UNIVERSITY OF RAJSHAHI

Rajshahi, BANGLADESH.

Course Code:

ICE-4221

Course Title :

Cryptography and Network security

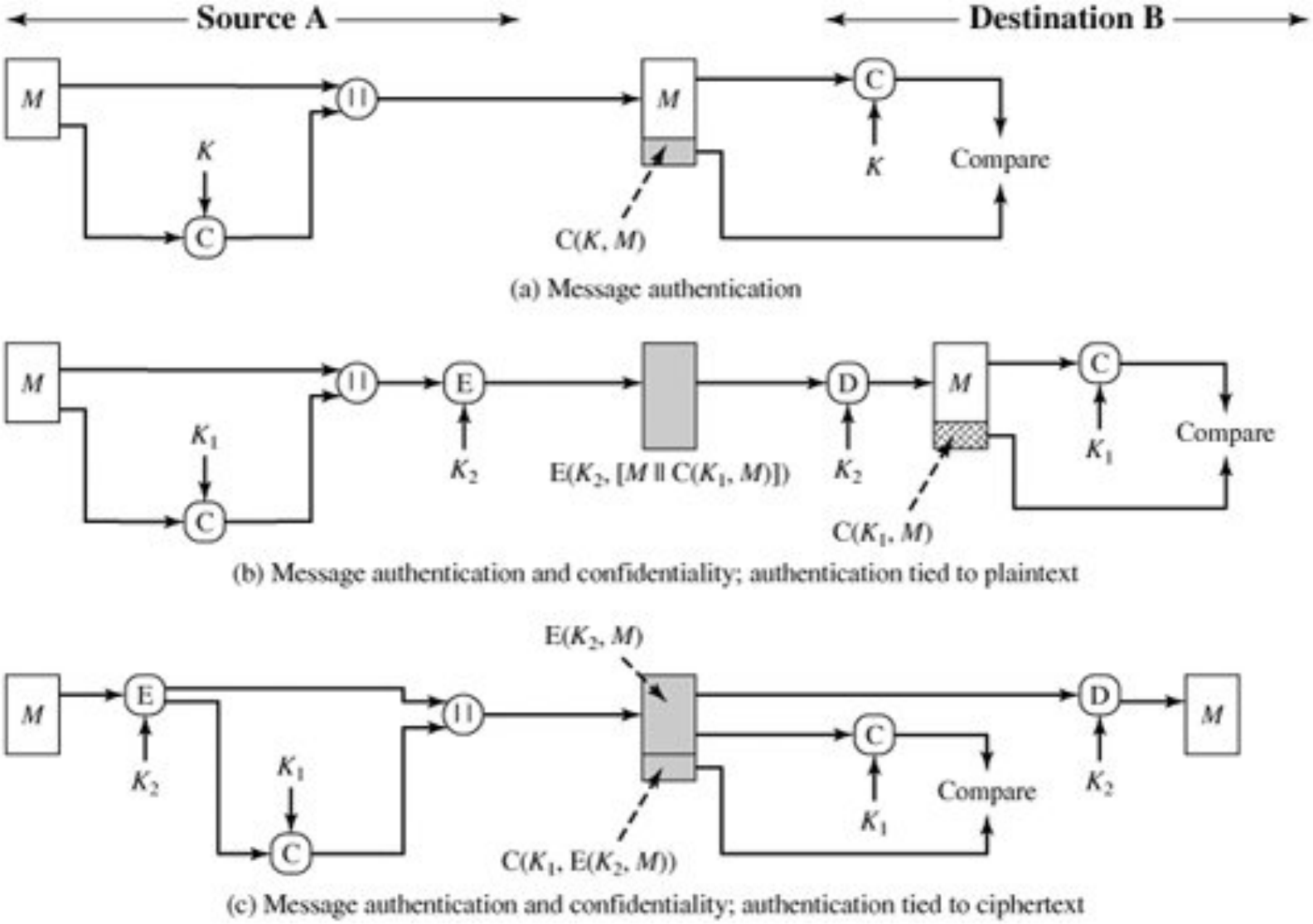
Digital Signatures,
And
Authentication Protocols

Hash Algorithm, Digital Signatures and Authentication Protocols: Secure hash algorithm, HMAC, HMAC design objectives, Digital signature, Authentication protocols, Digital signature standard, Mutual authentication, One-way authentication, Digital signature standard.

- **Message Authentication (?)**
  - It protects two parties from any third party who exchange messages.

- **However, it does not protect the two parties against each other.**

Possible Scenarios to Send Authenticated Message



For example, suppose that **John** sends an authenticated message to **Mary** using the mentioned scenarios.

✓ Many disputes that can arise…..

- Mary may forge a different message and claim that it came from John.
    - **Mary** would simply have to create a message and append an authentication code using the key that **John** and **Mary** share.
- Example:
    - An electronic funds transfer takes place, and the receiver increases the amount of funds transferred and claims that the larger amount had arrived from the sender.

✓ Many disputes that can arise…..

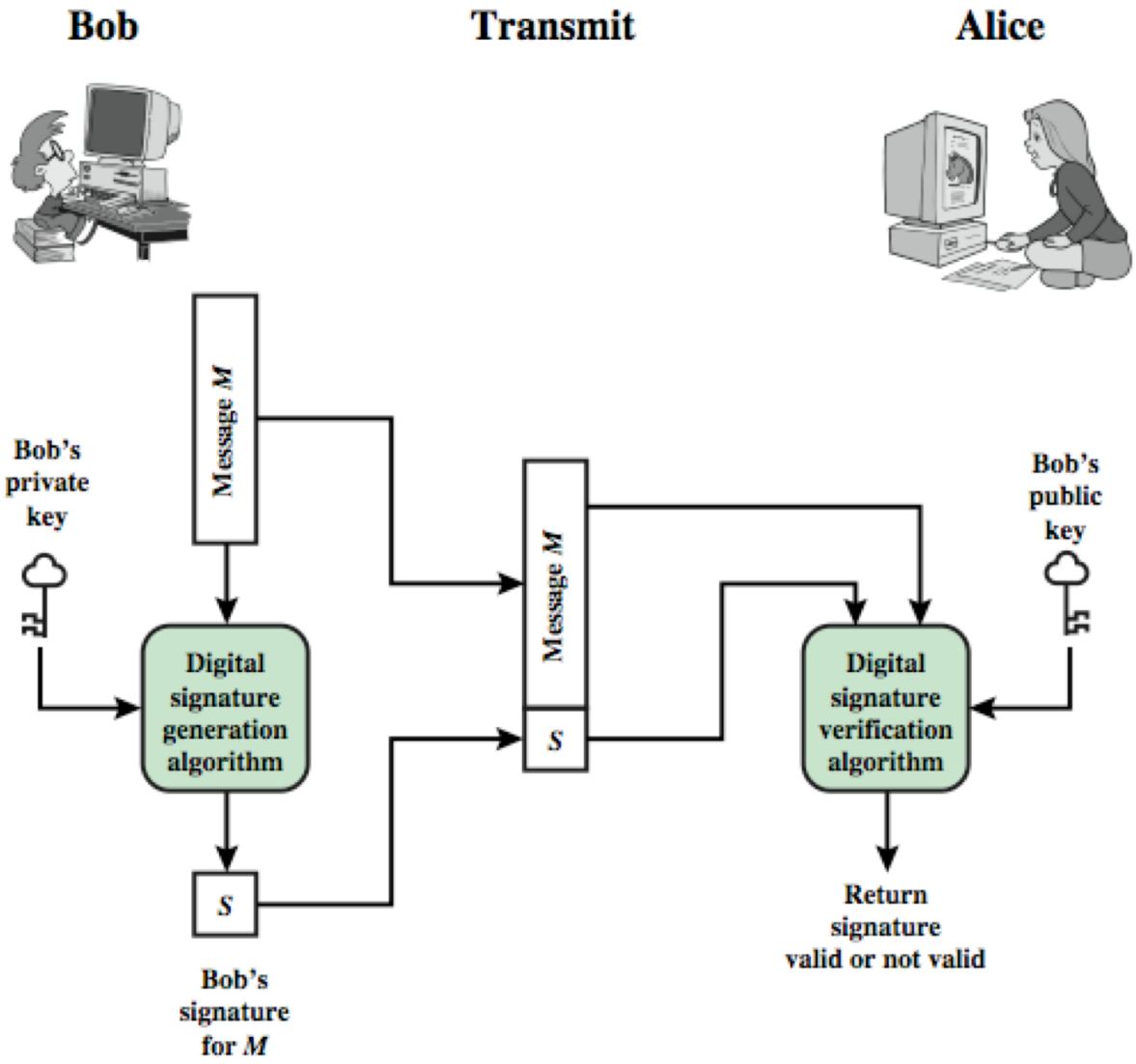
- John can deny sending the message.
    - Because it is possible for **Mary** to forge a message, there is no way to prove that **John** did in fact send the message.
- Example:
    - an electronic mail message contains instructions to a stockbroker for a transaction that subsequently turns out badly.
    - The sender pretends that the message was never sent.

- In situations where there is not complete trust between sender and receiver, **something more than authentication is needed.**
- The most attractive solution to this problem is the **Digital Signature**.
- The digital signature is analogous to the **Handwritten Signature**.

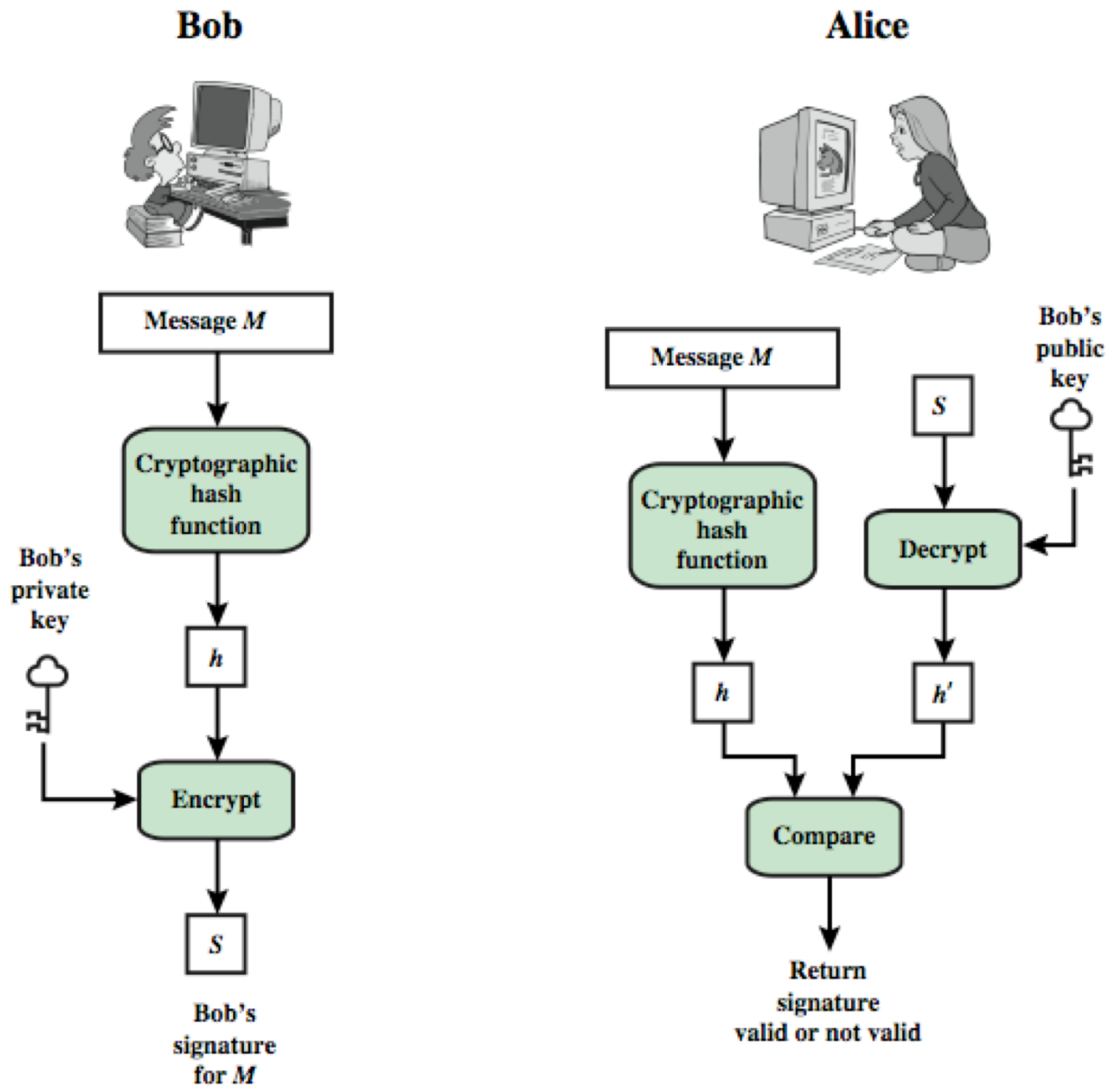
Digital Signatures

- have looked at message authentication
  - but does not address issues of lack of trust
- digital signatures provide the ability to:
  - verify author, date & time of signature
  - authenticate message contents
  - be verified by third parties to resolve disputes
- hence include authentication function with additional capabilities

Digital Signature Model



Digital Signature Model



Digital Signature Properties

Digital Signature function must have the following properties to ensure Authentication functions:

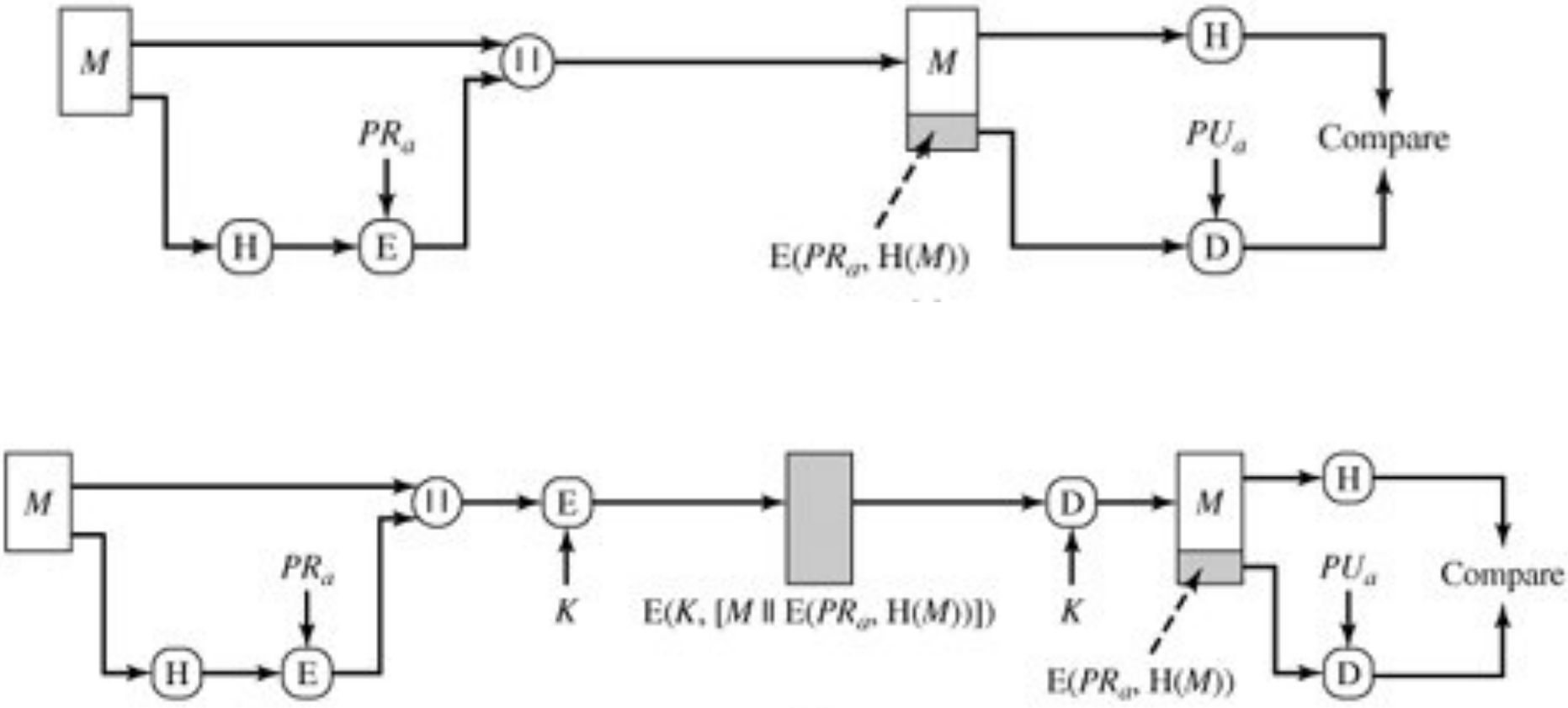
- It must verify the author and the date and time of the signature.
- It must to authenticate the contents at the time of the signature.
- It must be verifiable by third parties, to resolve disputes.

On the basis of these properties, there are some requirements for a Digital Signature.

Requirements for a Digital Signature

- The signature must be a bit pattern that depends on the message being signed.
- The signature must use some information unique to the sender, to prevent both forgery and denial.
- It must be relatively easy to produce the digital signature.
- It must be relatively easy to recognize and verify the digital signature.
- It must be computationally infeasible to forge a digital signature, either by constructing a new message for an existing digital signature or by constructing a fraudulent digital signature for a given message.
- It must be practical to retain a copy of the digital signature in storage.

A secure hash function, embedded in a scheme such as

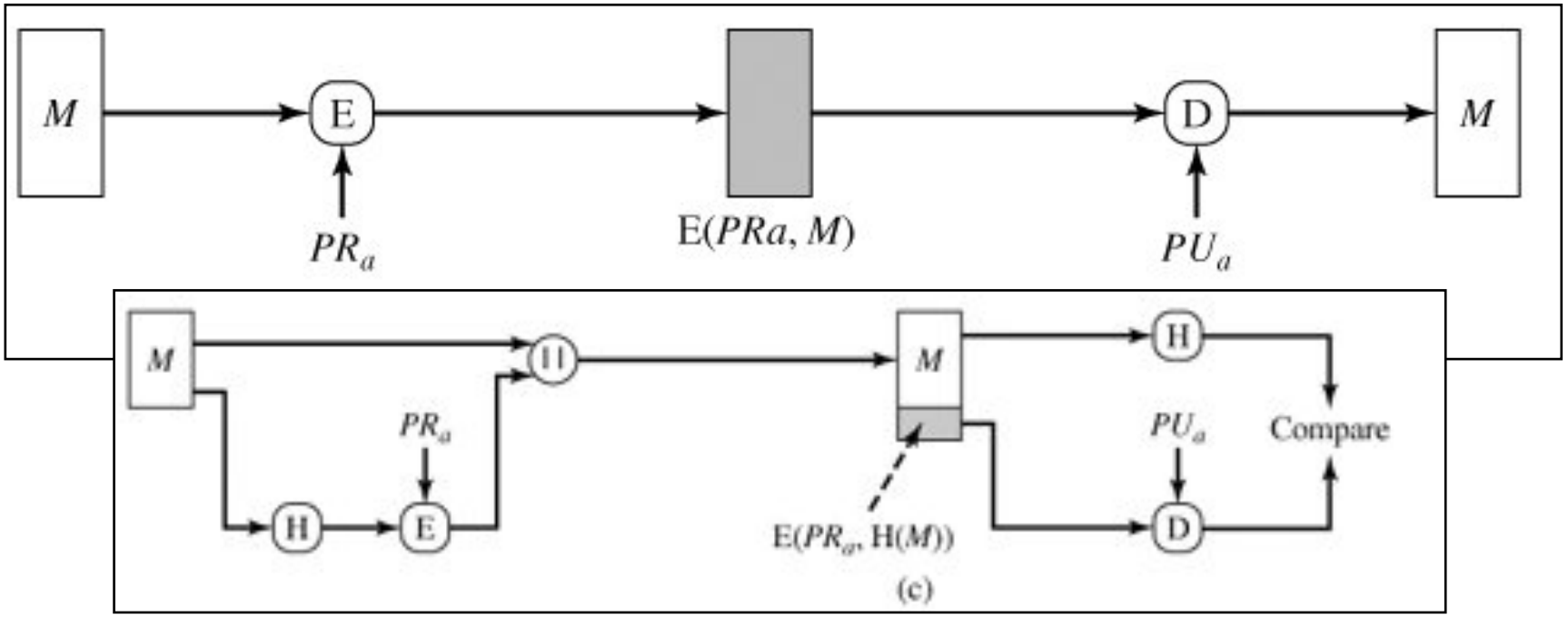


satisfies these requirements.

- A variety of approaches has been proposed for the digital signature function.
- These approaches fall into two categories:
    - Direct Digital Signature, and
    - Arbitrated Digital Signature.

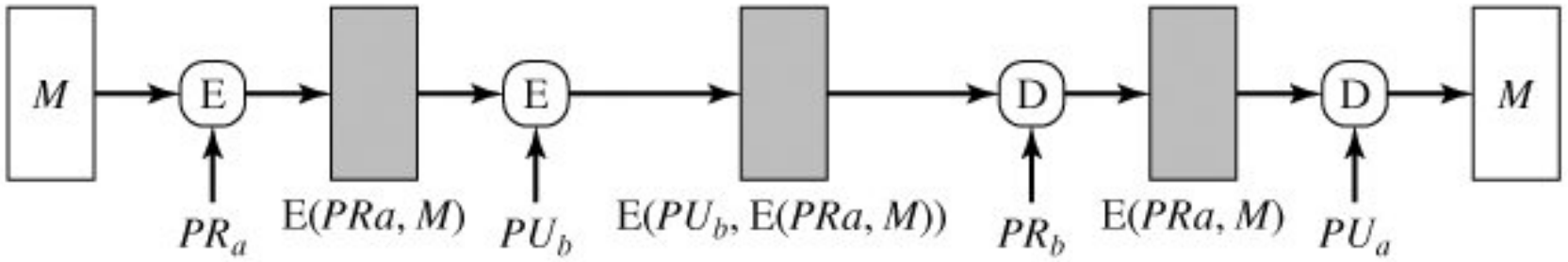
Direct Digital Signature

- The direct digital signature involves only the communicating parties (source, destination).
- It is assumed that the destination knows the public key of the source.

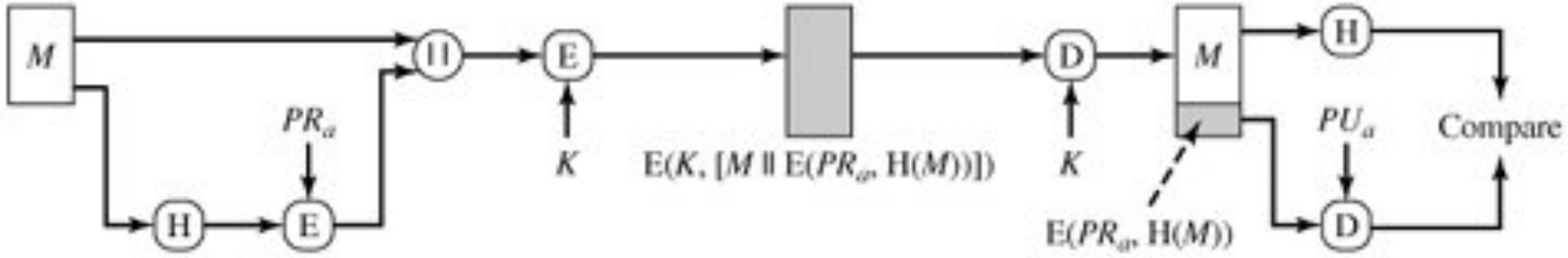


Direct Digital Signature

- Confidentiality can be provided by further encrypting the entire message plus signature with either the receiver's public key (public-key encryption) or a shared secret key (symmetric encryption);



(d) Public-key encryption: confidentiality, authentication, and signature



(d)

- It is important to perform the signature function first and then an outer confidentiality function.

Why?

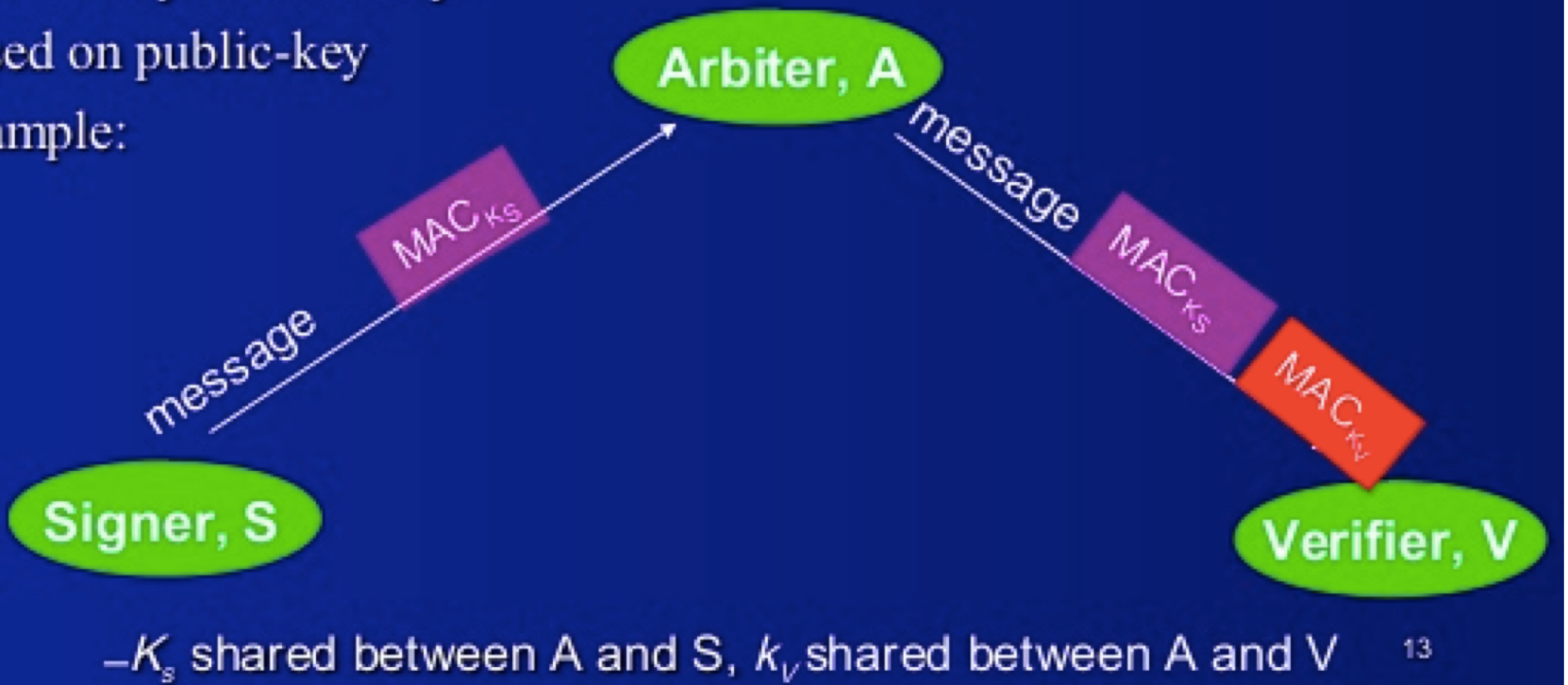
Common Weakness of DSS

- The validity of the scheme depends on the security of the sender's private key.
  - *If a sender later wishes to deny sending a particular message, the sender can claim that the private key was lost or stolen and that someone else forged his or her signature.*
- Some private key might actually be stolen from X at time T.
- The opponent can then send a message signed with X's signature and stamped with a time before or equal to T.

Arbitrated Digital Signature

Arbitrated digital signature is based on a trusted third party (arbiter). There are 2 types:

- Based on symmetric key
- Based on public-key
- Example:



– K_s shared between A and S, k_V shared between A and V

13

Arbitrated Digital Signature

Under this Digital Signature Scheme…..

- Every signed message from a sender X to a receiver Y goes first to an arbiter, who subjects the message and its signature to a number of tests to check its origin and contents.
- The message is then dated and sent to Y with the indication that it has been verified to the satisfaction of the arbiter.
- The presence of A solve the problem faced by DDS.

Arbitrated Digital Signature

- Involves use of arbiter A
  - validates any signed message
  - then dated and sent to recipient
- Requires a great deal of trust in arbiter
- Can be implemented with either private or public-key algorithms
- Arbiter may or may not see message

Arbitrated Digital Signature

(a) Conventional Encryption, Arbiter Sees Message
(1) X → A: $M \| \mathrm{E}_{K_{xa}}[ID_X \| \mathrm{H}(M)]$
(2) A → Y: $\mathrm{E}_{K_{ay}}[ID_X \| M \| \mathrm{E}_{K_{xa}}[ID_X \| \mathrm{H}(M)] \| T]$
(b) Conventional Encryption, Arbiter Does Not See Message
(1) X → A: $ID_X \| \mathrm{E}_{K_{xy}}[M] \| \mathrm{E}_{K_{xa}}[ID_X \| \mathrm{H}(\mathrm{E}_{K_{xy}}[M])]$
(2) A → Y: $\mathrm{E}_{K_{ay}}[ID_X \| \mathrm{E}_{K_{xy}}[M] \| \mathrm{E}_{K_{xa}}[ID_X \| \mathrm{H}(\mathrm{E}_{K_{xy}}[M])] \| T]$
(c) Public-Key Encryption, Arbiter Does Not See Message
(1) X → A: $ID_X \| \mathrm{E}_{KR_x}[ID_X \| \mathrm{E}_{KU_y}(\mathrm{E}_{KR_x}[M])]$
(2) A → Y: $\mathrm{E}_{KR_a}[ID_X \| \mathrm{E}_{KU_y}[\mathrm{E}_{KR_x}[M]] \| T]$

Notation:

X = sender
Y = recipient
A = Arbiter

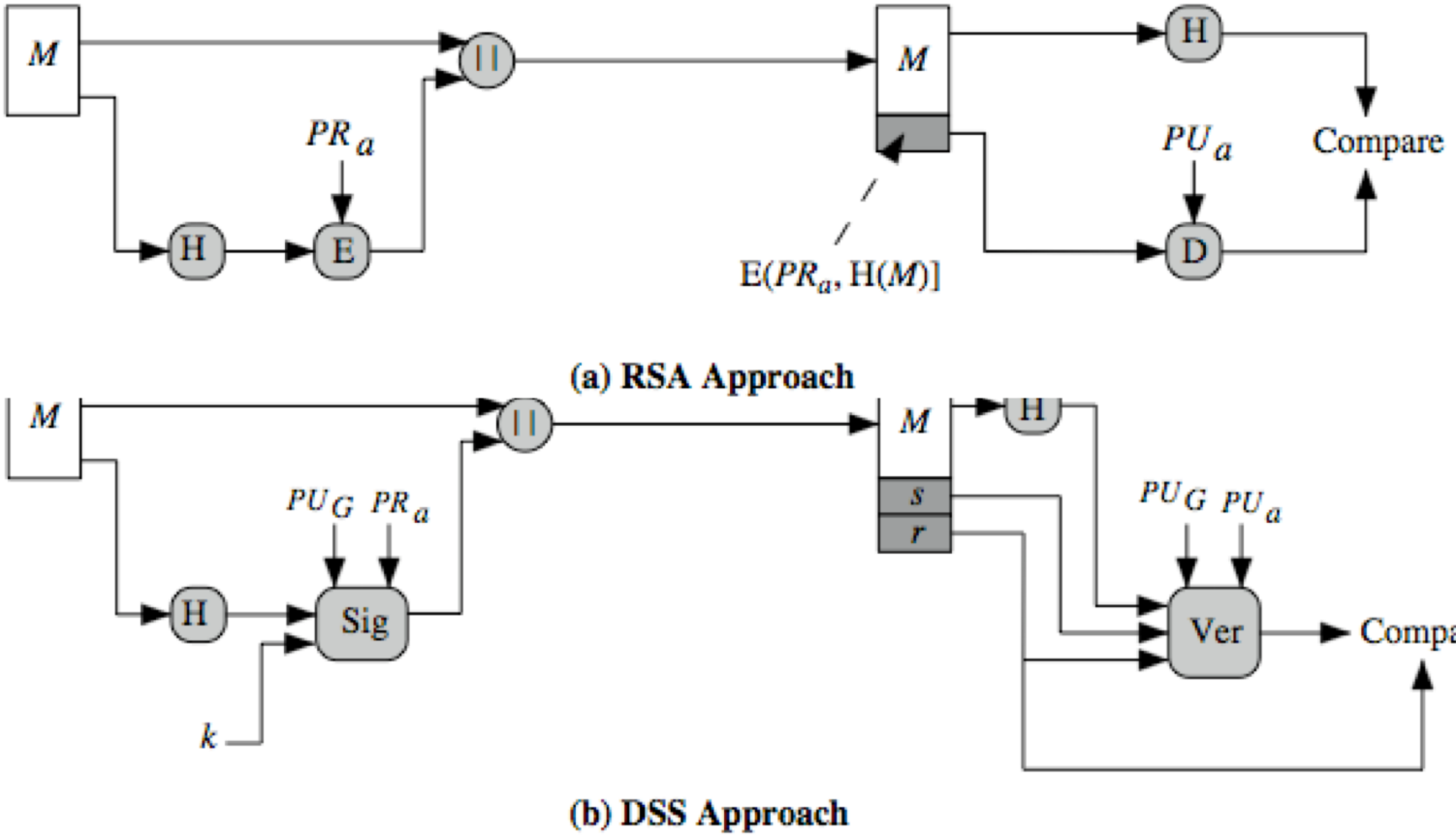
M = message
T = timestamp

- The timestamp informs Y that this message is timely and not a replay.

Digital Signature Standard (DSS)

- US Govt approved signature scheme
- designed by NIST & NSA in early 90's
- published as FIPS-186 in 1991
- revised in 1993, 1996 & then 2000
- uses the SHA hash algorithm
- DSS is the standard, DSA is the algorithm
- FIPS 186-2 (2000) includes alternative RSA & elliptic curve signature variants
- DSA is digital signature only unlike RSA
- is a public-key technique

DSS vs RSA Signatures



The signature function also depends on the sender's private key (PR_a) and a set of parameters known to a group of communicating principals. We can consider this set to constitute a global public key (PU_G). The result is a signature consisting of two components, labeled s and r.

Digital Signature Algorithm (DSA)

- creates a 320 bit signature
- with 512-1024 bit security
- smaller and faster than RSA
- a digital signature scheme only
- security depends on difficulty of computing discrete logarithms
- variant of ElGamal & Schnorr schemes

Digital Signature Algorithm (DSA)

Global Public-Key Components

p prime number where $2^{L-1} < p < 2^L$ for $512 \leq L \leq 1024$ and L a multiple of 64; i.e., bit length of between 512 and 1024 bits in increments of 64 bits

q prime divisor of $(p - 1)$, where $2^{159} < q < 2^{160}$; i.e., bit length of 160 bits

g $= h^{(p-1)/q} \bmod p$, where h is any integer with $1 < h < (p - 1)$ such that $h^{(p-1)/q} \bmod p > 1$

User's Private Key

x random or pseudorandom integer with $0 < x < q$

User's Public Key

$y = g^x \bmod p$

User's Per-Message Secret Number

k = random or pseudorandom integer with $0 < k < q$

Signing

$r = (g^k \bmod p) \bmod q$

$s = [k^{-1} (\mathrm{H(M)} + xr)] \bmod q$

Signature = (r, s)

Verifying

$w = (s')^{-1} \bmod q$

$u_1 = [\mathrm{H(M')}w] \bmod q$

$u_2 = (r')w \bmod q$

$v = [(g^{u1} y^{u2}) \bmod p] \bmod q$

TEST: $v = r'$

M = message to be signed

$\mathrm{H}(M)$ = hash of M using SHA-1

M', r', s' = received versions of M, r, s

Figure 13.4 The Digital Signature Algorithm (DSA)

DSA Key Generation

- have shared global public key values (p,q,g):
  - choose 160-bit prime number q
  - choose a large prime p with $2^{L-1} < p < 2^{L}$
    - where L= 512 to 1024 bits and is a multiple of 64
    - such that q is a 160 bit prime divisor of $(p-1)$
  - choose $g = h^{(p-1)/q}$
    - where $1<h<p-1$ and $h^{(p-1)/q} \bmod p > 1$
- users choose private & compute public key:
  - choose random private key: $x<q$
  - compute public key: $y = g^{x} \bmod p$

DSA Signature Creation

- to **sign** a message M the sender:
  - generates a random signature key k, $k<q$
  - nb. k must be random, be destroyed after use, and never be reused
- then computes signature pair:

$$r = (g^k \bmod p) \bmod q$$

$$s = [k^{-1}(H(M) + xr)] \bmod q$$

- sends signature (r,s) with message M

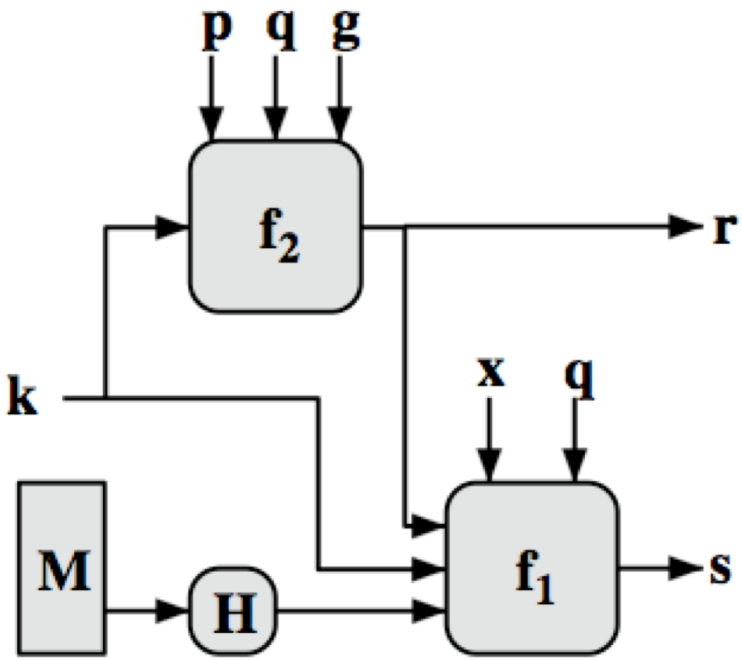
DSA Signature Verification

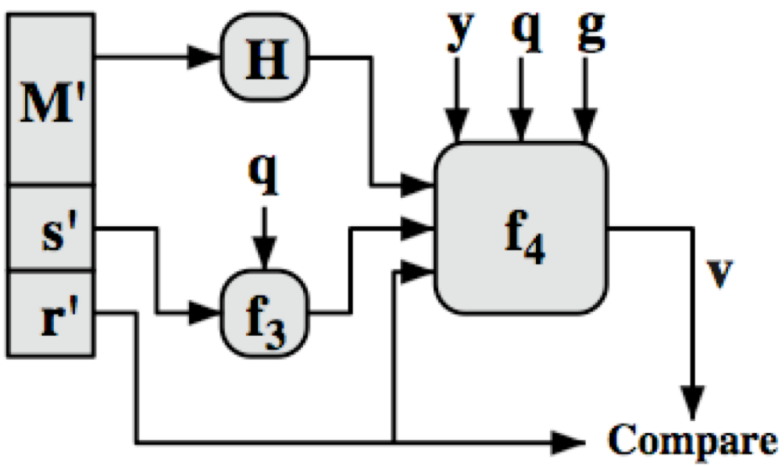
- having received M & signature (r,s)
- to **verify** a signature, recipient computes:

$$w = s^{-1} \bmod q$$
$$u1 = [H(M)w] \bmod q$$
$$u2 = (rw) \bmod q$$
$$v = [(g^{u1}\ y^{u2}) \bmod p] \bmod q$$

- if $v=r$ then signature is verified
- see Appendix A for details of proof why

DSS Overview





$s = f_1(H(M), k, x, r, q) = (k^{-1} (H(M) + xr)) \bmod q$

$r = f_2(k, p, q, g) = (g^k \bmod p) \bmod q$

(a) Signing

$w = f_3(s', q) = (s')^{-1} \bmod q$

$v = f_4(y, q, g, H(M'), w, r')$

$= ((g^{(H(M')w) \bmod q} y^{r'w \bmod q}) \bmod p) \bmod q$

(b) Verifying