Digital Signatures

Introduction

- A digital signature is an authentication mechanism that enables the creator of a message to attach a code that acts as a signature.
- The signature is formed by taking the hash of the message and encrypting the message with the creator’s private key.
- The signature guarantees the source and integrity of the message
- a signature on a document, when verified, is a sign of authentication – the document is authentic
- When Alice sends a message to Bob, Bob needs to check the authenticity of the sender; he needs to be sure that the message comes from Alice and not Eve. Bob can ask Alice to sign the message electronically.
- In other words, an electronic signature can prove the authenticity of Alice as the sender of the message. - this type of signature is referred to as a **digital signature**.

Differences between conventional signatures and digital signatures

- **Inclusion**-A conventional signature is included in the document; it is part of the document
- But when we sign a document digitally, we send the signature as a separate document.
- The sender sends two documents: the message and the signature.
- The recipient receives both documents and verifies that the signature belongs to the supposed sender.
- If this is proven,the message is kept; otherwise, it is rejected.

Verification Method

- Conventional signature
  - when the recipient receives a document, she compares the signature on the document with the signature on file. If they are the same, the document is authentic.
  - The recipient needs to have a copy of this signature on file for comparison.
- Digital signature
  - the recipient receives the message and the signature
  - Copy of the signature is not stored anywhere.
  - The recipient needs to apply a verification technique to the combination of the message and the signature to verify the authenticity.

Relationship

Conventional signature

- there is normally a one-to-many relationship between a signature and documents.
- A person uses the same signature to sign many documents.
- For a digital

Digital signature

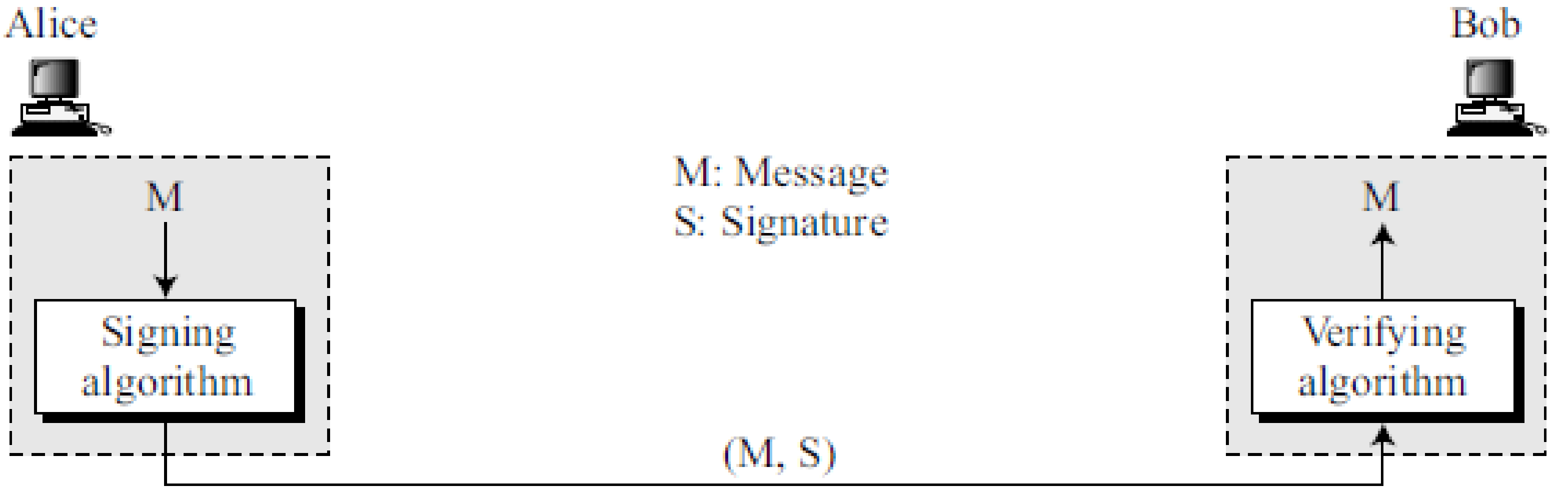
- there is a one-to-one relationship between a signature and a message.
- Each message has its own signature.
- The signature of one message cannot be used in another message.
- If Bob receives two messages, one after another, from Alice, he cannot use the signature of the first message to verify the second. Each message needs a new signature

Duplicity

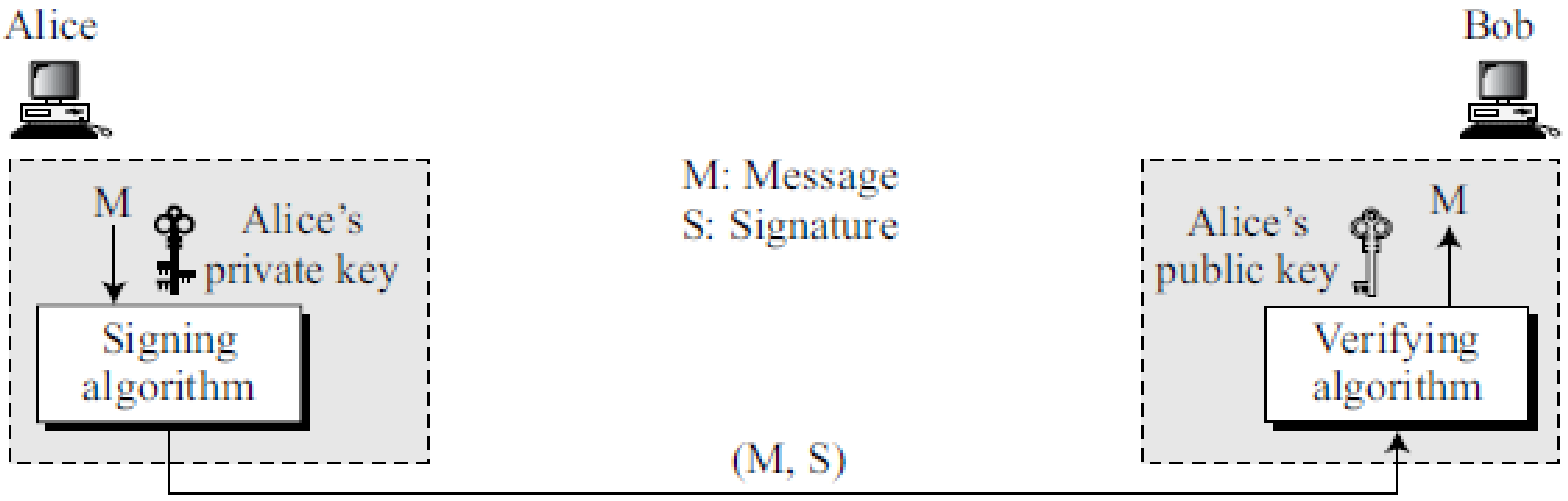
- conventional signature
    - a copy of the signed document can be distinguished from the original one on file.
    - In, there is no such distinction unless there is a
- digital signature
    - no such distinction unless there is aactor of time (such as a timestamp) on the document.
    - For example, suppose Alice sends a document instructing Bob to pay Eve. If Eve intercepts the document and the signature, she can replay it later to get money again from Bob.

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.



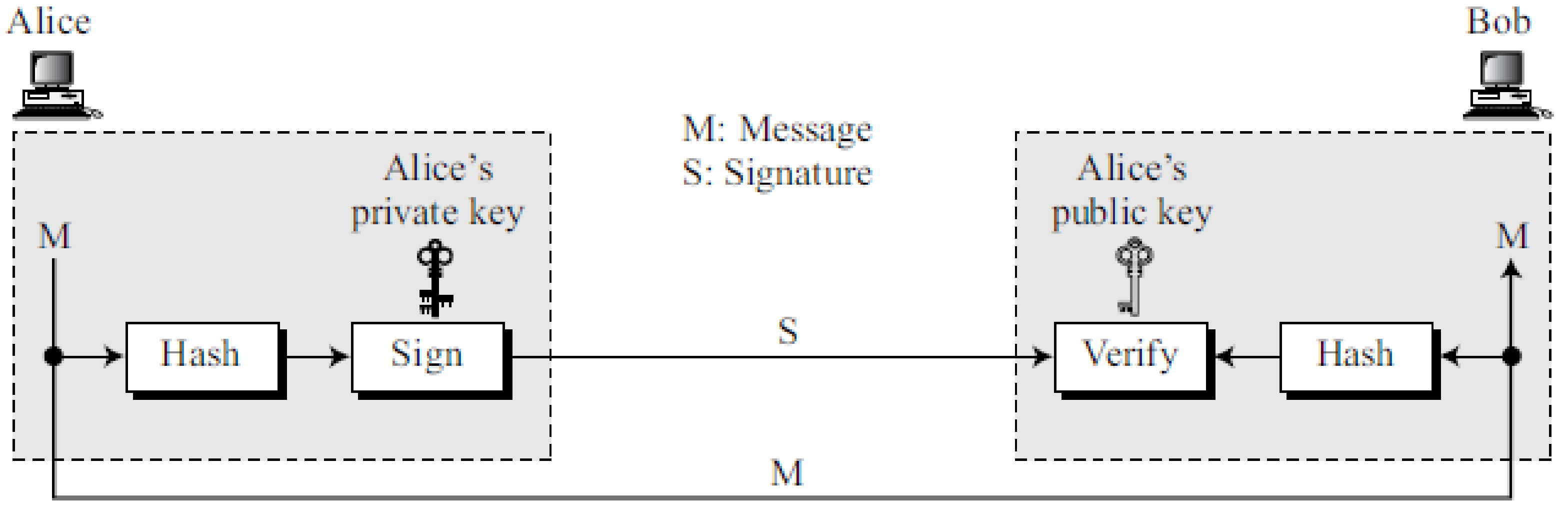
- the signer uses her private key, applied to a signing algorithm, to sign the document.
- The verifier, uses the public key of the signer, applied to the verifying algorithm, to verify the document



Can symmetric key be used to both sign and verify a signature?

- No.
- a secret key is known by only two entities (Alice and Bob, for example). So if Alice needs to sign another document and send it to Ted, she needs to use another secret key.
- Creating a secret key for a session involves authentication, which uses a digital signature.
- Bob could use the secret key between himself and Alice, sign a document, send it to Ted, and pretend that it came from Alice.

Signing the Digest



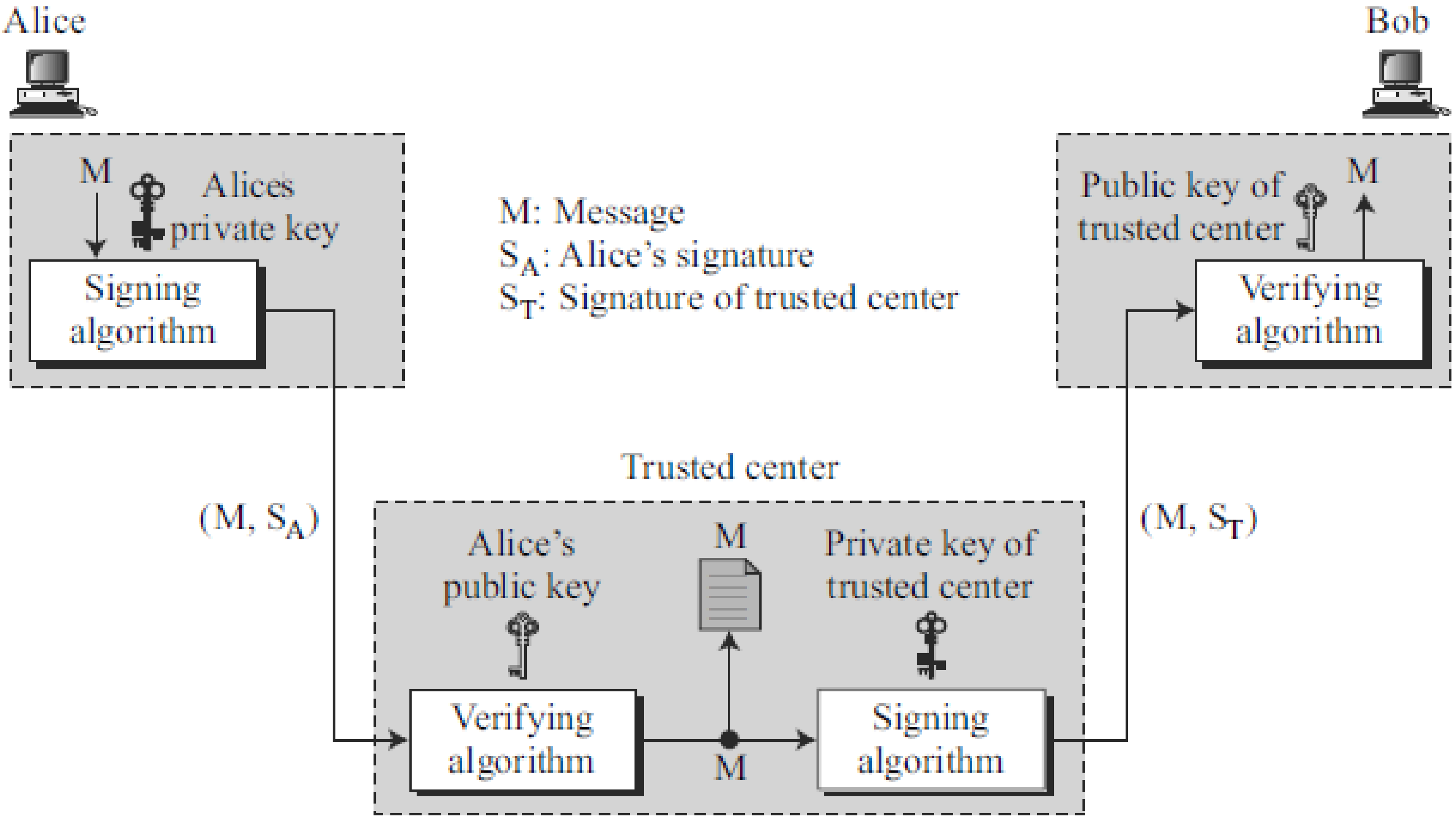
SERVICES

- A digital signature can directly provide
  - message authentication
  - message integrity
  - nonrepudiation.
- A secure digital signature scheme like a secure conventional signature can provide message authentication (also referred to as data-origin authentication).
- Bob can verify that the message is sent by Alice because Alice’s public key is used in verification.
- Alice’s public key cannot verify the signature signed by Eve’s private key.

- **Message Integrity-**
  - cannot get the same signature if the message is changed.
  - The digital signature schemes today use a hash function in the signing and verifying algorithms that preserve the integrity of the message
- **Non Repudiation**
- If Alice signs a message and then denies it, can Bob later prove that Alice actually signed it?
- Suppose Alice sends a message to a bank (Bob) and asks to transfer some amount from her account to Ted's account, can Alice later deny that she sent this message?
- With this scheme Bob must keep the signature on file and later use Alice's public key to create the original message to prove the message in the file and the newly created message are the same.
  - not feasible because Alice may have changed her private or public key during this time
  - may also claim that the file containing the signature is not authentic.

  Solution- Trusted third party

Trusted third party

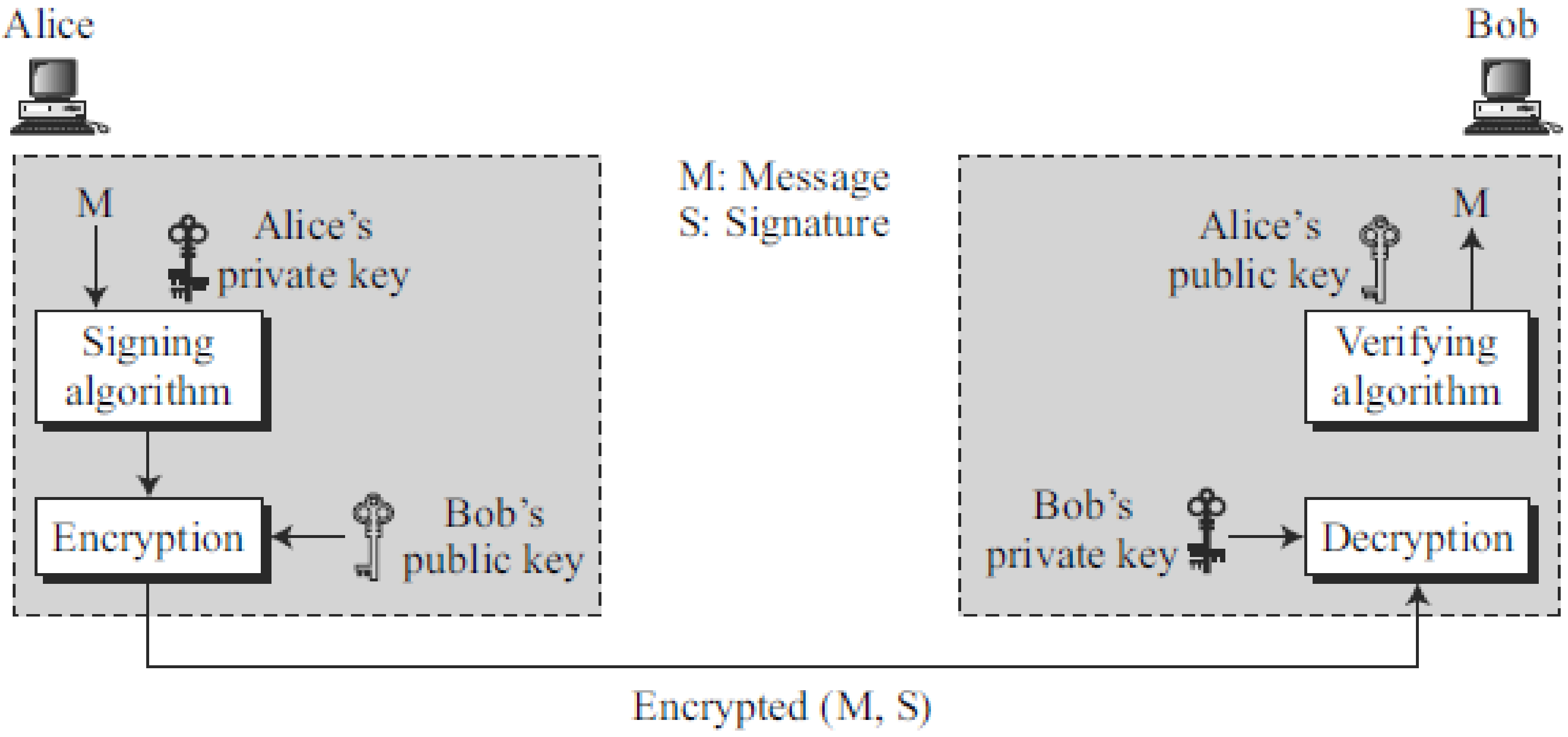


Trusted third party Contd…

- Alice creates a signature from her message (SA) and sends the message, her identity, Bob’s identity, and the signature to the center.
- The center, after checking that Alice’s public key is valid, verifies through Alice’s public key that the message came from Alice
- The center then saves a copy of the message with the sender identity, recipient identity, and a timestamp in its archive.
- The center uses its private key to create another signature (ST) from the message.
- The center then sends the message, the new signature, Alice’s identity, and Bob’s identity to Bob.
- Bob verifies the message using the public key of the trusted center
- If in the future Alice denies that she sent the message, the center can show a copy of the saved message.
- If Bob’s message is a duplicate of the message saved at the center, Alice will lose the dispute.

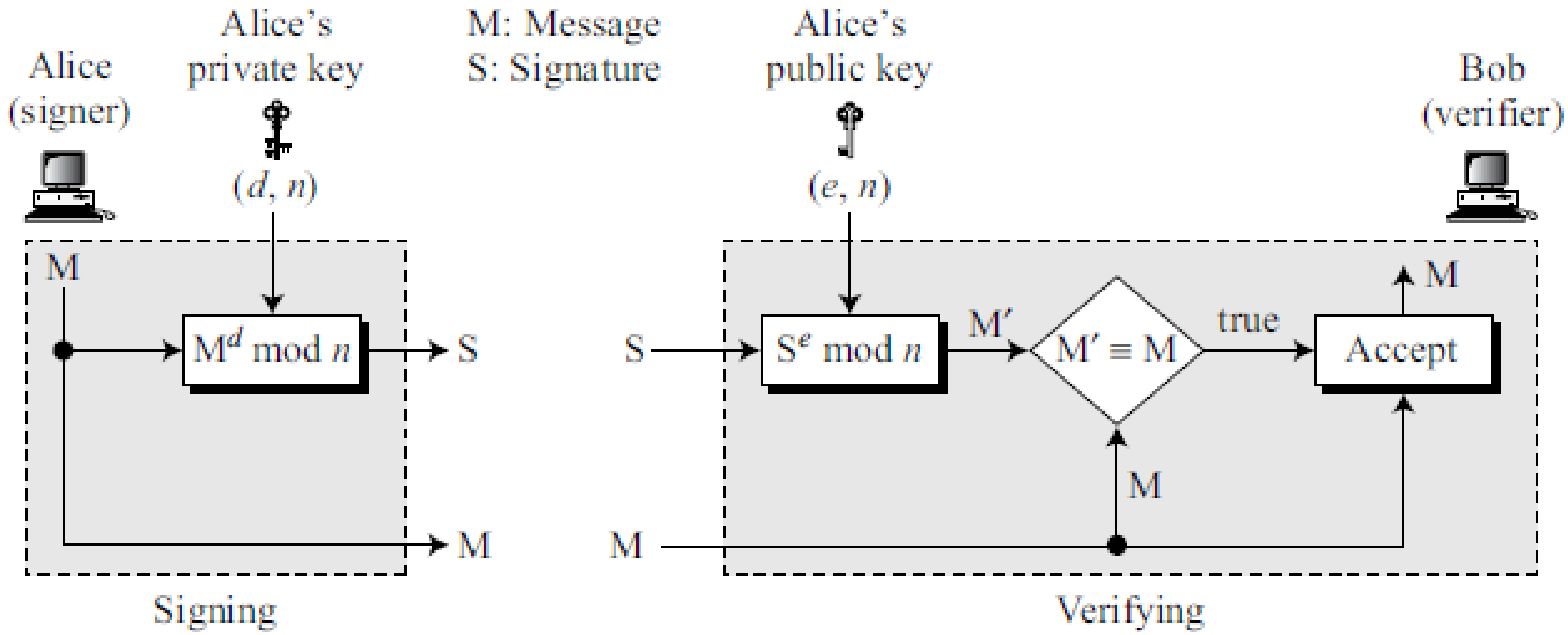
Confidentiality

- A digital signature does not provide confidential communication
- If confidentiality is required, the message and the signature must be encrypted using either a secret-key or public-key cryptosystem



DIGITAL SIGNATURE SCHEMES

- RSA Digital Signature Scheme



Example1

Let $p \leftarrow 7, q \leftarrow 13$, Choose $e \leftarrow 5$

Find d

Sign the message $m=35$ and verify the signature

Example1 -Solution

Let p←7,q←13, Choose e←5

Find d

Sign the message m=35

d←29

$s=m^{d}$ modn = 35^{29} mod n =42

(m,s)=(35,42)

M’ = $42^{5}\equiv 35$modn .

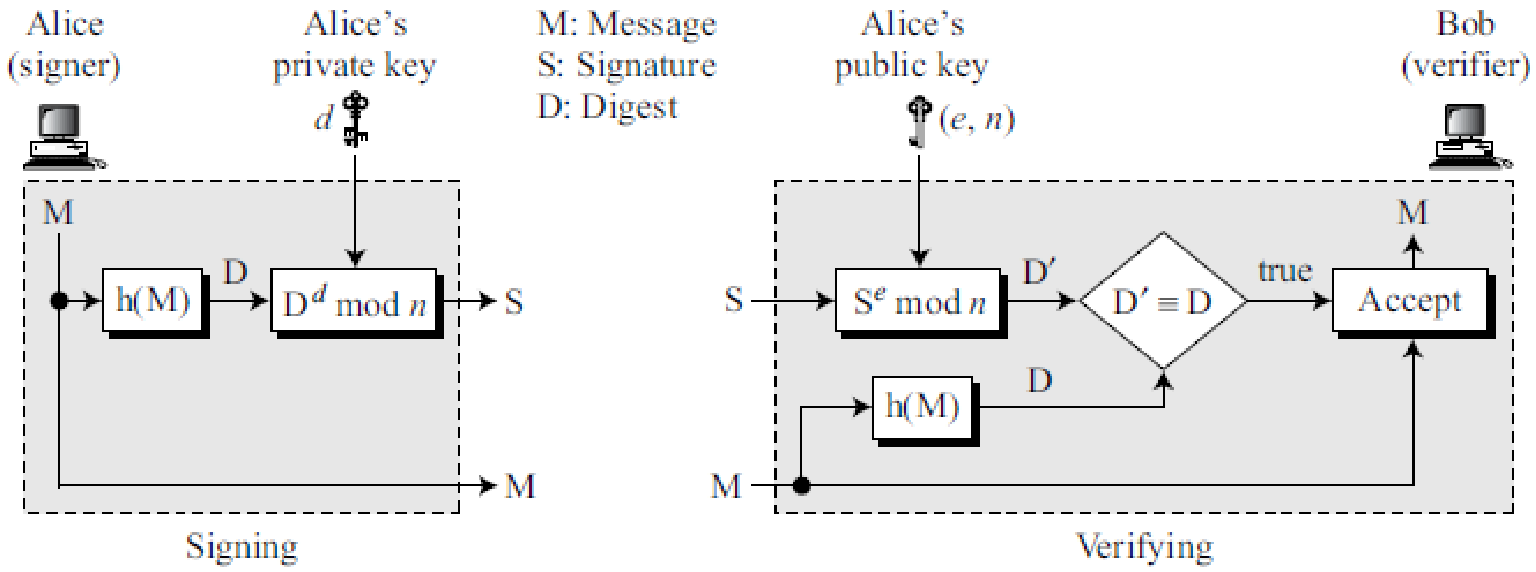
Example2

Let p←53,q←59, Choose e←5

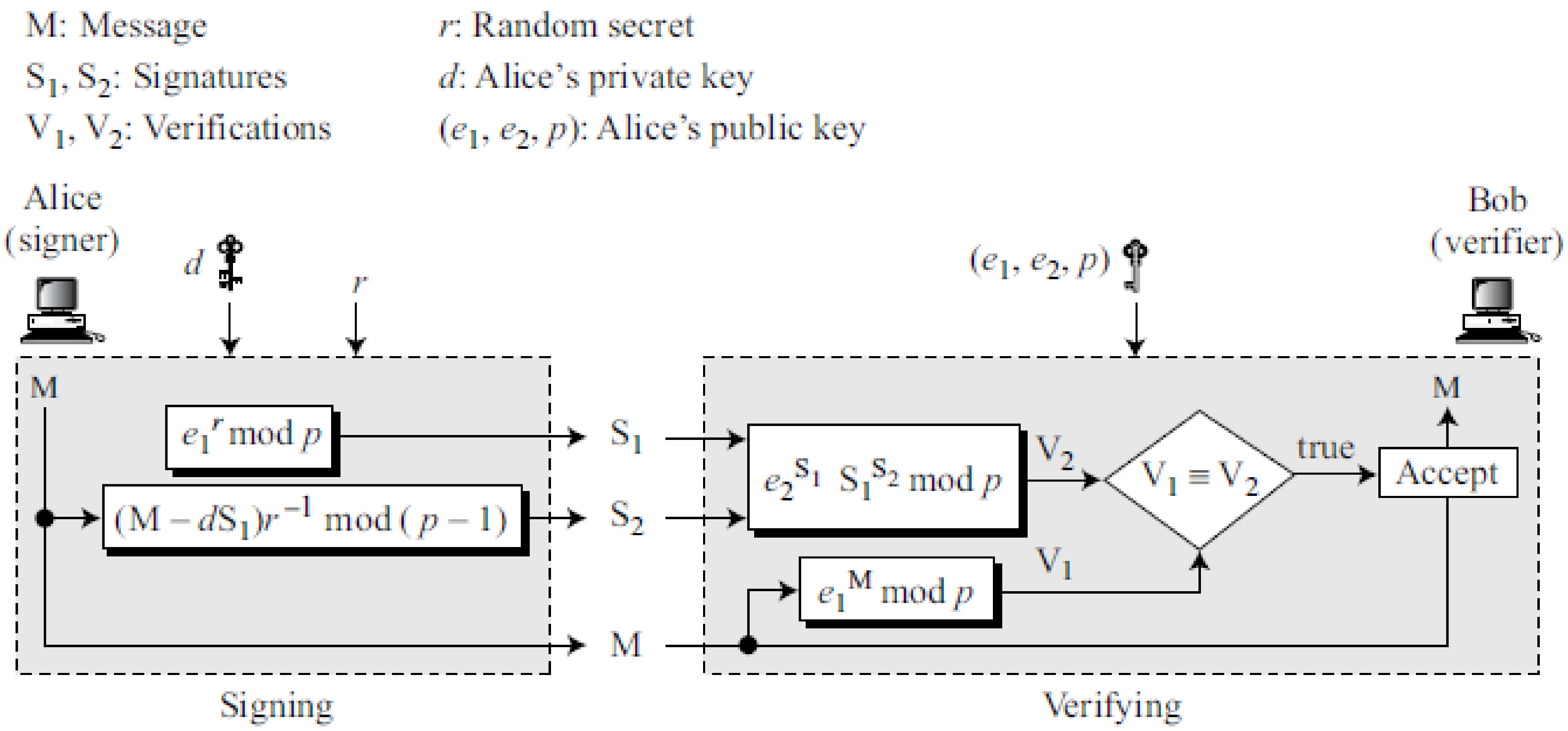
Find d

Sign the message m=1394 and verify the signature

The RSA signature on the message digest



ElGamal Digital Signature Scheme



Signing

- Alice chooses a secret random number r.
- Though public and private keys can be used repeatedly, Alice needs a new r each time she signs a new message
- Alice calculates the first signature $S1 = e1^{r} \bmod p$.
- Alice calculates the second signature $S2 = (M - d \times S1) \times r^{-1} \bmod (p - 1)$, where r^{-1} is the multiplicative inverse of r modulo p-1.
- Alice sends M, S1, and S2 to Bob.

Verifying

Bob receives M, S1, and S2, which can be verified as follows:

1. Bob checks to see if $0 < S1 < p$

2. Bob checks to see if $0 < S2 < p - 1$

3. Bob calculates $V1 = e1^{M} \bmod p$

4. Bob calculates $V2 = e2^{S1} \times S1^{S2} \bmod p$

5. If V1 is congruent to V2, the message is accepted; otherwise, it is rejected.

Example

q = 19, d=16, r =5, e1=10 sign the message M=14 and verify the same

Example

q = 19, d=16, r =5, e1=10 sign the message M=14 and verify the same

S1 = e1^{r} mod p = 10^{5} mod 19 =3

E2= e1^{d} mod p= 10^{16} mod 19 =4

r $^{-1}$ mod (p − 1) = 5 $^{-1}$ mod 18 =11

S2 = (M − d × S1) × r $^{-1}$ mod (p − 1)= (14 − 16 × 3) × 11 mod (18)= 4

Sends 14, 3, 4 to Bob

Bob calculates V1 = e1^{M} mod p = 10^{14} mod 19 =16

Bob calculates V2 = e2^{S1} × S1^{S2} mod p= 4^{3} × 3^{4} mod 19 =16

Therefore Signature is valid

Schnorr Digital Signature Scheme

- The problem with the ElGamal digital signature scheme is that p needs to be very large to guarantee that the discrete log problem is intractable in Zp*.
- The recommendation is a p of at least 1024 bits
- This could make the signature as large as 2048 bits
- Schnorr proposed a new scheme based on ElGamal, but with a reduced signature size.

Key Generation

- Before signing a message, Alice needs to generate keys and announce the public ones to the public.

1. Alice selects a prime p, which is usually 1024 bits in length.

2. Alice selects another prime q, which is the same size as the digest created by the cryptographic hash function. The prime q needs to divide (p − 1). In other words, (p − 1) = 0 mod q.

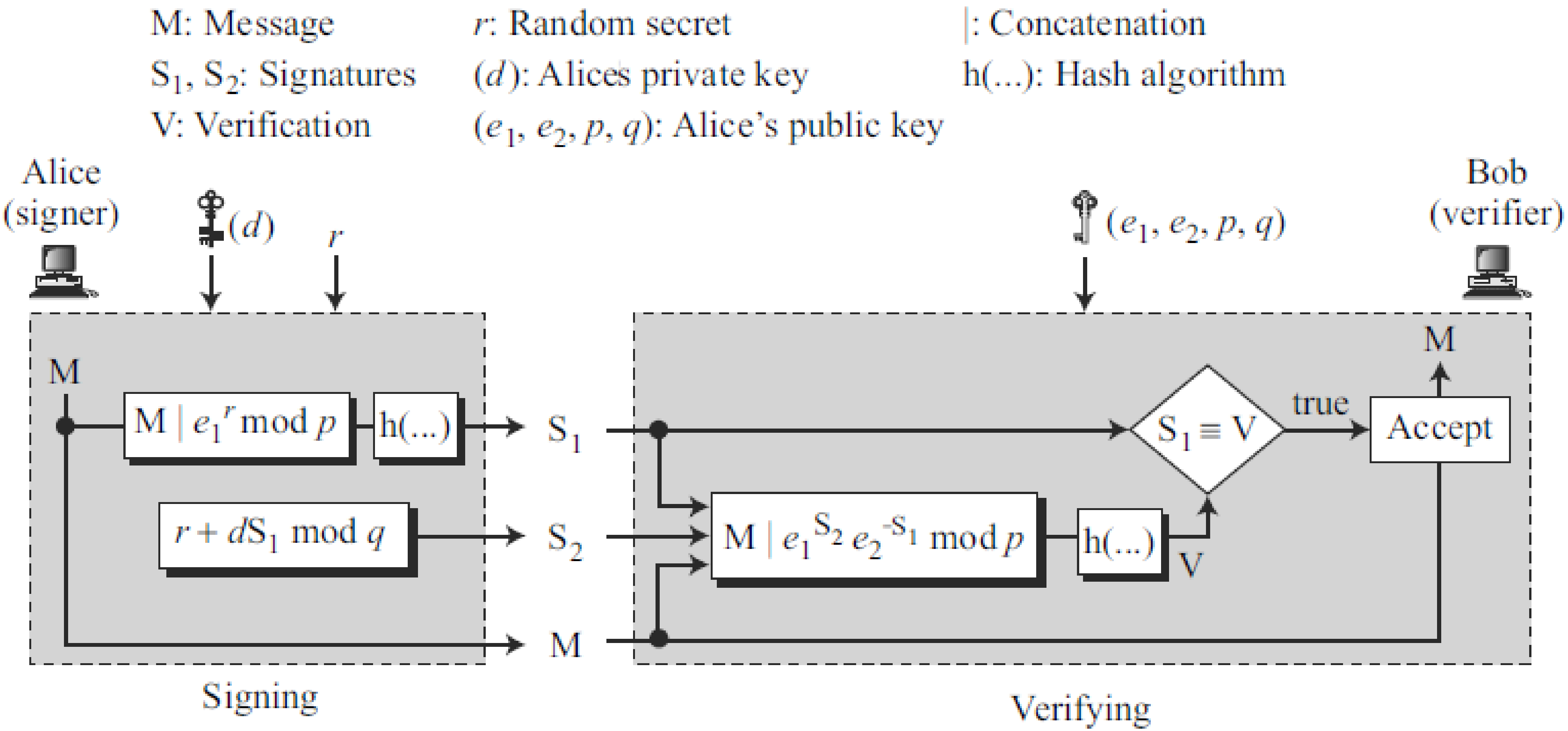
3. Alice chooses e1 to be the qth root of 1 modulo p. To do so, Alice chooses a primitive element in Zp, and calculates $e1 = e0^{(p-1)/q}$ mod p.

4. Alice chooses an integer, d, as her private key.

5. Alice calculates $e2 = e1^{d}$ mod p.

6. Alice’s public key is (e1, e2, p, q); her private key is (d);

Signing and Verifying



Signing

1. Alice chooses a random number r. Although public and private keys can be used to sign multiple messages, Alice needs to change r each time she sends a new message. Note also that r needs to be between 1 and q.

2. Alice calculates the first signature S1 = h(M| $e1^r$ mod p).

The message is prepended to the value of $e1^r$ mod p then the hash function is applied to create a digest.

Alice calculates the second signature S2 = r + d × S1 mod q.

4. Alice sends M, S1, and S2.

Verifying Message

The receiver, Bob receives M, S1, and S2.

1. Bob calculates V = h (M | $e1^{S2}$ $e2^{-S1}$mod p).

2. If S1 is congruent to V modulo p, the message is accepted; otherwise, it is rejected.

1. Alice selects a prime p=31
2. Alice selects another prime q, such that (p − 1) = 0 mod q. Let q=5
3. Let e0=3; e1 = e0$^{(p−1)/q}$ mod p= 3$^{(31−1)/5}$ mod 31 = 16
4. Alice chooses an integer, d, as her private key. Let d=7
5. Alice calculates e2 = e1^{d} mod p= 16^{7} mod 31 =8
6. Alice's public key is (e1, e2, p, q); her private key is (d);

- Alice chooses a random number r Let r=3
- S1 = h(M| e1^{r} mod p)= h(14| 16^{3} mod 31)= h(14 | 4 mod 31) Let it be = 9
- S2 = r + d × S1 mod q = 3 + 7 × 9 mod 5 =1
- Alice sends 14, 9, and 1

Verifying Message

The receiver, Bob receives 14, 9, and 1

1. Bob calculates V = h (M | $e1^{S2}$ $e2^{-S1}$mod p)= h (14 | 16^{1} 8^{-9}mod 31)=h(14|4 mod31) =9

2. If S1 is congruent to V modulo p, the message is accepted; otherwise, it is rejected.

Digital Signature Standard (DSS)

based on the ElGamal scheme with some ideas from the Schnorr scheme

Key Generation

1. Alice chooses a prime p, between 512 and 1024 bits in length. The number of bits in p must be a multiple of 64.

2. Alice chooses a 160-bit prime q in such a way that q divides (p − 1).

3. Alice uses two multiplication groups <Zp*, × > and <Zq*, ×>; the second is a subgroup of the first.

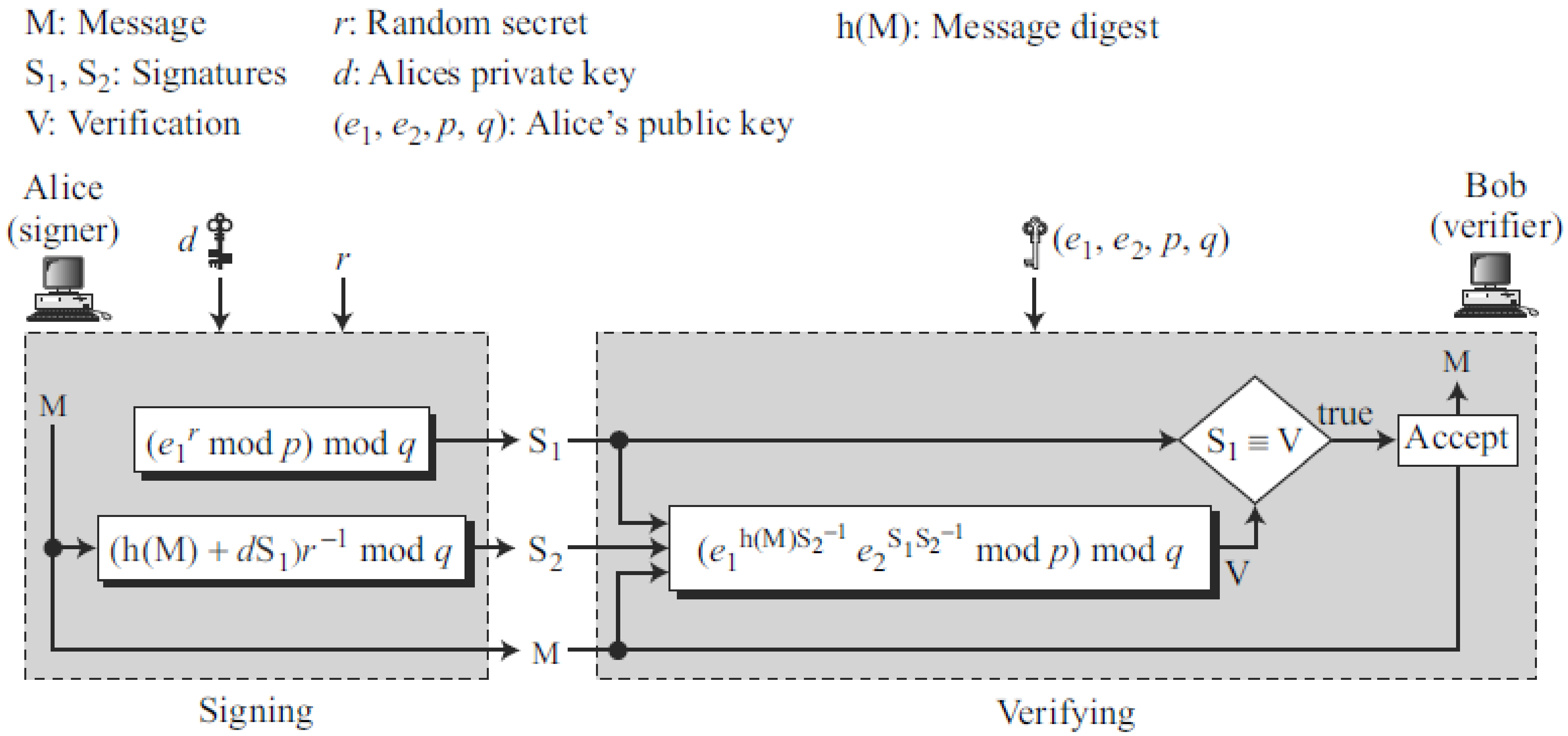
4. Alice creates e1 to be the qth root of 1 modulo p ($e1^{p} = 1$ mod p). To do so, Alice

chooses a primitive element in Zp, e0, and calculates $e1 = e0^{(p-1)/q}$ mod p.

5. Alice chooses d as the private key and calculates $e2 = e1^{d}$ mod p.

6. Alice’s public key is (e1, e2, p, q); her private key is (d).

Digital Signature Standard (DSS)



Signing

The following shows the steps to sign the message:

1. Alice chooses a random number r ($1 \leq r \leq q$). Although public and private keys can be chosen once and used to sign many messages, Alice needs to select anew r each time she needs to sign a new message.

2. Alice calculates the first signature S1 = ($e1^r$ mod p) mod q. Note that the value of the first signature does not depend on M, the message.

3. Alice creates a digest of message h(M).

4. Alice calculates the second signature S2 = (h(M) + d S1)r^{-1}mod q. Note that the calculation of S2 is done in modulo q arithmetic.

5. Alice sends M, S1, and S2 to Bob.

Verifying

Following are the steps used to verify the message when M, S1, and S2 are received:

1. Bob checks to see if $0 < S1 < q$.
2. Bob checks to see if $0 < S2 < q$.
3. Bob calculates a digest of M using the same hash algorithm used by Alice.
4. Bob calculates $V = [(e1^{h(M)}S2^{-1}e2^{S1\ S2^{-1}}) \bmod p] \bmod q$.
5. If S1 is congruent to V, the message is accepted; otherwise, it is rejected.

Example

Alice chooses q = 5 and p = 31. Alice selects e0 = 3 and calculates

e1 = $e0^{(p-1)/q}$ mod p= $3^{(31-1)/5}$ mod 31 = 16

Alice chooses d = 7 as the private key and calculates Alice calculates e2 = $e1^{d}$ mod p= 16^{7} mod 31 =8

Now Alice can send a message to Bob. Assume that h(M) = 50 and Alice chooses r = 3

h(M) = 50, r = 3

S1 = ($e1^{r}$ mod p)mod q = (16^{3} mod 31)mod 5 = 4

r^{-1}mod q = 3^{-1}mod 5 = 2

S2 = (h(M) + d S1)r^{-1}mod q = ((50 + 7. 4).2) mod 5 = 1

Alice sends M, S1, and S2 to Bob. Bob uses the public keys to calculate V.

$S2^{-1}$ mod q= 1^{-1} mod 5 =1

V = [($e1^{h(M)\ S2^{-1}}$ $e2^{S1\ S2^{-1}}$) mod p] mod q = [($16^{50.\ 1}$ $8^{4\ .1}$) mod 31] mod 5= 4

Because S1 is congruent to V, the message is accepted

Elliptic Curve Digital Signature Scheme(ECDSS)

- Referred as ECDSA (elliptic curve DSA

Key Generation: Key generation follows these steps:

1. Alice chooses an elliptic curve Ep(a, b) with p a prime number.

2. Alice chooses another prime number q to be used in the calculation.

3. Alice chooses the private key d, an integer.

4. Alice chooses e1(…, …), a point on the curve.

5. Alice calculates e2(…, …) = d × e1(…, …), another point on the curve.

6. Alice’s public key is (a, b, p, q, e1, e2); her private key is d..

The ECDSS scheme

M: Message

S_1, S_2: Signatures

V: Verification

r: Random secret

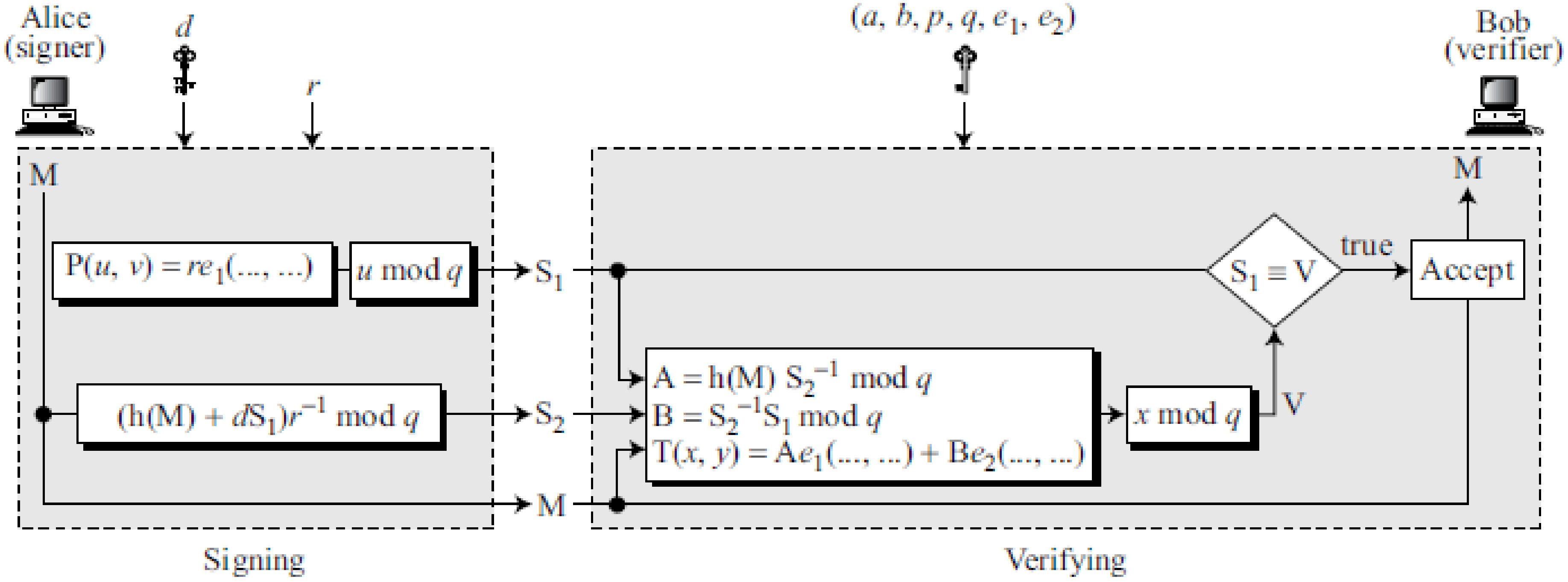
d: Alice's private key

(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


Alice
(signer)
d
r
(a, b, p, q, e_1, e_2)
Bob
(verifier)
M
$P(u, v) = re_1(..., ...)$
$u \bmod q$
S_1
$(h(M) + dS_1)r^{-1} \bmod q$
S_2
M
$A = h(M)\ S_2^{-1} \bmod q$
$B = S_2^{-1}S_1 \bmod q$
$T(x, y) = Ae_1(..., ...) + Be_2(..., ...)$
$x \bmod q$
V
$S_1 \equiv V$
true
Accept
M
Signing
Verifying

Signing

The signing process consists mainly of choosing a secret random number, creating a third point on the curve, calculating two signatures, and sending the message and signatures.

1. Alice chooses a secret random number r, between 1 and q − 1.

2. Alice selects a third point on the curve, P(u, v) = r × e1 (…, …).

3. Alice uses the first coordinates of P(u, v) to calculate the first signature S1. This means S1 = u mod q.

4. Alice uses the digest of the message, her private key, and the secret random number r, and the S1 to calculate the second signature S2 = (h(M) + d × S1) r^{-1}mod q.

5. Alice sends M, S1, and S2.

Verifying

The verification process consists mainly of reconstructing the third point and verifying that the first coordinate is equivalent to S1 in modulo q.

Note that the third point was created by the signer using the secret random number r.

The verifier does not have this value. He needs to make the third point from the message digest, S1 and S2:

1. Bob uses M, S1, and S2 to create two intermediate results, A and B:

A = h(M) $S2^{-1}$ mod q and B = $S2^{-1}$ S1 mod q

Bob then reconstructs the third point T(x, y) = A × e1 (…, …) + B × e2(…, …).

2. Bob uses the first coordinate of T(x, y) to verify the message. If x = S1 mod q, the signature is verified; otherwise, it is rejected.

Example

Consider the elliptic curve $y^2 = x^3 + x - 1 \pmod{11}$, and a point P(1,1) on the curve

Let q=5, r=4, d=6

H(M)=11

Sign and verify the signature

DIGITAL SIGNATURES-VARIATIONS AND APPLICATIONS

Time Stamped Signatures

- signed document needs to be timestamped to prevent it from being replayed by an adversary- called timestamped digital signature scheme
- Including the actual date and time on the documents may create a problem if the clocks are not synchronized and a universal time is not used.
- One solution is to use a nonce (a one-time random number).
- A nonce is a number that can be used only once.
- When the receiver receives a document with a nonce, he makes a note that the number is now used by the sender and cannot be used again.
- In other words, a new nonce defines the “present time”; a used nonce defines “past time”.

Blind Signatures

- Sometimes we have a document that we want to get signed without revealing the contents of the document to the signer.
- For example, a scientist, say Bob, might have discovered a very important theory that needs to be signed by a notary public, say Alice, without allowing Alice to know the contents of the theory.
- David Chaum has developed some blind digital signature schemes for this purpose.
- The main idea is as follows:

a. Bob creates a message and blinds it. Bob sends the blinded message to Alice.

b. Alice signs the blinded message and returns the signature on the blinded message.

c. Bob unblinds the signature to obtain a signature on the original message.

Blind Signature Based on the RSA Scheme

Bob selects a random number, b, and calculates the blinded message

B = M × b^e mod n ; b is sometimes called the blinding factor.

Bob sends B to Alice.

Alice signs the blinded message using the signing algorithm

S_b = B^d mod n

S_b is the signature on the blind version of the message

Bob uses the multiplicative inverse of his random number b to remove the blind from the signature.

The signature is S = S_b b^{-1} mod n.

We can prove that

S is the signature on the original message as defined in the RSA digital signature scheme:

$$\mathrm{S} \equiv \mathrm{S}_b\, b^{-1} \equiv \mathrm{B}^d\, b^{-1} \equiv (\mathrm{M} \times b^{\mathrm{e}})^d\, b^{-1} \equiv \mathrm{M}^d\, b^{ed}\, b^{-1} \equiv \mathrm{M}^{\mathrm{d}}\, b\, b^{-1} \equiv \mathrm{M}^d$$

S is the signature if Bob has sent the original message to be signed by Alice.

Example

- P=17, q=23, n=391, e=5, m=89, b=48

Undeniable Digital Signatures- inventions of Chaum and van Antwerpen.

- has three components: a signing algorithm, a verification protocol, and a disavowal protocol.
- signing algorithm allows Alice to sign a message.
- verification protocol uses the challenge-response mechanism to involve Alice for verifying the signature. This prevents the duplication and distribution of the signed message without Alice’s approval.
- disavowal protocol helps Alice deny a forged signature. To prove that the signature is a forgery, Alice needs to take part in the disavowal protocol.

Chaum and van Antwerpen signatures

Initialization – Based on discrete logarithm problem

- Bob selects a large prime modulus p such that p-1=2q where q is prime
- Takes an element which generates the cyclic group G of order q
- Chooses at random a secret k (0<k<q) and computes the public key g^k (mod p)
- Public parameters are (g^k , g, p)

Signing: for m Є G, Bob computes s= m^k (mod p)

Verification: Bob and Alice interact thru the following steps

- Challenge: Alice selects two random integers a, b Є Z_q^* and sends the challenge

c= s^a $(g^k)^b$ (mod p) to Bob

Response: Bob computes k^{-1} and sends

r= $c^{k^{-1}}$ = $m^a g^b$ (mod p) to Alice

Test: Alice checks whether r= $m^a g^b$ (mod p)

If test fails, Alice runs disavowal protocol. Otherwise signature is accepted

Disavowal Protocol

- Executed only when verification fails
- A-> B: Alice selects two a1, b1 Є Z_q^* and sends the challenge

c1= $s^{a1} (g^k)^{b1}$ (mod p)

B-> A: B replies r1= $c1^{k^{-1}}$

A checks whether r1!= $m^{a1} g^{b1}$ (mod p) . If yes, same process repeated

- A-> B: Alice selects two a2, b2 Є Z_q^* and sends the challenge

c2= $s^{a2} (g^k)^{b2}$ (mod p)

B-> A: B replies r2= $c2^{k^{-1}}$

A checks whether r2!= $m^{a2} g^{b2}$ (mod p) . If yes, concludes that the signature is a forgery if $(r1\, g^{-b1})^{a2} = (r1\, g^{-b2})^{a1}$(mod p)

Otherwise Bob cheats by giving inconsistent responses

Example

Suppose we take *p* = 467,q =233.Since 2 is a primitive element, 2^2 = 4 is a generator of *G*, the quadratic residues modulo 467.

So we can take g = 4. Suppose *k* = 101; then g^k (mod p)=449

Bob will sign the message *x* = 119 with the signature

s= m^k (mod p)= 119^{101} (mod 467)= 129

Now, suppose Alice wants to verify the signature *y*. Suppose she chooses the random values a= 38, *b*= 397.

She will compute *c* = 13, whereupon Bob will respond with *r* = 9.

Alice checks the response by verifying that $119^{38}4^{397} \equiv 9 \pmod{467}$

Hence, Hence, Alice accepts the signature as valid.

- suppose *p* = 467, g = 4, a= 101 and b = 449. Suppose the message *m* = 286 is signed with the (bogus) signature *s* = 83, and Bob wants to convince Alice that the signature is invalid
- Suppose Alice begins by choosing the random values a_1= 45, b2= 237. Alice computes *c* = 305 and Bob responds with *r* = 109.
- Then Alice computes $286^{45} 4^{237} \bmod 467 = 149$
- Since 149 not equal to 109, Alice proceeds
- Now suppose Alice chooses the random values a2= 125, b2= 9. Alice computes *c* = 270 and Bob responds with *r* = 68. Alice computes

$$286^{125} 4^{9} \bmod 467 = 25$$

- Since 25 not equal to 68, Alice proceeds and performs the consistency check. This check succeeds, since

$$(109 \times 4^{-237})^{125} \equiv 188 \pmod{467}$$

and

$$(68 \times 4^{-9})^{45} \equiv 188 \pmod{467}$$

Hence, Alice is convinced that the signature is invalid

Example RSA Blind signature

- p=29, q=31, e=11, m=19, b=23

Undeniable Digital Signatures- Chaum and van Antwerpen Example2

- P=983, q=491, e=7, $g=e^{(p-1)/2}$
- k=375
- Calculate public parameters
- M=413
- Compute s
- Verify the signature with a=119, b=227

References

- Behrouz A. Forouzan and Debdeep Mukhopadhyay – “Cryptography and Network Security”, McGraw Hill, 2nd Edition, 2008.
- William Stallings,“Cryptography And Network Security Principles And Practice”, Fifth Edition, Pearson Education, 2013