DIGITAL SIGNATURES

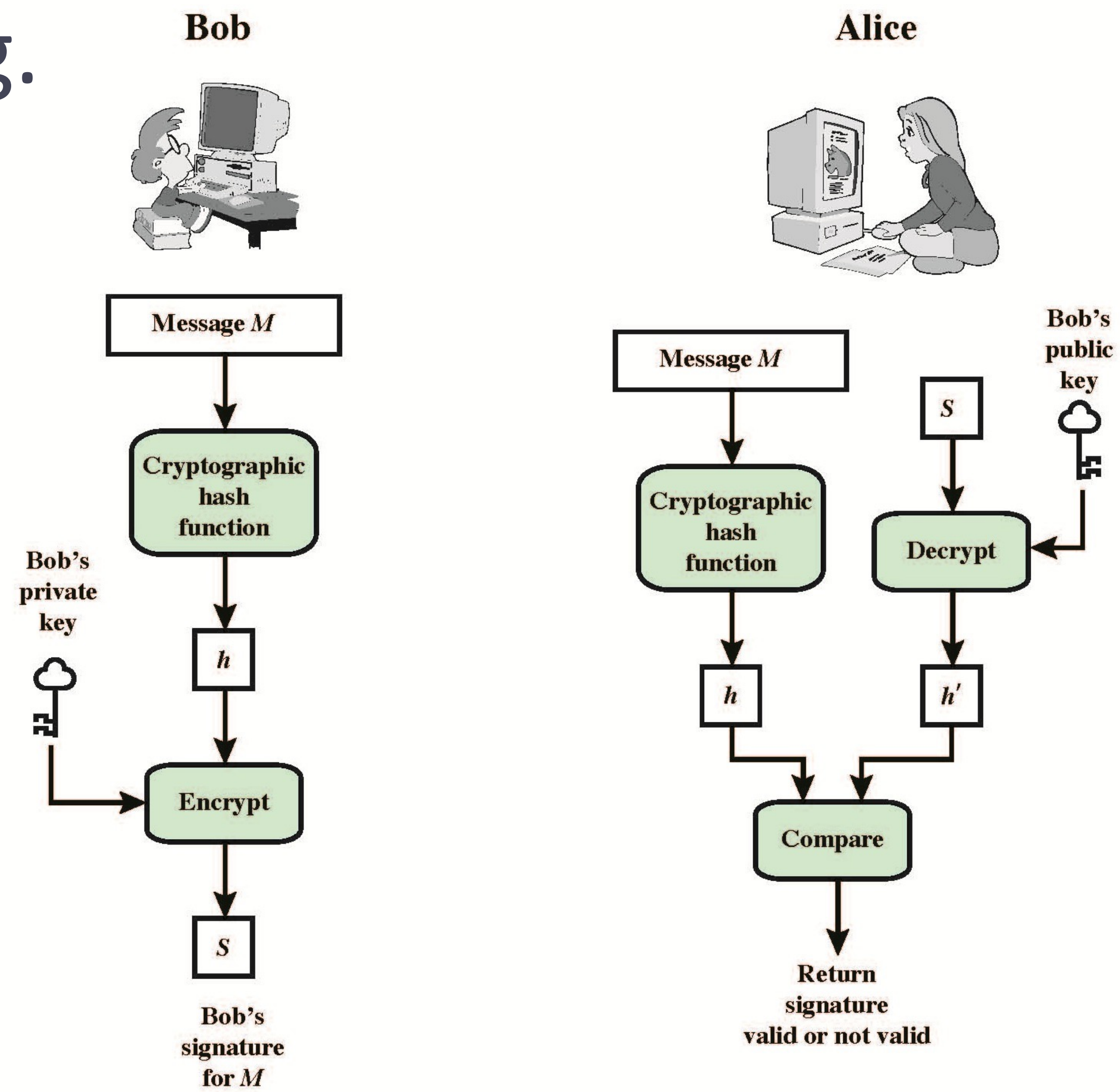
LECTURE 13

Cryptography

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 Sig.



Bob
Alice
Message M
Cryptographic hash function
h
Bob's private key
Encrypt
S
Bob's signature for M
Message M
S
Bob's public key
Cryptographic hash function
Decrypt
h
h′
Compare
Return signature valid or not valid

Attacks

- Key-only attack
  - knows one's public key
- Known message attack
  - a set of messages and their signatures
- Chosen message attack
  - obtains valid signatures for the chosen messages
- Adaptive chosen message attack
  - obtains valid signatures of messages that depend on previously obtained message-signature pairs

Forgeries

- Total break
  - determines one's private key
- Universal forgery
  - finds an efficient signing algorithm that provides an *equivalent* way of constructing signatures
- Selective forgery
  - forges a signature for a particular message
- Existential forgery
  - forges a signature for at least one message; but no control over the message

Digital Signature Requirements

- Must depend on the message signed
- Must use information unique to sender
  - to prevent both forgery and denial
- Must be relatively easy to produce
- Must be relatively easy to recognize & verify
- Must be computationally infeasible to forge
  - with new message for existing digital signature
  - with fraudulent digital signature for given message
- Must be practical save digital signature in storage

ElGamal Digital Signature

- Signature variant of ElGamal, related to D-H
  - so uses exponentiation in a finite (Galois) field
  - with security based difficulty of computing discrete logarithms, as in D-H
- Use private key for encryption (signing)
- Uses public key for decryption (verification)
- Each user (e.g. A) generates their key
  - chooses a *private* key (number): $1 < x_A < q-1$
  - compute their *public key*: $y_A = a^{x_A} \bmod q$

ElGamal Digital Signature

- Alice signs a message M to Bob by computing
  - the hash $m = \mathrm{H}(M)$, $0 <= m <= (q\text{-}1)$
  - chose random integer K with $1 <= K <= (q\text{-}1)$ and $\gcd(K, q\text{-}1) = 1$
  - compute temporary key: $S_1 = a^K \bmod q$
  - compute K^{-1} the inverse of $K \bmod (q\text{-}1)$
  - compute the value: $S_2 = K^{-1}(m - x_A S_1) \bmod (q\text{-}1)$
  - signature is: (S_1, S_2)
- Any user B can verify the signature by computing
  - $V_1 = a^m \bmod q$
  - $V_2 = y_A^{S_1} S_1^{S_2} \bmod q$
  - signature is valid if $V_1 = V_2$

ElGamal Signature Example

- Use field GF(19) q = 19 and a = 10
- Alice computes her key:
  - A chooses x_A=16 & computes y_A = 10^{16} mod 19 = 4
- Alice signs message with hash m = 14 as (3,4):
  - choosing random K = 5 which has gcd(18,5) = 1
  - computing S_1 = 10^5 mod 19 = 3
  - finding K^{-1} mod (q-1) = 5^{-1} mod 18 = 11
  - computing S_2 = $K^{-1}(m - x_A S_1)$ mod (q-1) = 11*(14-16*3) mod 18 = 4
- Any user B can verify the signature by computing
  - $V_1 = a^m$ mod q = 10^{14} mod 19 = 16
  - $V_2 = y_A^{S_1} S_1^{S_2}$ mod q = $4^3 * 3^4$ = 5184 = 16 mod 19
  - since 16 = 16, signature is valid

Schnorr Digital Signatures

- Choose suitable primes p , q
- Choose a such that $a^q \equiv 1 \bmod p$
- (a,p,q) are global parameters for all

- Each user (e.g. A) generates a key
  - chooses a *private* key (number): $0 < s < q$
  - compute their *public* *key*: $v = a^{-s} \bmod p$

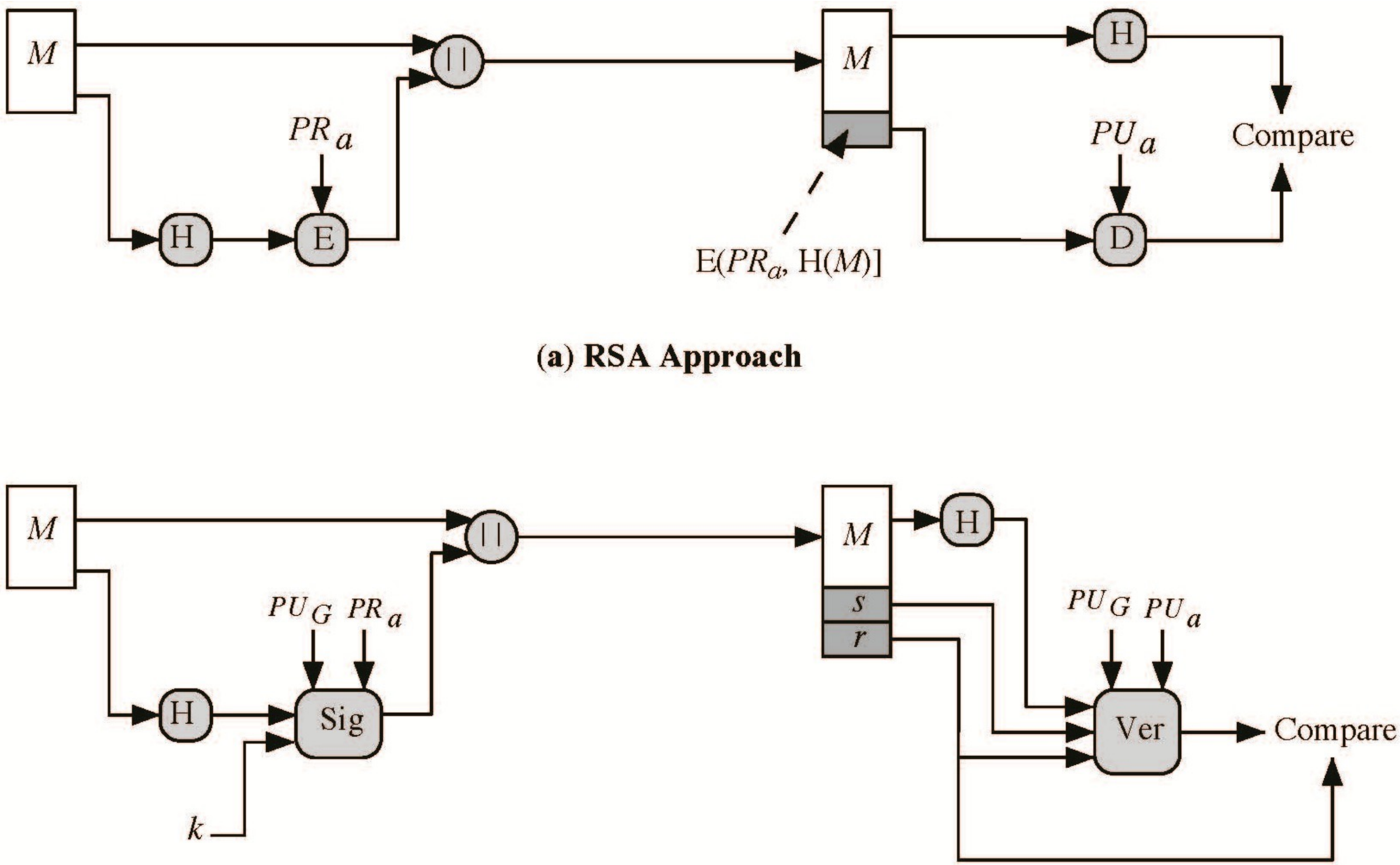
Schnorr Signature

- User signs message by
  - choosing random r with $0 < r < q$ and computing $x = a^r \bmod p$
  - concatenate message with x and hash result to computing: $e = H(M||x)$
  - 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: $e = H(M||x')$

Digital Signature Standard (DSS)

- US Government approved signature scheme
  - designed by NIST & NSA in early 90's
  - published as FIPS-186 in 1991
  - revised in 1993, 1996, 2000, 2009 & then 2013
- 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

DSS vs RSA Signatures



(a) RSA Approach

(b) DSS Approach

Digital Signature Algorithm (DSA)

- Creates a 320 or more bit signature
- With 1024-3072 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

DSA Key Generation

- Have shared global public key values (p,q,g):
  - choose a large prime p with $2^{L-1} < p < 2^L$
  - q is a N bit prime divisor of $(p-1)$ with $2^{N-1} < q < 2^N$
    - where $(L, N) \in \{(1024, 160), (2048, 224), (2048, 256), (3072, 256)\}$
  - choose $g = h^{(p-1)/q} \bmod p$
    - 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, $k < q$
  - 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:

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

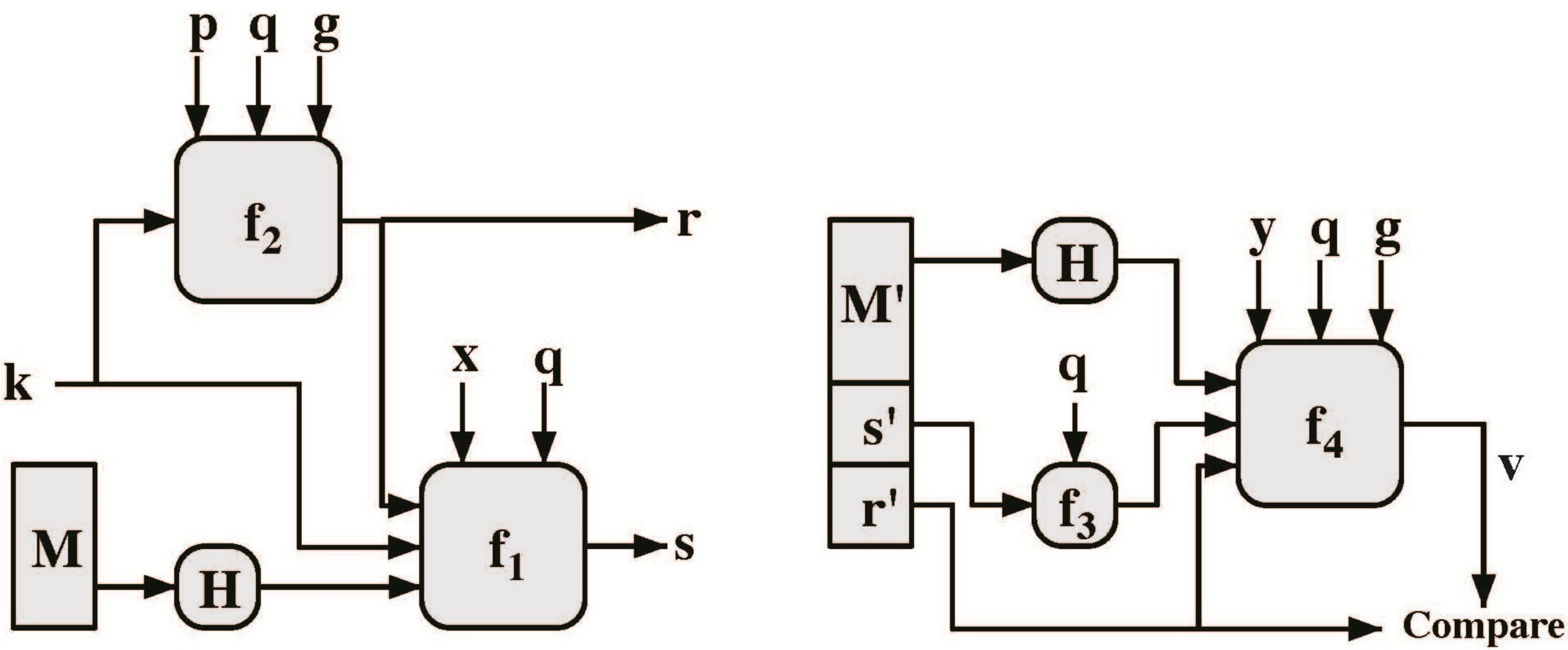
$u1 = [H(M)w] \bmod q$

$u2 = (rw) \bmod q$

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

- If $v = r$ then signature is verified

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

Elliptic
Curve
Digital
Signature
Algorithm

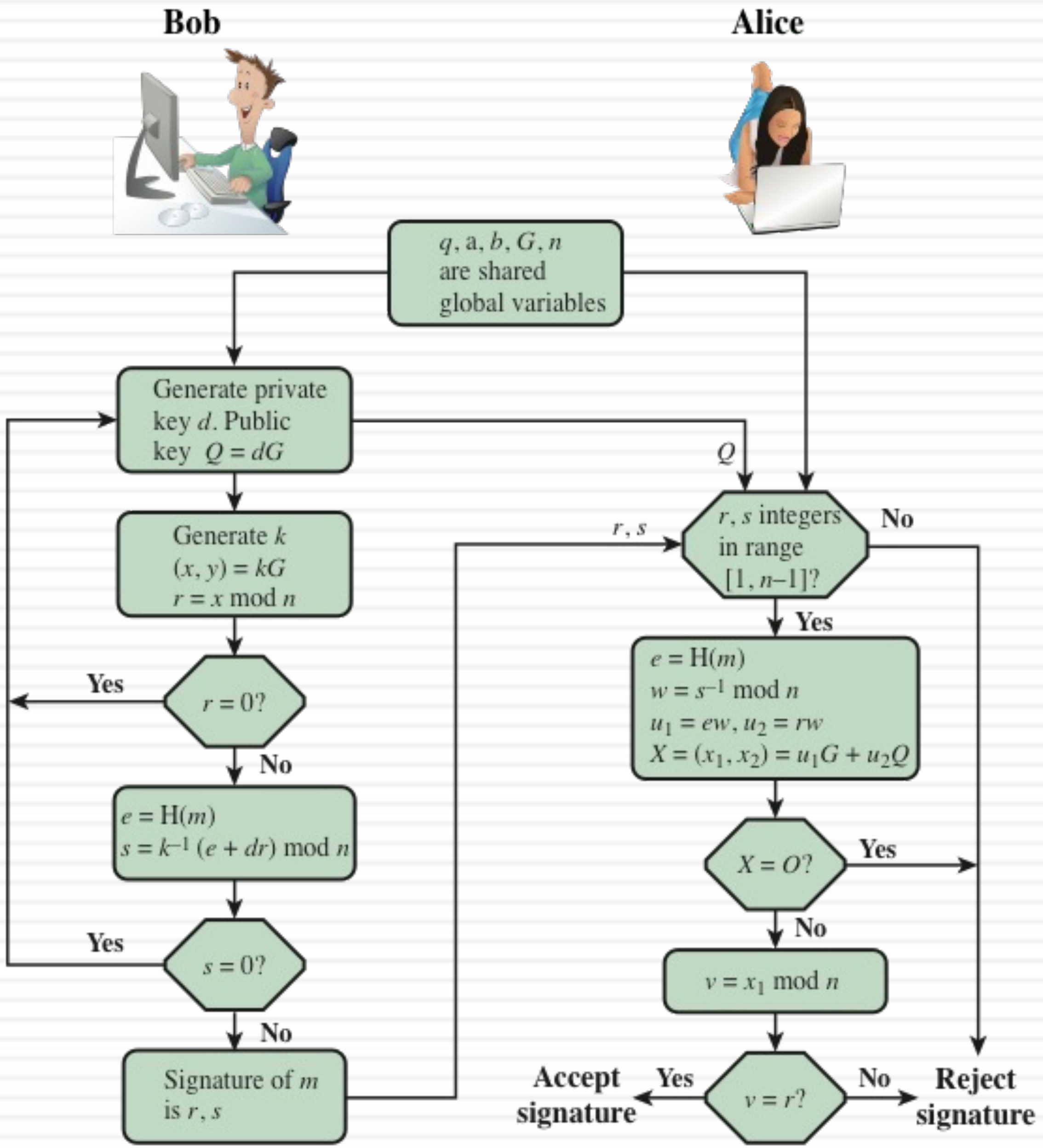


Figure 13.6 ECDSA Signing and Verifying

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, it has not been possible to develop a mathematical proof that the signature scheme is as secure as the underlying RSA encryption/decryption primitive

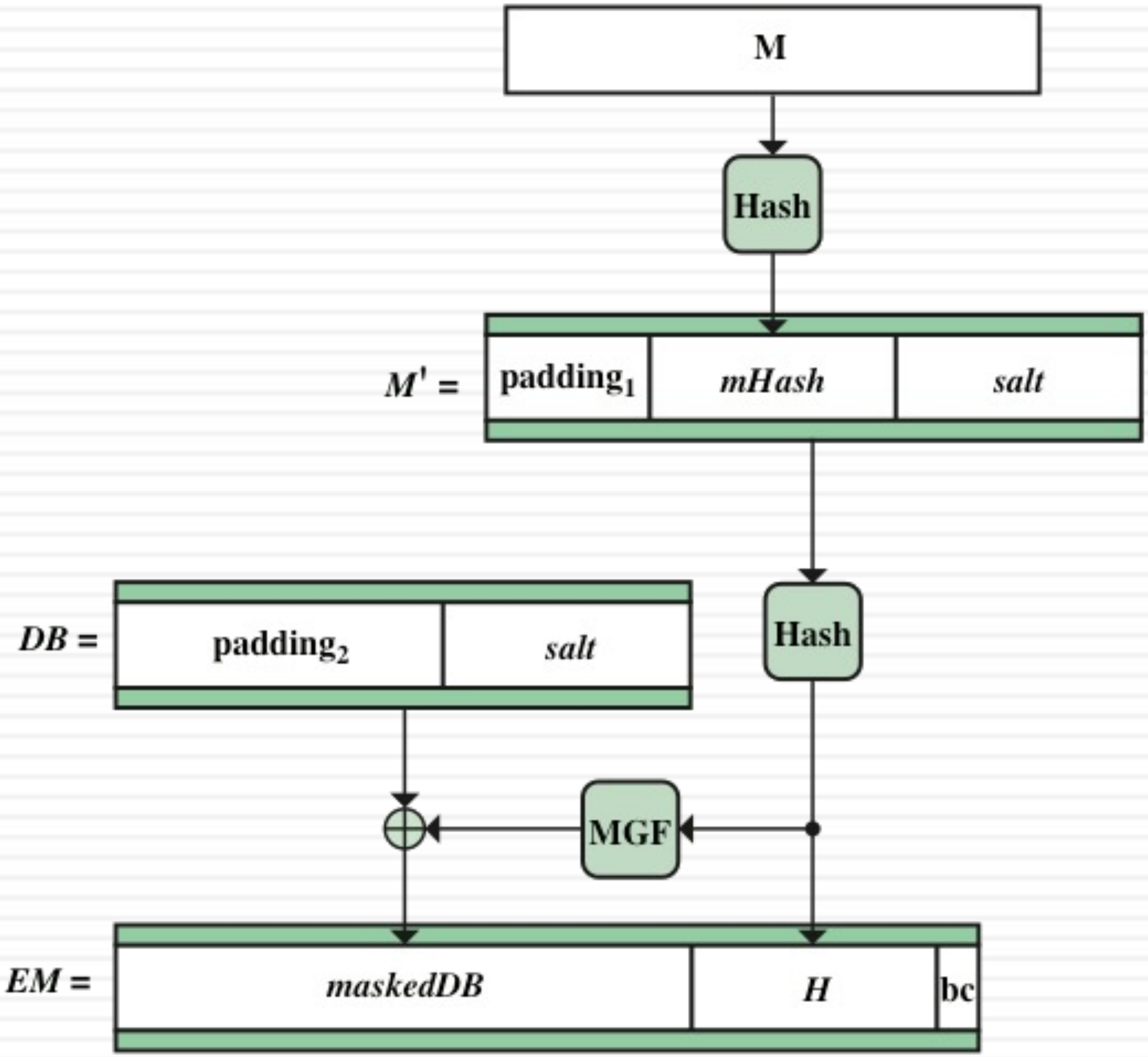


Figure 13.7 RSA-PSS Encoding

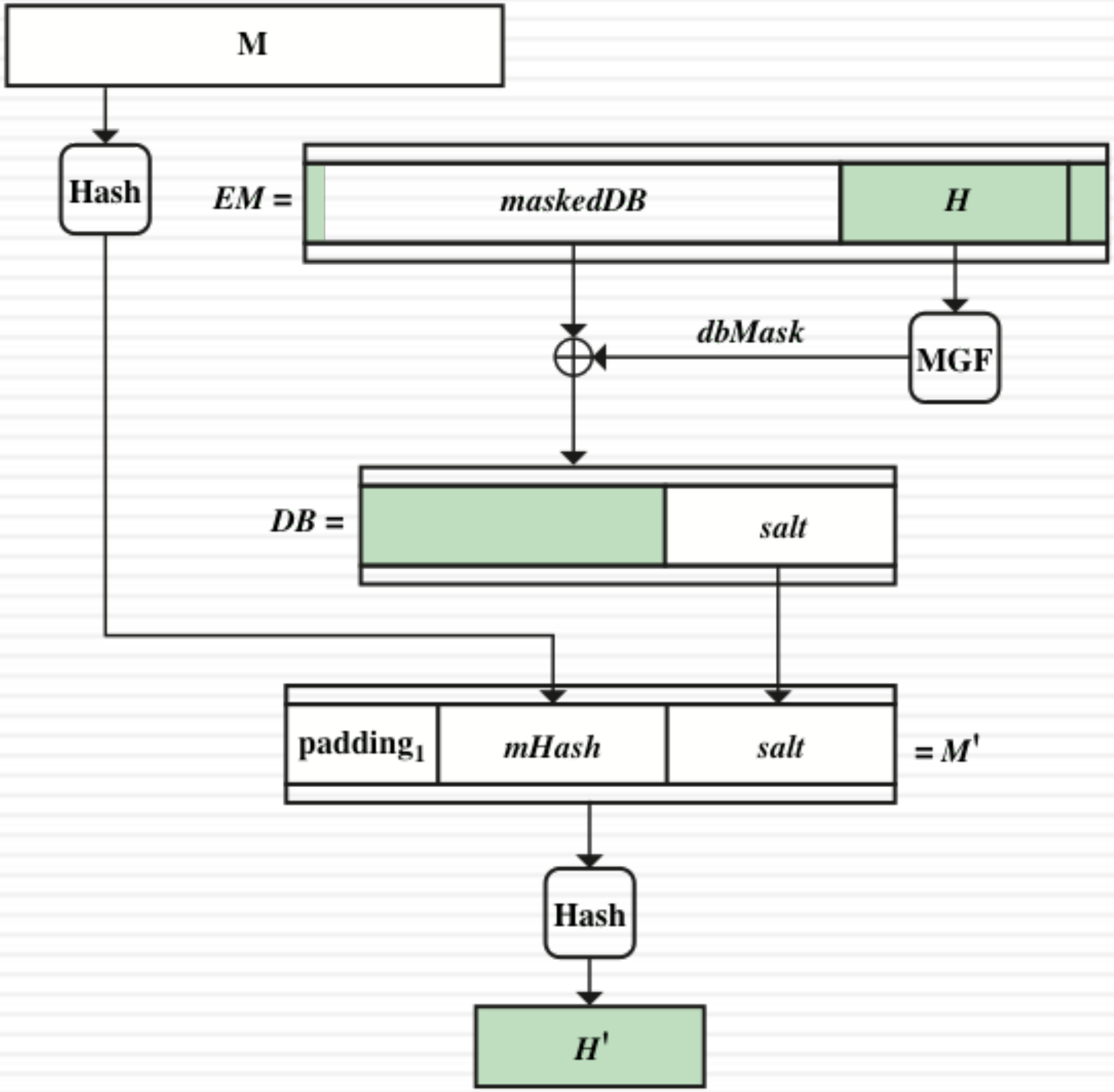


Figure 13.8 RSA-PSS EM Verification