of the two keys in a public register or other accessible file. This is the public key. The companion key is kept private. As Figure 9.1a suggests, each user maintains a collection of public keys obtained from others.
3. If Bob wishes to send a confidential message to Alice, Bob encrypts the message using Alice's public key.
4. When Alice receives the message, she decrypts it using her private key. No other recipient can decrypt the message because only Alice knows Alice's private key.

Conventional Encryption	Public-Key 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 no other information is available. 3. Knowledge of the algorithm plus samples of ciphertext must be insufficient to determine the key.	*Needed to Work:* 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 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 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 cryptosystem-secrecv

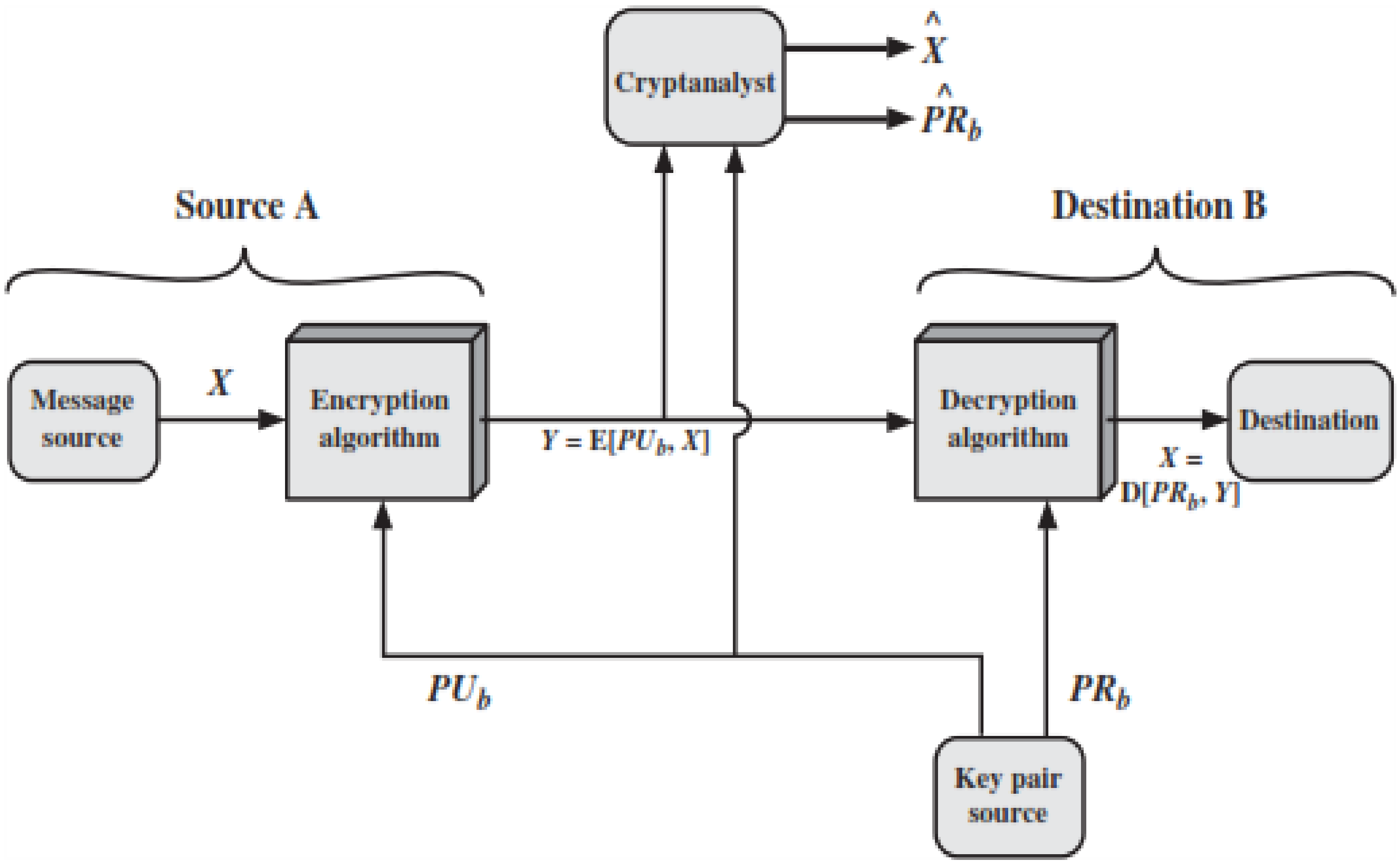


Figure 9.2 Public-Key Cryptosystem: Secrecy

Public key cryptosystem-authentication

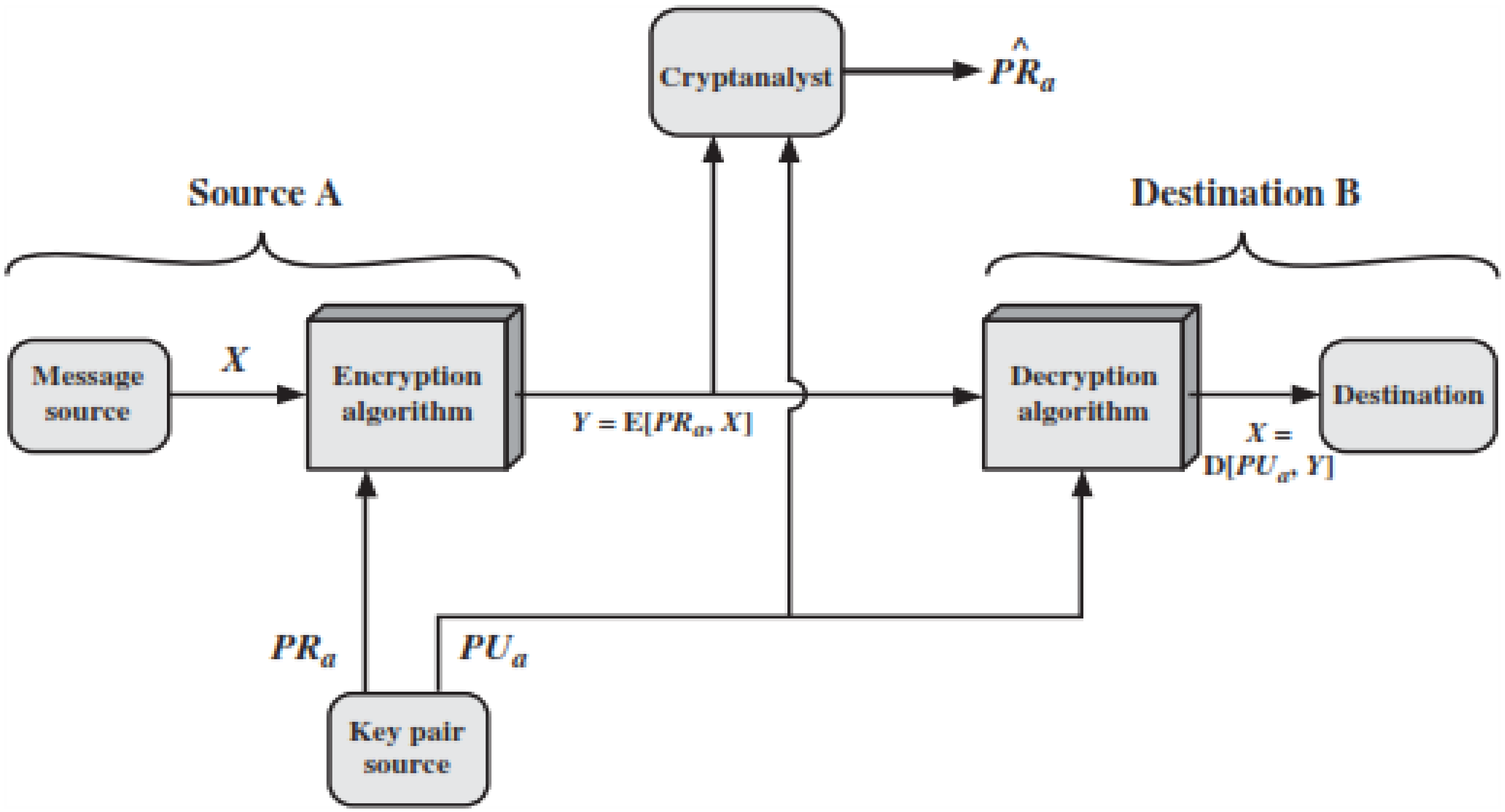
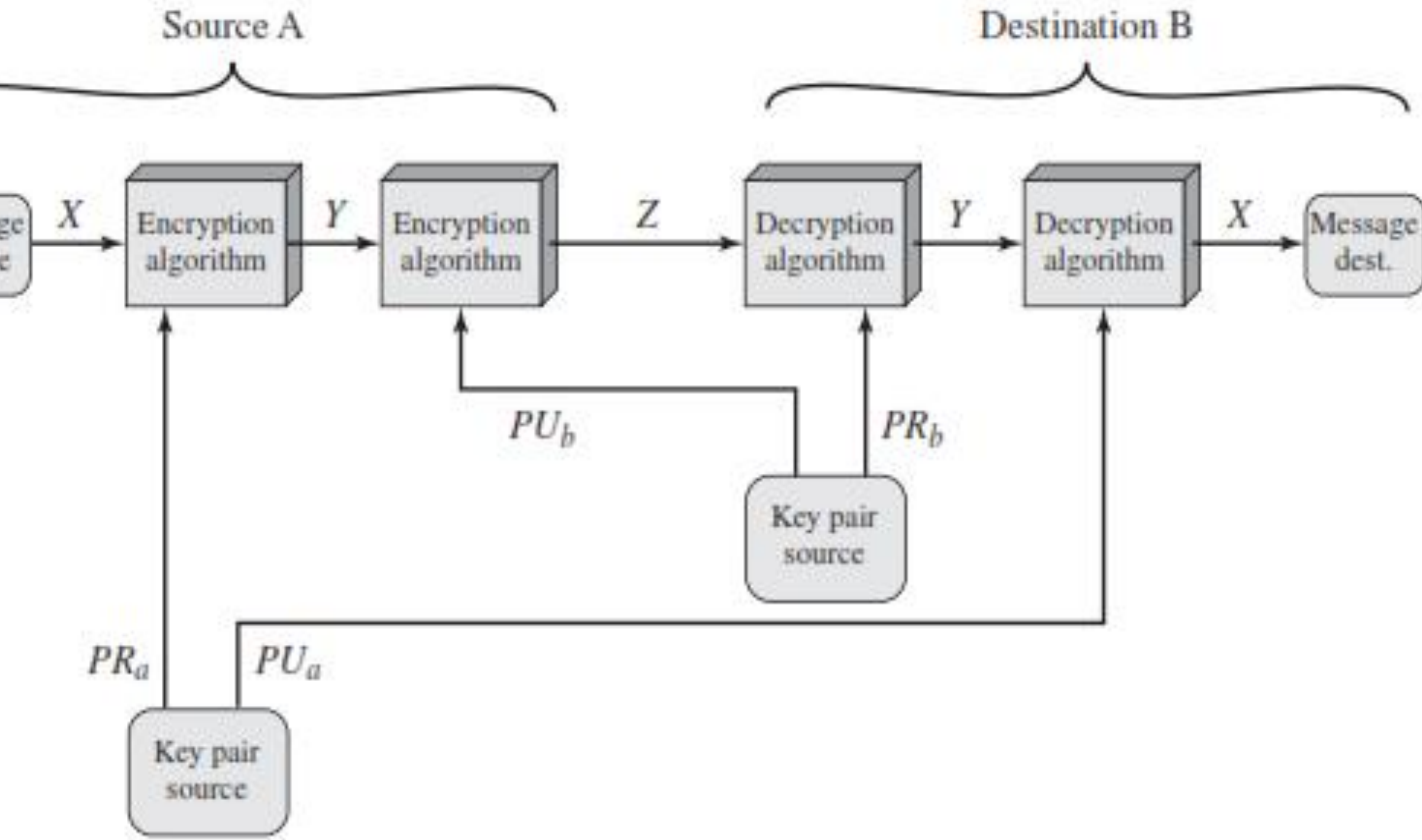


Figure 9.3 Public-Key Cryptosystem: Authentication

Public key cryptosystem-



e 9.4 Public-Key Cryptosystem: Authentication and Secrecy

Applications of public key cryptosystems

1. *Encryption / Decryption*

   Sender encrypts a message with the recipient’s public key

2. *Digital signature*

   sender signs a message with private key

3. *Key exchange*

   two sides cooperate to exchange a session key

Applications of public key cryptosystems

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

Requirements of PKC

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, M, to generate the corresponding ciphertext:

$$C = \mathrm{E}(PU_b, M)$$

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

$$M = \mathrm{D}(PR_b, C) = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$$

4. It is computationally infeasible for an adversary, knowing the public key, PU_b, to determine the private key, PR_b.
5. It is computationally infeasible for an adversary, knowing the public key, PU_b, and a ciphertext, C, to recover the original message, M.

We can add a sixth requirement that, although useful, is not necessary for all public-key applications:

6. The two keys can be applied in either order:

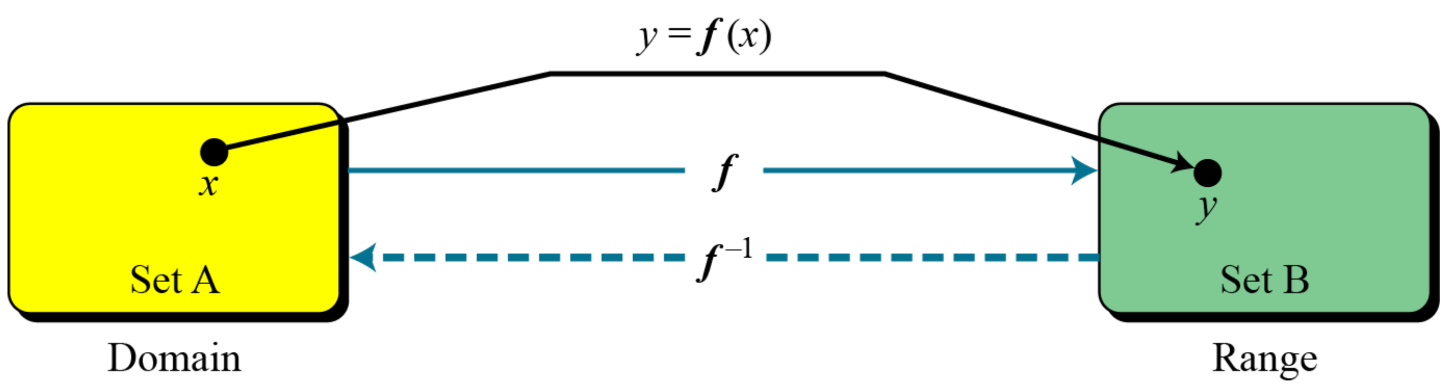
$$M = \mathrm{D}[PU_b, \mathrm{E}(PR_b, M)] = \mathrm{D}[PR_b, \mathrm{E}(PU_b, M)]$$

10.1.4 Trapdoor One-Way Function

The main idea behind asymmetric-key cryptography is the concept of the trapdoor one-way function.

Functions

Figure 10.3 ***A function as rule mapping a domain to a range***



10.1.4 Continued

One-Way Function (OWF)

1. f is easy to compute.
2. f^{-1} is difficult to compute.

Trapdoor One-Way Function (TOWF)

3. Given y and a trapdoor(secret), x can be computed easily.

10.1.4 Continued

Example 10. 1

When n is large, $n = p \times q$ is a one-way function. Given p and q , it is always easy to calculate n ; given n, it is very difficult to compute p and q. This is the factorization problem.

Example 10. 2

When n is large, the function $y = x^k \bmod n$ is a trapdoor one-way function. Given x, k, and n, it is easy to calculate y. Given y, k, and n, it is very difficult to calculate x. This is the discrete logarithm problem.

We now turn to the definition of a **trap-door one-way function**, which is easy to calculate in one direction and infeasible to calculate in the other direction unless certain additional information is known. With the additional information the inverse can be calculated in polynomial time. We can summarize as follows: A trap-door one-way function is a family of invertible functions f_k, such that

$$
\begin{array}{ll}
Y = f_k(X) & \text{easy, if } k \text{ and } X \text{ are known} \\
X = f_k^{-1}(Y) & \text{easy, if } k \text{ and } Y \text{ are known} \\
X = f_k^{-1}(Y) & \text{infeasible, if } Y \text{ is known but } k \text{ is not known}
\end{array}
$$

Thus, the development of a practical public-key scheme depends on discovery of a suitable trap-door one-way function.

Requirements for public key cryptography

1. **Pair of keys (public key KU_b, private key KR_b)**
2. **Easy to encrypt the message $C=E_{KUb}(M)$**
3. **Easy to decrypt the ciphertext**

   $$M = D_{KRb}(C)=D_{KRb}[\,E_{KUb}(M)\,]$$
4. **Knowing KU_b, it is infeasible to determine KR_b**
5. **Knowing C & KU_b, it is infeasible to determine M**
6. **Either of 2 keys can be used for encryption**

   $$M = D_{KRb}[\,E_{KUb}(M)\,] = D_{KUb}[\,E_{KRb}(M)\,]$$

Public Key Cryptanalysis

Brute force Attack

Complexity in invertible mathematical functions, key size is large enough for brute force impractical, small enough for ease of enc/dec

Compute Private key with Public key – Not mathematically proven that it is infeasible for PKC.

Probable message Attack:

RSA ALGORITHM

Block Cipher, PT and CT are integers between 0 to n-1 for some n. {typical size 1024 bits/309 Decimal digits}

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. Thus, this is a public-key encryption algorithm with a public key of $PU = \{e, n\}$ and a private key of $PR = \{d, n\}$. 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 Algorithm

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

Encryption by Bob with Alice's Public Key	
Plaintext:	$M < n$
Ciphertext:	$C = M^e \bmod n$

Decryption by Alice with Alice's Public Key	
Ciphertext:	C
Plaintext:	$M = C^d \bmod n$

Figure 9.5 The RSA Algorithm

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 private key consists of $\{d, n\}$ and the public key consists of $\{e, n\}$. Suppose that user A has published its public key and that user B wishes to send the message M to A. Then B calculates $C = M^e \bmod n$ and transmits C. On receipt of this ciphertext, user A decrypts by calculating $M = C^d \bmod n$.

1. Select two prime numbers, $p = 17$ and $q = 11$.
2. Calculate $n = pq = 17 \times 11 = 187$.
3. Calculate $\phi(n) = (p - 1)(q - 1) = 16 \times 10 = 160$.
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 $de \equiv 1 \pmod{160}$ and $d < 160$. The correct value is $d = 23$, because $23 \times 7 = 161 = (1 \times 160) + 1$; d can be calculated using the extended Euclid's algorithm (Chapter 4).

The resulting keys are public key $PU = \{7, 187\}$ and private key $PR = \{23, 187\}$. The example shows the use of these keys for a plaintext input of $M = 88$. For encryption, we need to calculate $C = 88^7 \bmod 187$. Exploiting the properties of modular arithmetic, we can do this as follows.

$$
\begin{aligned}
88^7 \bmod 187 = {} & [(88^4 \bmod 187) \times (88^2 \bmod 187) \\
& \times (88^1 \bmod 187)] \bmod 187
\end{aligned}
$$

$$88^1 \bmod 187 = 88$$

$$88^2 \bmod 187 = 7744 \bmod 187 = 77$$

$$88^4 \bmod 187 = 59{,}969{,}536 \bmod 187 = 132$$

$$88^7 \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894{,}432 \bmod 187 = 11$$

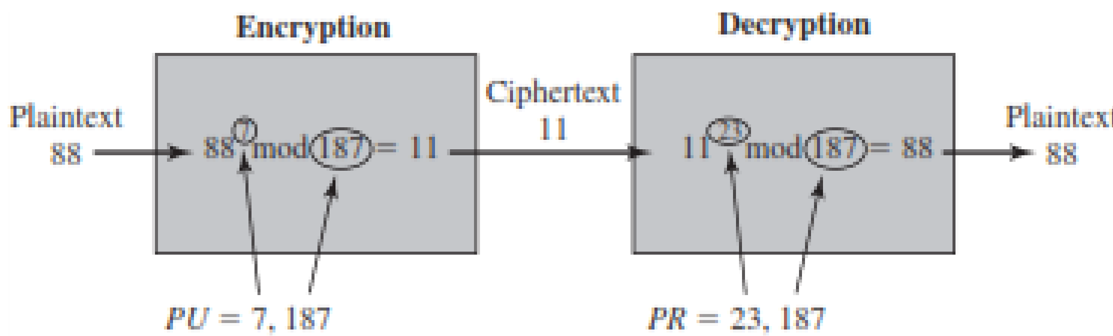


Figure 9.6 Example of RSA Algorithm

For decryption, we calculate $M = 11^{23} \bmod 187$:

$$11^{23} \bmod 187 = [(11^1 \bmod 187) \times (11^2 \bmod 187) \times (11^4 \bmod 187) \times (11^8 \bmod 187) \times (11^8 \bmod 187)] \bmod 187$$

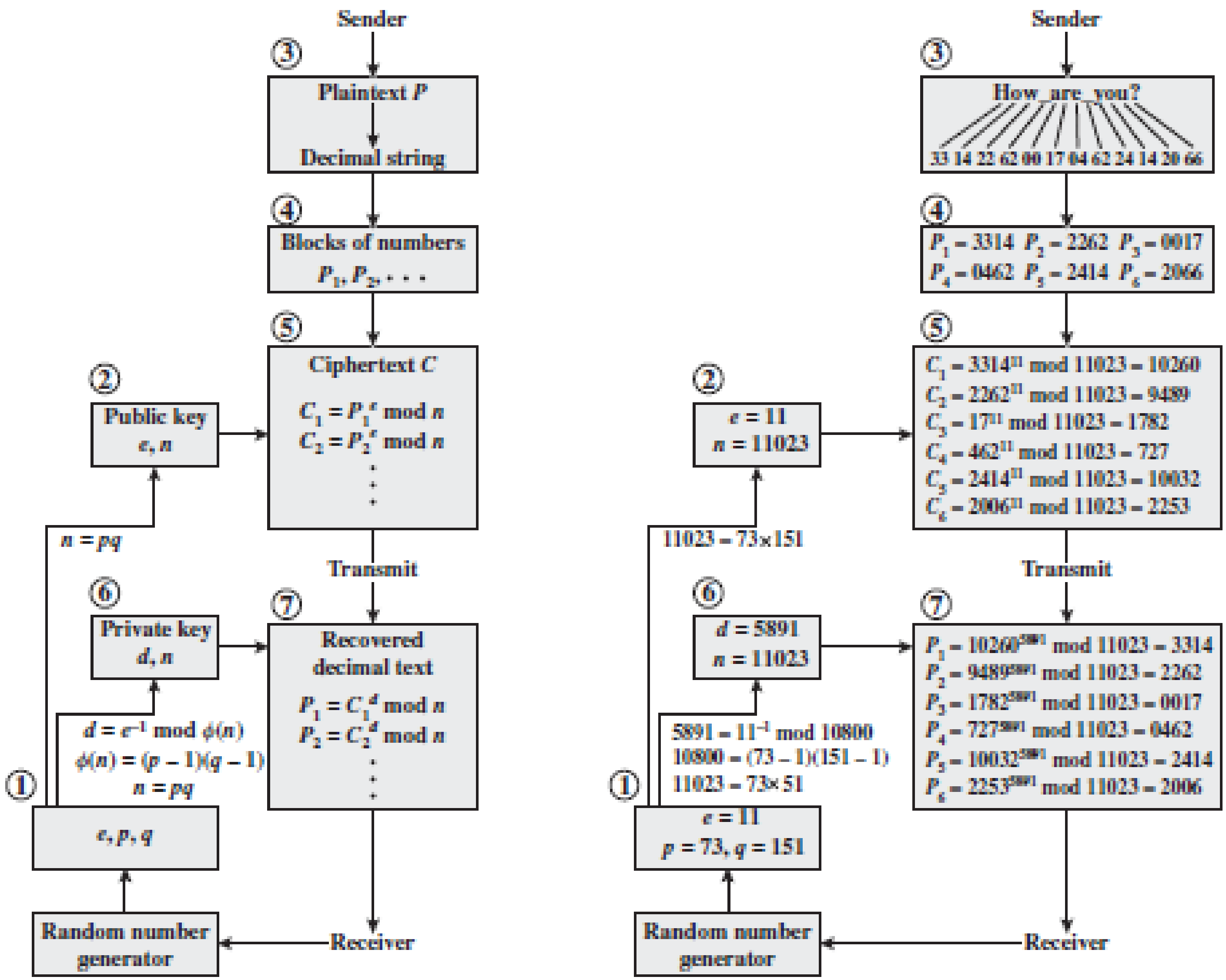
$$11^1 \bmod 187 = 11$$

$$11^2 \bmod 187 = 121$$

$$11^4 \bmod 187 = 14{,}641 \bmod 187 = 55$$

$$11^8 \bmod 187 = 214{,}358{,}881 \bmod 187 = 33$$

$$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 79{,}720{,}245 \bmod 187 = 88$$



(a) General approach **(b) Example**

Figure 9.7 RSA Processeing of Multiple Blocks

Security of RSA

The Security of RSA

Four possible approaches to attacking the RSA algorithm are

- **Brute force:** This involves trying all possible private keys.
- **Mathematical attacks:** There are several approaches, all equivalent in effort to factoring the product of two primes.
- **Timing attacks:** These depend on the running time of the decryption algorithm.
- **Chosen ciphertext attacks:** This type of attack exploits properties of the RSA algorithm.

DIFFIE HELLMAN KEY EXCHANGE

-Discrete Logarithms

4-1 ALGEBRAIC STRUCTURES

Cryptography requires sets of integers and specific operations that are defined for those sets. The combination of the set and the operations that are applied to the elements of the set is called an algebraic structure.

4.1.1 Groups

A **group** (G) is a set of elements with a binary operation "•" that satisfies four properties (or axioms). A **commutative group,** also called an **abelian group,** is a group in which the operator satisfies the four properties for groups plus an extra property, commutativity. The four properties for groups plus commutativity are defined as follows:

- **Closure:** If a and b are elements of **G**, then $c = a \bullet b$ is also an element of **G**. This means that the result of applying the operation on any two elements in the set is another element in the set.
- **Associativity:** If a, b, and c are elements of **G**, then $(a \bullet b) \bullet c = a \bullet (b \bullet c)$. In other words, it does not matter in which order we apply the operation on more than two elements.
- **Commutativity:** For all a and b in **G**, we have $a \bullet b = b \bullet a$. Note that this property needs to be satisfied only for a commutative group.
- **Existence of identity:** For all a in **G**, there exists an element e, called the identity element, such that $e \bullet a = a \bullet e = a$.
- **Existence of inverse:** For each a in **G**, there exists an element a', called the inverse of a, such that $a \bullet a' = a' \bullet a = e$.

Figure 4.2 shows the concept of a group.

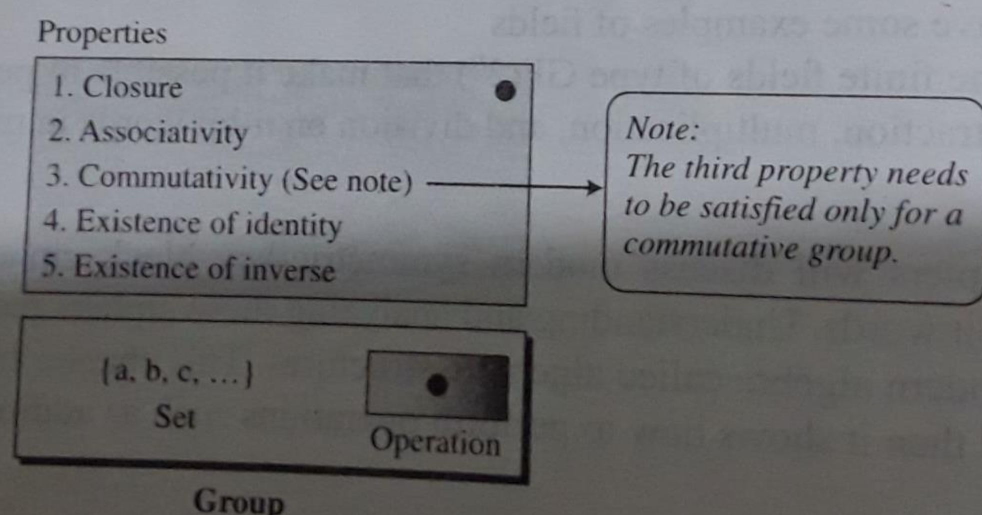


Fig. 4.2 Group

Finite Multiplicative Group In cryptography, we often use the multiplicative finite group: $\mathbf{G} = \langle \mathbf{Z}_n^*, \times \rangle$ in which the operation is multiplication. The set $\mathbf{Z}_n^*$ contains those integers from 1 to $n-1$ that are relatively prime to n; the identity element is $e = 1$. Note that when the modulus of the group is a prime, we have $\mathbf{G} = \langle \mathbf{Z}_p^*, \times \rangle$. This group is the special case of the first group, so we concentrate on the first group in this section.

9.6.2 Continued

Order of the Group is number of elements in the group. ***In*** **G = <Z$_n$∗, ×> it is proved that, order of the group is φ(n).**

What is the order of group G = <Z$_{21}$∗, ×>? |G| = φ(21) = φ(3) × φ(7) = 2 × 6 =12. There are 12 elements in this group: 1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, and 20. All are relatively prime with 21.

9.6.2 Continued

Order of an Element: **in G = <Z_n*, ×>, the order of an element ‘a’ is the smallest integer ‘i’ such that** $a^i \equiv e \bmod (n)$, where e is identity element ie 1.

Lagrange's Theorem: order of an element divides order of group

Example 9.47 Find the order of all elements in $G = <Z_{10}^*, \times>$.

Solution This group has only $\phi(10) = 4$ elements: 1, 3, 7, 9. We can find the order of each element by trial and error. However, recall from Chapter 4 that the order of an element divides the order of the group (Lagrange theorem). The only integers that divide 4 are 1, 2, and 4, which means in each case we need to check only these powers to find the order of the element.

a. $1^1 \equiv 1 \bmod (10) \rightarrow \text{ord}(1) = 1$.
b. $3^1 \equiv 3 \bmod (10)$; $3^2 \equiv 9 \bmod (10)$; $3^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(3) = 4$.
c. $7^1 \equiv 7 \bmod (10)$; $7^2 \equiv 9 \bmod (10)$; $7^4 \equiv 1 \bmod (10) \rightarrow \text{ord}(7) = 4$.
d. $9^1 \equiv 9 \bmod (10)$; $9^2 \equiv 1 \bmod (10) \rightarrow \text{ord}(9) = 2$.

Primitive Roots A very interesting concept in multiplicative group is that of **primitive root,** which is used in the ElGamal cryptosystem in Chapter 10. In the group $\mathbf{G} = <\mathbf{Z}_n{}^*, \times>$, when the order of an element is the same as $\phi(n)$, that element is called the primitive root of the group.

Example 9.50 Table 9.5 shows the result of $a^i \equiv x \pmod 7$ for the group $G = \langle Z_7^*, \times \rangle$. In this group, $\phi(7) = 6$.

Table 9.5 *Example 9.50*

		$i = 1$	$i = 2$	$i = 3$	$i = 4$	$i = 5$	$i = 6$
	$a = 1$	$\boxed{x: 1}$	$x: 1$	$x: 1$	$x: 1$	$x: 1$	$x: 1$
	$a = 2$	$x: 2$	$x: 4$	$\boxed{x: 1}$	$x: 2$	$x: 4$	$x: 1$
Primitive root →	$a = 3$	$x: 3$	$x: 2$	$x: 6$	$x: 4$	$x: 5$	$\boxed{x: 1}$
	$a = 4$	$x: 4$	$x: 2$	$\boxed{x: 1}$	$x: 4$	$x: 2$	$x: 1$
Primitive root →	$a = 5$	$x: 5$	$x: 4$	$x: 6$	$x: 2$	$x: 3$	$\boxed{x: 1}$
	$a = 6$	$x: 6$	$\boxed{x: 1}$	$x: 6$	$x: 1$	$x: 6$	$x: 1$

The orders of elements are ord(1) = 1, ord(2) = 3, ord(3) = <u>**6**</u>, ord(4) = 3, ord(5) = <u>**6**</u>, and ord(6) = 1. Table 9.5 shows that only two elements, 3 and 5, have the order at $i = \phi(n) = 6$. Therefo
only two primitive roots: 3 and 5.

Diffie-Hellman Key Exchange

Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

User A Key Generation	
Select private X_A	$X_A < q$
Calculate public Y_A	$Y_A = \alpha^{XA} \bmod q$

User B Key Generation	
Select private X_B	$X_B < q$
Calculate public Y_B	$Y_B = \alpha^{XB} \bmod q$

Calculation of Secret Key by User A
$K = (Y_B)^{XA} \bmod q$

Calculation of Secret Key by User B
$K = (Y_A)^{XB} \bmod q$

Figure 10.1 The Diffie-Hellman Key Exchange Algorithm

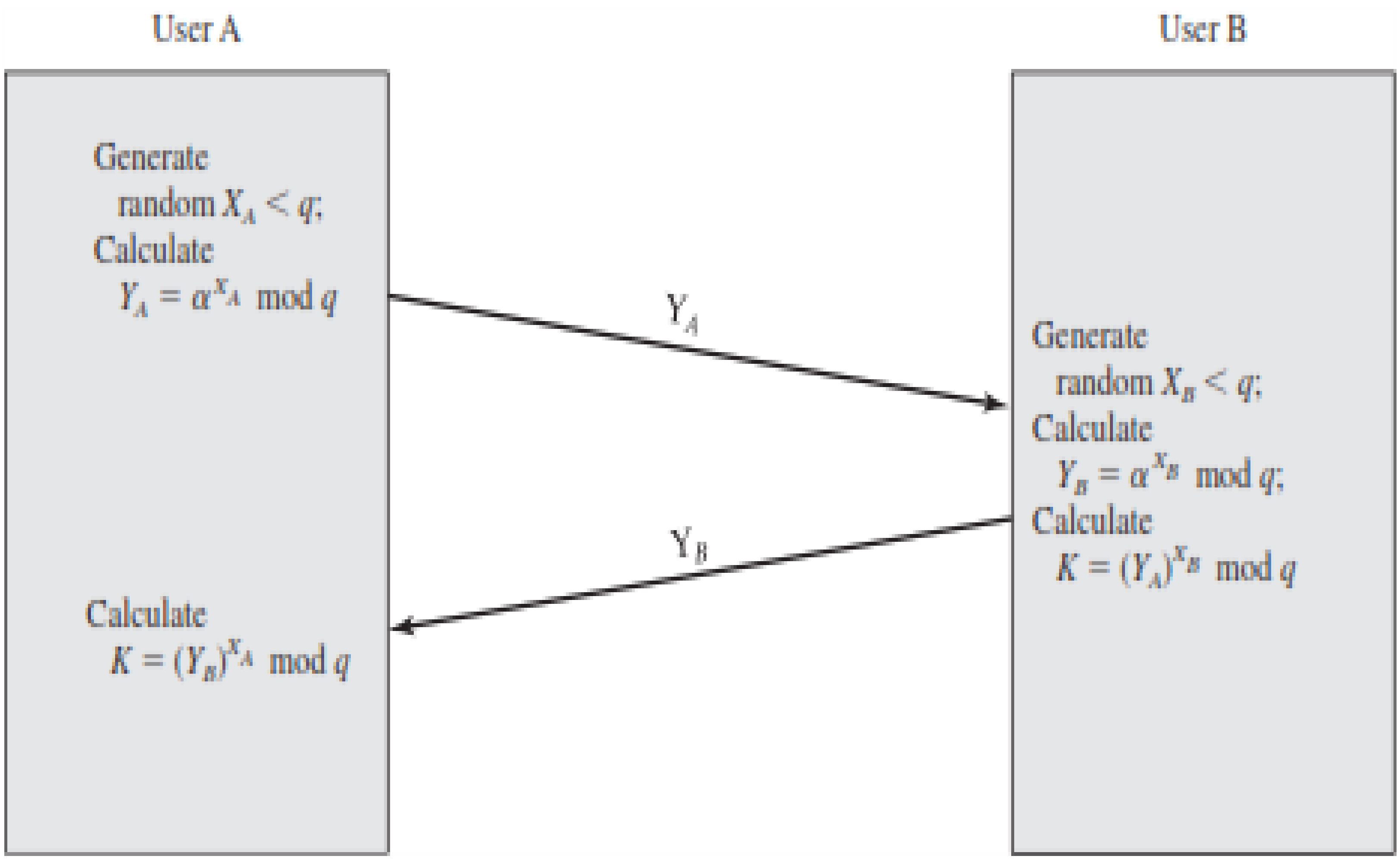


Figure 10.2 Diffie-Hellman Key Exchange

The Algorithm

Figure 10.1 summarizes the Diffie-Hellman key exchange algorithm. For this scheme, there are two publicly known numbers: a prime number q and an integer α that is a primitive root of q. Suppose the users A and B wish to exchange a key. User A selects a random integer $X_A < q$ and computes $Y_A = \alpha^{X_A} \bmod q$. Similarly, user B independently selects a random integer $X_B < q$ and computes $Y_B = \alpha^{X_B} \bmod q$. Each side keeps the X value private and makes the Y value available publicly to the other side. User A computes the key as $K = (Y_B)^{X_A} \bmod q$ and user B computes the key as $K = (Y_A)^{X_B} \bmod q$. These two calculations produce identical results:

$$
\begin{aligned}
K &= (Y_B)^{X_A} \bmod q \\
&= (\alpha^{X_B} \bmod q)^{X_A} \bmod q \\
&= (\alpha^{X_B})^{X_A} \bmod q && \text{by the rules of modular arithmetic} \\
&= \alpha^{X_B X_A} \bmod q \\
&= (\alpha^{X_A})^{X_B} \bmod q \\
&= (\alpha^{X_A} \bmod q)^{X_B} \bmod q \\
&= (Y_A)^{X_B} \bmod q
\end{aligned}
$$

The result is that the two sides have exchanged a secret value. Furthermore, because X_A and X_B are private, an adversary only has the following ingredients to work with: q, α, Y_A, and Y_B. Thus, the adversary is forced to take a discrete logarithm to determine the key. For example, to determine the private key of user B, an adversary must compute

$$X_B = \mathrm{dlog}_{\alpha,q}(Y_B)$$

Example

$q=11 \qquad \alpha=3$

$X_A = 5$

$Y_A = 3^5 \bmod 11 = 1$

$X_B = 3$

$Y_B = 3^3 \bmod 11 = 27 \bmod 11 = 5$

$K1 = 5^5 \bmod 11 = 1$

$K2 = 1^3 \bmod 11 = 1$

A & B can share 1 without transmitting

$q=23 \qquad \alpha=5$

$X_A = 6 \qquad X_B = 15$

$K=?$

Here is an example. Key exchange is based on the use of the prime number $q = 353$ and a primitive root of 353, in this case $\alpha = 3$. A and B select secret keys $X_A = 97$ and $X_B = 233$, respectively. Each computes its public key:

A computes $Y_A = 3^{97} \bmod 353 = 40$.

B computes $Y_B = 3^{233} \bmod 353 = 248$.

After they exchange public keys, each can compute the common secret key:

A computes $K = (Y_B)^{X_A} \bmod 353 = 248^{97} \bmod 353 = 160$.

B computes $K = (Y_A)^{X_B} \bmod 353 = 40^{233} \bmod 353 = 160$.

Man –in- the- Middle attack

1. Darth prepares for the attack by generating two random private keys X_{D1} and X_{D2} and then computing the corresponding public keys Y_{D1} and Y_{D2}.
2. Alice transmits Y_A to Bob.
3. Darth intercepts Y_A and transmits Y_{D1} to Bob. Darth also calculates $K2 = (Y_A)^{X_{D2}} \bmod q$.
4. Bob receives Y_{D1} and calculates $K1 = (Y_{D1})^{X_B} \bmod q$.
5. Bob transmits Y_B to Alice.
6. Darth intercepts Y_B and transmits Y_{D2} to Alice. Darth calculates $K1 = (Y_B)^{X_{D1}} \bmod q$.
7. Alice receives Y_{D2} and calculates $K2 = (Y_{D2})^{X_A} \bmod q$.

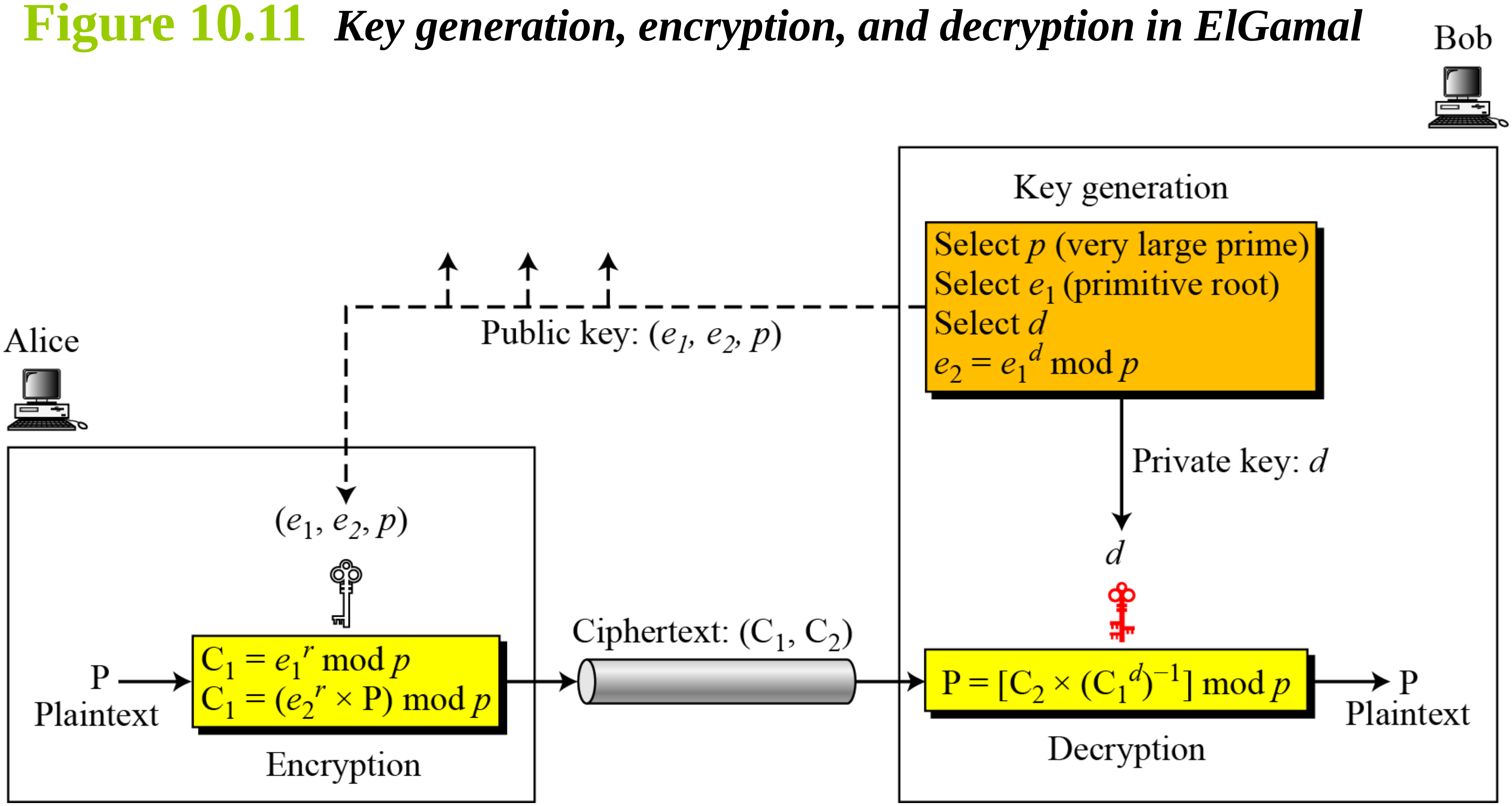
Middle man attack / bucket bridage attack

Alice	Carol	Bob
$q=11 \quad \alpha=7$	$q=11 \quad \alpha=7$	$q=11 \quad \alpha=7$
$X_A = 3$	$M_{XA}=8 \quad M_{XB}=6$	$X_B = 9$
$Y_A = 7^3 \bmod 11$	$Y_A= 7^8 \bmod 11$	$Y_B = 7^9 \bmod 11$
$= 2$	$= 9$	$= 8$
$Y_B = 4$	$Y_B= 7^6 \bmod 11$	$Y_A = 9$
	$= 4$	
$K1= 4^3 \bmod 11$	$Y_A=2 \quad Y_B=8$	$K2= 9^9 \bmod 11$
$= 9$	$K1= 8^8 \bmod 11$	$= 5$
	$= 5$	
	$K2= 2^6 \bmod 11$	
	$= 9$	

10-4 ELGAMAL CRYPTOSYSTEM

10.4.2 Procedure

Figure 10.11 ***Key generation, encryption, and decryption in ElGamal***



10.4.2 Continued

Key Generation

Algorithm 10.9 *ElGamal key generation*

ElGamal_Key_Generation

{

Select a large prime p

Select d to be a member of the group $\mathbf{G} = <\mathbf{Z}_p{}^*, \times>$ such that $1 \leq d \leq p - 2$

Select e_1 to be a primitive root in the group $\mathbf{G} = <\mathbf{Z}_p{}^*, \times>$

$e_2 \leftarrow e_1{}^d \bmod p$

Public_key $\leftarrow (e_1, e_2, p)$ // To be announced publicly

Private_key $\leftarrow d$ // To be kept secret

return Public_key and Private_key

}

10.4.2 Continued

Algorithm 10.10 *ElGamal encryption*

```
ElGamal_Encryption (e1, e2, p, P)                    // P is the plaintext
{
    Select a random integer r in the group G = < Zp*, × >
    C1 ← e1^r mod p
    C2 ← (P × e2^r) mod p                            // C1 and C2 are the ciphertexts
    return C1 and C2
}
```

$$C_1 \leftarrow e_1^{\,r} \bmod p \qquad C_2 \leftarrow (P \times e_2^{\,r}) \bmod p \qquad \mathbf{G} = \langle \mathbf{Z}_p^*, \times \rangle$$

10.4.2 Continued

Algorithm 10.11 *ElGamal decryption*

```
ElGamal_Decryption (d, p, C1, C2)                // C1 and C2 are the ciphertexts
{
    P ← [C2 (C1^d)^-1] mod p                     // P is the plaintext
    return P
}
```

$$P \leftarrow [C_2\,(C_1^{\,d})^{-1}] \bmod p$$

10.4.3 Continued

Example 10. 10

Bob chooses $p = 11$ and $e_1 = 2$. and $d = 3$ $e_2 = e_1^d = 8$. So the public keys are (2, 8, 11) and the private key is 3. Alice chooses $r = 4$ and calculates C1 and C2 for the plaintext 7.

Plaintext: 7

$C_1 = e_1^r \bmod 11 = 16 \bmod 11 = 5 \bmod 11$

$C_2 = (P \times e_2^r) \bmod 11 = (7 \times 4096) \bmod 11 = 6 \bmod 11$

Ciphertext: (5, 6)

Bob receives the ciphertexts (5 and 6) and calculates the plaintext.

$[C_2 \times (C_1^d)^{-1}] \bmod 11 = 6 \times (5^3)^{-1} \bmod 11 = 6 \times 3 \bmod 11 = 7 \bmod 11$

Plaintext: 7

Q4) ElGamal, $p = 37$, $e_1 = 3$, $d = 2$: Encrypt "SUNSHINE" $(r = 7)$

→ Key Generation: $e_2 = e_1^d \bmod p = 3^2 \bmod 37 = 9$

Public key $= (3, 9, 37)$, Private key $= (2)$

→ Encryption: $C_1 = e_1^r \bmod p = 3^7 \bmod 37 = 4$

$C_2 = (P \times e_2^r) \bmod p = P \times 9^7 \bmod 37 = P \times 16 \bmod 37$

$P = \text{SUNSHINE} = [20, 22, 13, 20, 7, 8, 13, 4]$

$C_2 = 16 \times [20, 22, 13, 20, 7, 8, 13, 4] \bmod 37$

$= [24, 19, 23, 24, 1, 17, 23, 27] =$ ~~YROPB~~

Y R X Y B P X B

$= [24, 19, 23, 24, 1, 17, 23, 1]$ $\left\{ 27 \equiv 1 \right.$ because $27 \bmod \underline{\underline{26}} \equiv 1$ # alphabets

$C_2 = \underline{\text{YRXYBPXB}}$

→ Decryption: $P = \left[C_2 (C_1^d)^{-1} \right] \bmod p$

$= \left[C_2 (4^2)^{-1} \right] \bmod 37 = C_2 \times 16^{-1} \bmod 37$

$= C_2 \times 7 \bmod 37$

[Inverse Calc^n]

2	37	16	5	0	1	-2
3	16	5	1	1	-2	7
5	5	1	0	-2	7	-37
	1	0		$\boxed{7}$		

Wrong! Hence, the value before doing mod 26 must be considered

$P = 7 \times [24, 19, 23, 24, 1, 17, 23, 1 \text{ or } 27] \bmod 37$

$= [20, 22, 13, 20, 7, 8, 13, 7 \text{ or } 4]$

$P = \underline{\text{SUNSHINE}}$

Analysis A very interesting point about the ElGamal cryptosystem is that Alice creates r and keeps it secret; Bob creates d and keeps it secret. The puzzle of this cryptosystem can be solved as follows:

a. Alice sends $C_2 = [e_2^r \times P] \bmod p = [(e_1^{rd}) \times P] \bmod p$. The expression (e_1^{rd}) acts as a mask that hides the value of P. To find the value of P, Bob must remove this mask.
b. Because modular arithmetic is being used, Bob needs to create a replica of the mask and invert it (multiplicative inverse) to cancel the effect of the mask.
c. Alice also sends $C_1 = e_1^r$ to Bob, which is a part of the mask. Bob needs to calculate C_1^d to make a replica of the mask because $C_1^d = (e_1^r)^d = (e_1^{rd})$. In other words, after obtaining the mask replica, Bob inverts it and multiplies the result with C_2 to remove the mask.
d. It might be said that Bob helps Alice make the mask (e_1^{rd}) without revealing the value of d (d is already included in $e_2 = e_1^d$); Alice helps Bob make the mask (e_1^{rd}) without revealing the value of r (r is already included in $C_1 = e_1^r$).

9-4 CHINESE REMAINDER THEOREM

The Chinese remainder theorem (CRT) is used to solve a set of congruent equations with one variable but different moduli, which are relatively prime, as shown
below:

$$x \equiv a_1 \pmod{m_1}$$
$$x \equiv a_2 \pmod{m_2}$$
$$\cdots$$
$$x \equiv a_k \pmod{m_k}$$

9-4 Continued

Example 9.35

The following is an example of a set of equations with different moduli:

$$x \equiv 2 \pmod{3}$$
$$x \equiv 3 \pmod{5}$$
$$x \equiv 2 \pmod{7}$$

The solution to this set of equations is given in the next section; for the moment, note that the answer to this set of equations is $x = 23$. This value satisfies all equations: $23 \equiv 2 \pmod{3}$, $23 \equiv 3 \pmod{5}$, and $23 \equiv 2 \pmod{7}$.

9-4 Continued

Solution To Chinese Remainder Theorem

1. ***Find $M = m_1 \times m_2 \times \ldots \times m_k$. This is the common modulus.***
2. ***Find $M_1 = M/m_1$, $M_2 = M/m_2$, …, $M_k = M/m_k$.***
3. ***Find the multiplicative inverse of M_1, M_2, …, M_k using the corresponding moduli (m_1, m_2, …, m_k). Call the inverses M_1^{-1}, M_2^{-1}, …, M_k^{-1}.***
4. ***The solution to the simultaneous equations is***

$$x = (a_1 \times M_1 \times M_1^{-1} + a_2 \times M_2 \times M_2^{-1} + \cdots + a_k \times M_k \times M_k^{-1}) \bmod M$$

9-4 Continued

Example 9.36

Find the solution to the simultaneous equations:

$$x \equiv 2 \pmod 3$$
$$x \equiv 3 \pmod 5$$
$$x \equiv 2 \pmod 7$$

Solution

We follow the four steps.

1. $M = 3 \times 5 \times 7 = 105$

2. $M_1 = 105 / 3 = 35$, $M_2 = 105 / 5 = 21$, $M_3 = 105 / 7 = 15$

3. The inverses are $M_1^{-1} = 2$, $M_2^{-1} = 1$, $M_3^{-1} = 1$

4. $x = (2 \times 35 \times 2 + 3 \times 21 \times 1 + 2 \times 15 \times 1) \bmod 105 = 23 \bmod 105$