Digital signature and authentication protocols

- General Model
- Properties of Digital Signature
- Attack Possibilities in DS
- Digital Signature Standard Algorithm
  - EIGamal
  - Schnorr.
- DSS

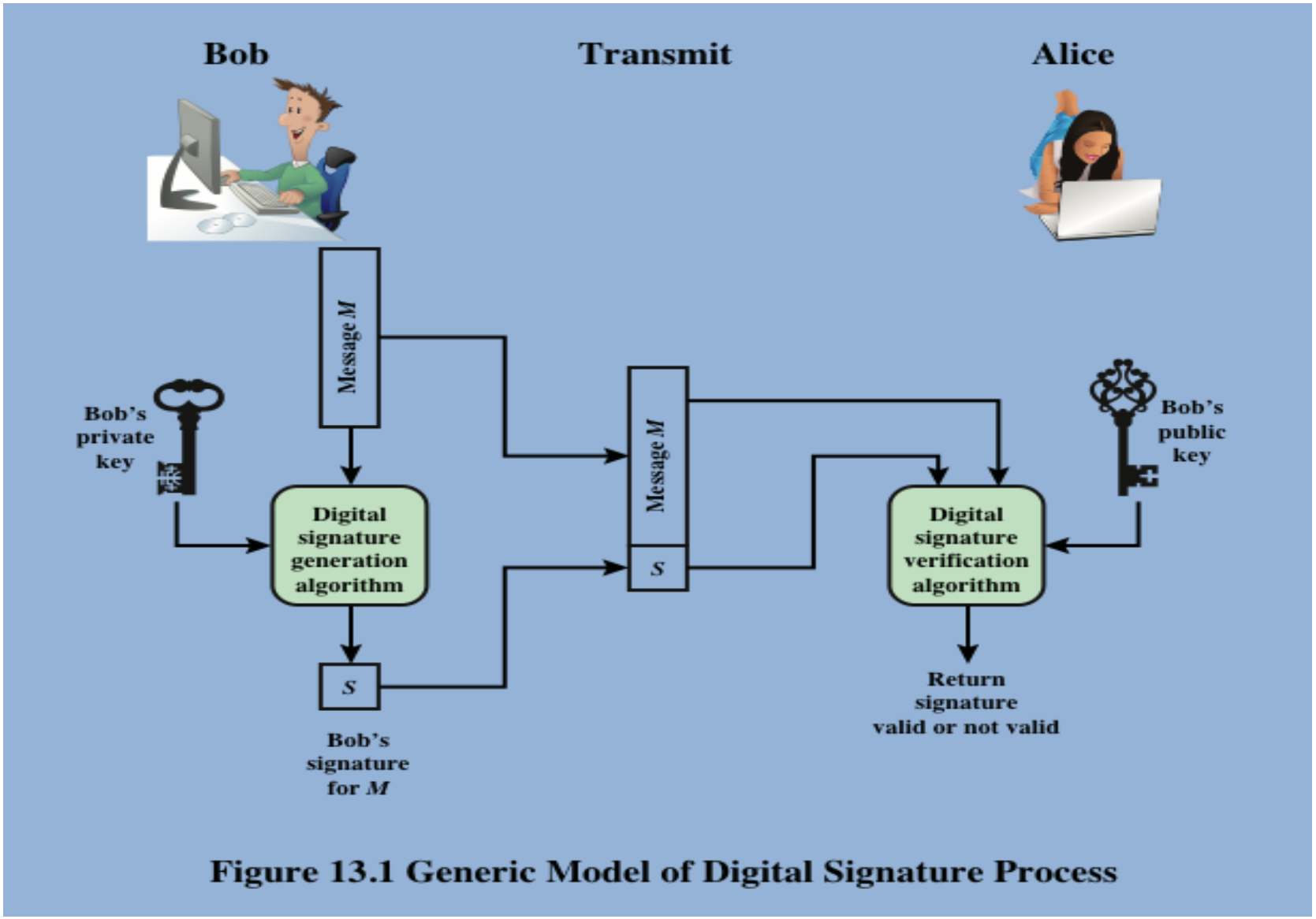


Figure 13.1 Generic Model of Digital Signature Process

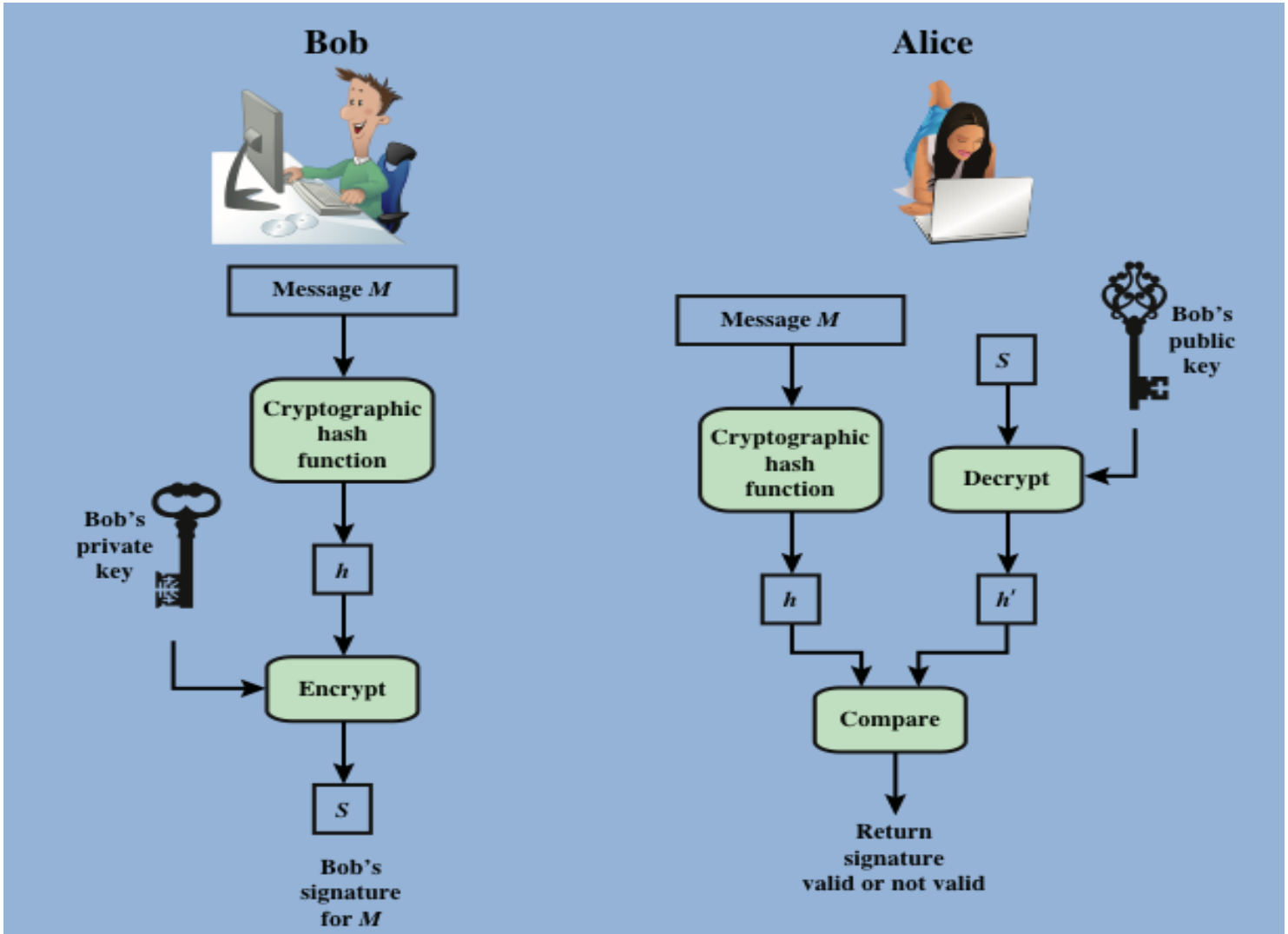
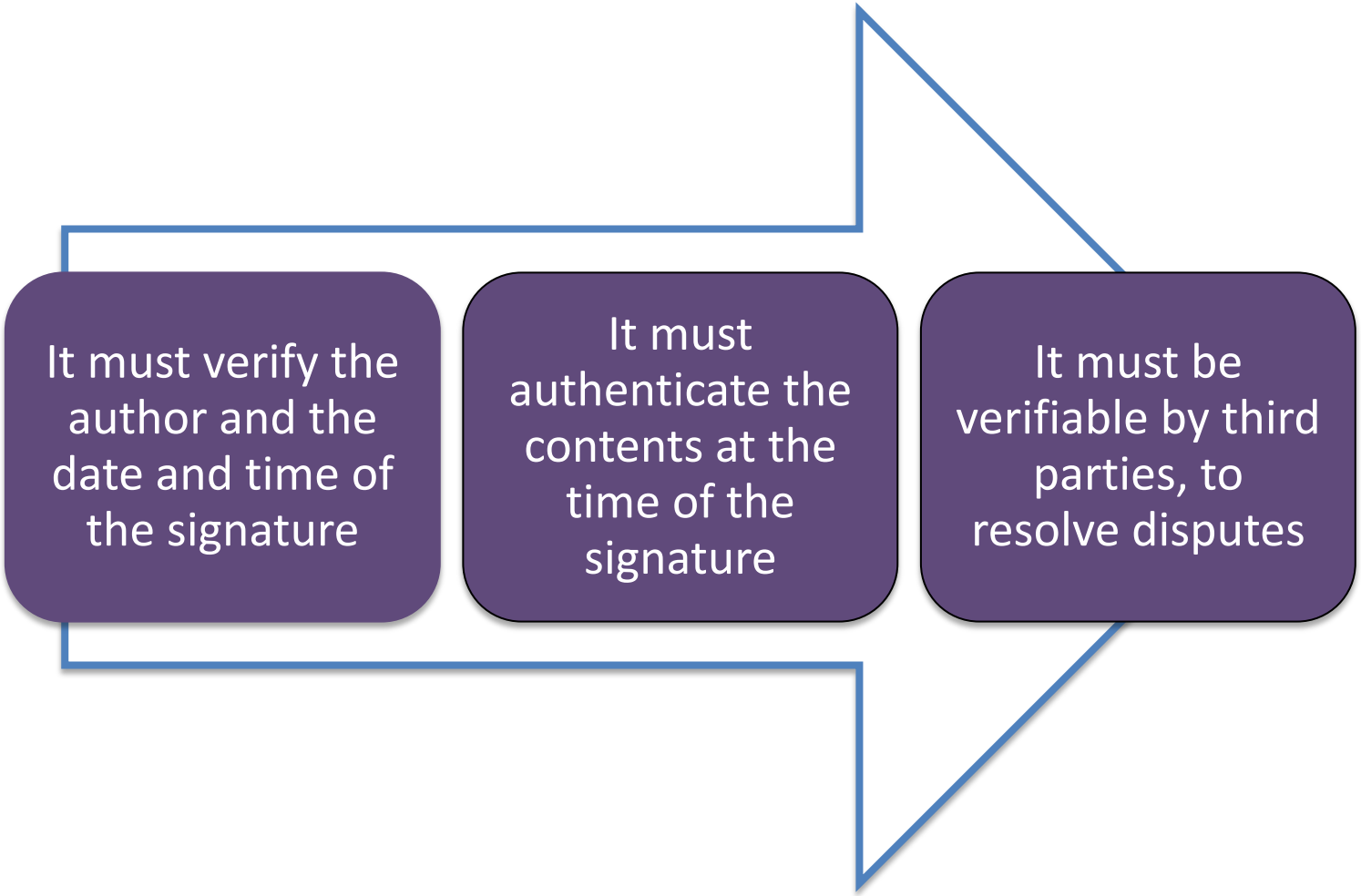
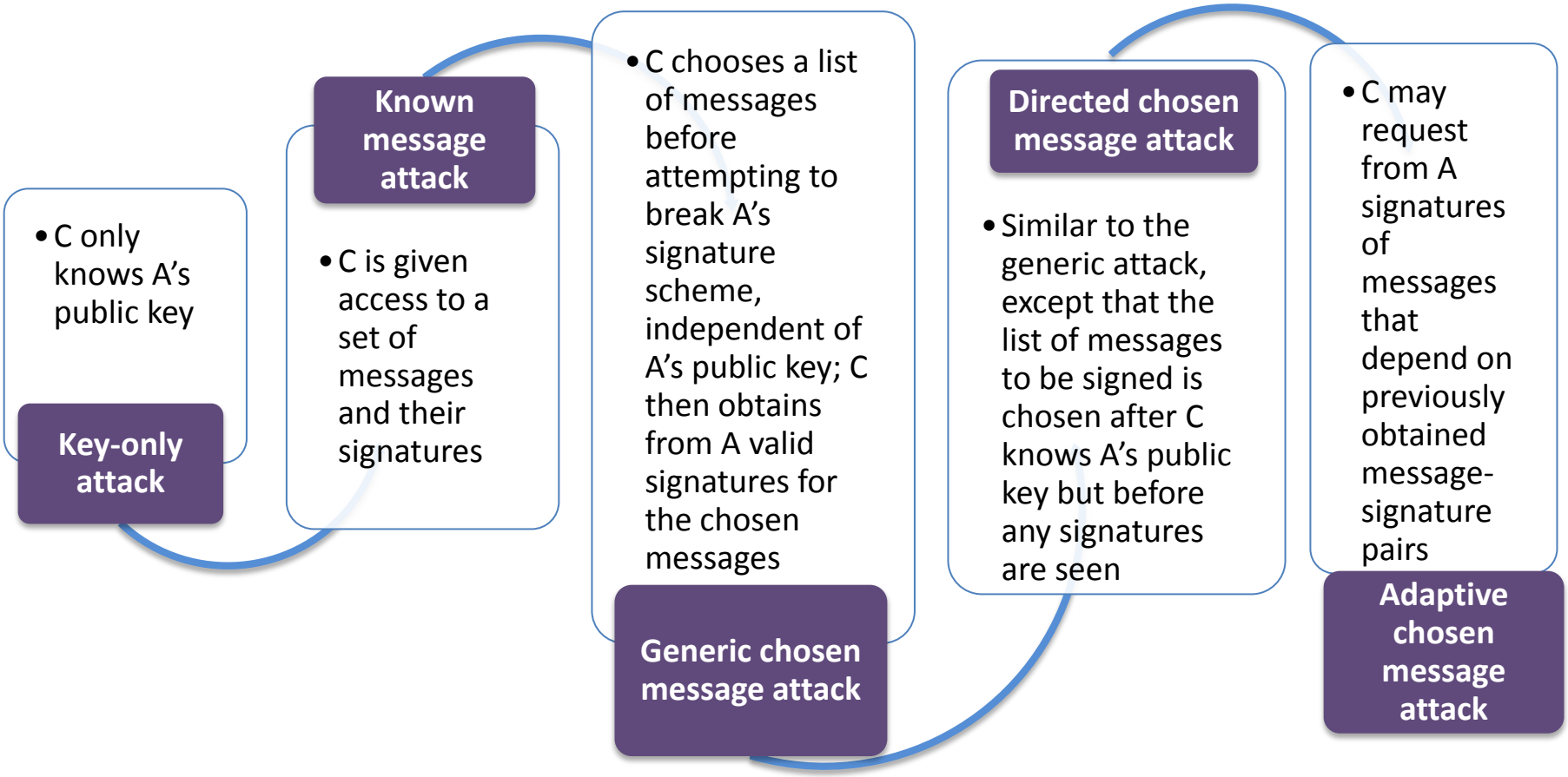


Figure 13.2 Simplified Depiction of Essential Elements of Digital Signature Process

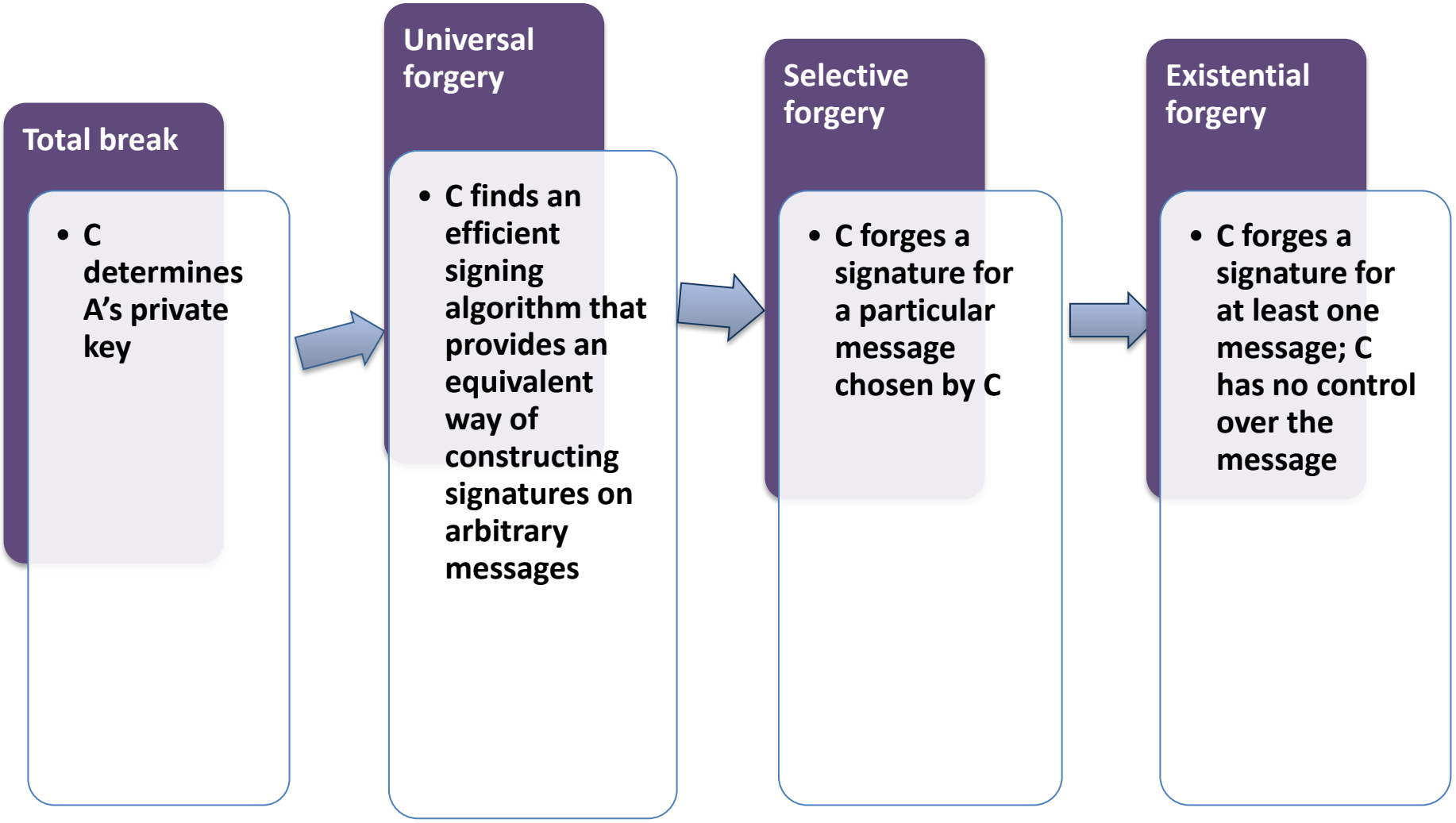
Digital Signature Properties



Attacks


• C only knows A's public key
Key-only attack
Known message attack
• C is given access to a set of messages and their signatures
• C chooses a list of messages before attempting to break A's signature scheme, independent of A's public key; C then obtains from A valid signatures for the chosen messages
Generic chosen message attack
Directed chosen message attack
• Similar to the generic attack, except that the list of messages to be signed is chosen after C knows A's public key but before any signatures are seen
• C may request from A signatures of messages that depend on previously obtained message-signature pairs
Adaptive chosen message attack

Forgeries


Total break
• C determines A's private key
Universal forgery
• C finds an efficient signing algorithm that provides an equivalent way of constructing signatures on arbitrary messages
Selective forgery
• C forges a signature for a particular message chosen by C
Existential forgery
• C forges a signature for at least one message; C has no control over the message

Digital Signature Requirements

- 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

Direct Digital Signature

- Refers to a digital signature scheme that involves only the communicating parties
  - It is assumed that the destination knows the public key of the source
- Confidentiality can be provided by encrypting the entire message plus signature with a shared secret key
  - It is important to perform the signature function first and then an outer confidentiality function
  - In case of dispute some third party must view the message and its signature
- 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
  - One way to thwart or at least weaken this ploy is to require every signed message to include a timestamp and to require prompt reporting of compromised keys to a central authority

ElGamal Digital Signature

- Scheme involves the use of the private key for encryption and the public key for decryption
- Global elements are a prime number q and a, which is a primitive root of q
- Use private key for encryption (signing)
- Uses public key for decryption (verification)

Elgamal digital signature

- Select a prime number q and a, where a is primitive root of q.

1. Generate a random integer X_A, such that $1 < X_A < q - 1$.
2. Compute $Y_A = \alpha^{X_A} \bmod q$.
3. A's private key is X_A; A's pubic key is $\{q, \alpha, Y_A\}$.

To sign a message M, user A first computes the hash $m = \mathrm{H}(M)$, such that m is an integer in the range $0 \leq m \leq q - 1$. A then forms a digital signature as follows.

1. Choose a random integer K such that $1 \leq K \leq q - 1$ and $\gcd(K, q - 1) = 1$. That is, K is relatively prime to $q - 1$.
2. Compute $S_1 = \alpha^K \bmod q$. Note that this is the same as the computation of C_1 for Elgamal encryption.
3. Compute $K^{-1} \bmod (q - 1)$. That is, compute the inverse of K modulo $q - 1$.
4. Compute $S_2 = K^{-1}(m - X_A S_1) \bmod (q - 1)$.
5. The signature consists of the pair (S_1, S_2).

Verification at Receiver Side

Any user B can verify the signature as follows.

1. Compute $V_1 = \alpha^m \bmod q$.
2. Compute $V_2 = (Y_A)^{S_1}(S_1)^{S_2} \bmod q$.

The signature is valid if $V_1 = V_2$. Let us demonstrate that this is so. Assume that the equality is true. Then we have

$\alpha^m \bmod q = (Y_A)^{S_1}(S_1)^{S_2} \bmod q$	assume $V_1 = V_2$
$\alpha^m \bmod q = \alpha^{X_A S_1}\alpha^{KS_2} \bmod q$	substituting for Y_A and S_1
$\alpha^{m-X_A S_1} \bmod q = \alpha^{KS_2} \bmod q$	rearranging terms
$m - X_A S_1 \equiv KS_2 \bmod (q-1)$	property of primitive roots
$m - X_A S_1 \equiv KK^{-1}(m - X_A S_1) \bmod (q-1)$	substituting for S_2

Example: Choose q = 19 & a = 10.

1. Alice chooses $X_A = 16$.
2. Then $Y_A = \alpha^{X_A} \bmod q = \alpha^{16} \bmod 19 = 4$.
3. Alice's private key is 16; Alice's pubic key is $\{q, \alpha, Y_A\} = \{19, 10, 4\}$.

Suppose Alice wants to sign a message with hash value $m = 14$.

1. Alice chooses $K = 5$, which is relatively prime to $q - 1 = 18$.
2. $S_1 = \alpha^K \bmod q = 10^5 \bmod 19 = 3$ (see Table 8.3).
3. $K^{-1} \bmod (q - 1) = 5^{-1} \bmod 18 = 11$.
4. $S_2 = K^{-1}(m - X_A S_1) \bmod (q - 1) = 11(14 - (16)(3)) \bmod 18 = -374 \bmod 18 = 4$.

Bob can verify the signature as follows.

1. $V_1 = \alpha^m \bmod q = 10^{14} \bmod 19 = 16$.
2. $V_2 = (Y_A)^{S_1}(S_1)^{S_2} \bmod q = (4^3)(3^4) \bmod 19 = 5184 \bmod 19 = 16$.

Thus, the signature is valid.

Schnorr Digital Signature

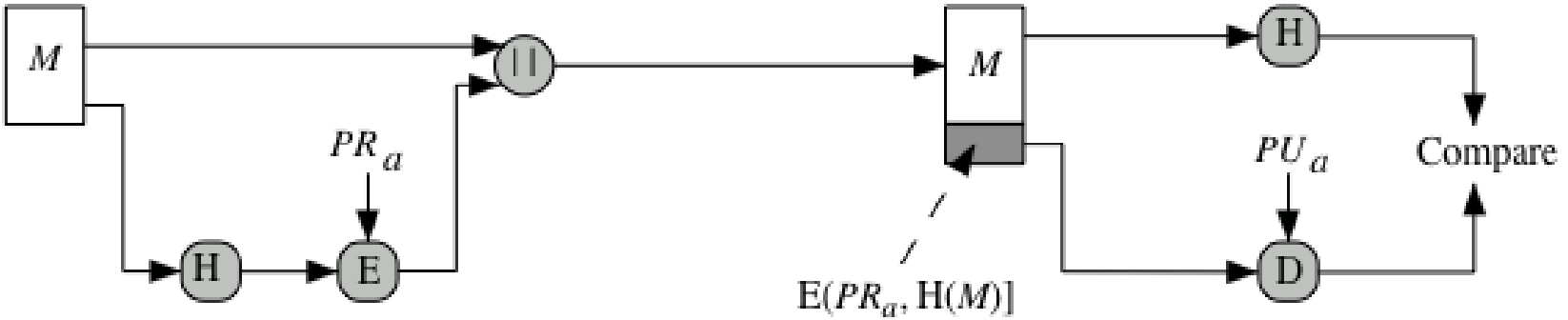
- Also uses exponentiation in a finite (Galois)
- Minimizes message dependent computation
  - Main work can be done in idle time
- Using a prime modulus p
  - $p-1$ has a prime factor q of appropriate size
  - typically p 1024-bit and q 160-bit (SHA-1 hash size)
- Schnorr Key Setup: Choose suitable primes p, q
  - Choose a such that $a^q = 1 \bmod p$
  - (a, p, q) are global parameters for all
  - Each user (e.g., A) generates a key
  - Chooses a secret key (number): $0 < s < q$
  - Computes his **public key**: $v = a^{-s} \bmod q$

Schnorr Digital Signature

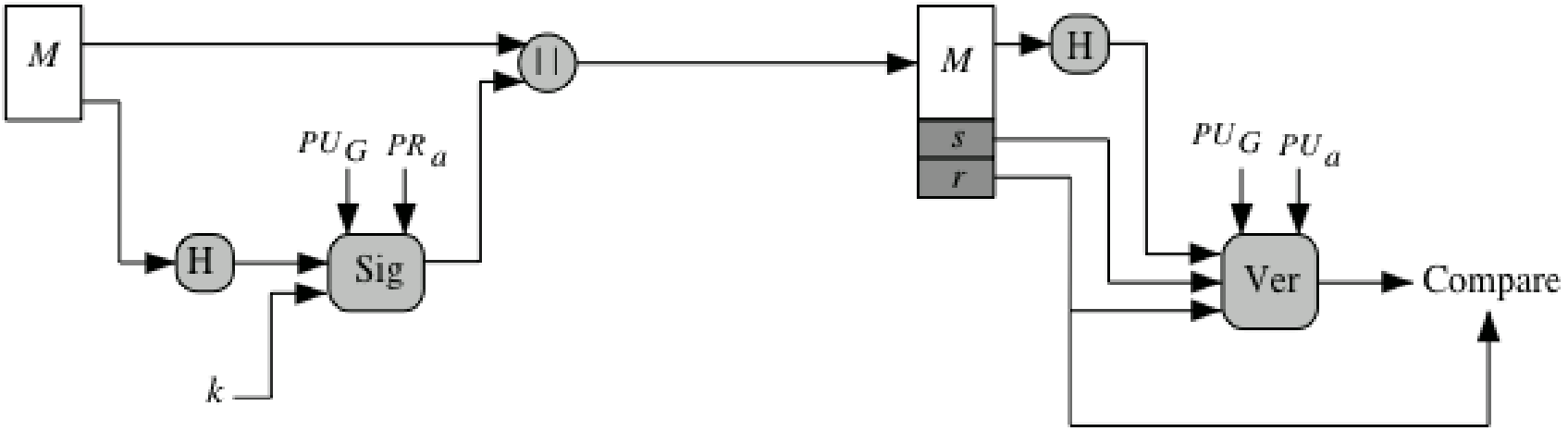
- User signs message by
  - Choosing random r with $0<r<q$ and computing $x = a^r \bmod p$
  - Concatenating message with x and hashing:
    $$e = H(M \,||\, x)$$
  - Computing: $y = (r + se) \bmod q$
  - Signature is pair (e, y)
- Any other user can verify the signature as follows:
  - Computing: $x' = a^y v^e \bmod p$
  - Verifying that: $e = H(M \,||\, x')$
  - $x' = a^y v^e = a^y a^{-se} = a^{y-se} = a^r = x \bmod p$
- Signature is valid only if x'=x.

NIST Digital Signature Algorithm

- Published by NIST as Federal Information Processing Standard FIPS 186
- Makes use of the Secure Hash Algorithm (SHA)
- The latest version, FIPS 186-3, also incorporates digital signature algorithms based on RSA and on elliptic curve cryptography



(a) RSA Approach



(b) DSA Approach

Figure 13.3 Two Approaches to Digital Signatures

Global Public Key Components

p prime number where $2^{L-1} < p < 2^L$ for $512 \le L \le 1024$ 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

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

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

Signature = (r, s)

Verifying

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

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

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

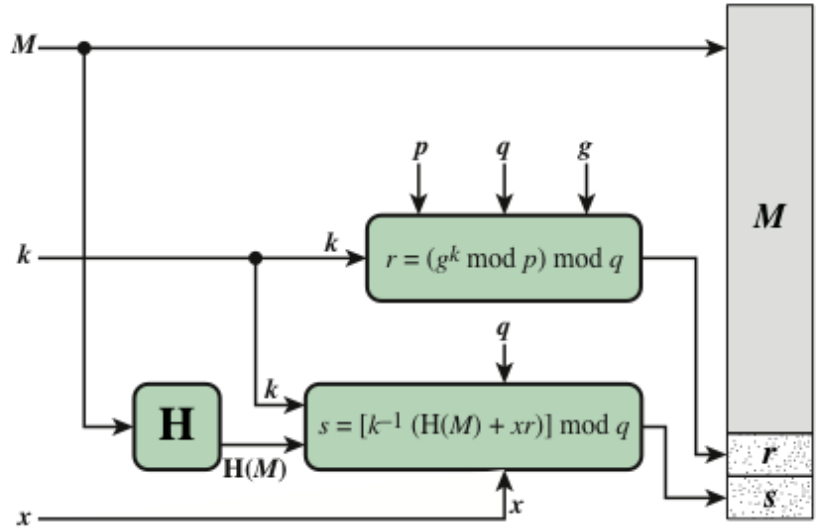
$$v = \left[\left(g^{u1} y^{u2}\right) \bmod p\right] \bmod q$$

TEST: $v = r'$

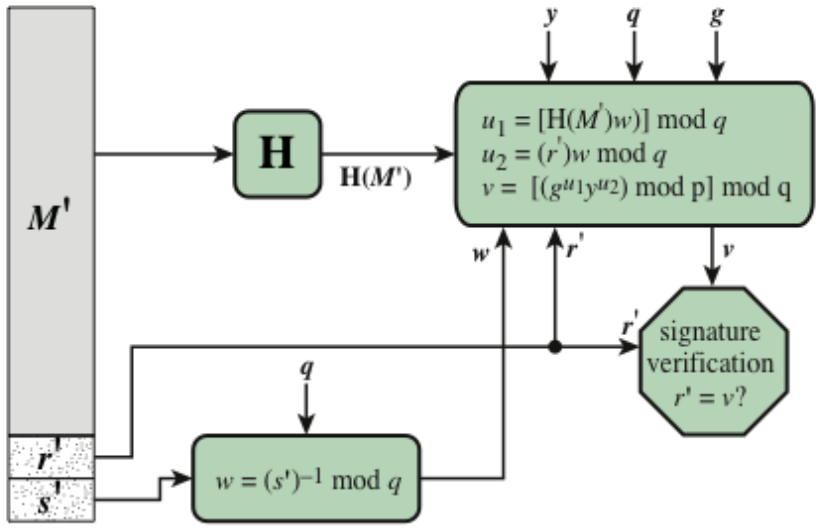
M = message to be signed
$\mathrm{H}(M)$ = hash of M using SHA-1
M', r', s' = received versions of M, r, s

Figure 13.4 The Digital Signature Algorithm (DSS)

DSA Signing and Verifying



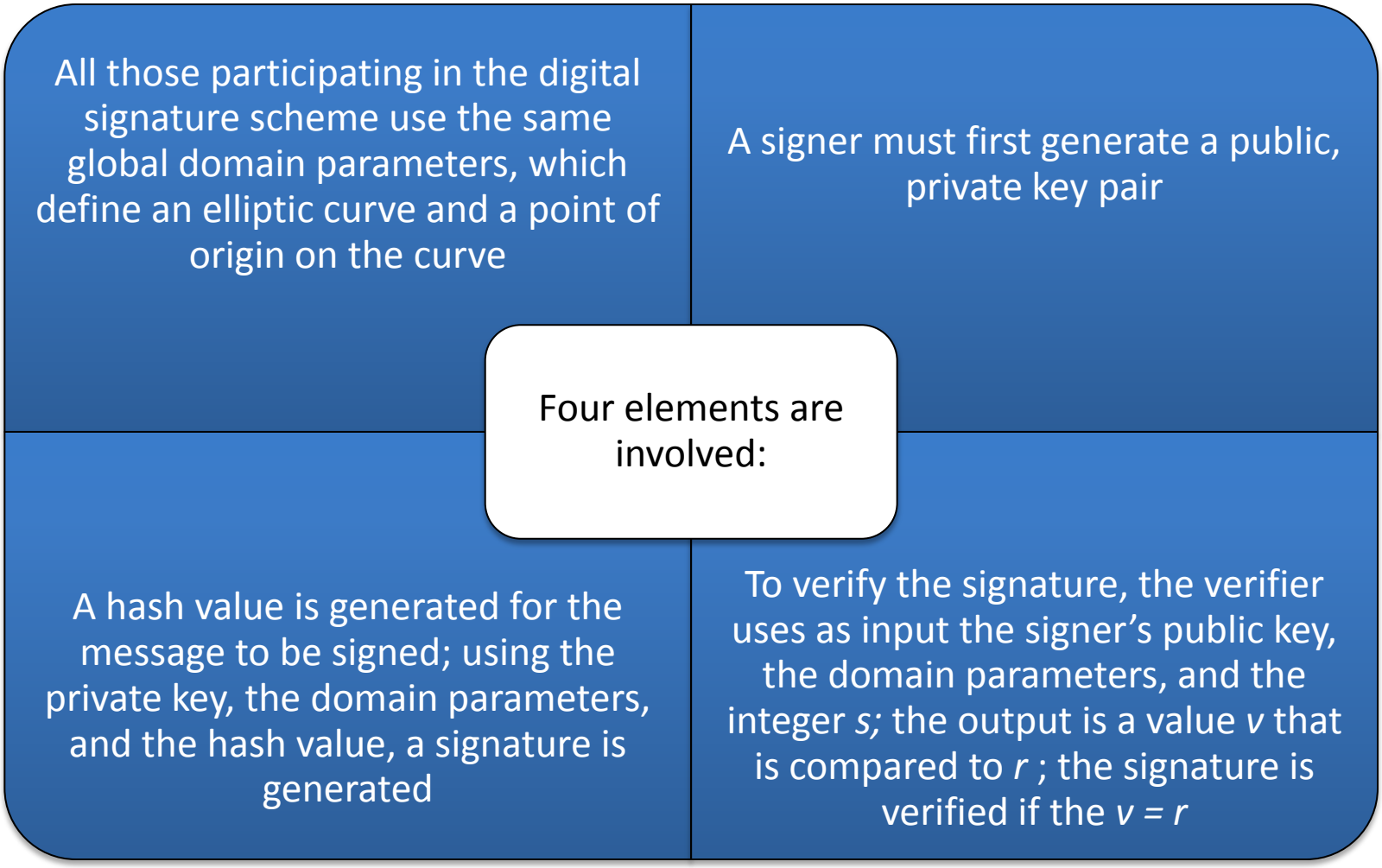
(a) Signing



(b) Verifying

Figure 13.5 DSA Signing and Verifying

Elliptic Curve Digital Signature Algorithm (ECDSA)



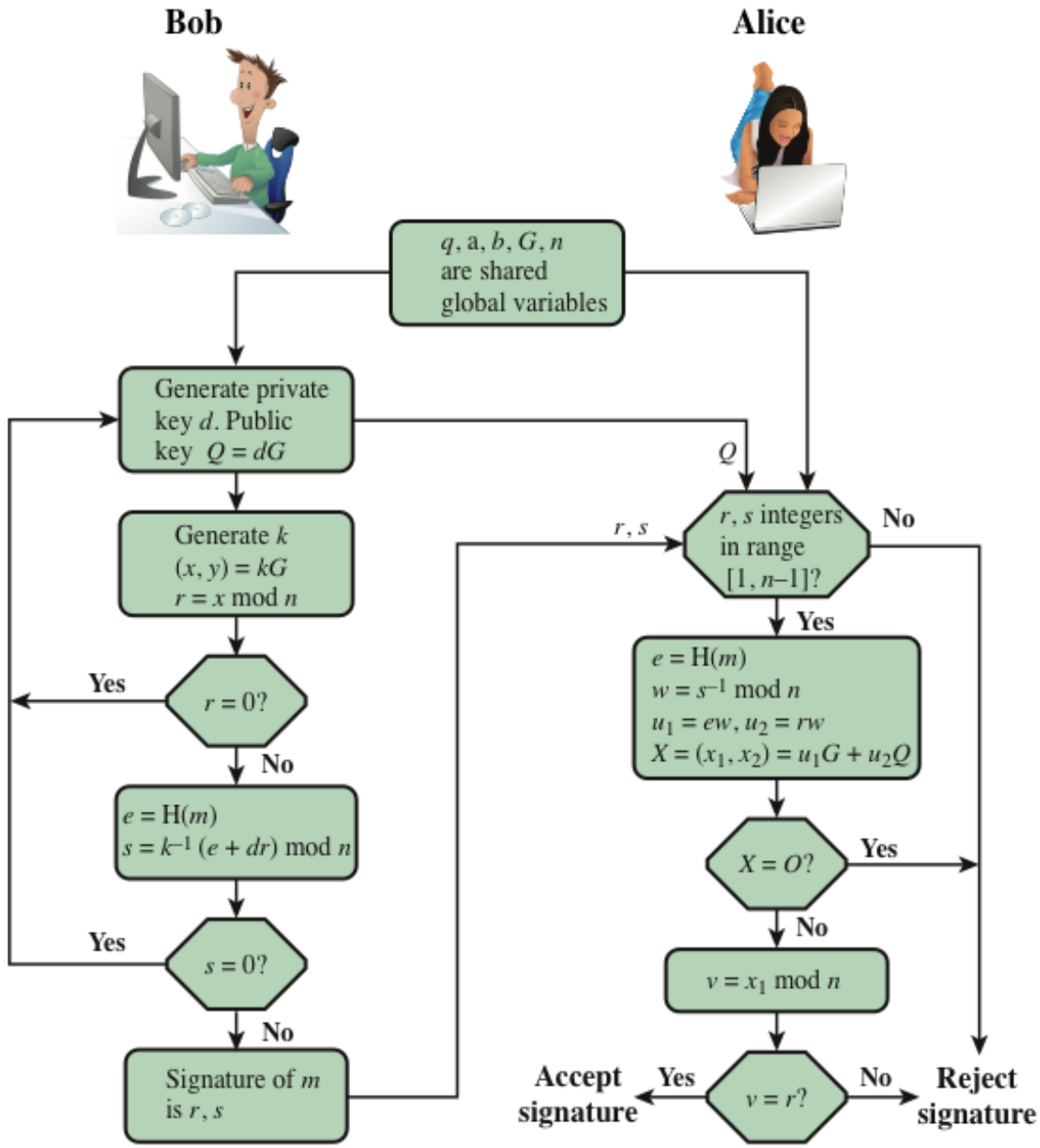


Figure 13.6 ECDSA Signing and Verifying

RSA-PSS

- RSA Probabilistic Signature Scheme
- Included in the 2009 version of FIPS 186
- Latest of the RSA schemes and the one that RSA Laboratories recommends as the most secure of the RSA schemes
- For all schemes developed prior to PSS is has not been possible to develop a mathematical proof that the signature scheme is as secure as the underlying RSA encryption/decryption primitive
- The PSS approach was first proposed by Bellare and Rogaway
- This approach, unlike the other RSA-based schemes, introduces a randomization process that enables the security of the method to be shown to be closely related to the security of the RSA algorithm itself

Mask Generation Function (MGF)

- Typically based on a secure cryptographic hash function such as SHA-1
  - Is intended to be a cryptographically secure way of generating a message digest, or hash, of variable length based on an underlying cryptographic hash function that produces a fixed-length output

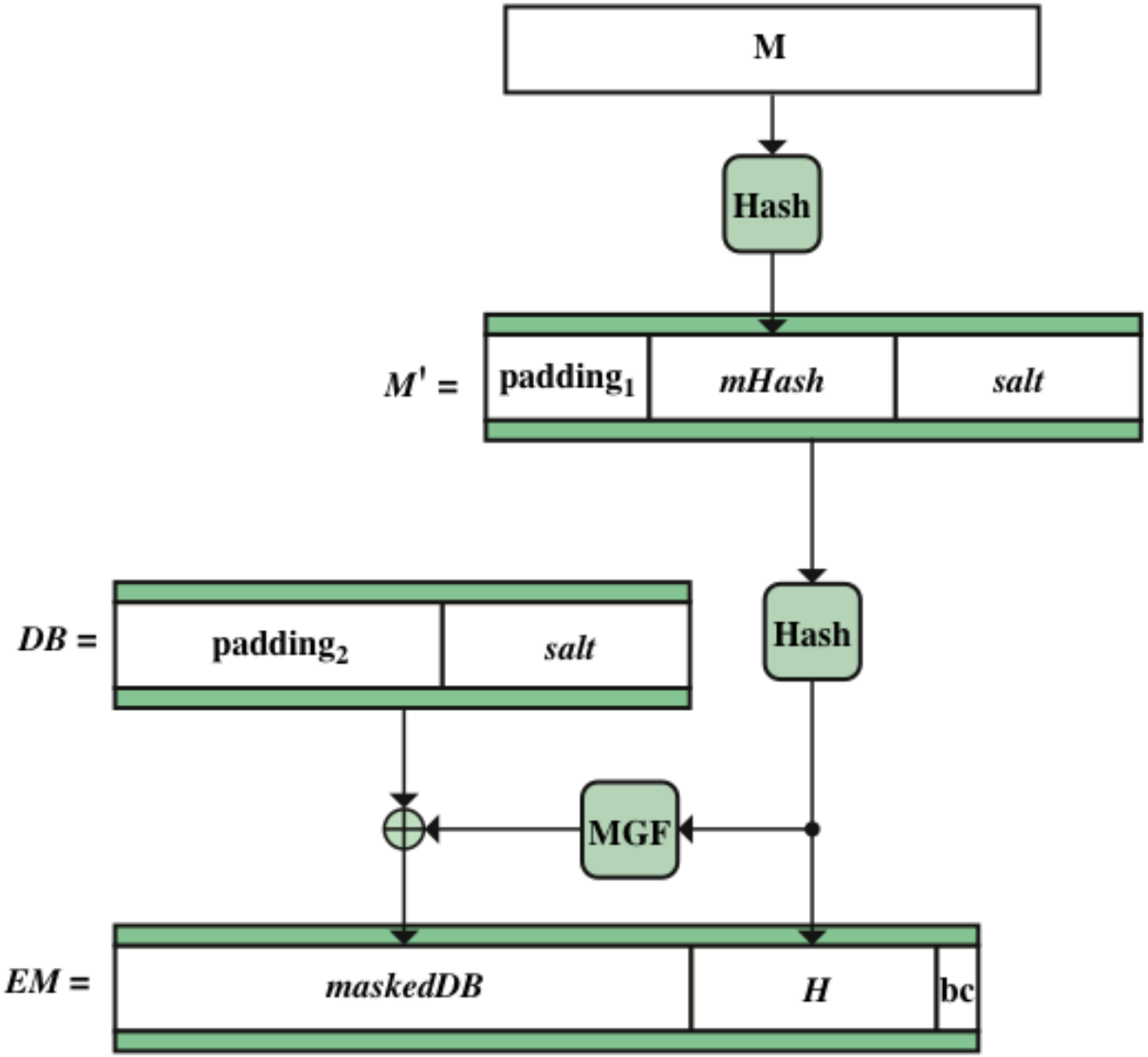


Figure 13.7 RSA-PSS Encoding

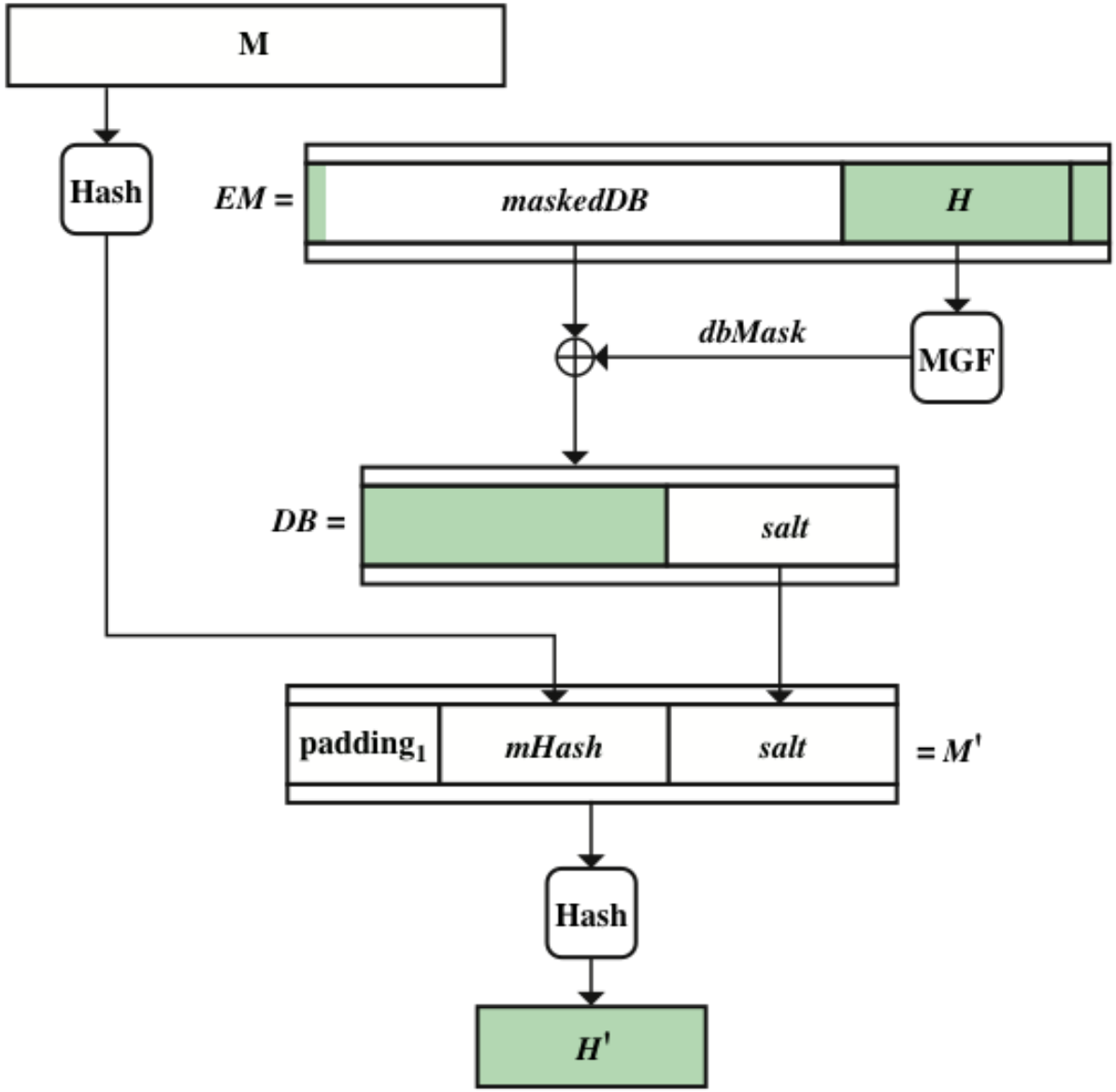
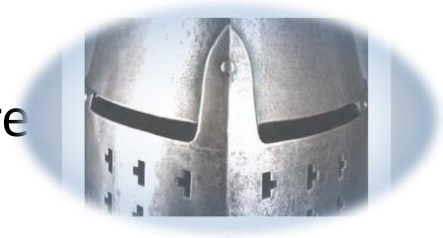


Figure 13.8 RSA-PSS EM Verification

Summary

- Digital signatures
  - Properties
  - Attacks and forgeries
  - Digital signature requirements
  - Direct digital signature
- Elgamal digital signature scheme
- RSA-PSS
  - Mask generation function
  - The signing operation
  - Signature verification



- NIST digital signature algorithm
  - The DSA approach
- Elliptic curve digital signature algorithm
  - Global domain parameters
  - Key generation
  - Digital signature generation and authentication
  - Schnorr digital signature scheme