Chapter3

Public-Key Cryptography and Message Authentication

OUTLINE

- Approaches to Message Authentication
- Secure Hash Functions (SHA) and Keyed-Hash Message Authentication Code (HMAC)
- Public-Key Cryptography Principles
- Public-Key Cryptography Algorithms
- Digital Signatures
- Key Management

Authentication

- Requirements - must be able to verify that:
    1. Message came from apparent source or author.
    2. Contents have not been altered.
    3. Sometimes, it was sent at a certain time or sequence.
- Protection against active attack (falsification of data and transactions)

Approaches to Message Authentication

- Authentication Using Conventional Encryption
  - Only the sender and receiver should share a key
- Message Authentication without Message Encryption
  - An authentication tag is generated and appended to each message
- Message Authentication Code
  - Calculate the MAC as a function of the message and the key. MAC = F(K, M)

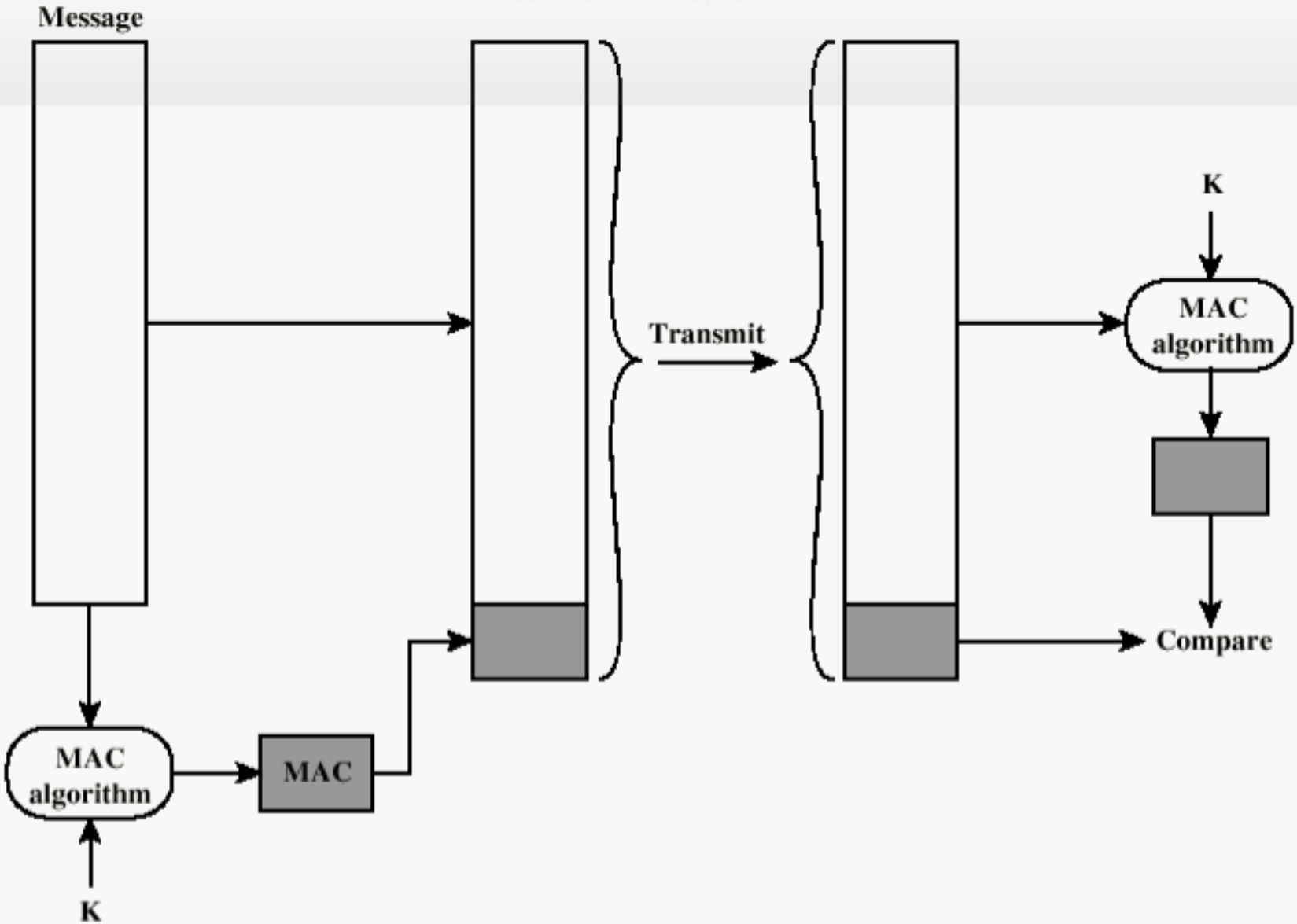
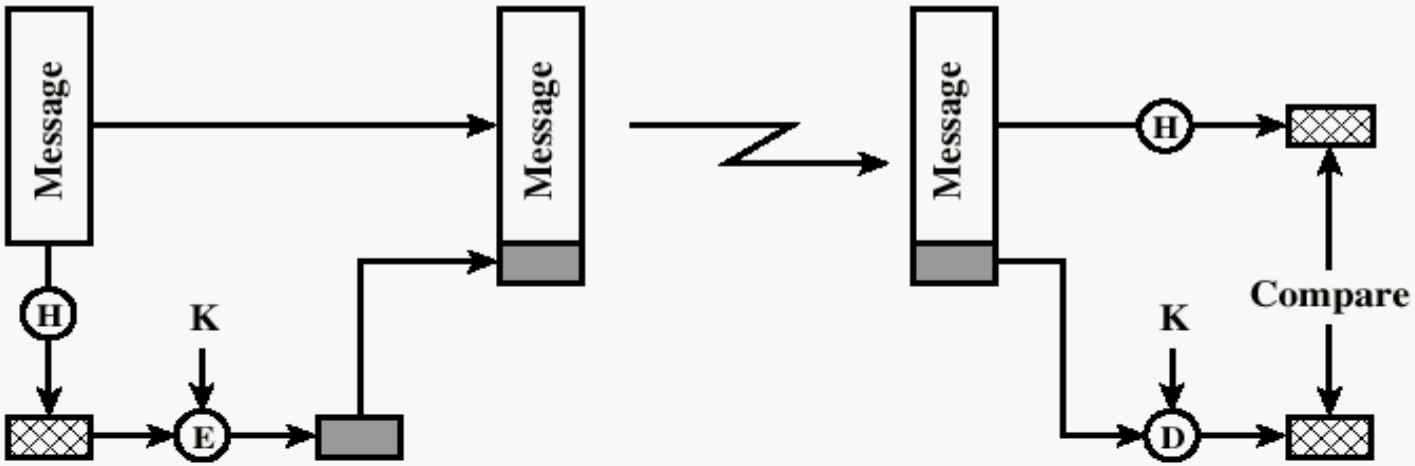
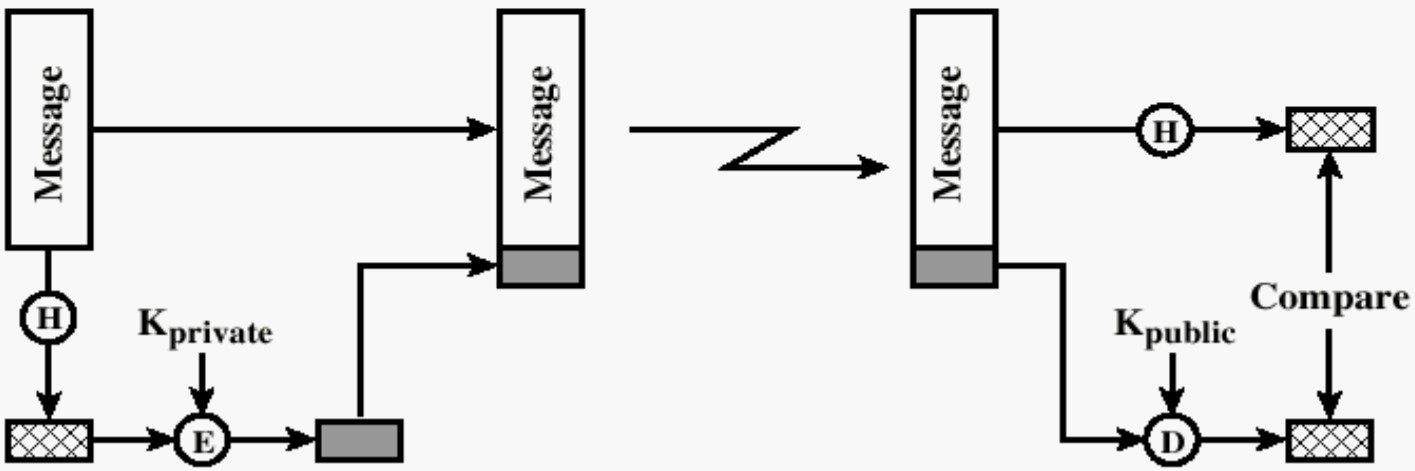


Figure 3.1 Message Authentication Using a Message Authentication Code (MAC)

One-way HASH function



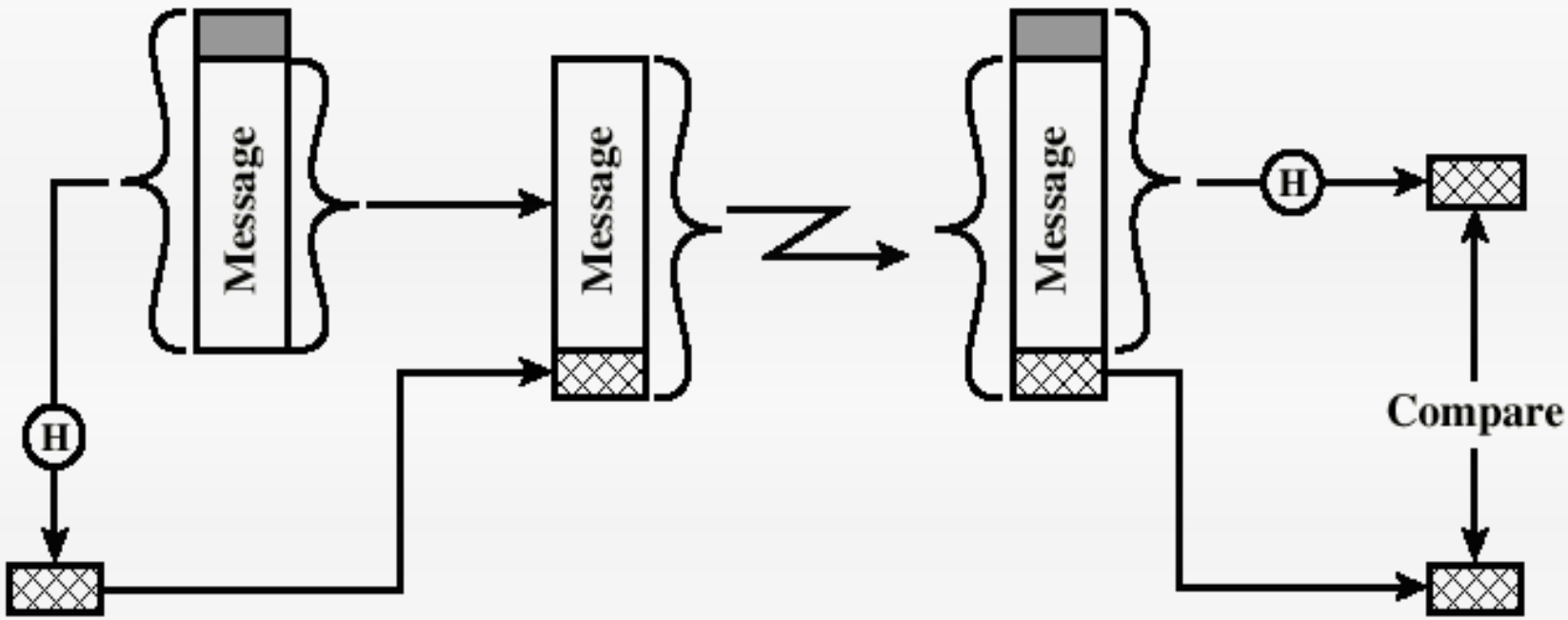
(a) Using conventional encryption



(b) Using public-key encryption

One-way HASH function

- Secret value is added before the hash and removed before transmission.



(c) Using secret value

Secure HASH Functions

- Purpose of the HASH function is to produce a "fingerprint"
- Used in message authentication and digital signatures
- Properties of a HASH function H :
  - H can be applied to a block of data at any size
  - H produces a fixed length output
  - H(x) is easy to compute for any given x.
  - For any given block x, it is computationally infeasible to find x such that H(x) = h (one-way property)
  - For any given block x, it is computationally infeasible to find with H(y) = H(x). (weak collision resistance)
  - It is computationally infeasible to find any pair (x, y) such that H(x) = H(y) (strong collsion resistance)

Simple Hash Function

	bit 1	bit 2	• • •	bit *n*
block 1	b_{11}	b_{21}		b_{n1}
block 2	b_{12}	b_{22}		b_{n2}
	•	•	•	•
	•	•	•	•
	•	•	•	•
block *m*	b_{1m}	b_{2m}		b_{nm}
hash code	C_1	C_2		C_n

Figure 3.3 Simple Hash Function Using Bitwise XOR

- One-bit circular shift on the hash value after each block is processed would improve

Secure Hash Algorithm

Table 3.1 Comparison of SHA Parameters

	SHA-1	SHA-256	SHA-384	SHA-512
Message digest size	160	256	384	512
Message size	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block size	512	512	1024	1024
Word size	32	32	64	64
Number of steps	80	64	80	80
Security	80	128	192	256

Notes: 1. All sizes are measured in bits.
2. Security refers to the fact that a birthday attack on a message digest of size n produces a collision with a workfactor of approximately $2^{n/2}$.

Message Digest Generation Using SHA-512

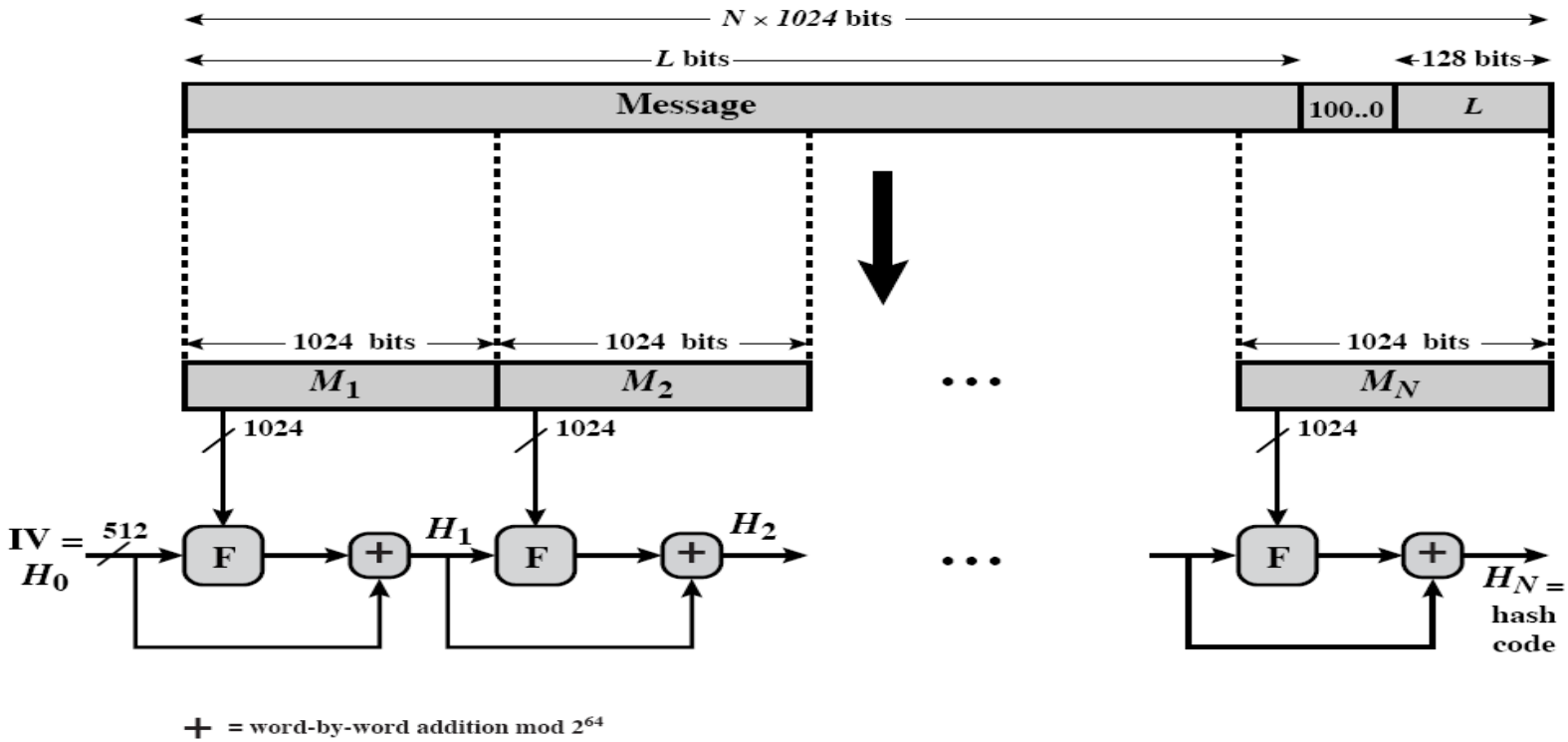
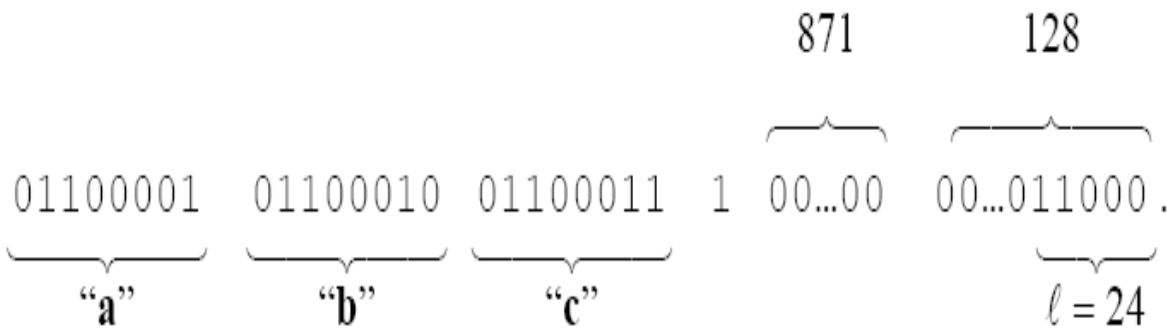


Figure 3.4 Message Digest Generation Using SHA-512

Single message Using SHA-512

Suppose the length of the message M, in bits, is ℓ bits. Append the bit "1" to the end of the message, followed by k zero bits, where k is the smallest non-negative solution to the equation $\ell + 1 + k \equiv 896 \bmod 1024$. Then append the 128-bit block that is equal to the number ℓ expressed using a binary representation. For example, the (8-bit ASCII) message "**abc**" has length $8 \times 3 = 24$, so the message is padded with a one bit, then $896 - (24 + 1) = 871$ zero bits, and then the message length, to become the 1024-bit padded message



The length of the padded message should now be a multiple of 1024 bits.

SHA-512 Processing of single 1024-Bit Block

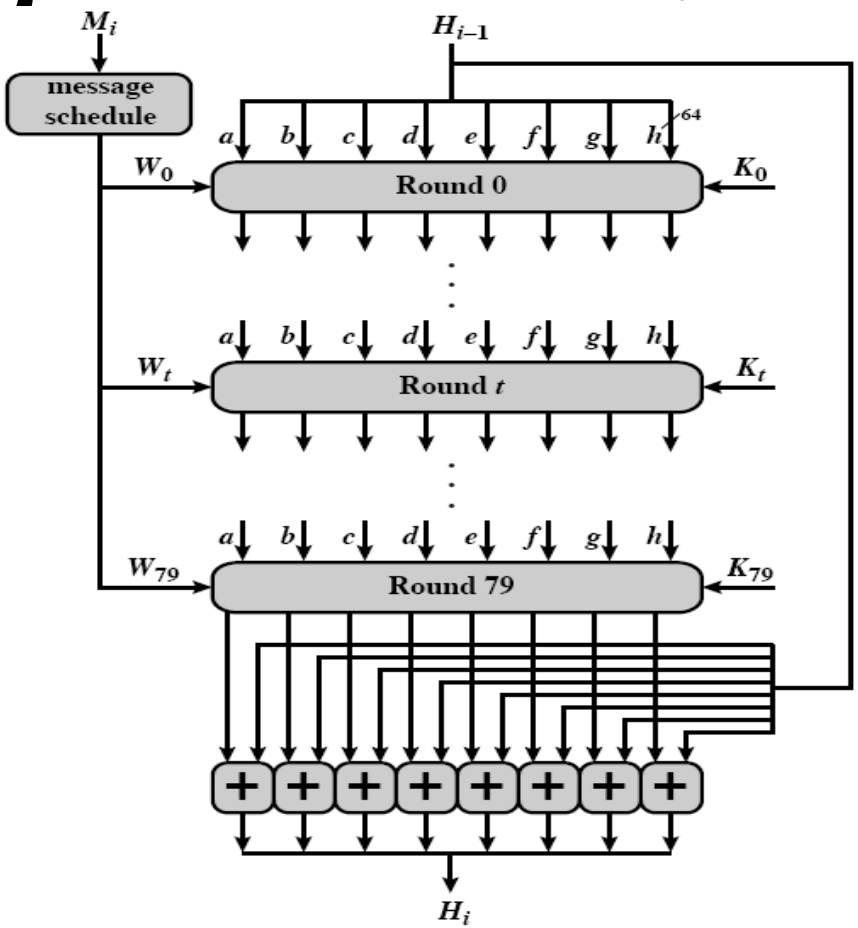


Figure 3.5 SHA-512 Processing of a Single 1024-Bit Block

Other Secure HASH functions

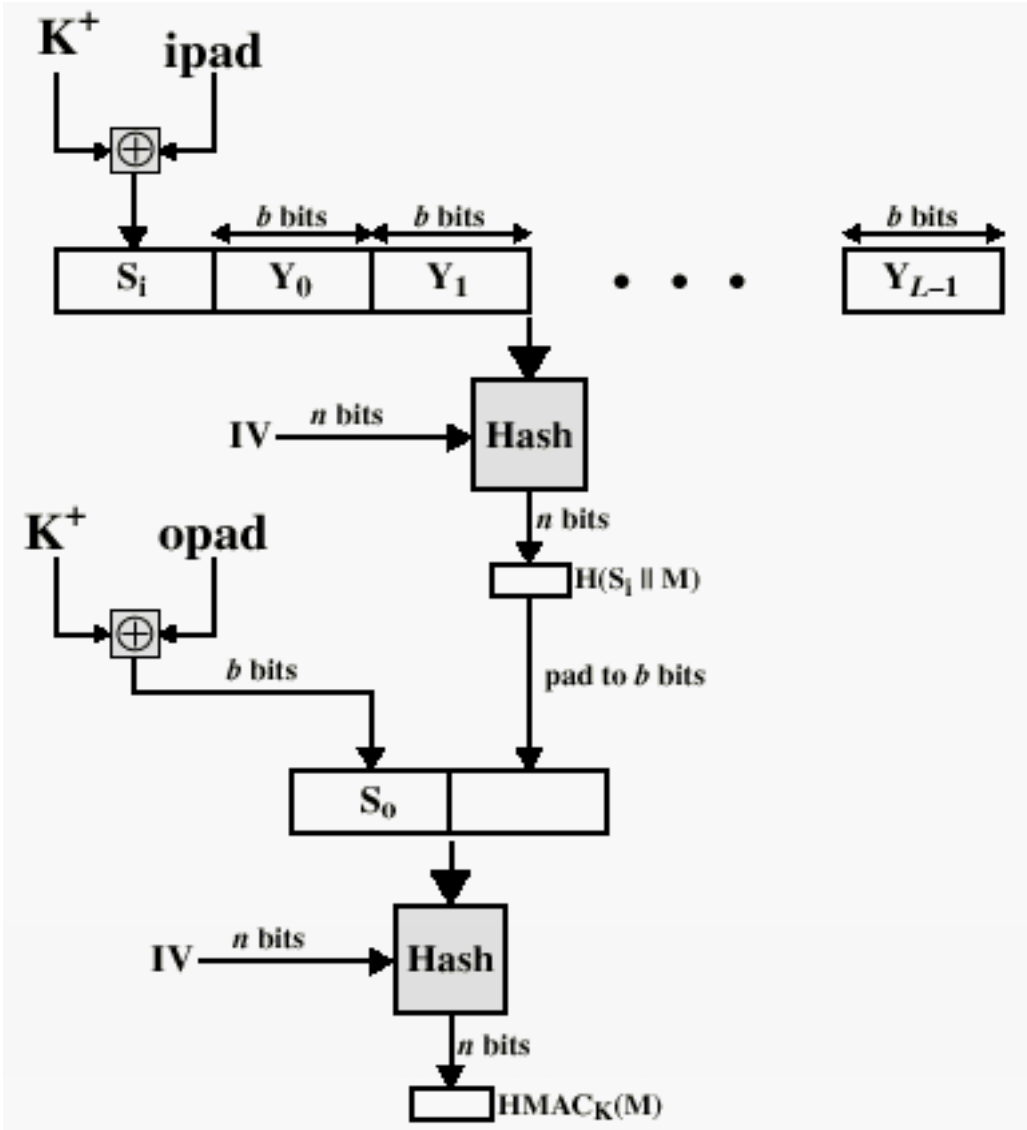
	SHA-1	MD5	RIPEMD-160
Digest length	160 bits	128 bits	160 bits
Basic unit of processing	512 bits	512 bits	512 bits
Number of steps	80 (4 rounds of 20)	64 (4 rounds of 16)	160 (5 paired rounds of 16)
Maximum message size	$2^{64}-1$ bits	∞	∞

H e

HMAC

- Use a MAC derived from a cryptographic hash code, such as SHA-1.
- **Motivations:**
  - Cryptographic hash functions executes faster in software than encryptoin algorithms such as DES
  - Library code for cryptographic hash functions is widely available
  - No export restrictions from the US

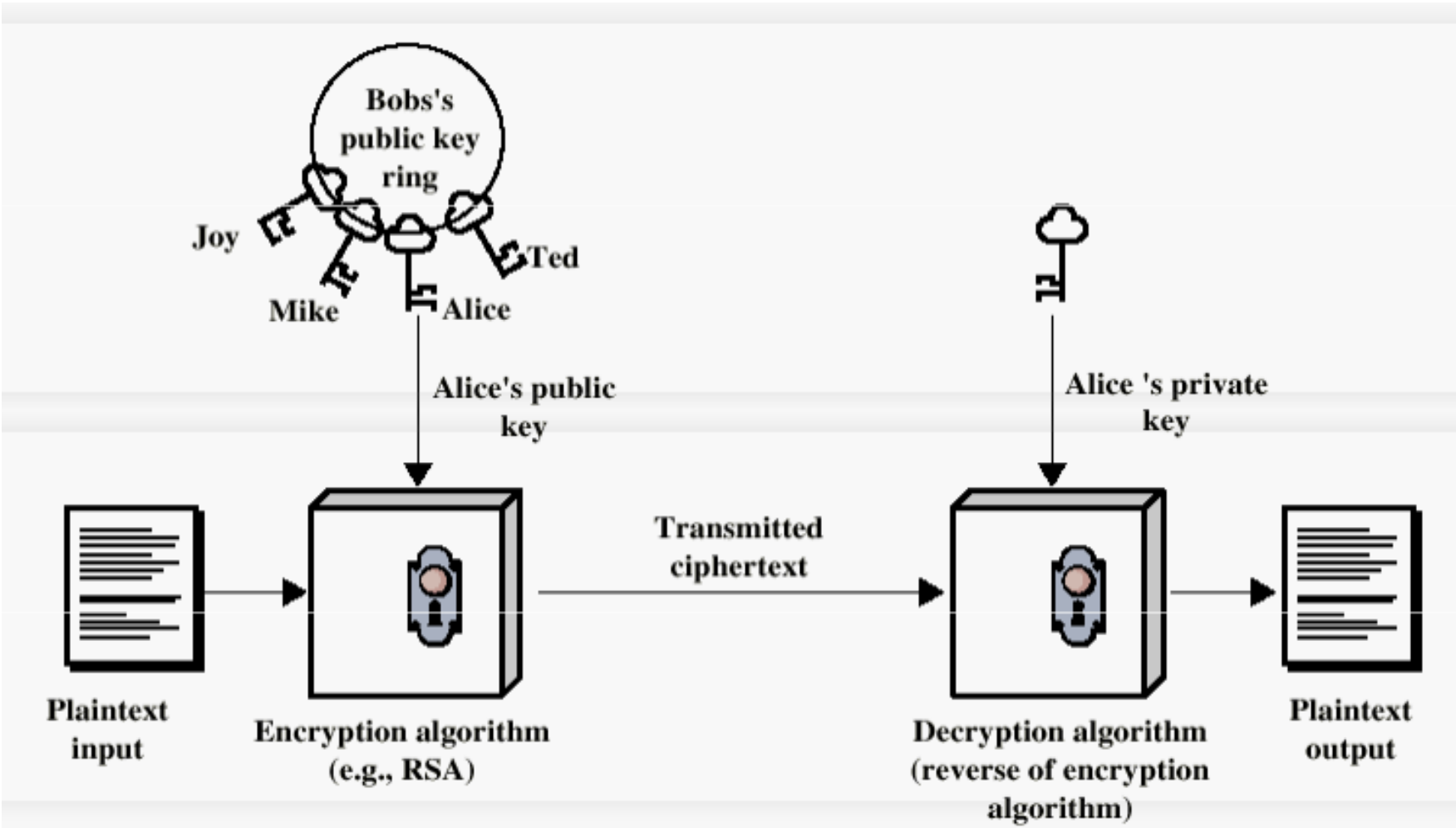
HMAC Structure



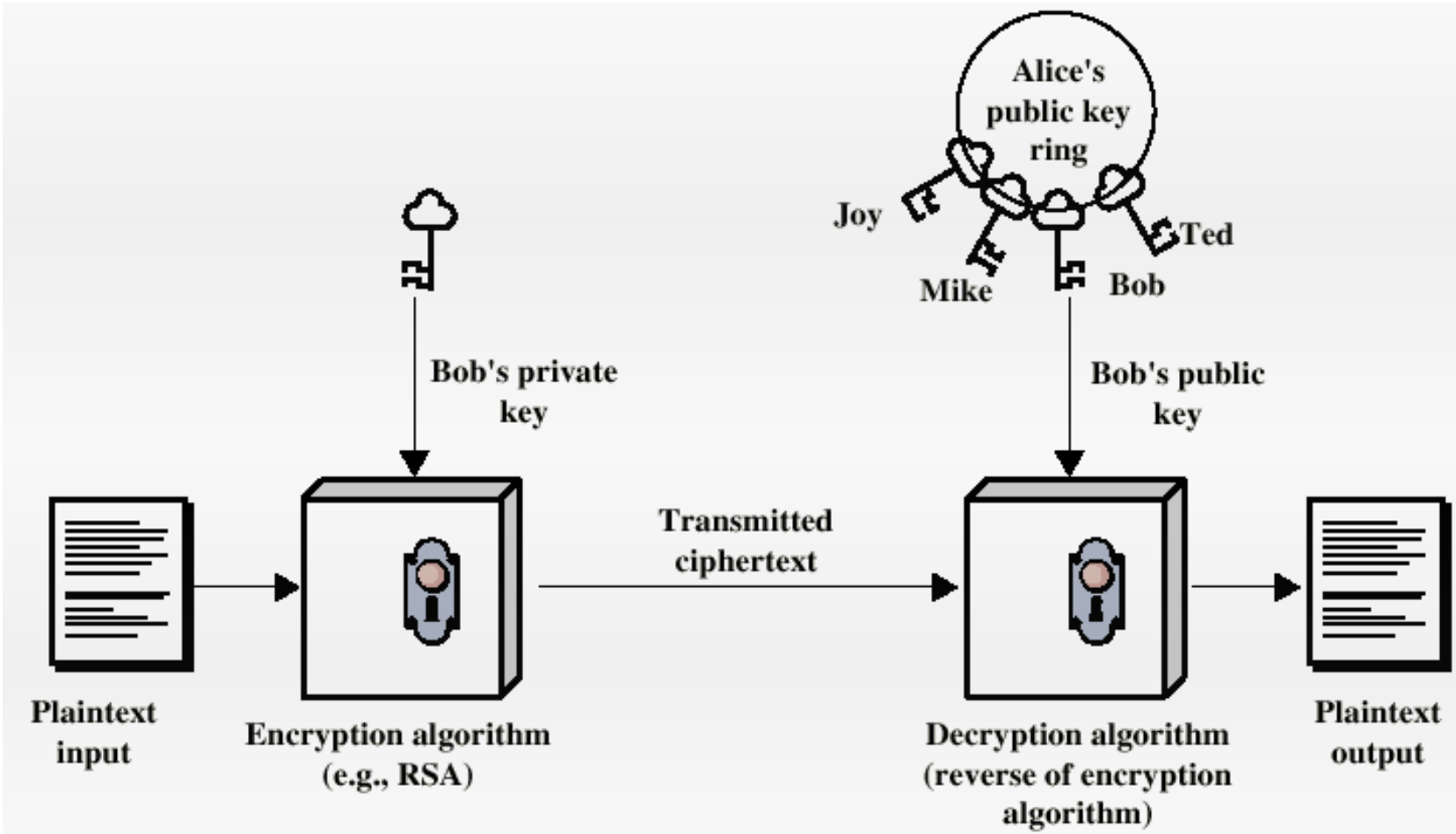
Public-Key Cryptography Principles

- The use of two keys has consequences in: key distribution, confidentiality and authentication.
- The scheme has six ingredients (see Figure 3.7)
  - Plaintext
  - Encryption algorithm
  - Public and private key
  - Ciphertext
  - Decryption algorithm

Encryption using Public-Key system



Authentication using Public-Key System



Applications for Public-Key Cryptosystems

- 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 echange:** Two sides cooperate to exhange a session key.

Requirements for Public-Key Cryptography

1. Computationally easy for a party B to generate a pair (public key KU_b, private key KR_b)
2. Easy for sender to generate ciphertext: $C = E_{KUb}(M)$
3. Easy for the receiver to decrypt ciphertect using private key:

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

Requirements for Public-Key Cryptography

4. Computationally infeasible to determine private key (KR_b) knowing public key (KU_b)
5. Computationally infeasible to recover message M, knowing KU_b and ciphertext C
6. Either of the two keys can be used for encryption, with the other used for decryption:

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

Public-Key Cryptographic Algorithms

- RSA and Diffie-Hellman
- **RSA** - Ron Rives, Adi Shamir and Len Adleman at MIT, in 1977.
  - RSA is a block cipher
  - The most widely implemented
- **Diffie-Hellman**
  - Echange a secret key securely
  - Compute discrete logarithms

The RSA Algorithm – Key Generation, Encryption & Decryption

1.	Select *p,q*	*p* and *q* both prime
2.	Calculate	$n = p \times q$
3.	Calculate	$\Phi(n) = (p-1)(q-1)$
4.	Select integer *e*	$\gcd(\Phi(n), e) = 1; 1 < e < \Phi(n)$
5.	Calculate *d*	$d = e^{-1} \bmod \Phi(n)$
6.	Public Key	KU = {e,n}
7.	Private key	KR = {d,n}
8.	Plaintext: M<n	Ciphertext: $C = M^e \pmod n$
9.	Ciphertext: *C*	Plaintext: $M = C^d \pmod n$

Example of RSA Algorithm

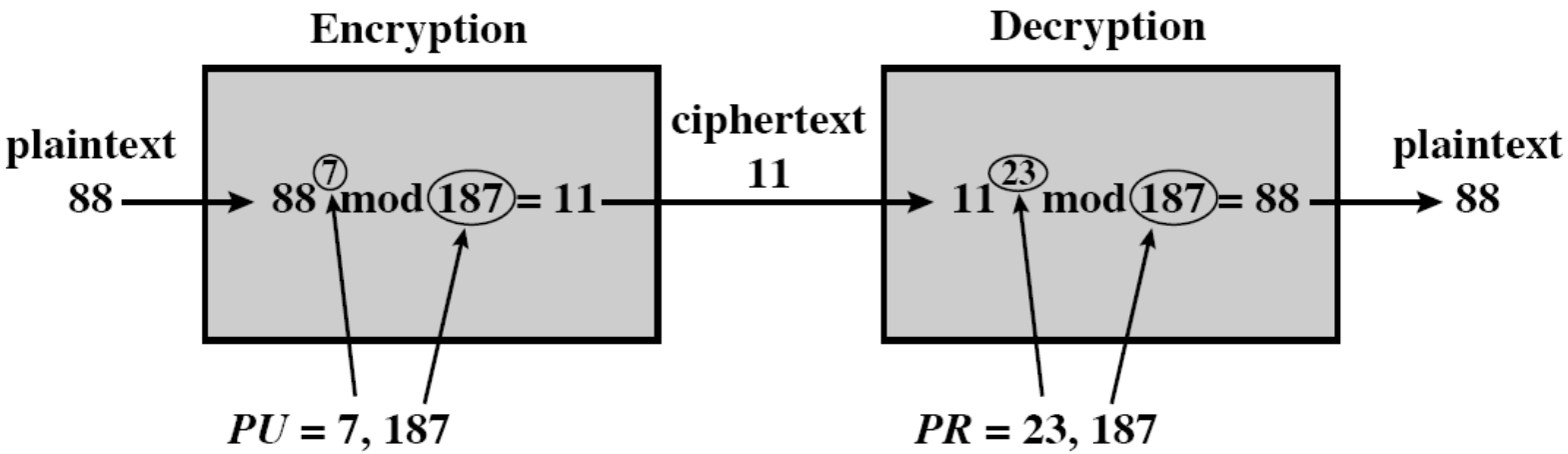
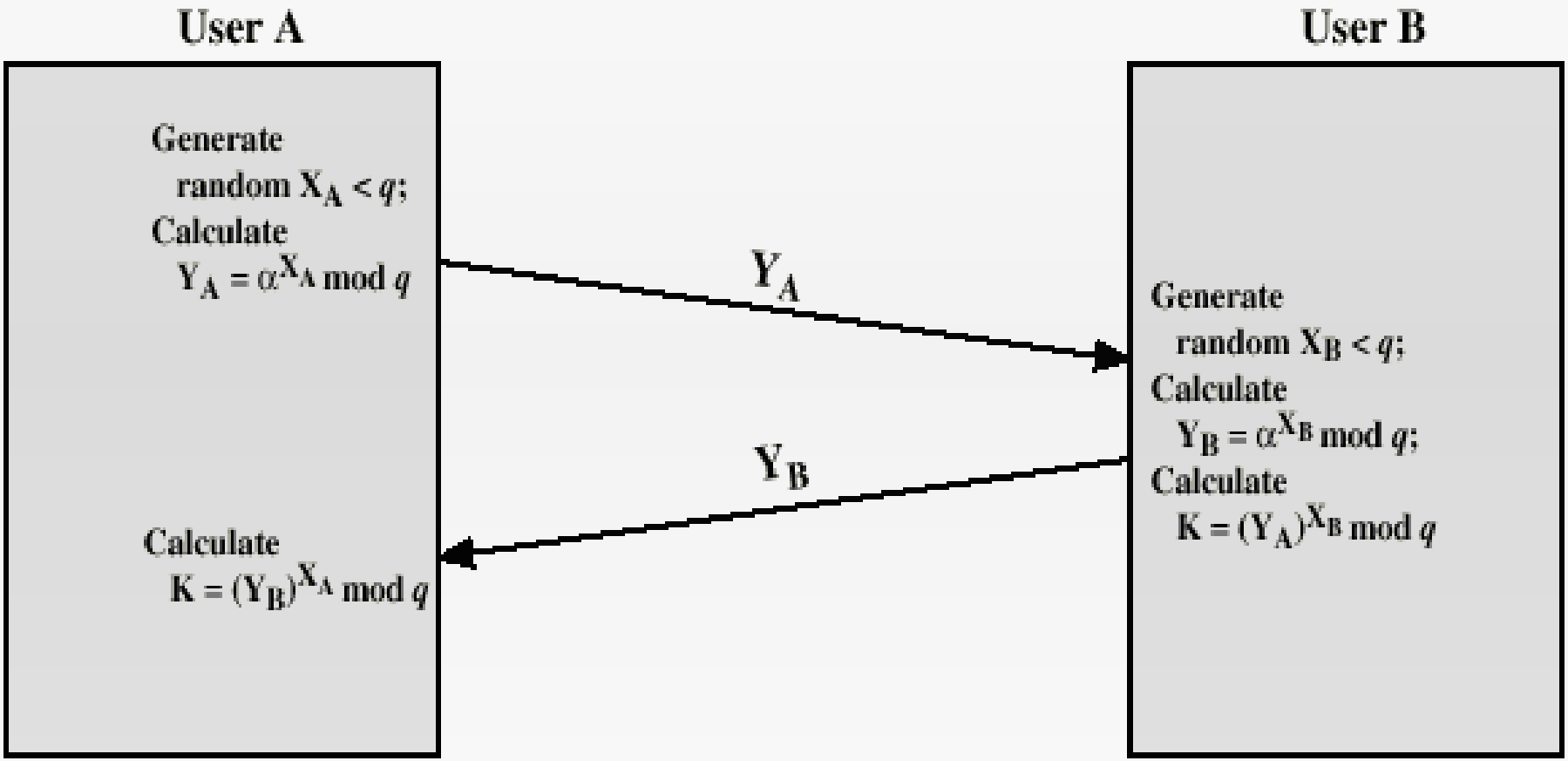


Figure 3.9 Example of RSA Algorithm

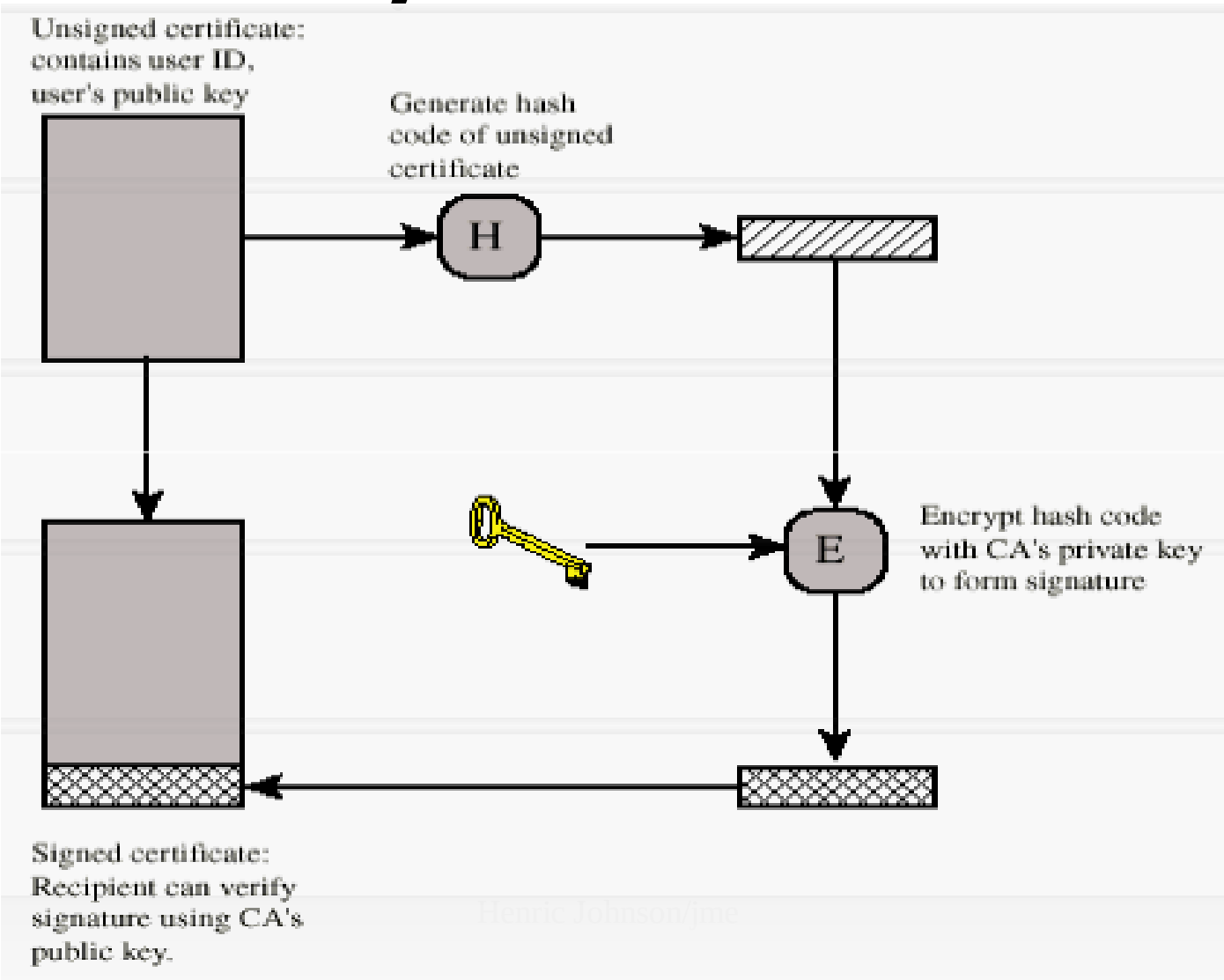
Diffie-Hellman Key Echange



Other Public-Key Cryptographic Algorithms

- Digital Signature Standard (DSS)
  - Makes use of the SHA-1
  - Not for encryption or key echange
- Elliptic-Curve Cryptography (ECC)
  - Good for smaller bit size
  - Low confidence level, compared with RSA
  - Very complex

Key Management
Public-Key Certificate Use



Digital signature = signed message digest