Public Key Encryption

Diffie and Hellman – 1976 *Famous Paper:* New Directions In Cryptography - 1976

First revolutionary advance in cryptography in thousands of years

Based on mathematical functions not bit manipulation

Asymmetric, two separate key

Profound effect on confidentiality, key distribution and authentication

Public Key Structure

Plaintext: message input into the algorithm

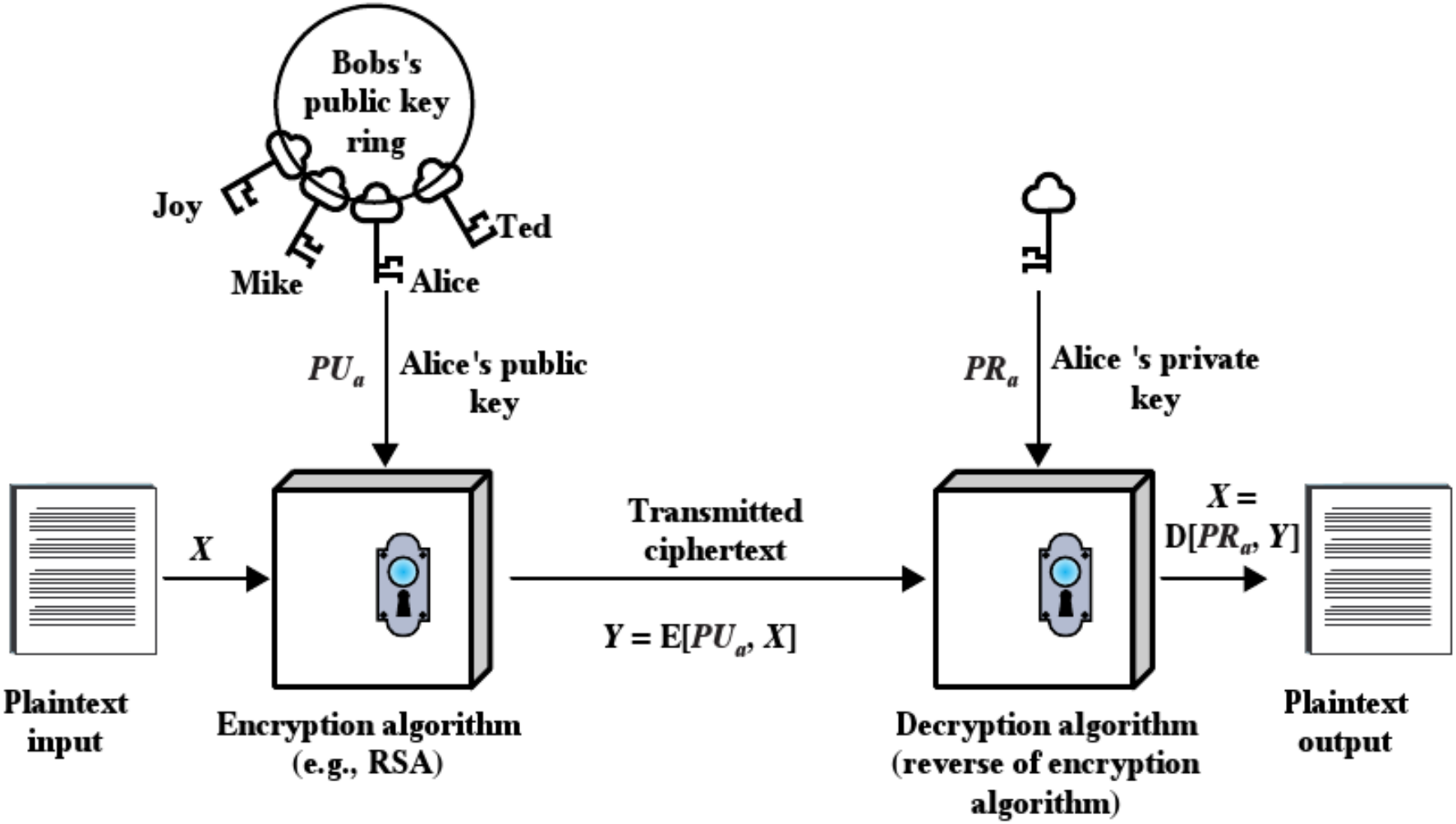
Encryption algorithm: transformations on plaintext

Public & Private Key: pair of keys, one for encryption; one for decryption

Ciphertext: scrambled message

Decryption algorithm: produces original plaintext

Public Key Encryption



(a) Encryption

The Basic Steps

Each user generates a pair of keys

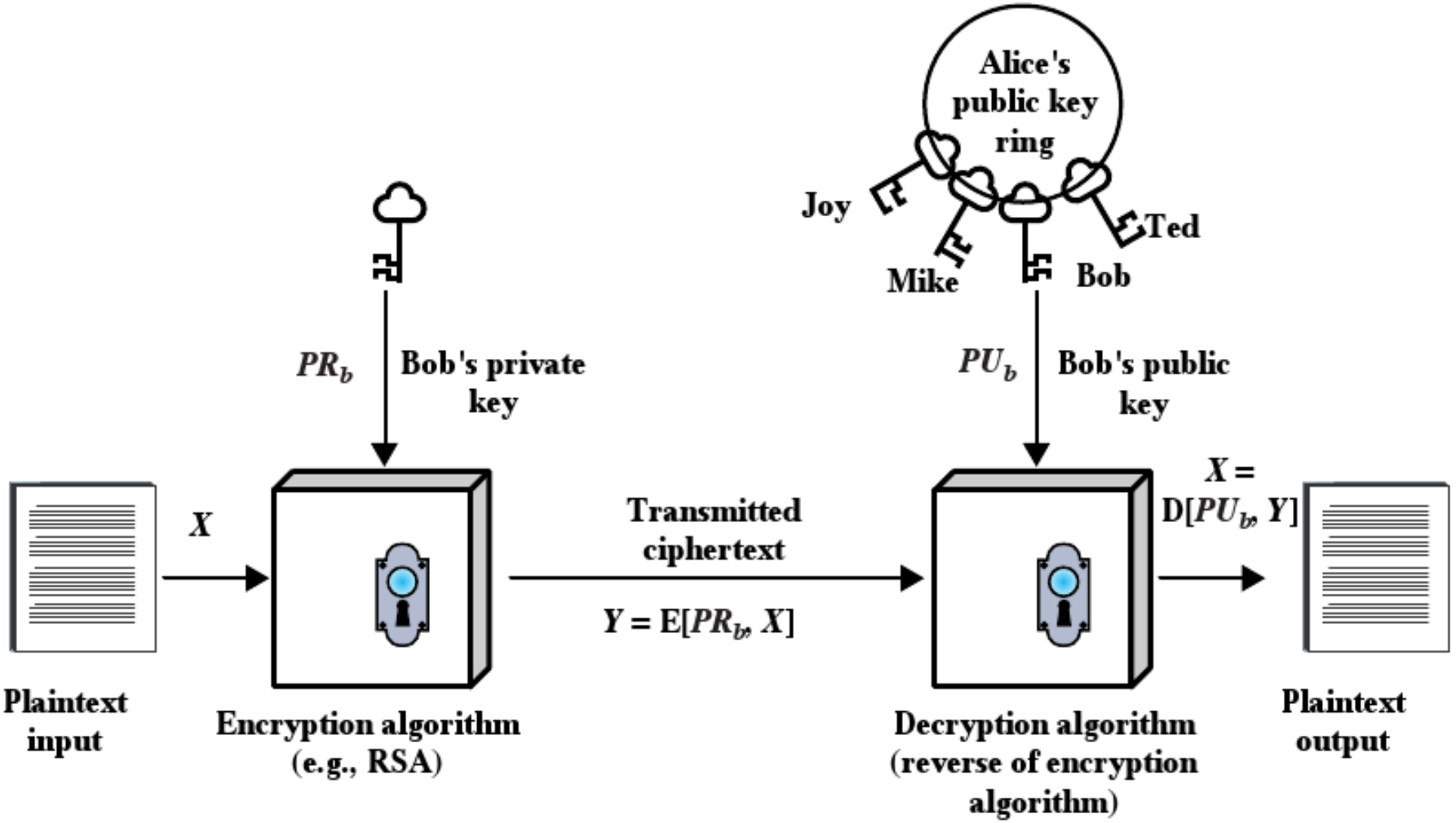
The public key goes in a public register

The private key is kept private

If Bob wishes to send a private message to Alice, Bob *encrypts* the message using Alice's *public* key

When Alice receives the message, she *decrypts* using her *private* key

Public Key Authentication



(b) Authentication

Figure 3.7 Public-Key Cryptography

Public Key Applications

Encryption/decryption – encrypts a message with the recipient's public key

Digital signature – sender *signs* a message with private key

Key Exchange – two sides cooperate to exchange a session key

Requirements For Public Key

Easy for party B to generate pairs: public key KU_b ; private key KR_b

Easy for sender A to generate ciphertext using public key:

$$C = E_{KUb}(M)$$

HINT:
PUBLIC
PRIVATE

Easy for receiver B to decrypt using the private key to recover original message

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

Requirements For Public Key

It is computationally infeasible for an opponent, knowing the public key KUb to determine the private key KR_b

It is computationally infeasible for an opponent, knowing the public key KUb and a ciphertext, C, to recover the original message, M

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

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

RSA Algorithm

Ron Rivest, Adi Shamir, Len Adleman – 1978

Most widely accepted and implemented approach to public key encryption

M and C are integers between 0 and $n-1$ for some n :

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

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

Table 3.2 Applications for Public-Key Cryptosystems

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

RSA Algorithm

Sender and receiver know the values of n and e, but only the receiver knows the value of d

Public key: $KU = \{e,n\}$

Private key: $KR = \{d,n\}$

RSA Requirements

It is possible to find values of e, d, n such that $M^{ed} = M \bmod n$ for all $M<n$

It is relatively easy to calculate M^e and C for all values of $M<n$

It is infeasible to determine d given e and n

RSA Algorithm

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	$KR = \{d, n\}$

RSA Algorithm

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

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

Figure 3.8 The RSA Algorithm

RSA Example

Select two prime numbers, $p=7$ and $q=17$

Calculate $n = pq = 7 \text{ x } 17 = 119$ ← this is the modulus

Calculate $\Phi(n) = (p-1)(q-1) = 96$ ← Euler totient

Select e such that e is relatively prime to $\Phi(n) = 96$ and less than $\Phi(n)$; in this case, $e= 5$

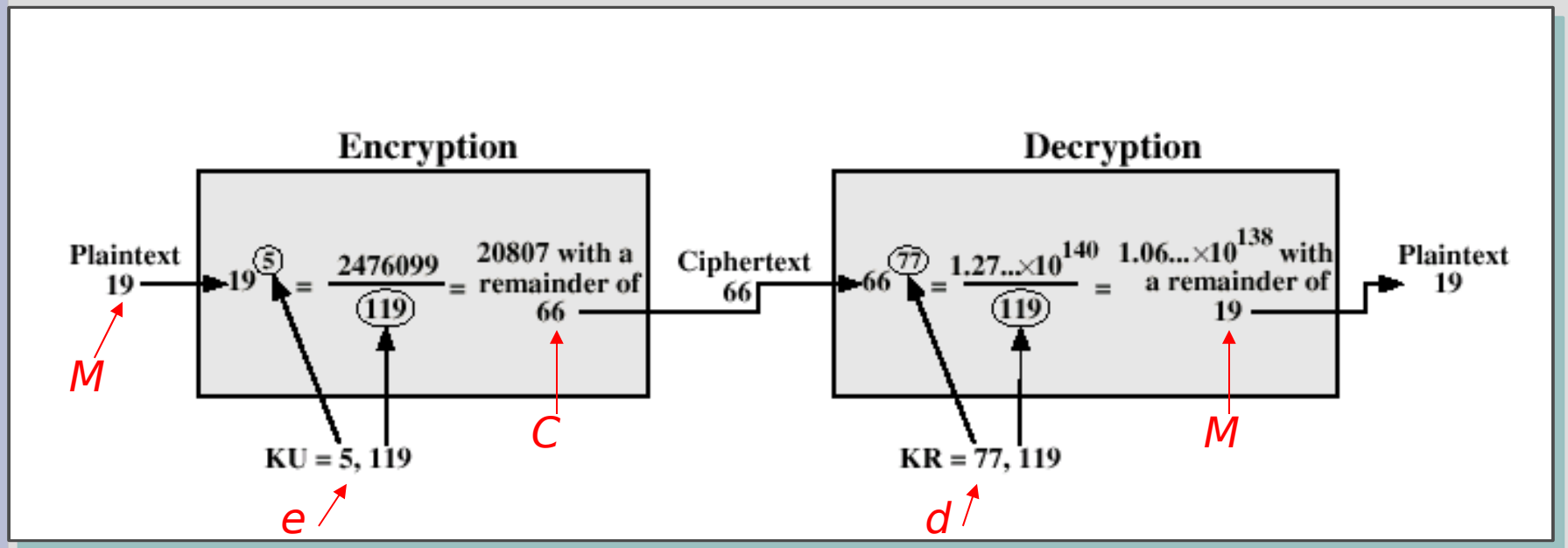
Determine d such that $de = 1 \bmod 96$ and $d<96$. The correct value is $d = 77$, because

$77 \text{ x } 5 = 385 = 4 \text{ x } 96 + 1$

↑ multiplicative inverse of e

See also "a very simple example of RSA encryption": http://www.di-mgt.com.au/rsa_alg.html

RSA Example


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

RSA Strength

Brute force attack: try all possible keys – the larger e and d the more secure

The larger the key, the slower the system

For large n with large prime factors, factoring is a hard problem

Cracked in 1994 a 428 bit key; $100

As of 2008 , the largest (known) number factored by a general-purpose factoring algorithm was 663 bits long (see RSA-200), using a state-of-the-art distributed implementation (Wikipedia)

It was estimated that the factoring for 512-bit RSA can be completed in 10 minutes by a $10 000 device and 1024-bit RSA in less than 1 year with a $10 million device (2003).

RSA keys are typically 1024–2048 bits long. Some experts

A primitive root of the prime number p

A primitive root of the prime number p, namely a is positive integer less than p and its powers (from 1 to p-1) generates all the integers from 1 to p-1, that is

$$S = \{ a^1 \bmod p, a^2 \bmod p, \ldots, a^{p-1} \bmod p\} = \{ 1,2,\ldots,p-1\}$$

(Here the first set is a permutation on the integers from 1 to p-1).

Note that a^i not equivalent to $a^j \bmod p$ when i is different than j, where $0<i<j<p$.

Diffie-Hellman Key Exchange

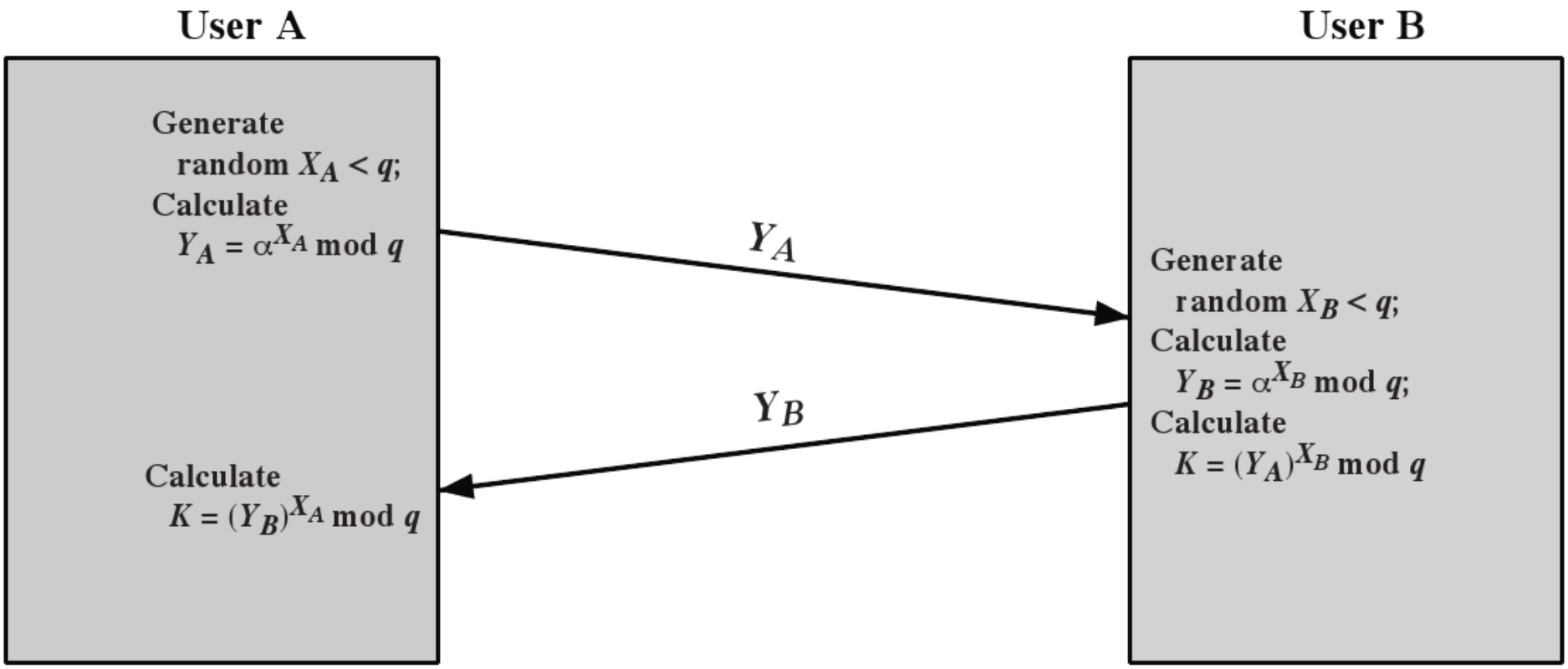
Global Public Elements	
q	prime number
α	$\alpha < q$ and α a primitive root of q

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

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

Enables two users to exchange a secret key (of a block cipher or a stream cipher) for securely.

Exchange



Visualization of "Diffie Hellman Key Exchange Protocol"
Use Cryptool Software (www.cryptool.org) and choose Menu --> Indv. Procedures -->Protocols --> Diffie-Hellman Demonstration

Figure 3.11 Diffie-Hellman Key Exchange

Security of Diffie-Hellman Key Exchange

It is relatively easy to calculate exponentials modulo a prime but very difficult to calculate discrete logarithms (for large prime q, the latter task is computationally infeasible).

The adversary (opponent such as Darth or Oscar) has only α,q, and Y_A and Y_B and he should calculate the following discrete logarithm in order to find the private key of part A, X_A (respectively X_B):

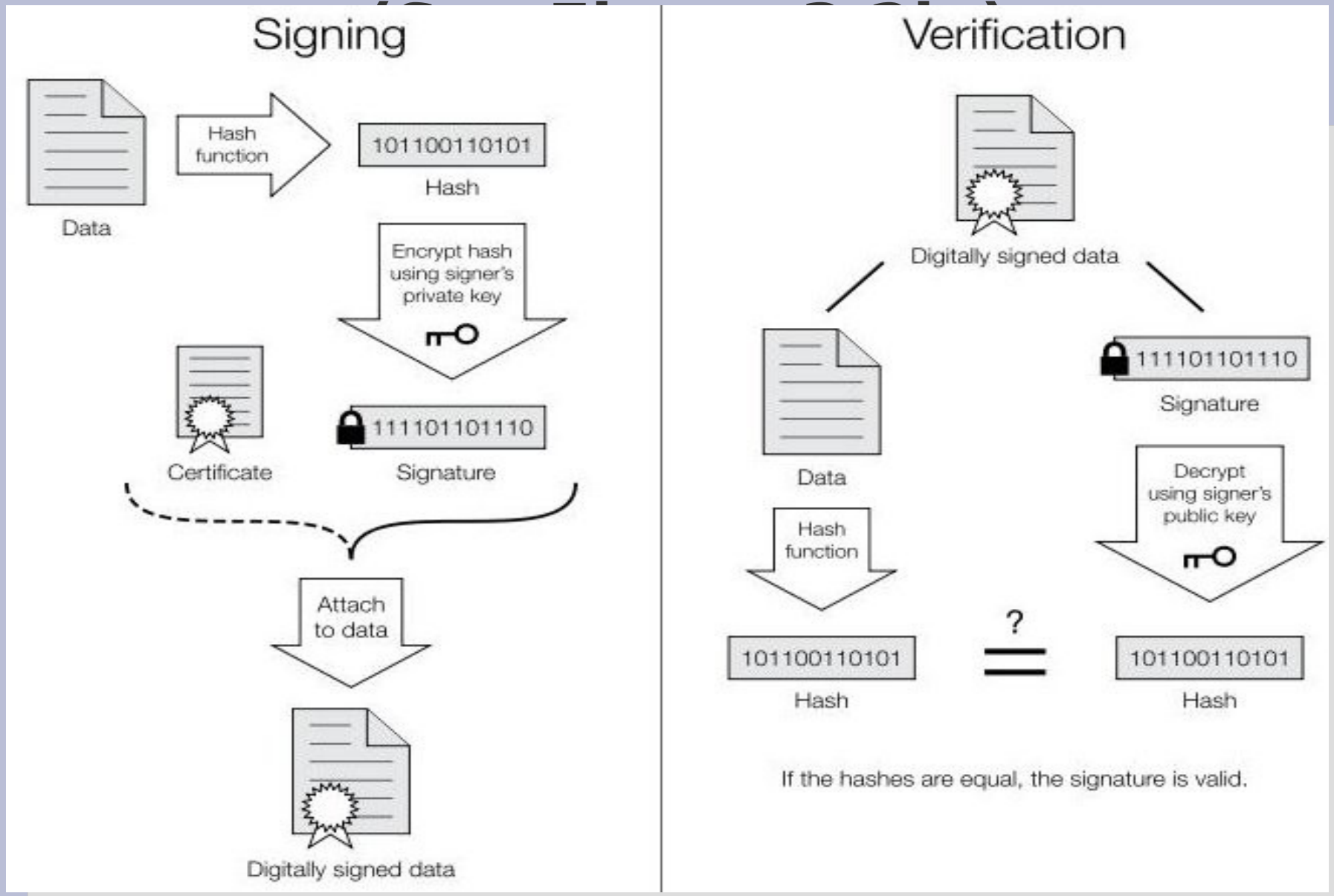
$$X_A = \text{dlog}_{\alpha,q}(Y_A)$$

Other Public Key Algorithms

Digital Signature Standard (DSS) – makes use of SHA-1 and presents a new digital signature algorithm (DSA)

Only used for digital signatures not encryption or key exchange

Digital Signatures



Digital Signatures (See Figure 3.2b)

Digital signatures are equivalent to traditional handwritten signatures in many respects; properly implemented digital signatures are more difficult to forge than the handwritten type.

Digital signatures can also provide non-repudiation

Digitally signed messages may be anything representable as a bitstring: examples include electronic mail, contracts, or a message sent via some other cryptographic protocol.

Digital Signatures (See Figure 3.2b)

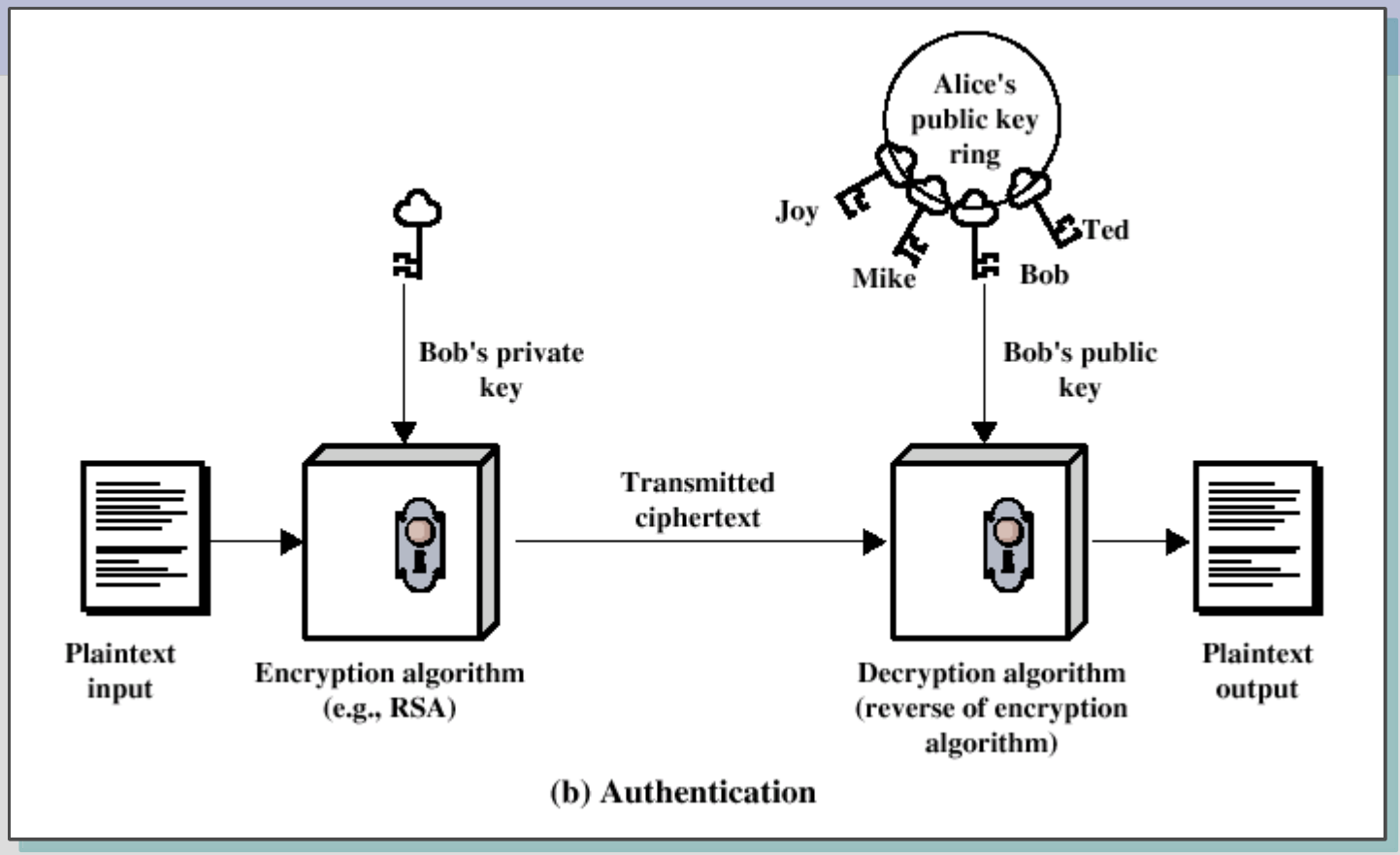
A digital signature scheme typically consists of three algorithms:

A key generation algorithm that selects a private key uniformly at random from a set of possible private keys. The algorithm outputs the private key and a corresponding public key.

A signing algorithm which, given a message and a private key, produces a signature.

A signature verifying algorithm which given a message, public key and a signature, either accepts or rejects.

Public Key Authentication



Key Management
Digital Certificate

Certificate consists of a *public key* plus a *user ID* of the key owner (other user information may also be added), with the whole block signed by a trusted third party, the certificate authority (CA) (trusted by user community such as a government or a financial institution)

CA has public key (and associated certificate also) and private key pair.

User publishes his/her certificate and public key can be obtained from this certificate (trusted in a sense that CA signature verified by using CA's public key)

Hash code of unsigned user certificate signed by the

Key Management
Digital Certificate

Certificate Fundamental and important object in X.509 standard.

X.509 public-key certificates universally accepted.

X.509 public-key certificates are used in most network security applications: IP Security, SSL, SET and S/MIME

Verisign is primary vendor

In Turkey, legally valid user certificates are delivered by (see http://tr.wikipedia.org/wiki/Elektronik_imza)

Tübitak Kamu Sertifikasyon Merkezi (http://www.kamusm.gov.tr)

TurkTrust Elektronik Sertifika Sağlayıcısı (https://www.turktrust.com.tr)

E-Güven Elektronik İmza (e-imza) ve Elektronik Sertifika Hizmetleri

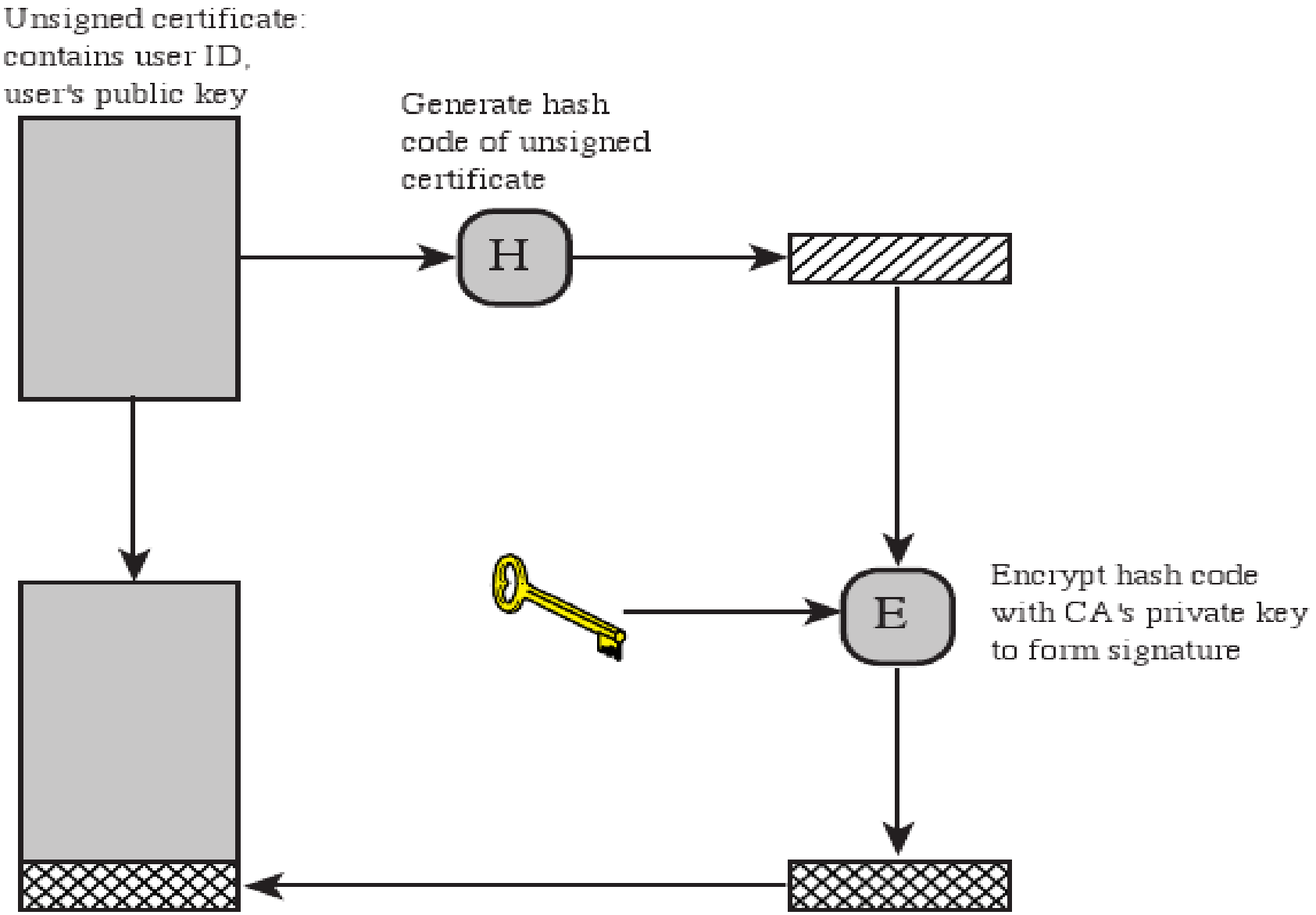


Figure 3.12 Public Key Certificate Use