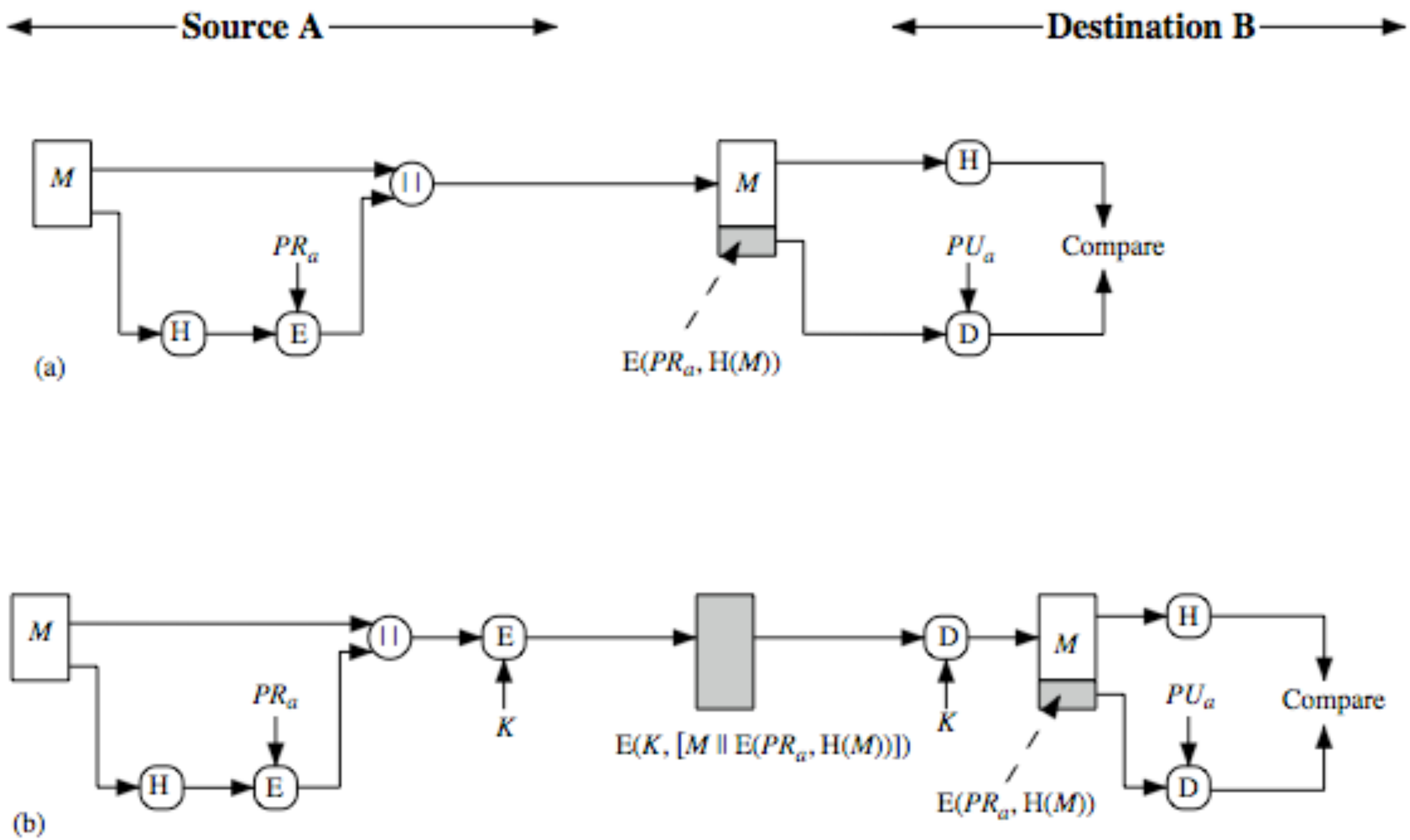
Information Security CENG418

week-5

Cryptography: Digital Signatures

Hash Functions & Digital Signatures



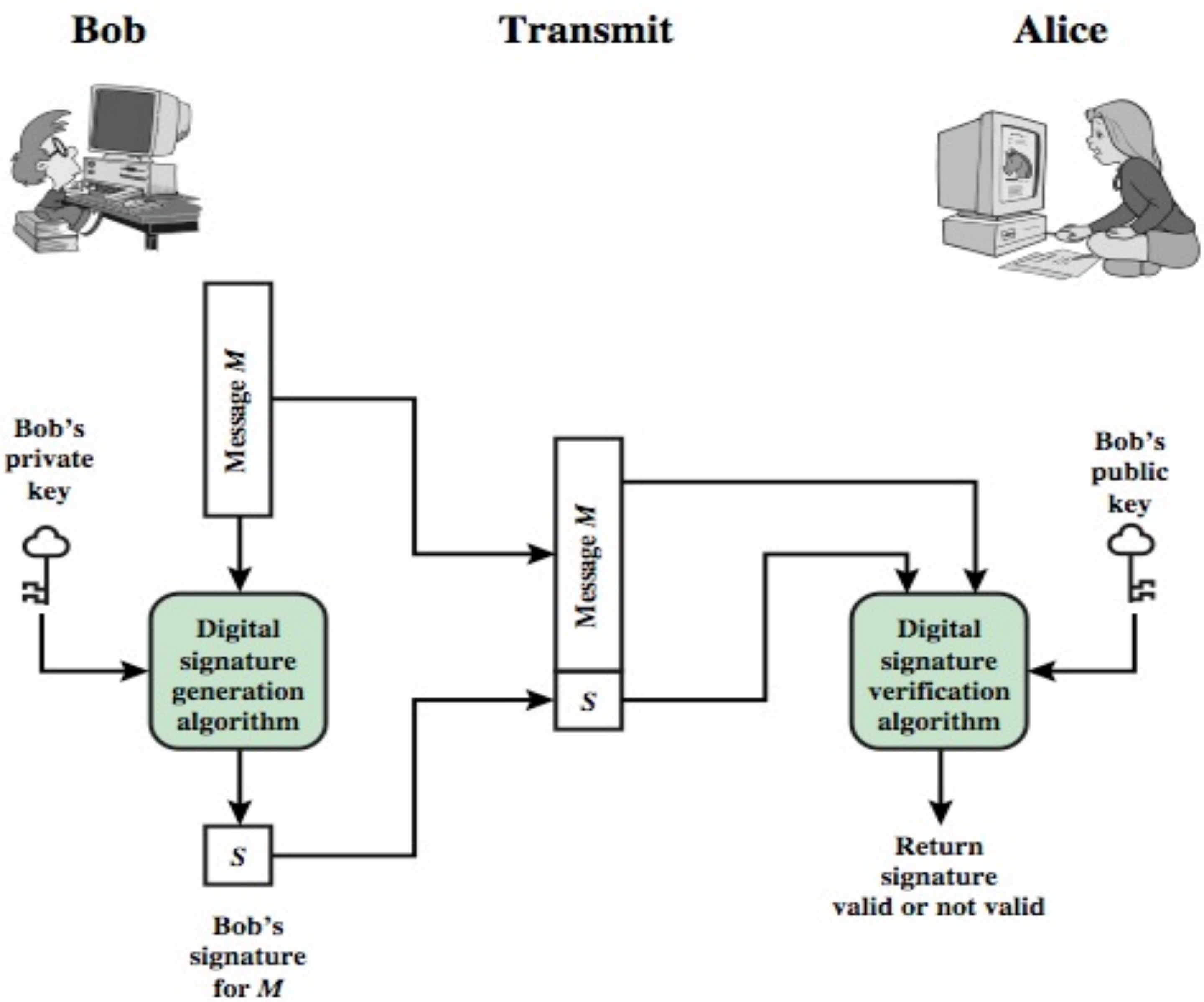
Digital Signatures – message authentication

- Message authentication protects the communication of two parties from any third party.
- **It does not protect the two parties against either fraudulently creating or denying the creation of a message**.
- A **digital signature is analogous to a handwritten signature** and provides a set of security capabilities that would be difficult to implement in any other way. It must have the following properties:
  - It must verify the author and the date and time of the signature
  - It must authenticate the contents at the time of the signature
  - It must be verifiable by third parties to resolve disputes
  - thus, the digital signature function includes the authentication function.

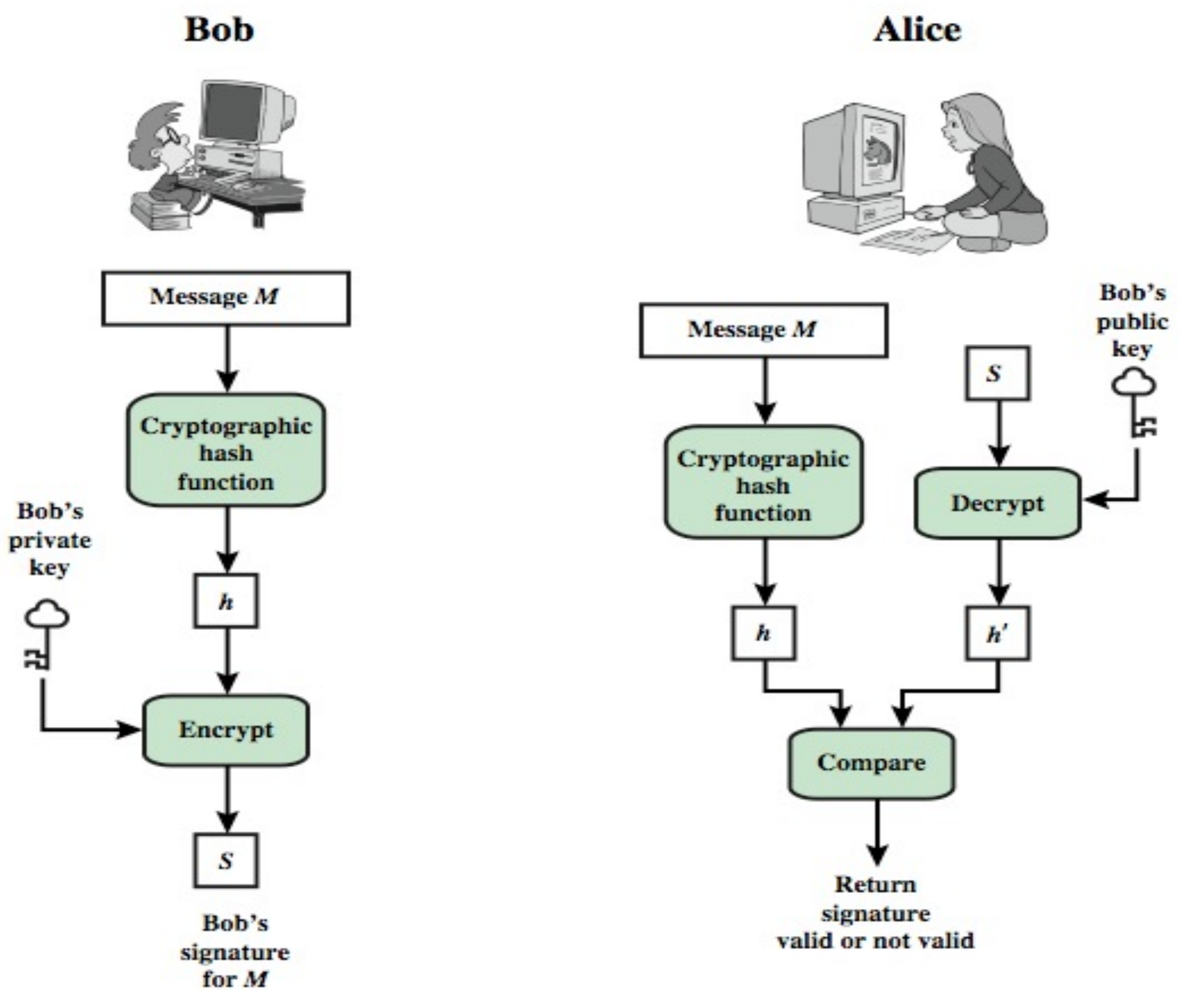
Digital Signatures

- 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 Signature Model



Digital Signature Model



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

Direct Digital Signatures

- Involve only sender & receiver
- Assumed that the receiver has the sender's public key to verify the signature of the received message
- The sender's signing makes the digital signature on either the entire message or hash with the sender's private key
- Before sending; the message, it can be encrypted using the receiver's public key
- Important that sign first then encrypt message & signature
- Security of signature (authentication of the message) depends on the secrecy of the sender's private key

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
- uses 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: $S_1 = a^k \bmod q$
  - compute K^{-1} the inverse of $K \bmod (q-1)$
  - compute the value: $S_2 = K^{-1}(m-x_A S_1) \bmod (q-1)$
  - 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 $y_A=10^{16} \bmod 19 = 4$
- Alice signs message with hash $m=14$ as $(3,4)$:
  - choosing random $K=5$ which has $\gcd(18,5)=1$
  - computing $S_1 = 10^5 \bmod 19 = 3$
  - 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
  - $V_1 = 10^{14} \bmod 19 = 16$
  - $V_2 = 4^3.3^4 = 5184 = 16 \bmod 19$
  - since $16 = 16$ signature is valid

ElGamal Signature

$\mathbf{y_A = a^{x_A} \bmod q}$ and X_A **is private key** of signer.
$\mathbf{y_A}$ **is public key of signer.**

$m = H(M)$
$1 <= k <= (q-1)$ and $\gcd(k,q-1)=1$
$\mathbf{S_1 = a^k \bmod q}$
k^{-1} the inverse of $k \bmod (q-1)$
$\mathbf{S_2 = k^{-1}(m - x_A S_1) \bmod (q-1)}$
signature is: (S_1, S_2)

Verification of Signature

$\mathbf{V_1 = a^m \bmod q}$
$\mathbf{V_2 = y_A^{S1}\ S_1^{S2} \bmod q}$
signature is valid if $\mathbf{V_1 = V_2}$

$$V_2 = y_A^{S1}\ S_1^{S2} \bmod q$$

$$\boldsymbol{V_2 = a^{X_A . a^k} . a^{k.k^{-1}.(m - X_A . a^k)} = a^m \bmod q}$$

Schnorr Digital Signatures

- also uses exponentiation in a finite (Galois)
  - security based on discrete logarithms, as in D-H
- The main work for signature generation does not depend on the message and can be done during the idle time of the processor. Minimizes message dependent computation,
  - The message dependent part of the signature generation requires multiplying a 2*n-bit* integer with an *n-bit* integer
- main work can be done in idle time
- have using a prime modulus *p*
  - *p–1* has a prime factor *q* of appropriate size
  - typically *p* 1024-bit and *q* 160-bit numbers

Schnorr Key Setup

- choose suitable primes *p* , *q;* , such that *q* is a prime factor of p – 1
- choose ***a*** such that $\mathbf{a^q = 1 \bmod p}$
- (**a,p,q**) are **global parameters for all**
- each user (eg. A) generates a key
  - chooses **a** secret key (number): $\mathbf{0 < s_A < q}$
  - compute their **public key:** $\mathbf{v_A = a^{-s_A} \bmod q}$

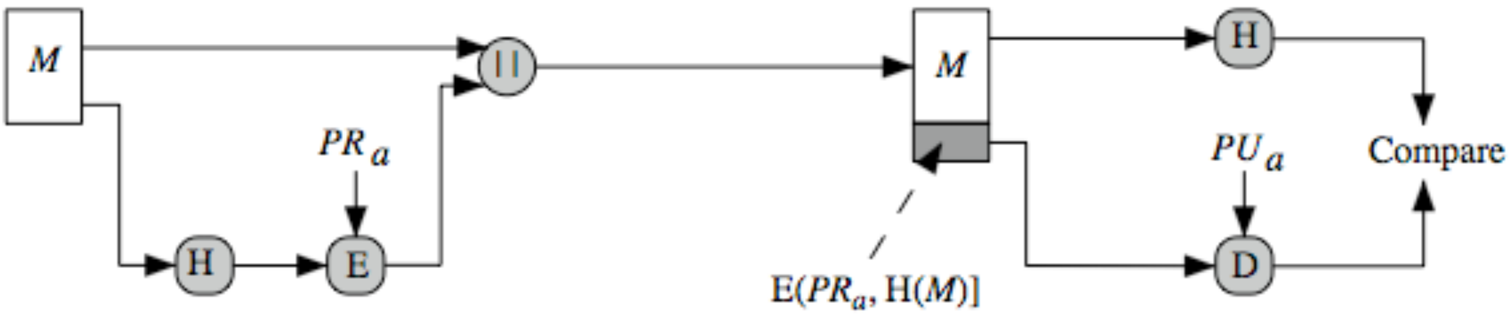
Schnorr Signature

- A user with public key v and private key s generates a signature as follows:
- 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) mod q
  - signature is pair (**e, y**)
- any other user can verify the signature as follows:
  - computing: **x' =** $\mathbf{a^y v^e}$ **mod p**
  - verifying that: **e = H(M || x')**

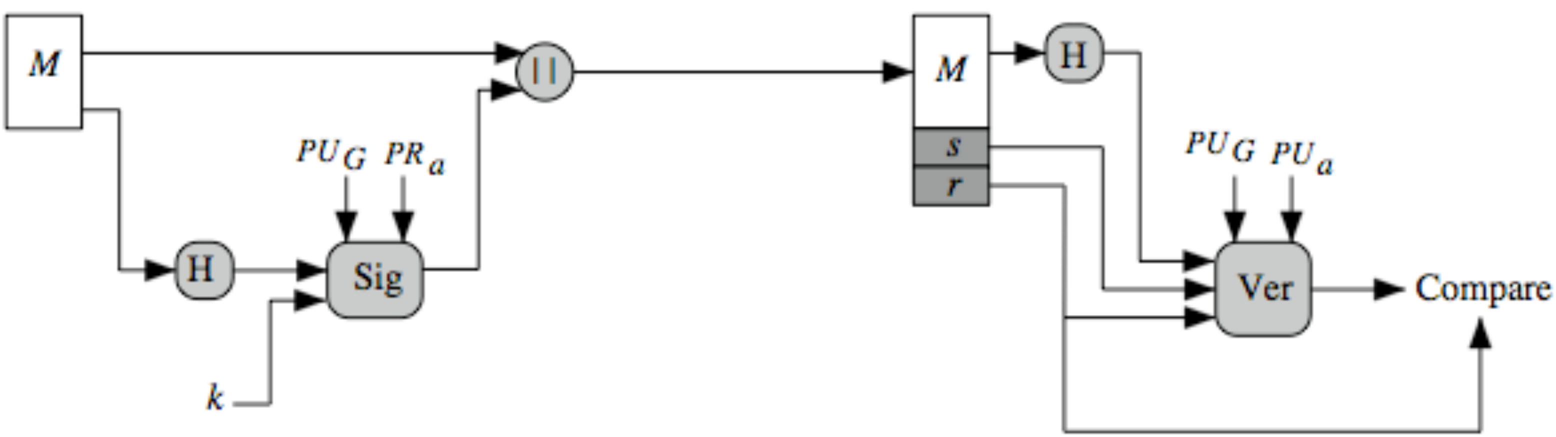
Digital Signature Standard (DSS)

- 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
- DSA is digital signature only unlike RSA
- is a public-key technique

DSS vs RSA Signatures



(a) RSA Approach



(b) DSS Approach

PU_G; a set of parameters known to a group of communicating principals. Global public key.

Digital Signature Algorithm (DSA)

The DSA is based on the difficulty of computing discrete logarithms and is based on schemes originally presented by ElGamal and Schnorr.

- 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

The DSA signature scheme has advantages, being both smaller (320 vs 1024bit) and faster (much of the computation is done modulo a 160 bit number), over RSA. Unlike RSA, it cannot be used for encryption or key exchange. Nevertheless, it is a public-key technique.

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, $k<q$
  - k must be random, be destroyed after use, and never be reused
  - the public key components (p,q,g), the user's private key (x)
- 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**

Note that computing r only involves calculation mod p and does not depend on message, hence can be done in advance. Similarly with randomly choosing k's and computing their inverses.

DSA Signature Verification

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

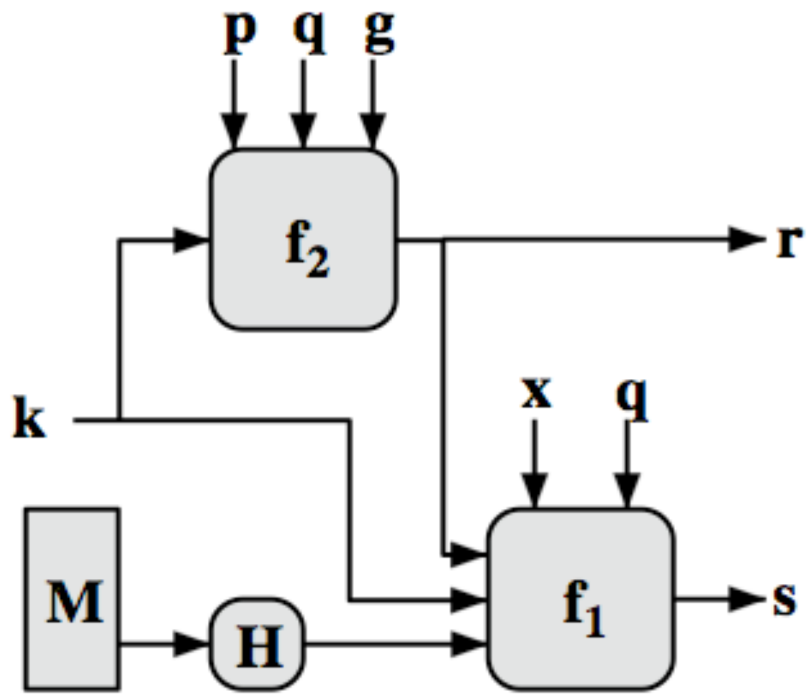
  $\mathbf{w = s^{-1} \bmod q}$

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

  $\mathbf{u2 = (rw) \bmod q}$

  $\mathbf{v = [(g^{u1}\ y^{u2}) \bmod p\ ] \bmod q}$
- **if `v=r` then signature is verified**

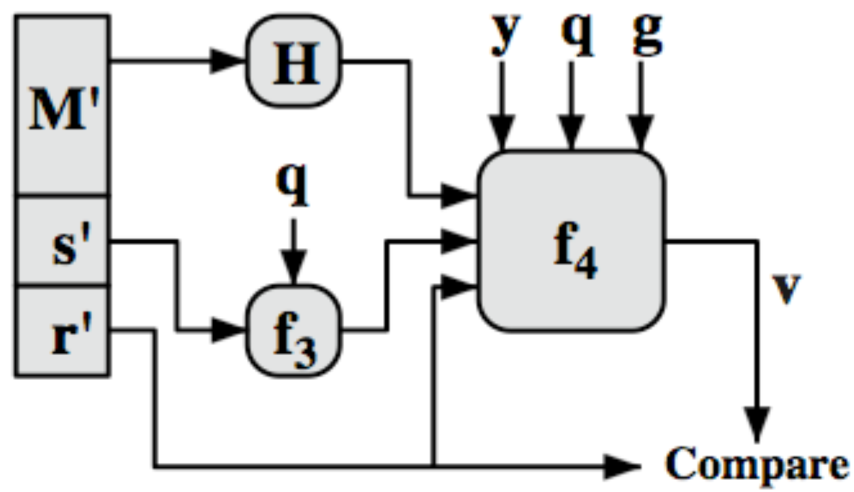
DSS Overview



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

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

(a) Signing

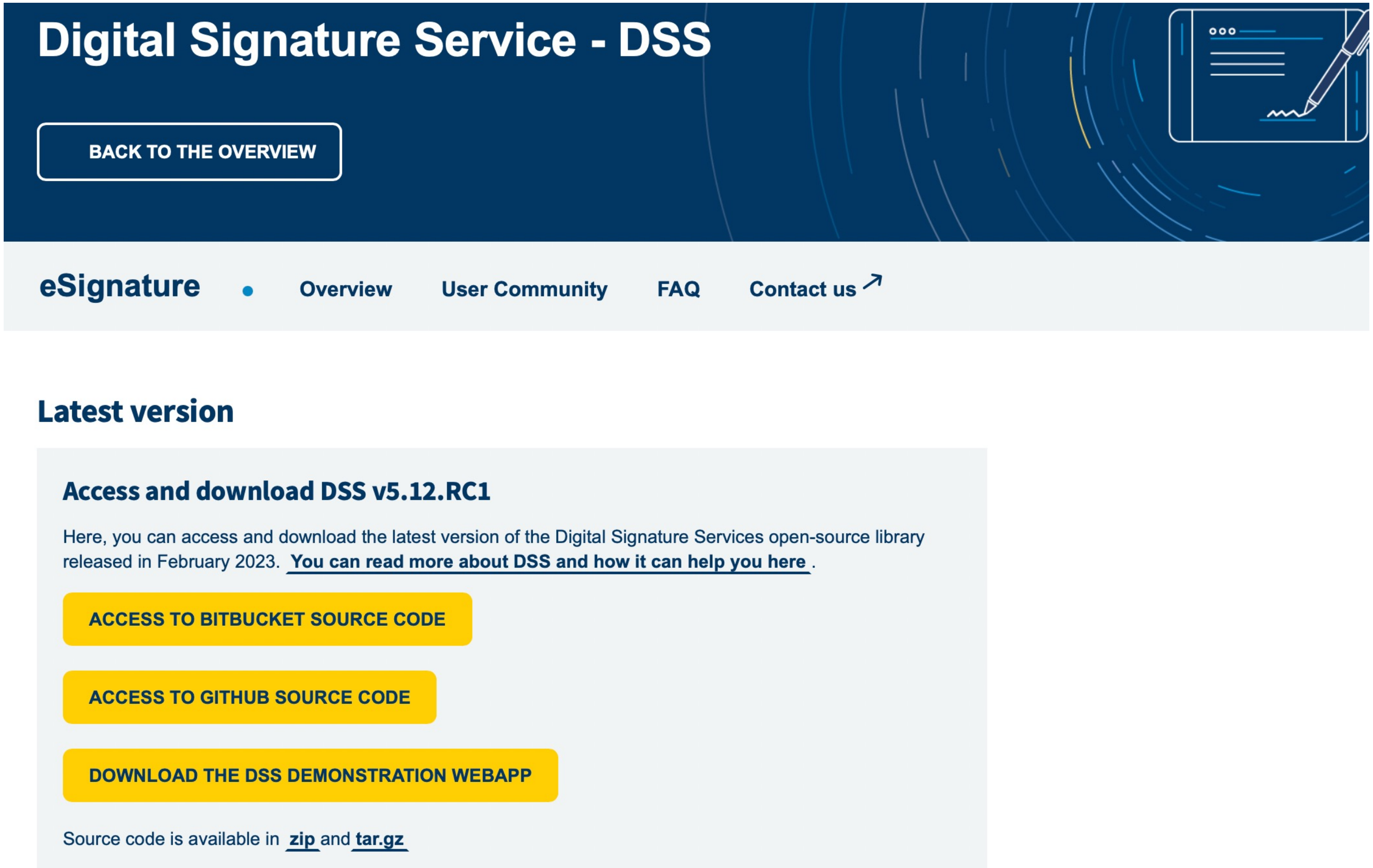


$$w = f_3(s', q) = (s')^{-1} \bmod q$$

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

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

(b) Verifying



https://ec.europa.eu/digital-building-blocks/wikis/display/DIGITAL/Digital+Signature+Service+-++DSS

Summary

- have discussed:
  - digital signatures
  - digital signature algorithm and standard

Next Lecture

- Key Distribution and Management

Sample questions

Which of the following is a characteristic of digital signatures?

A. They use symmetric key cryptography.

B. They require a physical signature on a document.

C. They use public key cryptography to provide authentication and integrity.

D. They are not secure for transmitting sensitive information.

What is correct about digital signatures?

A. A digital signature cannot be moved from one signed document to another because it is the hash of the original document encrypted with the private key of the signing party.

B. Digital signatures may be used in different documents of the same type.

C. A digital signature cannot be moved from one signed document to another because it is a plain hash of the document content.

D. Digital signatures are issued once for each user and can be used everywhere until they expire.

Which of the following is a characteristic of public key cryptography?

A. It uses the same key for encryption and decryption.

B. It uses two different keys for encryption and decryption.

C. It can only be used for encrypting small messages.

D. It is not suitable for use in electronic commerce.