# Chapter 13

# Digital Signature

## Objectives

- To define a digital signature
- To define security services provided by a digital signature
- To define attacks on digital signatures
- To discuss some digital signature schemes, including RSA, ElGamal,
- Schnorr, DSS, and elliptic curve
- To describe some applications of digital signatures

# 13-2 PROCESS

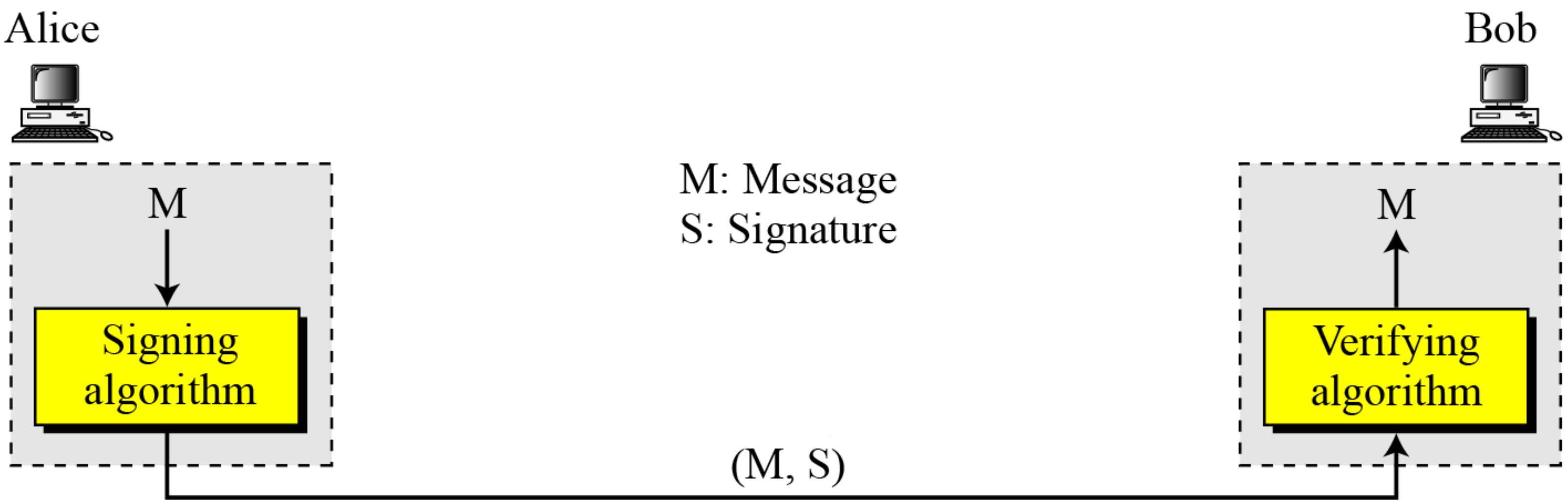
***Figure 13.1 shows the digital signature process. The sender uses a signing algorithm to sign the message. The message and the signature are sent to the receiver. The receiver receives the message and the signature and applies the verifying algorithm to the combination. If the result is true, the message is accepted; otherwise, it is rejected.***

***Topics discussed in this section:***

**13.2.1 Need for Keys**
**13.2.2 Signing the Digest**

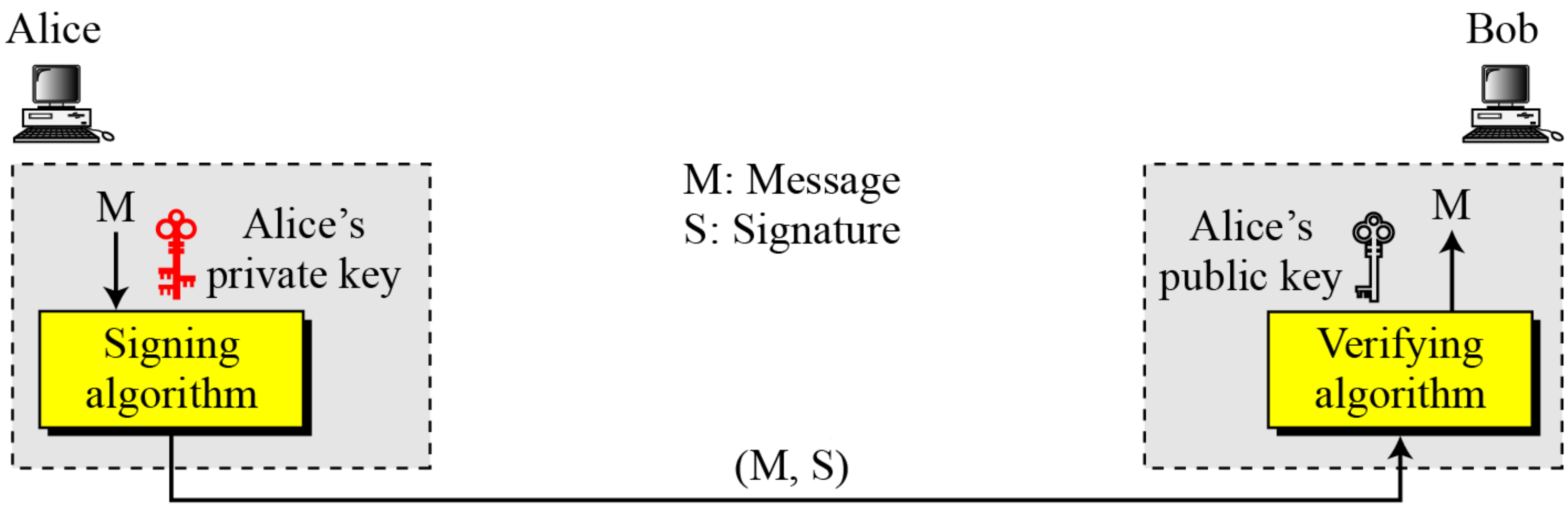
# 13-2 Continued

**Figure 13.1** ***Digital signature process***



# 13.2.1 Need for Keys

**Figure 13.2** *Adding key to the digital signature process*

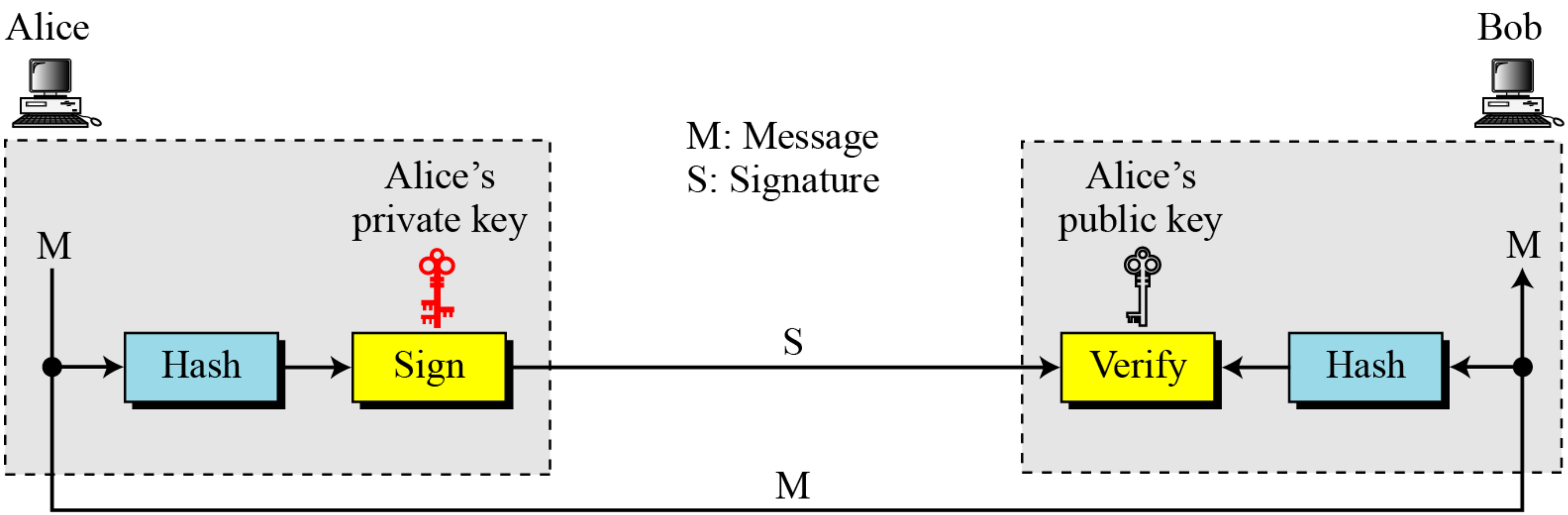



**Note**

**A digital signature needs a public-key system. The signer signs with her private key; the verifier verifies with the signer's public key.**

# 13.2.2 Signing the Digest

**Figure 13.3** ***Signing the digest***



# 13-3 SERVICES

***We discussed several security services in Chapter 1 including message confidentiality, message authentication, message integrity, and nonrepudiation. A digital signature can directly provide the last three; for message confidentiality we still need encryption/decryption.***

***Topics discussed in this section:***

**13.3.1 Message Authentication**
**13.3.2 Message Integrity**
**13.3.3 Nonrepudiation**
**13.3.4 Confidentiality**

# 13.3.1 Message Authentication

***A secure digital signature scheme, like a secure conventional signature can provide message authentication.***

*Note*

**A digital signature provides message authentication.**

## *13.3.2 Message Integrity*

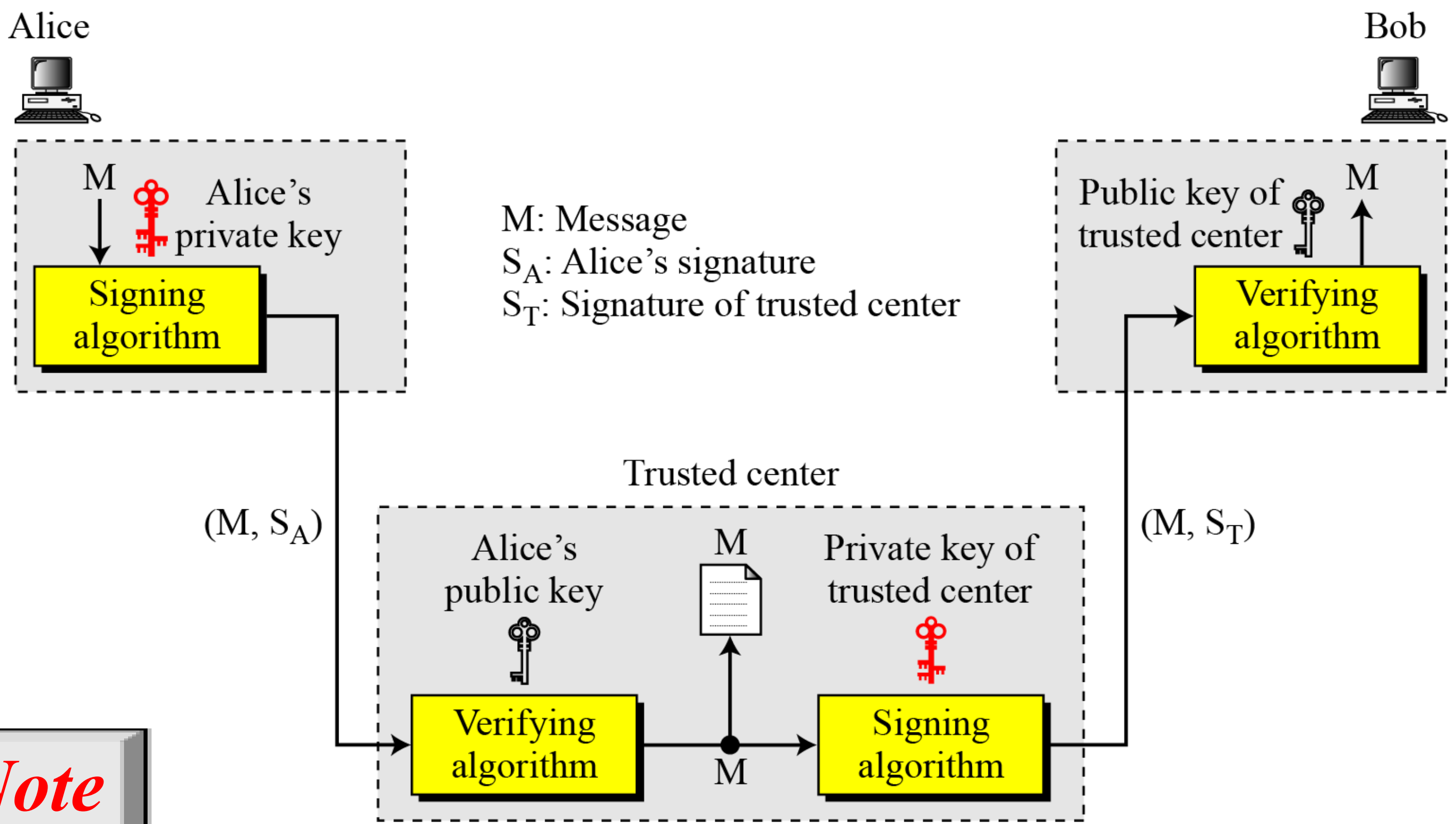
***The integrity of the message is preserved even if we sign the whole message because we cannot get the same signature if the message is changed.***

*Note*

**A digital signature provides message integrity.**

# 13.3.3 Nonrepudiation

**Figure 13.4** *Using a trusted center for nonrepudiation*

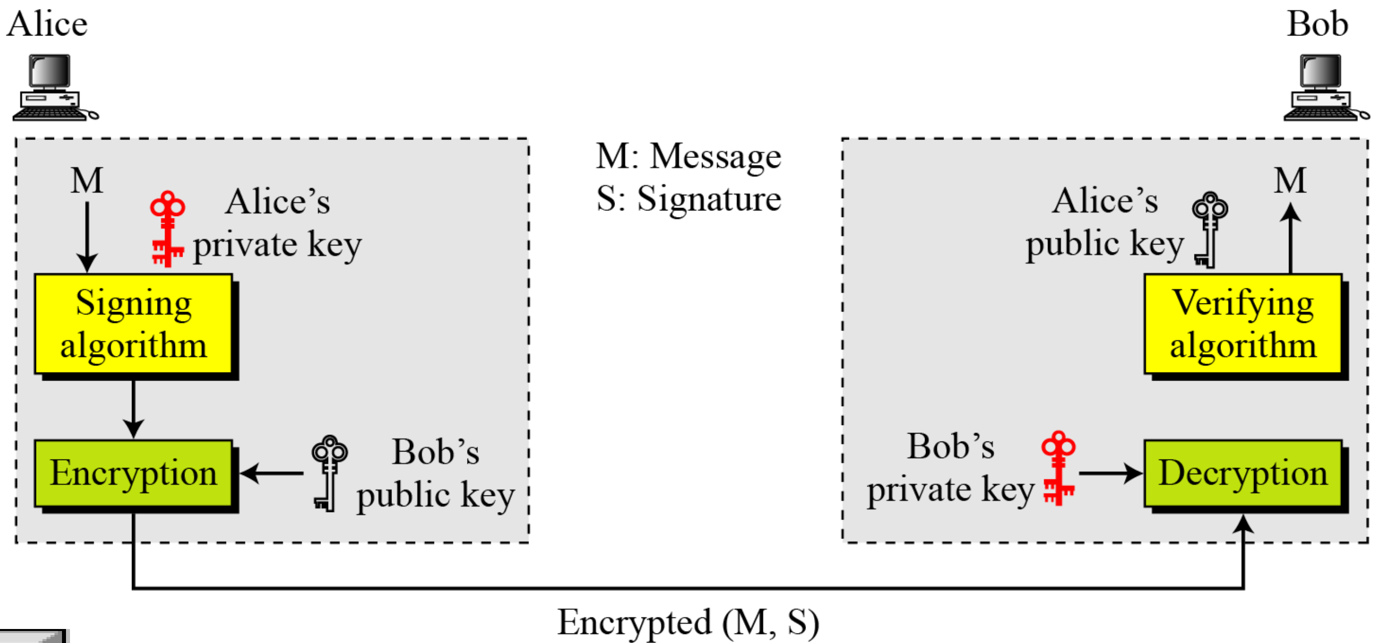



**Note**

**Nonrepudiation can be provided using a trusted party.**

# 13.3.4 Confidentiality

**Figure 13.5** *Adding confidentiality to a digital signature scheme*




*Note*

**A digital signature does not provide privacy. If there is a need for privacy, another layer of encryption/decryption must be applied.**

# 13-5 DIGITAL SIGNATURE SCHEMES

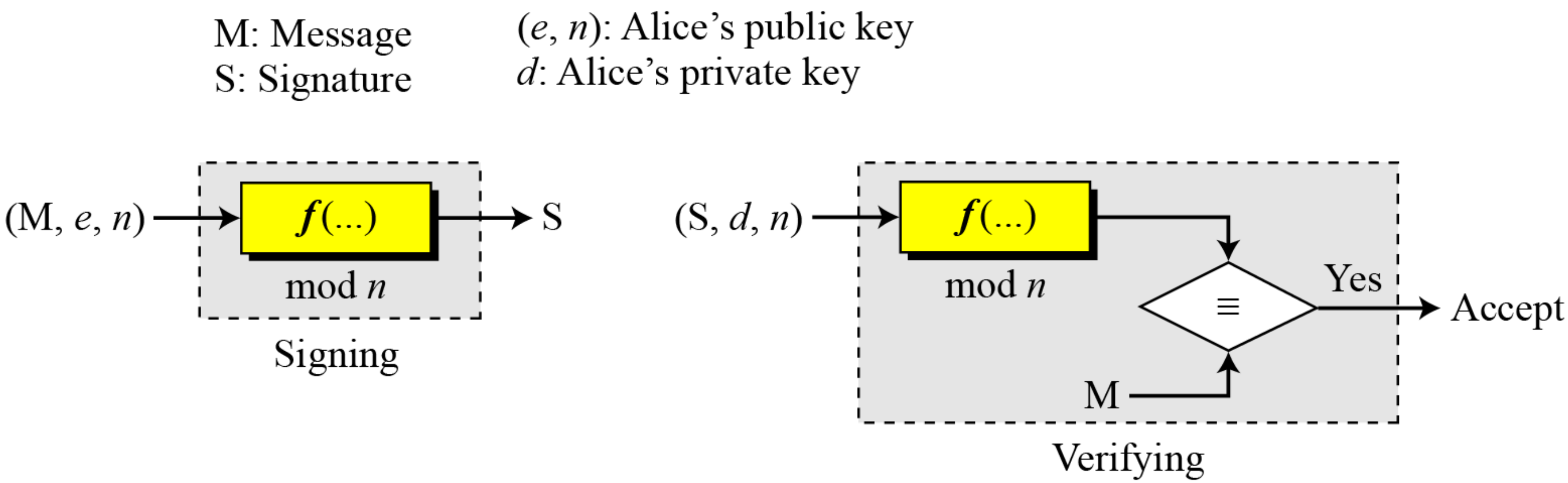
***Several digital signature schemes have evolved during the last few decades. Some of them have been implemented.***

***Topics discussed in this section:***

**13.5.1 RSA Digital Signature Scheme**
**13.5.2 ElGamal Digital Signature Scheme**
**13.5.3 Schnorr Digital Signature Scheme**
**13.5.4 Digital Signature Standard (DSS)**
**13.5.5 Elliptic Curve Digital Signature Scheme**

# 13.5.1 RSA Digital Signature Scheme

**Figure 13.6** *General idea behind the RSA digital signature scheme*



# *13.5.1 Continued*

## *Key Generation*

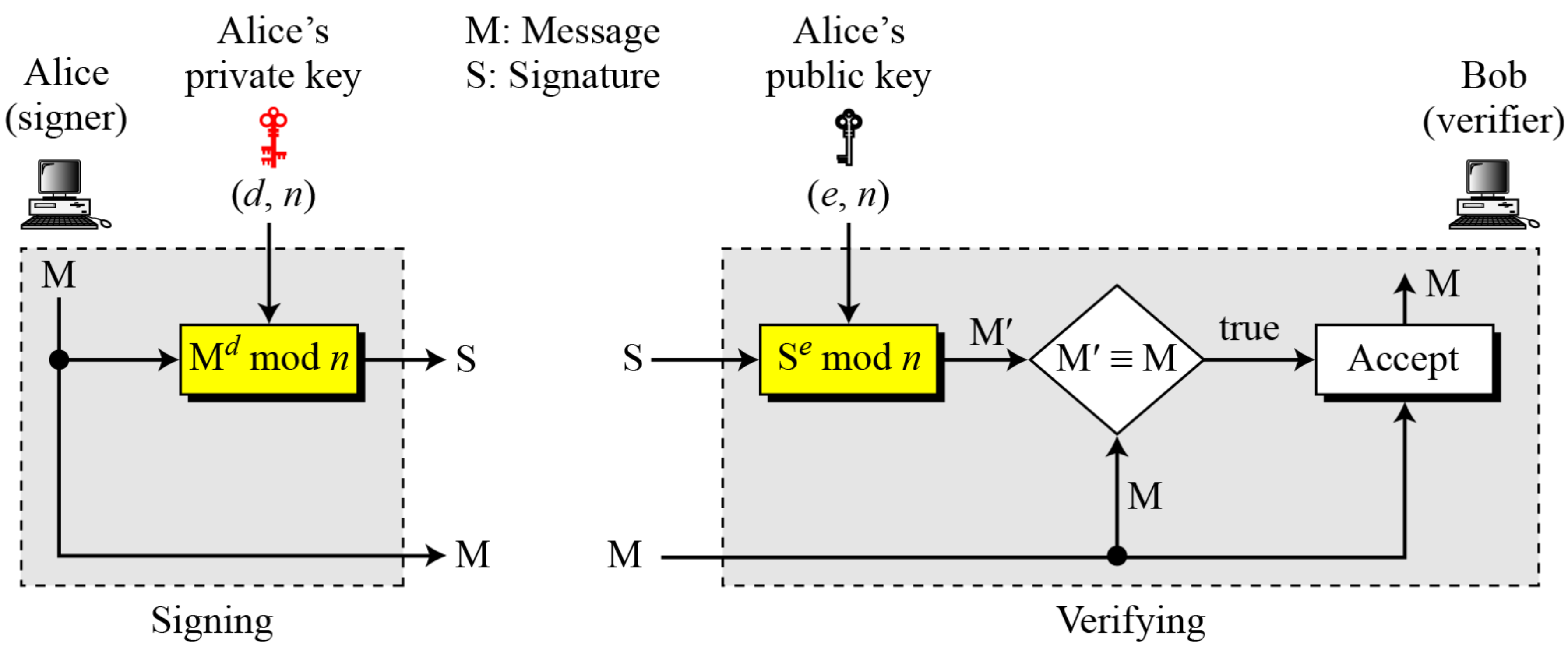
***Key generation in the RSA digital signature scheme is exactly the same as key generation in the RSA***

**Note**

**In the RSA digital signature scheme, *d* is private; *e* and *n* are public.**

## Signing and Verifying

**Figure 13.7** ***RSA digital signature scheme***



# 13.5.1 *Continued*

**Example 13.1**

**As a trivial example, suppose that Alice chooses $p$ = 823 and $q$ = 953, and calculates $n$ = 784319. The value of ϕ(n) is 782544. Now she chooses $e$ = 313 and calculates $d$ = 160009. At this point key generation is complete. Now imagine that Alice wants to send a message with the value of M = 19070 to Bob. She uses her private exponent, 160009, to sign the message:**

$$\text{M: } 19070 \quad \rightarrow \quad \text{S} = (19070^{160009}) \bmod 784319 = 210625 \bmod 784319$$

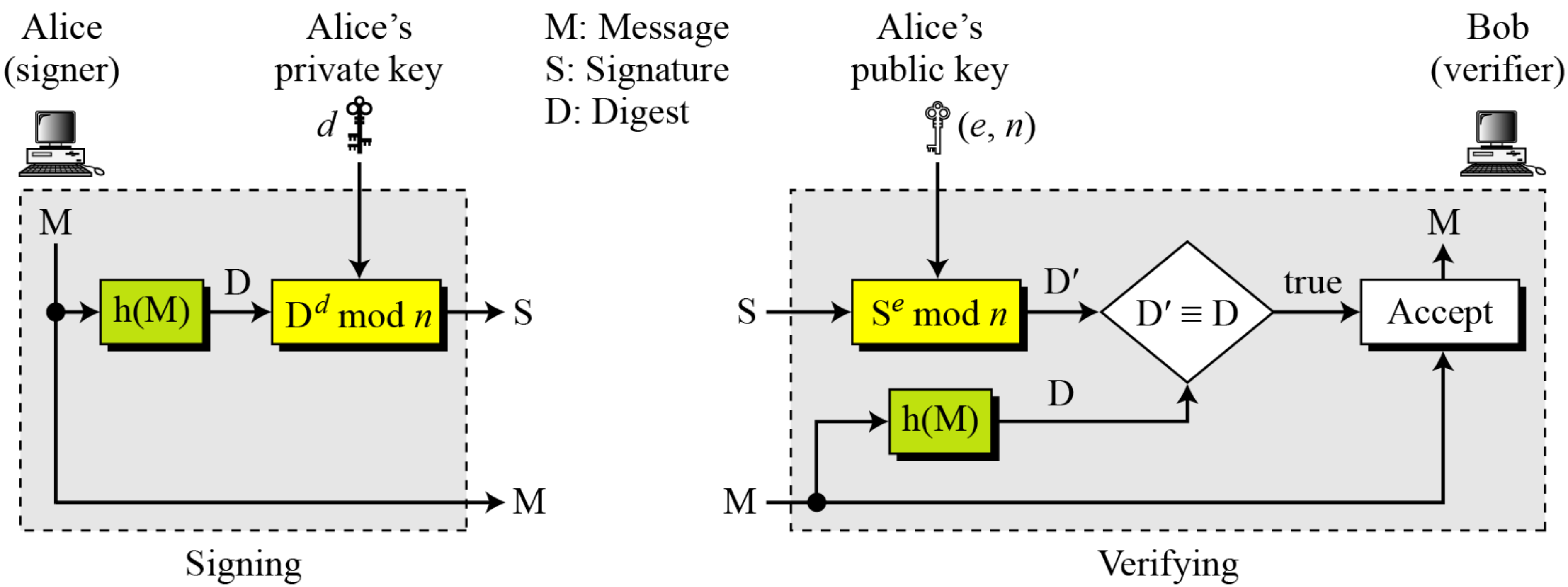
**Alice sends the message and the signature to Bob. Bob receives the message and the signature. He calculates**

$$\text{M}' = 210625^{313} \bmod 784319 = 19070 \bmod 784319 \quad \rightarrow \quad \text{M} \equiv \text{M}' \bmod n$$

**Bob accepts the message because he has verified Alice's signature.**

## RSA Signature on the Message Digest

**Figure 13.8** *The RSA signature on the message digest*



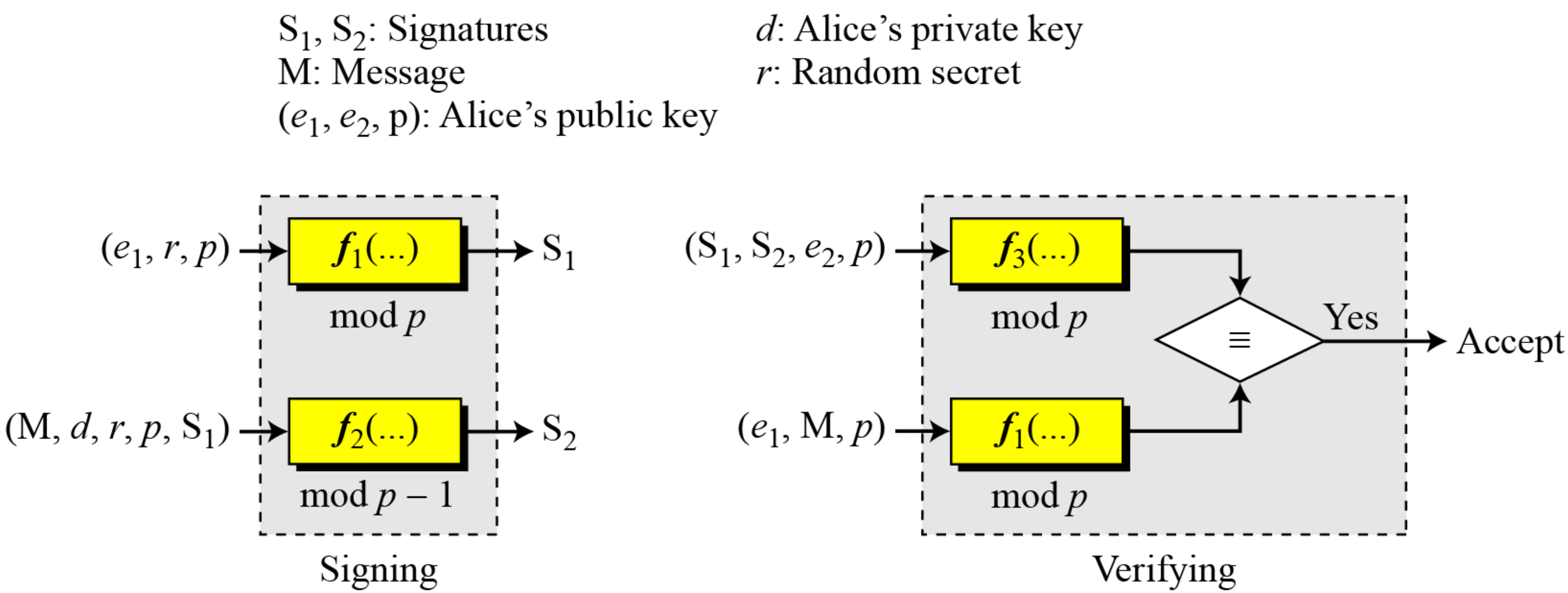
# 13.5.1 Continued

**Note**

**When the digest is signed instead of the message itself, the susceptibility of the RSA digital signature scheme depends on the strength of the hash algorithm.**

# 13.5.2 ElGamal Digital Signature Scheme

**Figure 13.9** ***General idea behind the ElGamal digital signature scheme***



# *13.5.2 Continued*

### *Key Generation*

***The key generation procedure here is exactly the same as the one used in the cryptosystem.***

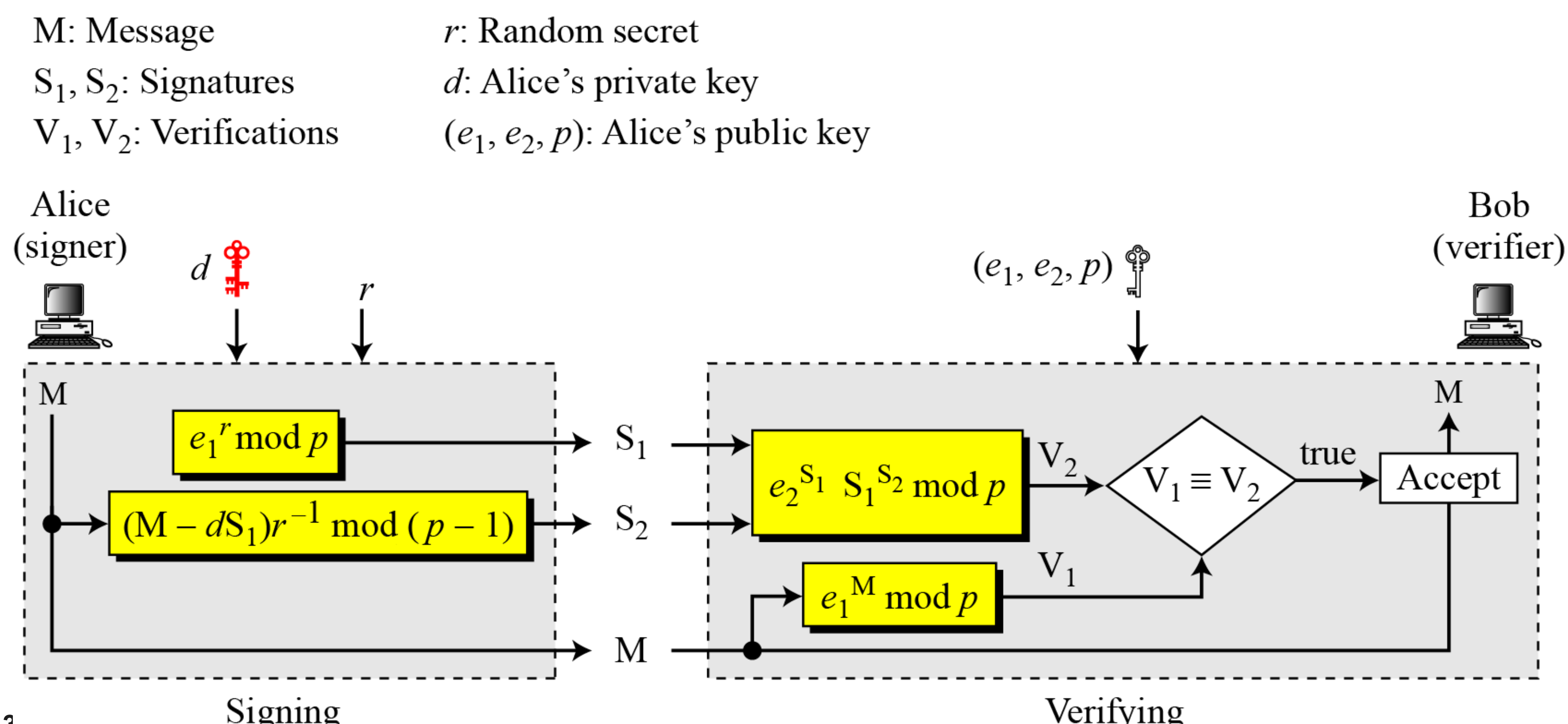
*Note*

**In ElGamal digital signature scheme, ($e_1$, $e_2$, $p$) is Alice's public key; $d$ is her private key.**

# 13.5.2 Continued

## Verifying and Signing

**Figure 13.10** *ElGamal digital signature scheme*



# 13.5.1 *Continued*

## Example 13.2

**Here is a trivial example. Alice chooses $p$ = 3119, $e_1$ = 2, $d$ = 127 and calculates $e_2 = 2^{127} \bmod 3119 = 1702$. She also chooses $r$ to be 307. She announces e*1*, e*2*, and $p$ publicly; she keeps $d$ secret. The following shows how Alice can sign a message.**

$$M = 320$$

$$S_1 = e_1{}^r = 2^{307} = 2083 \bmod 3119$$

$$S_2 = (M - d \times S_1) \times r^{-1} = (320 - 127 \times 2083) \times 307^{-1} = 2105 \bmod 3118$$

**Alice sends M, $S_1$, and $S_2$ to Bob. Bob uses the public key to calculate $V_1$ and $V_2$.**

$$V_1 = e_1{}^M = 2^{320} = 3006 \bmod 3119$$

$$V_2 = d^{\,S_1} \times S_1{}^{S_2} = 1702^{2083} \times 2083^{2105} = 3006 \bmod 3119$$

# 13.5.1 Continued

**Example 13.3**

**Now imagine that Alice wants to send another message, M = 3000, to Ted. She chooses a new *r*, 107. Alice sends M, $S_1$, and $S_2$ to Ted. Ted uses the public keys to calculate $V_1$ and $V_2$.**

$$M = 3000$$

$$S_1 = e_1^{\,r} = 2^{107} = 2732 \bmod 3119$$

$$S_2 = (M - d \times S_1)\, r^{-1} = (3000 - 127 \times 2083) \times 107^{-1} = 2526 \bmod 3118$$

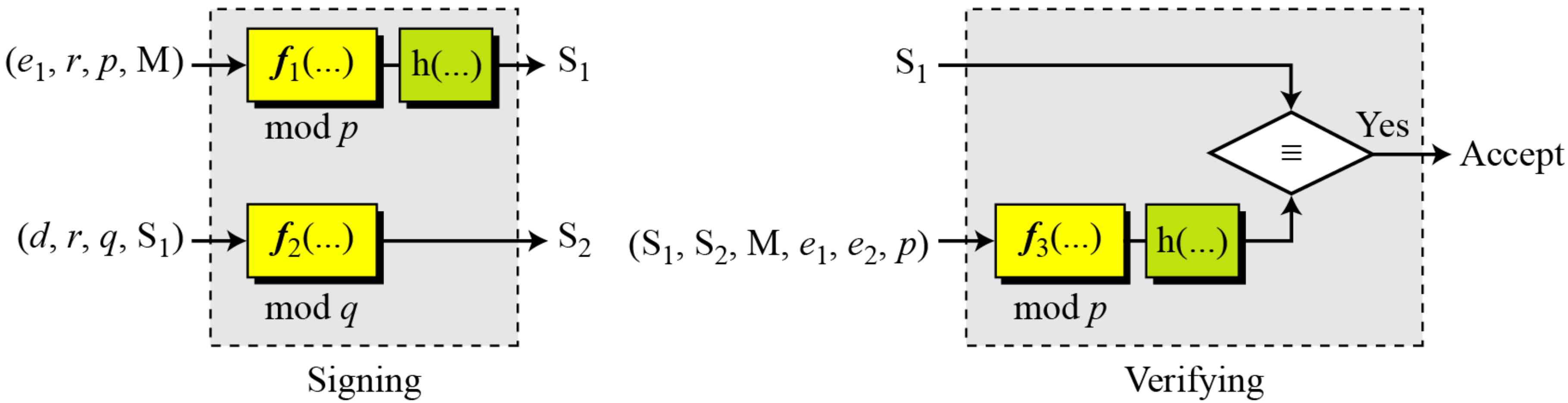
$$V_1 = e_1^{\,M} = 2^{3000} = 704 \bmod 3119$$

$$V_2 = d^{\,S_1} \times S_1^{\,S} = 1702^{2732} \times 2083^{2526} = 704 \bmod 3119$$

# 13.5.3 Schnorr Digital Signature Scheme

**Figure 13.11** *General idea behind the Schnorr digital signature scheme*

$S_1$, $S_2$: Signatures
M: Message
$(e_1, e_2, p, q)$: Alice's public key
$(d)$: Alice's private key
$r$: Random secret



# 13.5.3 Continued

## Key Generation

1) *Alice selects a prime p, which is usually 1024 bits in length.*
2) *Alice selects another prime q.*
3) *Alice chooses $e_1$ to be the qth root of 1 modulo p.*
4) *Alice chooses an integer, d, as her private key.*
5) *Alice calculates $e_2 = e_1^d \bmod p$.*
6) *Alice's public key is ($e_1$, $e_2$, p, q); her private key is (d).*

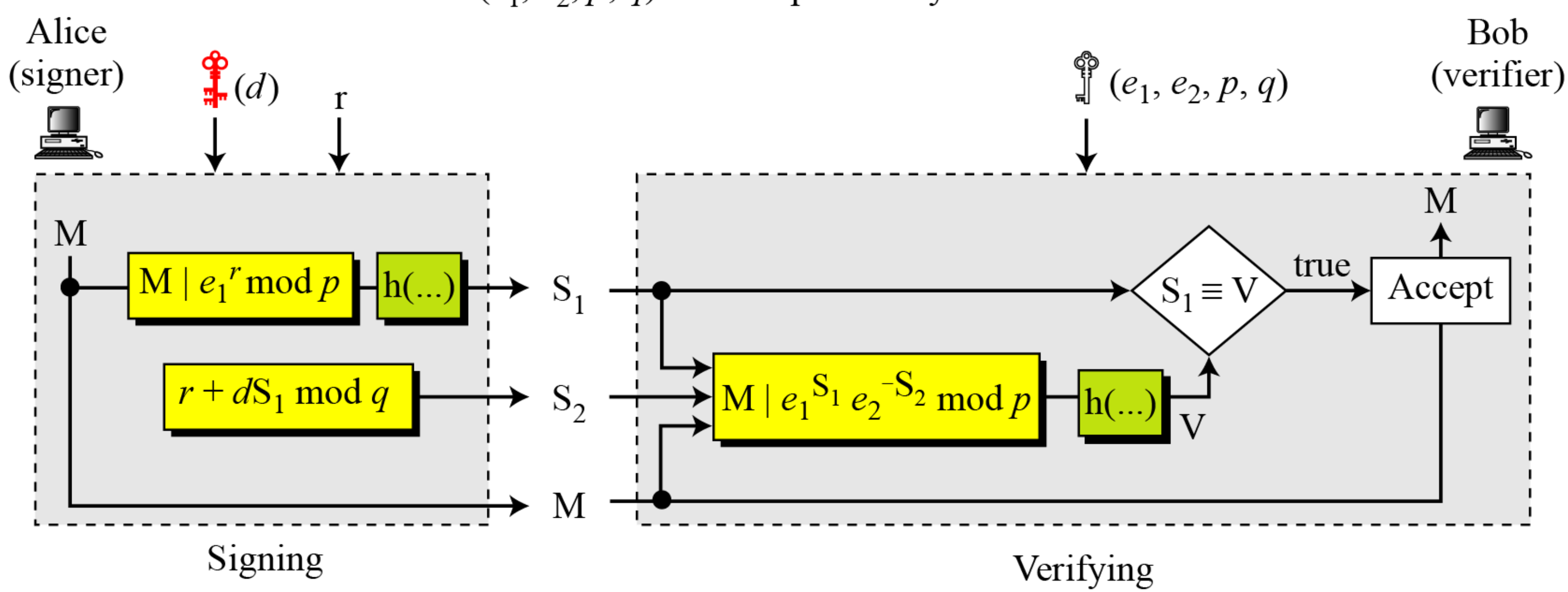
**Note**

**In the Schnorr digital signature scheme, Alice's public key is ($e_1$, $e_2$, $p$, $q$); her private key ($d$).**

## Signing and Verifying

**Figure 13.12** *Schnorr digital signature scheme*

M: Message
$S_1$, $S_2$: Signatures
V: Verification
$r$: Random secret
($d$): Alice's private key
($e_1$, $e_2$, $p$, $q$): Alice's public key
| : Concatenation
h(...): Hash algorithm



## 13.5.3 Continued

### *Signing*

*1. Alice chooses a random number r.*

*2. Alice calculates $S_1 = h(M|e_1^r \bmod p)$.*

*3. Alice calculates $S_2 = r + d \times S_1 \bmod q$.*

*4. Alice sends M, $S_1$, and $S_2$.*

### *Verifying Message*

*1. Bob calculates $V = h\ (M \mid e_1^{S2}\ e_2^{-S1} \bmod p)$.*

*2. If $S_1$ is congruent to V modulo p, the message is accepted;*

# 13.5.1 Continued

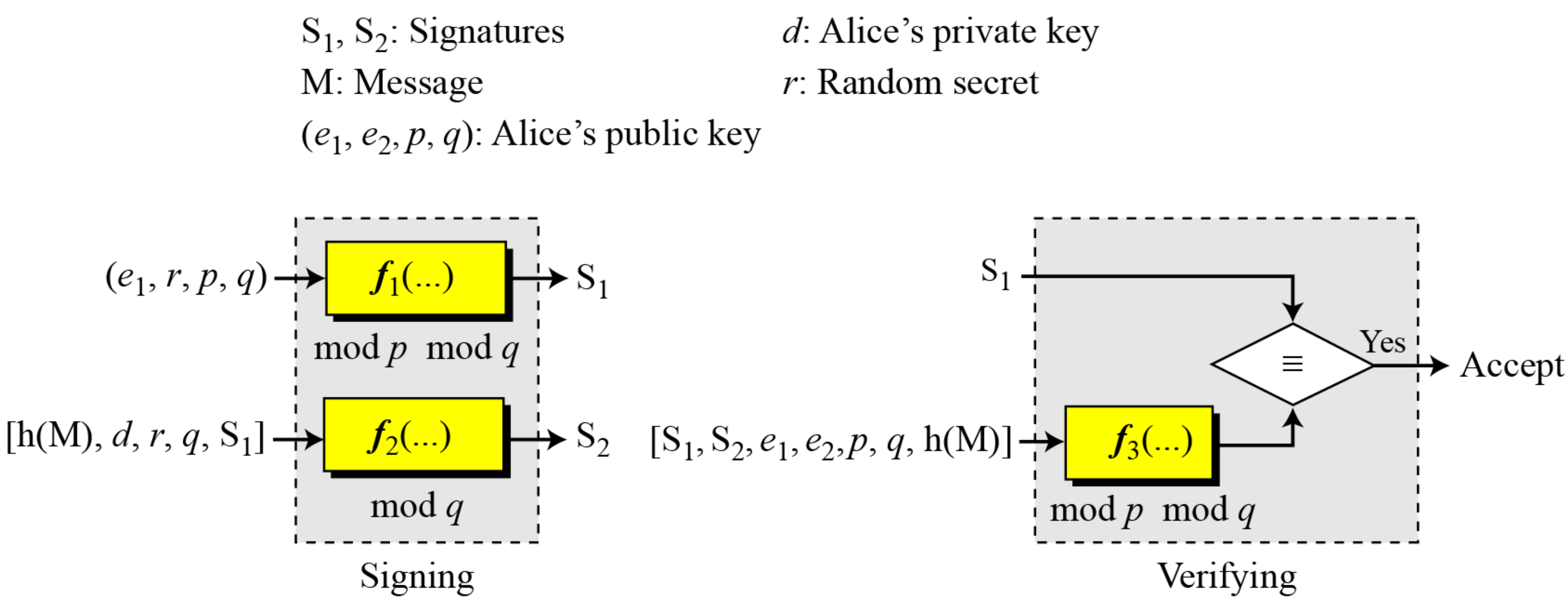
## Example 13.4

Here is a trivial example. Suppose we choose $q = 103$ and $p = 2267$. Note that $p = 22 \times q + 1$. We choose $e_0 = 2$, which is a primitive in $Z_{2267}$*. Then $(p - 1) / q = 22$, so we have $e_1 = 2^{22} \bmod 2267 = 354$. We choose $d = 30$, so $e_2 = 354^{30} \bmod 2267 = 1206$. Alice's private key is now ($d$); her public key is ($e_1$, $e_2$, $p$, $q$).

Alice wants to send a message M. She chooses $r = 11$ and calculates $e_2{}^r = 354^{11} = 630 \bmod 2267$. Assume that the message is 1000 and concatenation means 1000630. Also assume that the hash of this value gives the digest h(1000630) = 200. This means S1 = 200. Alice calculates $S2 = r + d \times S_1 \bmod q = 11 + 1026 \times 200 \bmod 103 = 35$. Alice sends the message M =1000, $S_1 = 200$, and $S_2 = 35$. The verification is left as an exercise.

# 13.5.4 Digital Signature Standard (DSS)

**Figure 13.13** ***General idea behind DSS scheme***



# 13.5.4 Continued

***Key Generation*.**

1) ***Alice chooses primes p and q.***

2) ***Alice uses*** $\langle Z_p^*, \times \rangle$ ***and*** $\langle Z_q^*, \times \rangle$.

3) ***Alice creates*** $e_1$ ***to be the qth root of 1 modulo p.***

4) ***Alice chooses d and calculates*** $e_2 = e_1^d$.

5) ***Alice's public key is*** $(e_1, e_2, p, q)$***; her private key is (d).***

# 13.5.4 *Continued*

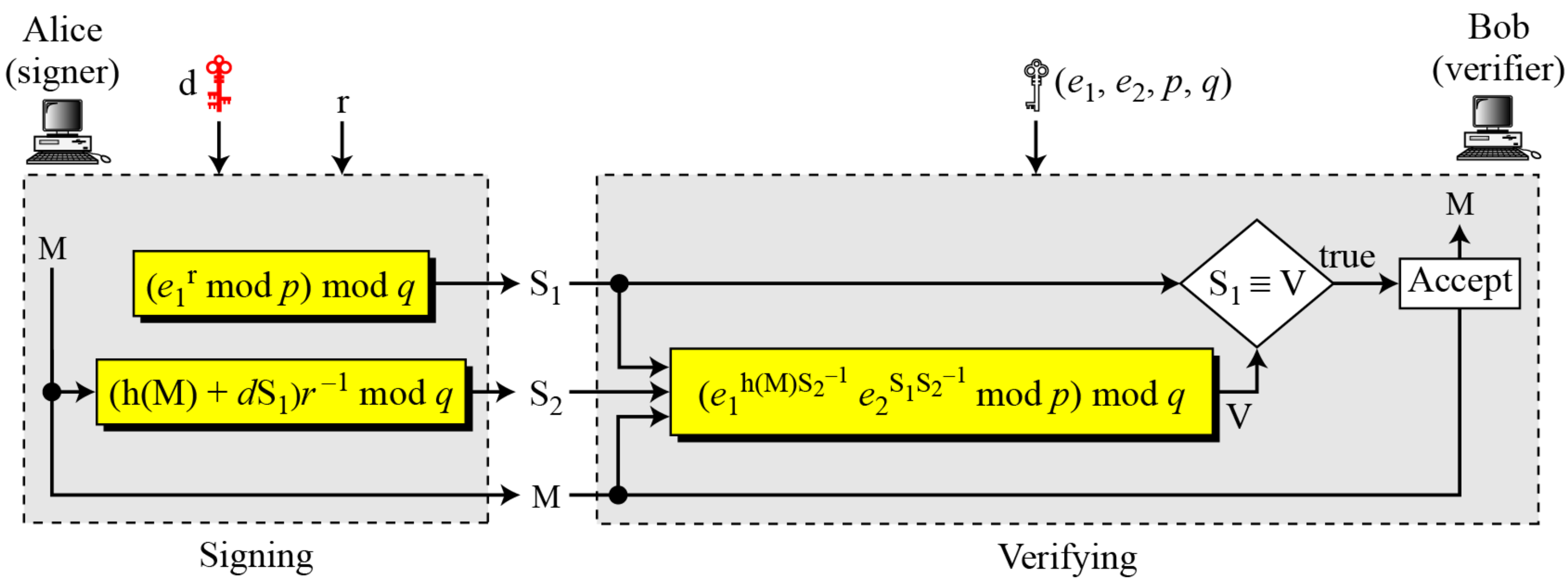
## *Verifying and Signing*

**Figure 13.14** ***DSS scheme***

M: Message | $r$: Random secret | h(M): Message digest

$S_1$, $S_2$: Signatures | $d$: Alice's private key

V: Verification | $(e_1, e_2, p, q)$: Alice's public key



# 13.5.1 *Continued*

**Example 13.5**

**Alice chooses $q$ = 101 and p = 8081. Alice *selects* $e_0$ = 3 and calculates $e^1 = e_0^{(p-1)/q} \bmod p = 6968$. Alice chooses $d$ = 61 as the private key and calculates $e_2 = e_1^d \bmod p = 2038$. Now Alice can send a message to Bob. Assume that h(M) = 5000 and Alice chooses $r$ = 61:**

$$\mathrm{h(M)} = 5000 \qquad r = 61$$

$$S_1 = (e_1^{\,r} \bmod p) \bmod q = 54$$

$$S_2 = ((\mathrm{h(M)} + d\, S_1)\, r^{-1}) \bmod q = 40$$

**Alice sends M, $S_1$, and $S_2$ to Bob. Bob uses the public keys to calculate V.**

$$S_2^{\,-1} = 48 \bmod 101$$

$$V = [(6968^{5000 \times 48} \times 2038^{54 \times 48}) \bmod 8081] \bmod 101 = 54$$

# *13.5.4 Continued*

***DSS Versus RSA***

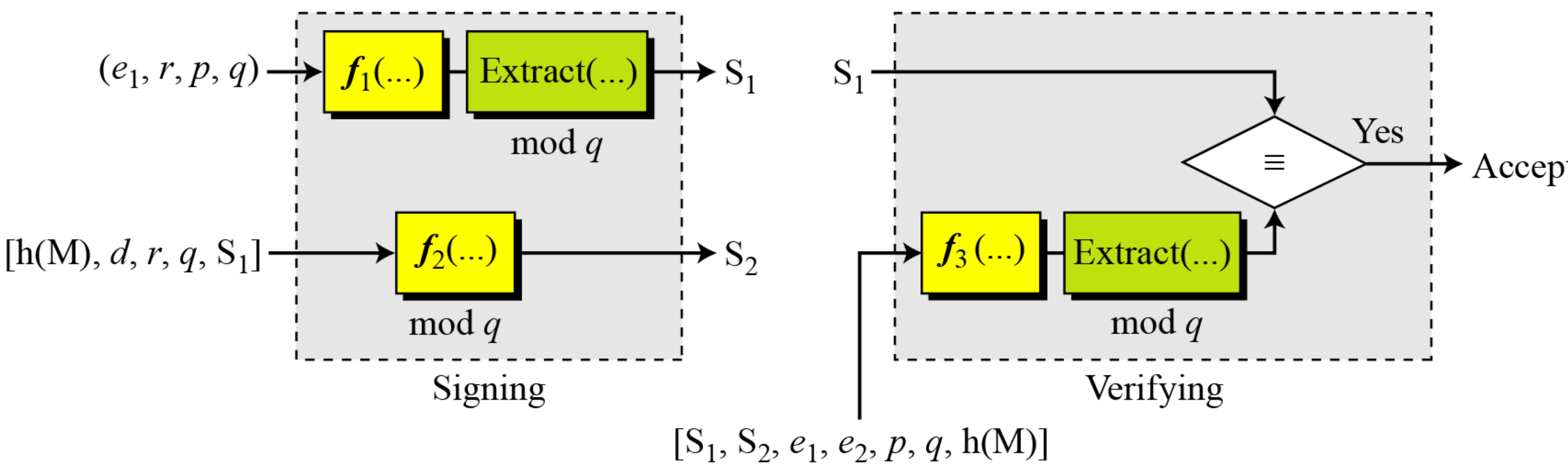
***Computation of DSS signatures is faster than computation of RSA signatures when using the same p.***

***DSS Versus ElGamal***

***DSS signatures are smaller than ElGamal signatures because q is smaller than p.***

**Figure 13.15** *General idea behind the ECDSS scheme*

$S_1$, $S_2$: Signatures
M: Message
$(a, b, p, q, e_1, e_2)$: Alice's public key
$d$: Alice's private key
$r$: Random secret



# 13.5.5 Continued

## Key Generation

***Key generation follows these steps:***

1) ***Alice chooses an elliptic curve $E_p(a, b)$.***

2) ***Alice chooses another prime q the private key d.***

3) ***Alice chooses $e_1(\ldots, \ldots)$, a point on the curve.***

4) ***Alice calculates $e_2(\ldots, \ldots) = d \times e_1(\ldots, \ldots)$.***

5) ***Alice's public key is (a, b, p, q, e1, e2); her private key is d.***

# 13.5.5 Continued

## Signing and Verifying

**Figure 13.16** ***The ECDSS scheme***

M: Message
$S_1$, $S_2$: Signatures
V: Verification
$r$: Random secret
$d$: Alice's private key
$(a, b, p, q, e_1, e_2)$: Alice's public key
$P(u, v)$, $T(x, y)$: Points on the curve
h(M): Message digest
A, B: Intermediate results