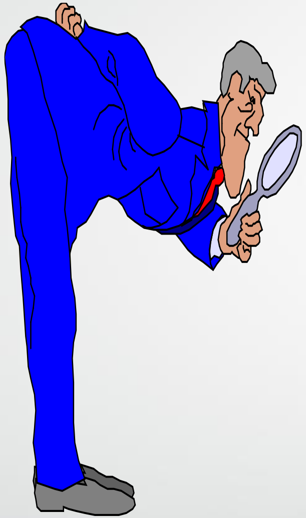
Digital Signatures



Digital Signatures

What is a Digital Signature?

- A **Digital Signature** is a type of asymmetric cryptography used to simulate the security properties of a handwritten signature on paper.
- Sometimes also used: **Electronic Signature** (here synonymic)

Digital Signatures

- Why is it important for E-Government?
  - Handwritten signature often required in public law
  - Digital signature can replace it
  - More possibilities of electronic services:
    - Cost savings
    - Saving Time

Law

- Germany: “Signaturgesetz” in 1997
  - Precondition for safe and legally binding electronic signatures
  - Regulates specifications for using digital signatures
- Europe: EU Signature Directive
  - Unification of different signature laws in the EU (especially different security levels)
  - Basis for changes of the German law in 2001, 2005 and 2007
  - Changes made the law conform to the European directive

Digital Signature

- Digital signature can be used in all electronic communications:
    - Web, e-mail, e-commerce
- It is an electronic stamp or seal that append to the document.
- Ensure the document being unchanged during transmission.

Digital Signatures (DS)

- Message authentication 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
- DS includes message authentication function with additional capabilities

Digital Signature Properties

- 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
- be computationally infeasible to forge
  - with new message for existing digital signature
  - with fraudulent digital signature for given message
- be practical to save digital signature in storage

Digital Signature

So, Digital Signatures can provide

- **Authentication.** As a drawing recipient, I can be sure that the person or corporation that sent me the drawing is who they claim to be.
- **Data Integrity.** I can be sure that the drawing has not changed in any way, either intentionally or accidentally, since it was signed.
- **Non-Repudiation.** The signed drawing cannot be repudiated: The signer cannot later disown it, claiming the signature was forged.

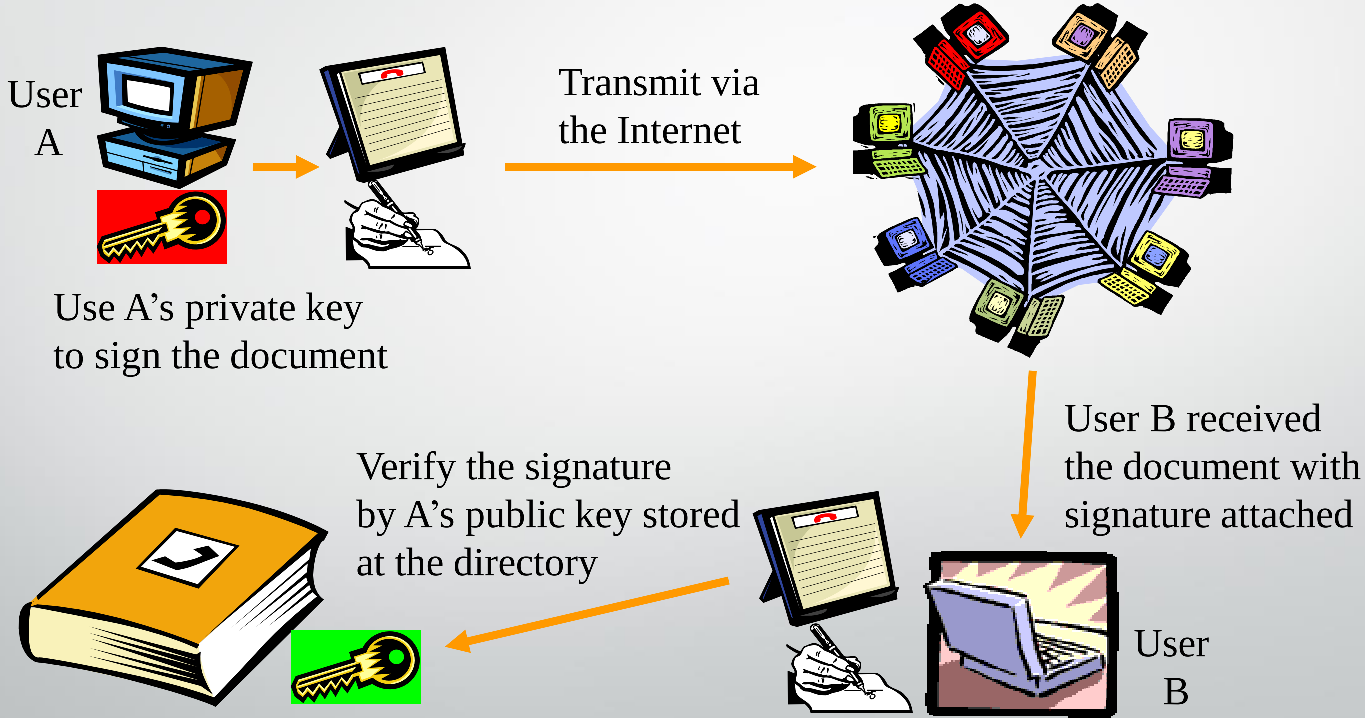
Digital Signature

Parameter	Paper	Electronic
Authenticity	May be forged	Can not be copied
Integrity	Signature independent of the Document	Signature depends on the contents of the Document
Non-repudiation	a. Handwriting expert needed b. Error prone	a. Any computer user b. Error free

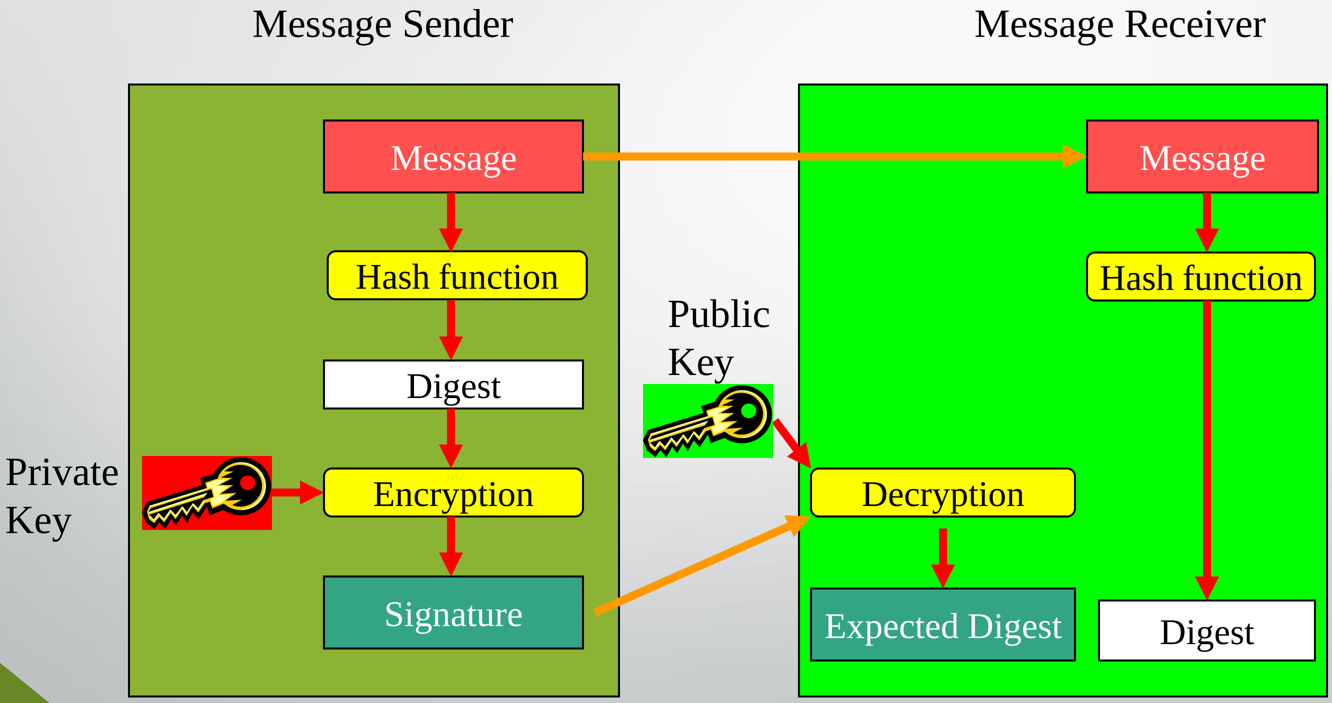
Digital Signatures

- Digital signatures employ a type of asymmetric (public-key) cryptography.
- For messages sent through an insecure channel, a properly implemented digital signature gives the receiver reason to believe the message was sent by the claimed sender.
- Digital signatures are equivalent to traditional handwritten signatures in many respects; properly implemented digital signatures are more difficult to forge than the handwritten type.
- Digital signature schemes in the sense used here are cryptographically based, and must be implemented properly to be effective.

How Digital Signature works?



Digital Signature
Generation and Verification

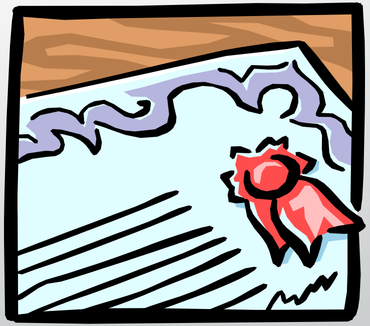


Digital Signatures

- It is possible to use the entire message, encrypted with the private key, as the digital signature
  - But, this is computationally expensive
  - And, anyone can then decrypt the original message
- Alternatively, a *digest* can be used
  - Should be short
  - Prevent decryption of the original message
  - Prevent modification of original message
  - Difficult to fake signature for
- If message authentication (integrity) is needed, we may use the hash code of the message
- If only source authentication is needed, a different message can be used (certificate)

Digital Certificates

- Digital Certificate is a data with digital signature from one trusted Certification Authority (CA).
- This data contains:
  - Who owns this certificate
  - Who signed this certificate
  - The expired date
  - User name & email address



Digital Signature Scheme

- Alice generates secret key k_1 and public k_2
- Alice publishes k_2
- Alice signs plaintext P: (P, S = $E_{k1}(P)$)
- Alice sends P, S to Bob
- Bob verifies that $D_{k2}(S) = P$
  (since only Alice knows k_1)

Combining Public Key Encryption and Authentication (case 1)

- Alice generates public key k_{1a} and private key k_{2a}, n_a;
- Bob generates public key k_{1b} and private key k_{2b} , n_b;
- Alice encrypts with Bob's public key k_{1b} :
  $C = E_{k_{1b}}(P)$;
- Alice signs with her secret key k_{2a} :
  $S = E_{k_{2a}}(C)$;
- Alice sends S, C to Bob;
- Bob verifies $D_{k_{1a}}(S) = Cb$; Cb = = C??
- Bob decrypts: $P = D_{k_{2b}}(C)$.

Combining Public Key Encryption and Authentication (case 2)

- Alice generates public key k_{1a} and private key $k_{2a,}$ n_a;
- Bob generates public key k_{1b} and private key k_{2b} , n_b;
- Alice encrypts with Bob's public key k_{1b} :
  $C = E_{k_{1b}}(P)$;
- Alice signs with her secret key k_{2a} :
  $S = E_{k_{2a}}(H(P))$;
- Alice sends S, C to Bob;
- Bob decrypts: $Pb = D_{k_{2b}}(C)$
- Bob verifies $D_{k_{1a}}(S) = H(P)$; $H(P) == H(Pb)$??

Combining Public Key Encryption and Authentication – Hybrid algorithm(case 3)

- Alice generates public key k_{1a} and private key k_{2a}, n_a;
- Bob generates public key k_{1b} and private key k_{2b} , n_b;
- Alice encrypts the message P with symmetric encryption $C=E_K(P)$; K – private key for encr/decr
- Alice encrypts the key K with Bob's public key k_{1b} : $CK = E_{k_{1b}}(K)$;
- Alice signs with her secret key k_{2a} : $S = E_{k_{2a}}(C)$;
- Alice sends S, C, CK to Bob;
- Bob verifies $D_{k_{1a}}(S) = Cb$; Cb = = C??
- Bob decrypts the key CK: $K = D_{k_{2b}}(CK)$
- Bob decrypts the message C: $P = D_K(C)$

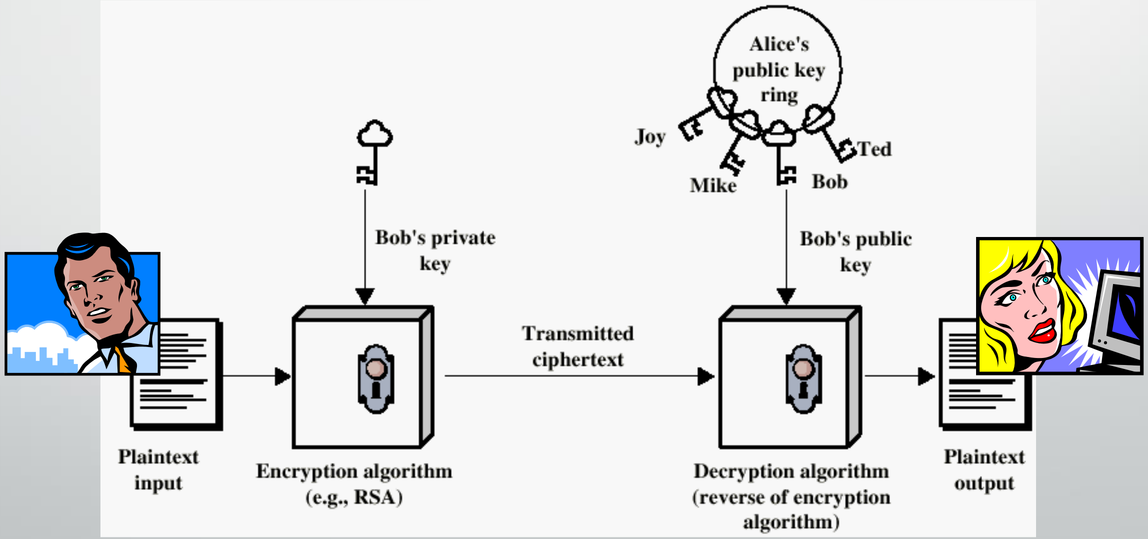
Combining Public Key Encryption and Authentication

The scheme suggests indirectly that Alice and Bob use the same base of the RSA.

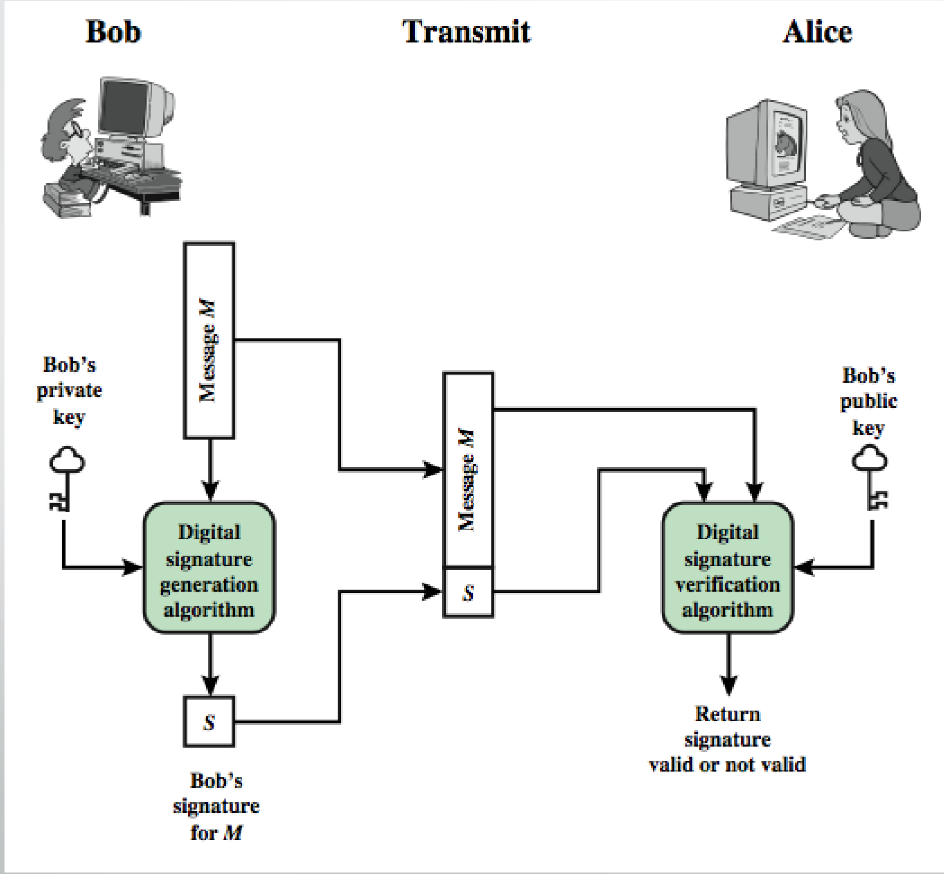
What should be done if it is not true?

Public-Key Digital Signature

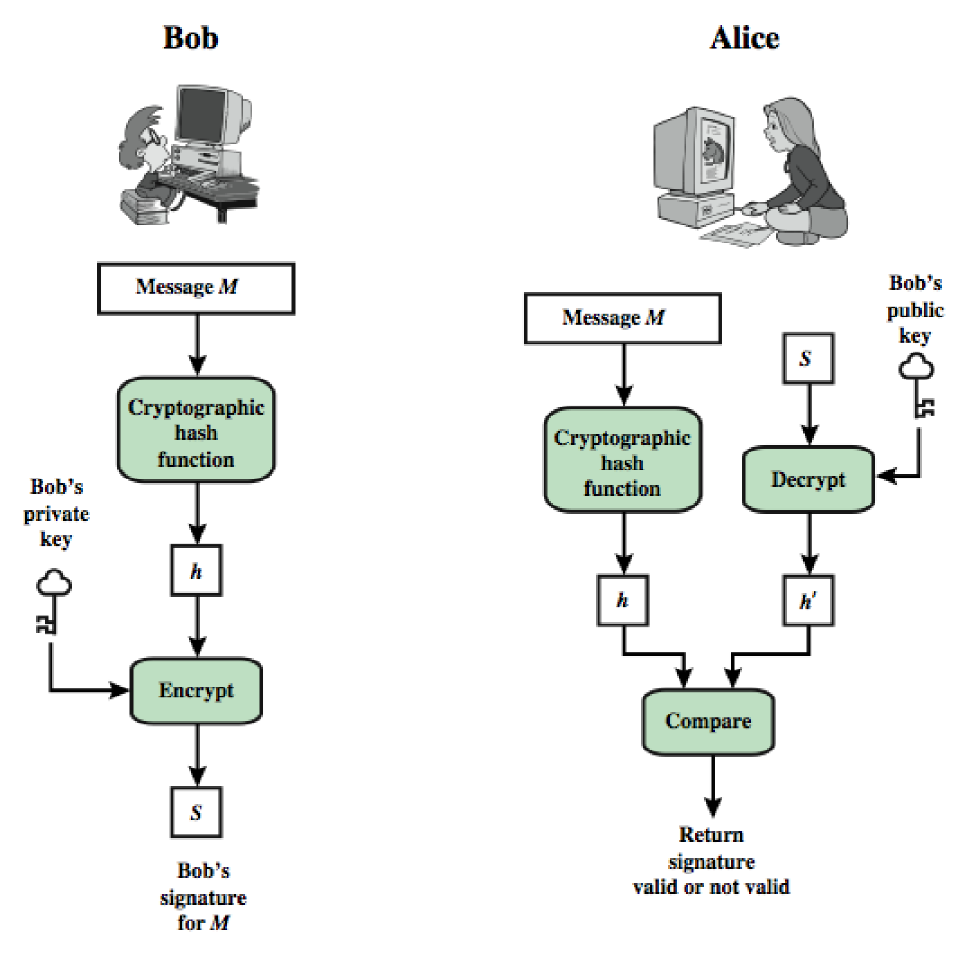
- Same as authentication
  - The sender encrypts a message with his own private key
  - The receiver, by decrypting, verifies key possession



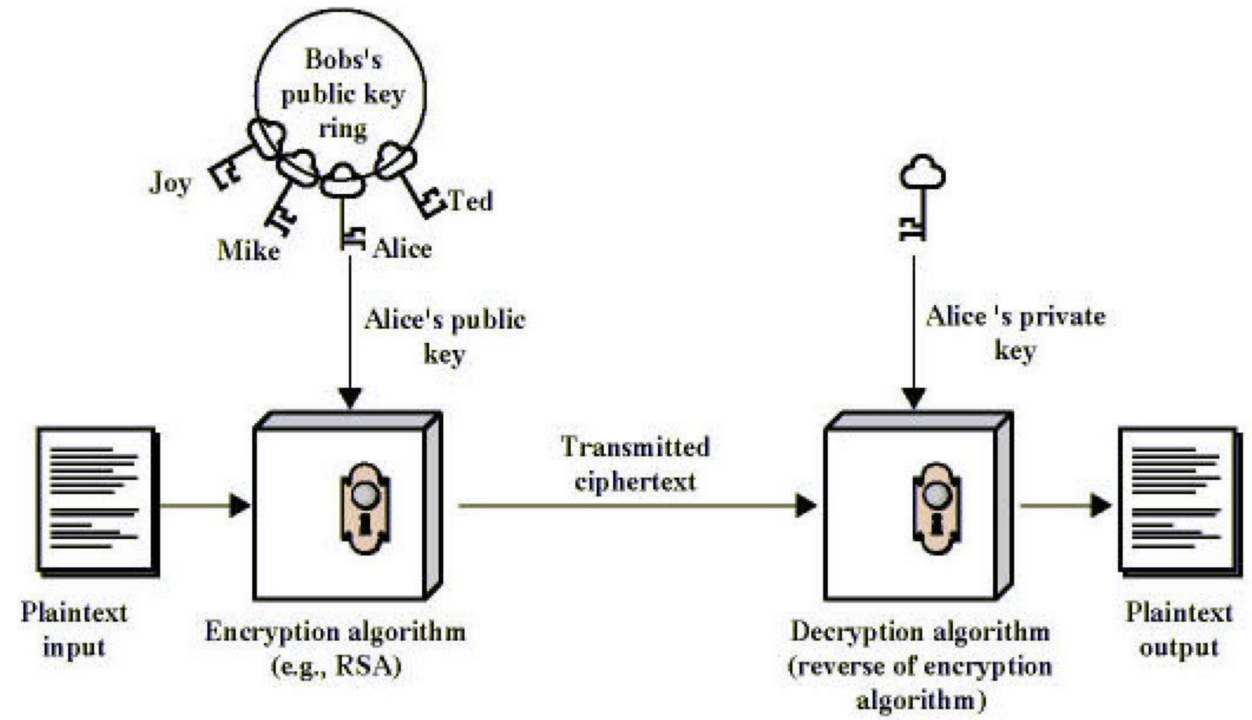
Digital Signature Model



Digital Signature Model

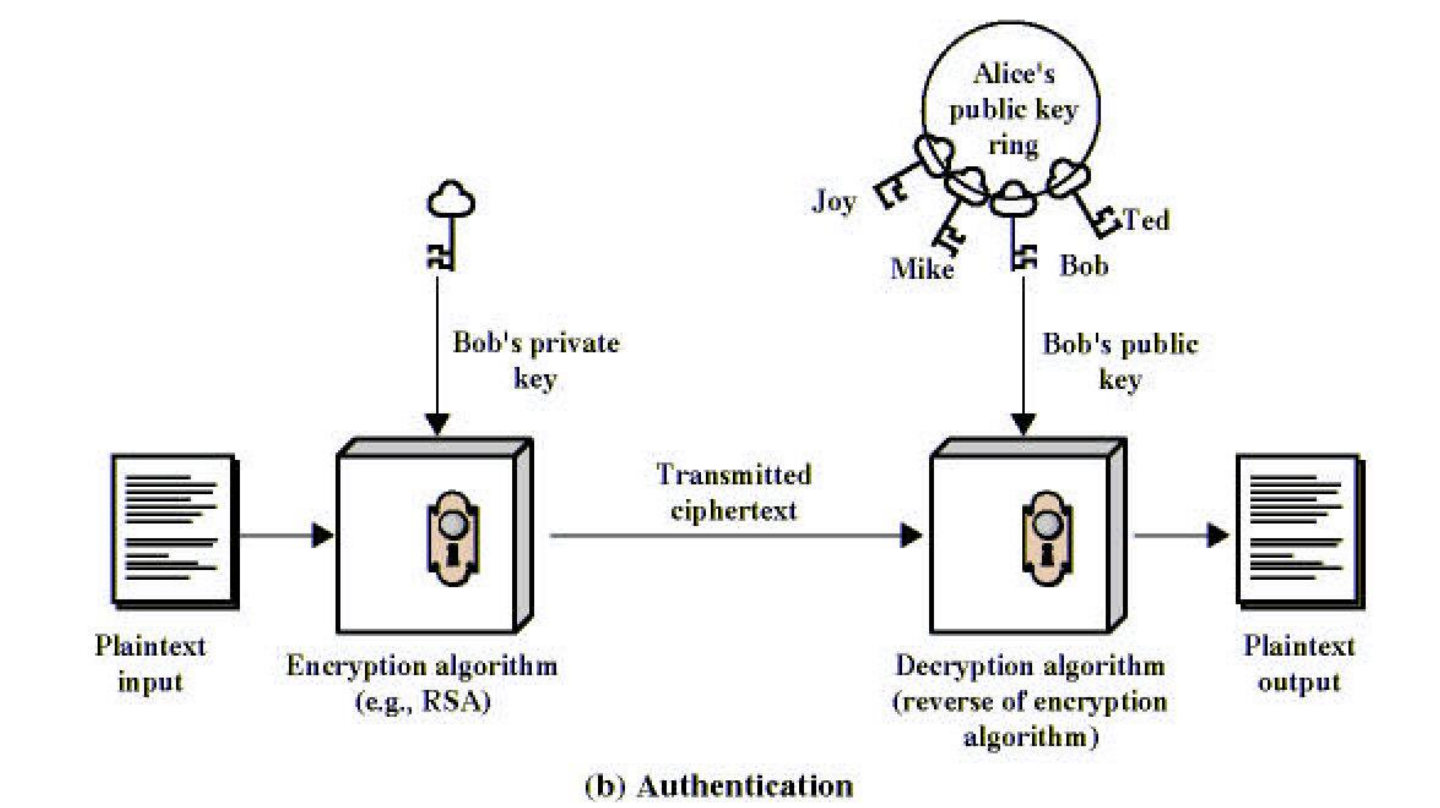


Public-Key Cryptography



(a) Encryption

Public-Key Cryptography



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

Digital Signature Algorithm (DSA)

- Proposed in 1991 by NIST as a standard (DSS)
  - Based on difficulty of computing discrete logarithms (like Diffie-Hellman and El Gamal)

- Encountered resistance because RSA was already de-facto standard, and already drew significant investment
  - DSA cannot be used for encryption or key distribution
  - RSA is advantageous in most applications (exc. smart cards)
    - RSA is 10x faster in signature
    - DSA is faster in verification
  - Concerns about NSA backdoor (table can be built for some primes)

- Key size was increased from 512 to 2048 and 3072 bits
  - In DSA, the key size needs to be 4 times the security level
- DSA has an Elliptic Curve version
  - Faster to compute, and requires half the bits

Description of DSA

- **Parameters**
  - p is a prime number with up to 1024 bits — public key
  - q is a 160-bit factor of (p-1), and itself prime — public key
  - $g=h^{(p-1)/q} \bmod p$ (h is random) — public key
  - x is the private key and is smaller than q -- **private key**
  - $y=g^x \bmod p$ is part of the public key — public key
- **Signature**
  - Given a message M, generate a random $k<q$ -- **keep secret**
  - Signature is a pair (r,s)
    - send $r=(g^k \bmod p) \bmod q$ — signature
    - send $s=k^{-1}(H(M)+x*r) \bmod q$ — signature
    - If r=0 or s=0, choose a new k
- **Verification**
  - Compute $w=s^{-1} \bmod q$
  - Compute $u1=H(M)*w \bmod q$; $u2=r*w \bmod q$
  - Compute $v=(g^{u1}*y^{u2} \bmod p) \bmod q$
  - If v=r then the signature is verified — verification

Key Generation in DSA

- Generate q as a SHA on an arbitrary 160-bit string
  - If not prime, try another string
  - Use Rabin method for primality testing

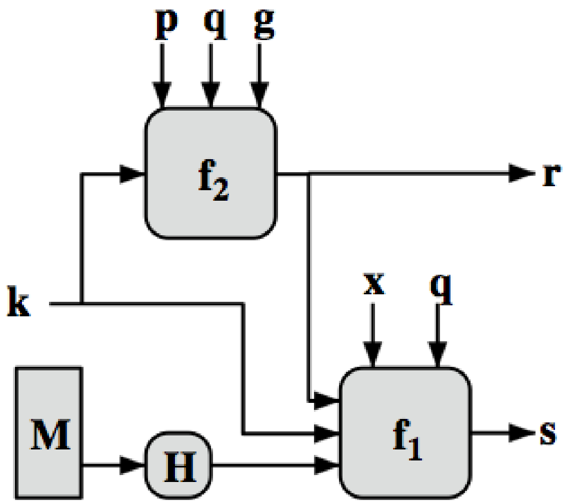
- To get (p-1)
  - Concatenate additional 160 bit numbers until you get to the right size (e.g., 1024)
  - Subtract the remainder after division by 2q
    - q is a factor from construction
    - Since p-1 is even, then 2 is also a factor

- If p is not prime, repeat the process

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

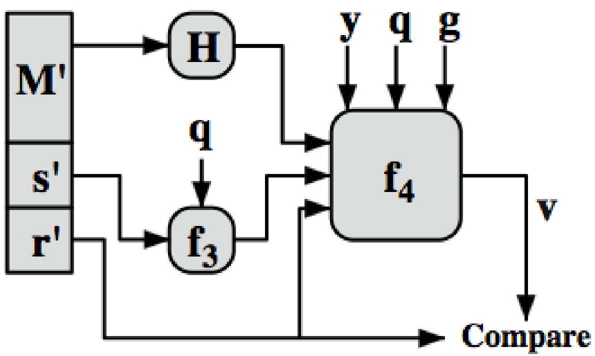
Digital Signature Standard (DSS)



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

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

(a) Signing



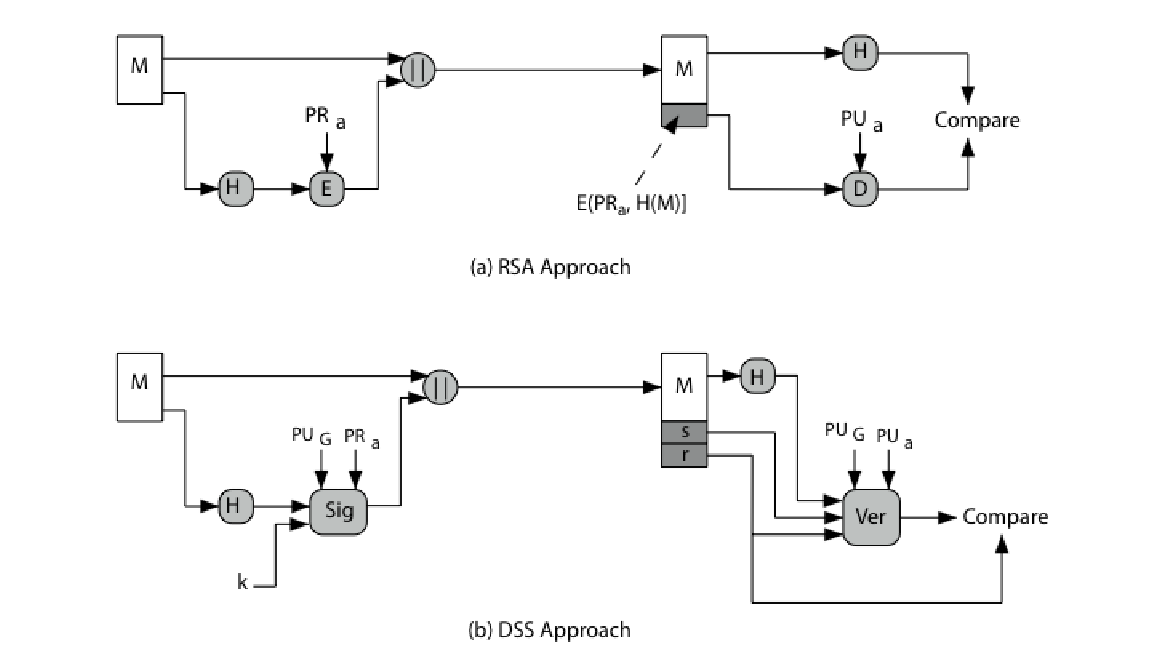
$w = f_3(s', q) = (s')^{-1} \bmod q$

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

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

(b) Verifying

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:

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

  $\mathbf{s = [k^{-1}(H(M) + xr)] \bmod q}$
- sends signature (r,s) with message M

DSA Signature Verification

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

  $w = s^{-1} (\text{mod } q)$

  $u1 = (H(M) * w)(\text{mod } q)$

  $u2 = (r * w)(\text{mod } q)$

  $v = (g^{u1}\, y^{u2} (\text{mod } p))\ (\text{mod } q)$

- if **v = r** then signature is verified
- see book web sitehe text for details!

ElGamal Digital Signatures

- signature variant of ElGamal, related to D-H
  - so uses exponentiation in a finite (Galois)
  - 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 (eg. A) generates their key
  - chooses a secret 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 $m = H(M)$, 0 <= m <= (q-1)
  - chose random integer K with 1 <= K <= (q-1) and gcd(K,q-1)=1
  - compute temporary key: $\mathbf{S_1 = a^k \bmod q}$
  - compute K^{-1} the inverse of K mod (q-1)
  - compute the value: $\mathbf{S_2 = K^{-1}(m - x_A S_1) \bmod (q-1)}$
  - signature is: $\mathbf{DS=(S_1, S_2)}$
- 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 a private key $x_A=16$
  - computes the public $y_A=10^{16} \bmod 19 = 4$
- Alice signs message with hash $m = 14$ as $DS = (3,4)$:
  - choosing random $K=5$ which has $\gcd(18,5)=1$
  - computing $S_1 = 10^5 \bmod 19 = 3$
  - finding $K^{-1} \bmod (q-1) = 5^{-1} \bmod 18 = 11$
  - computing $S_2 = 11*(14-16*3) \bmod 18 = 4$
- any user B can verify the signature by computing
  - $V_1 = a^m \bmod q = 10^{14} \bmod 19 = 16$
  - $V_2 = y_A^{S1}\, S_1^{S2} \bmod q = 4^3*3^4 = 5184 = 16 \bmod 19$
  - since $V_1 = V_2$ signature is valid