MANCHESTER 1824

The University of Manchester

Topic 6: Digital Signature Algorithms

Understand two mostly used digital signature schemes

Source: Stalling's book, chapter 13

Overview

- Part 1
  - Digital Signature Overview
  - Digital Signature using RSA
- Part 2
  - DSS (Digital Signature Standard, also called DSA - digital signature algorithm)
  - RSA vs DSA
  - Conclusion

Digital Signature Overview

- A **digital signature** is a technique for establishing the origin of a particular message such that any future disputes with regard to what message was sent and who sent it could be resolved by any third party.

- According to the European Community Directive on digital signatures, a digital signature should be:
  - uniquely linked to the signatory
  - capable of identifying the signatory
  - created using means under the sole control of the signatory
  - linked to data to which it relates in such a way that subsequent changes in the data is detectable.

Digital Signature Overview

- A digital signature associates a mark unique to an individual with a body of text.
- Security requirements:
  - message-dependent, inc date/time
    - unreusable
    - ensures content integrity
  - signer-dependent
    - unforgeable
    - ensures origin authentication

Both integrity and origin authenticity are necessary to ensure **non-repudiation**, i.e. the signer can not falsely deny that he/she has generated the signature.

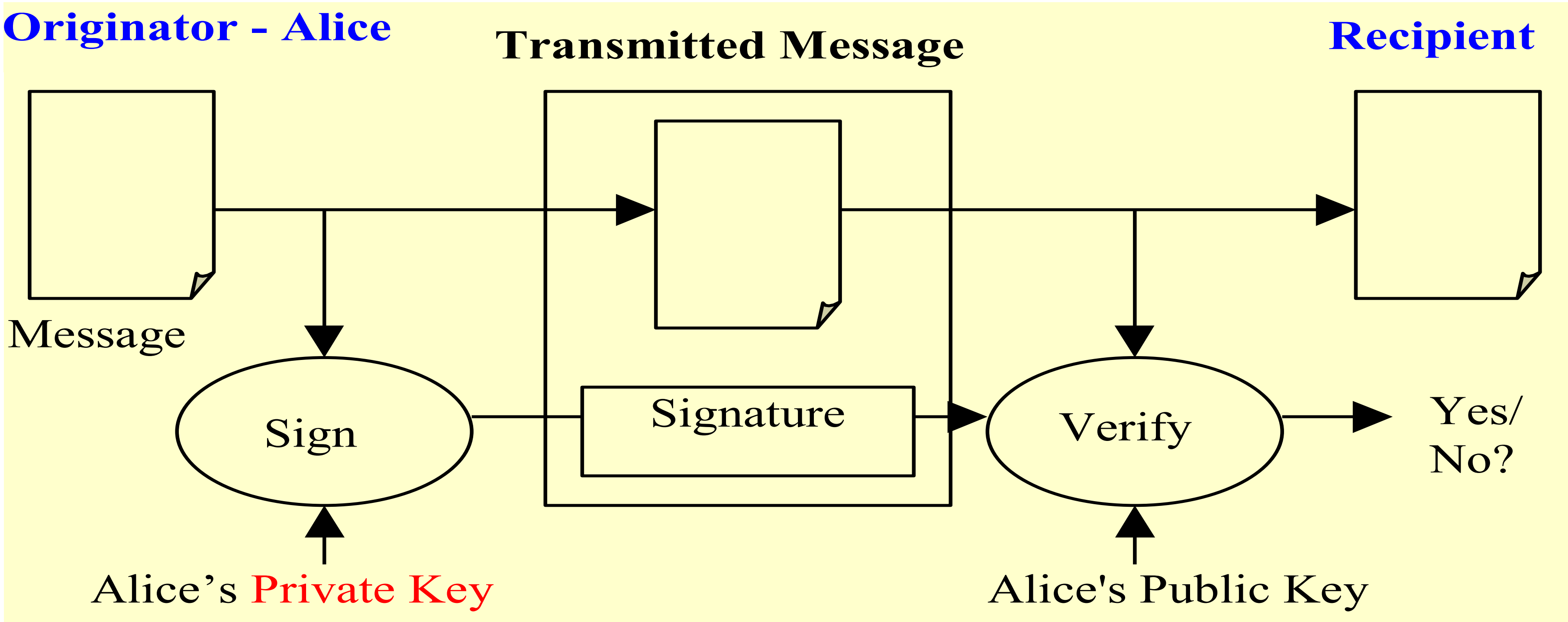
  - verifiable: others should be able to verify the validity of a signature.
  - anti-forgery: computationally infeasible to forge.

Digital Signature Overview

- Forgery types
  - Existential forgery: the creation (by an adversary) of any message/signature pair (*M, S*), where *S* was not produced by any legitimate signer.
  - Selective forgery: the creation (by an adversary) of a message/signature pair (*M, S*), where *S* has been *chosen* by the adversary prior to the attack.

Digital Signature Overview

- There are arbitrated digital signatures.
- BUT in most cases, a digital signature is generated using a public-key algorithm.
- PKC based digital signature model:



Digital Signature Overview

- The main idea is
  - only A can sign a message, since only A has access to the private key.
  - anyone can verify A's signature, since everyone has access to her public key.

- A digital signature scheme consists of:
  - A key generation algorithm
  - A signature (generation) algorithm
  - A signature verification algorithm

Digital Signature Overview – Improper way

What *Bob* knows/does	**What public sees**	**What Alice knows/does**
	Alice's public key $PU_a=\{e, n\}$	*A's public key* $PU_a=\{e, n\}$ *A's private key* $PR_a=\{d, n\}$
		Step 1: Signature generation $S=M^d \bmod n$
	$M'\|\|S$	**Step 2: send *M and S to Bob.***
Step 3: Signature verification $M=S^e \bmod n$ See if $M=M'$		

Digital Signature Overview - Improper way

- Performance concern: public-key cipher operations are time consuming and signing long messages is costly.
- Security concern: loopholes for signature forgery
  - *Example 1*
    - Let s be a random value, apply the public key (e, n) *to s:* $P= s^e \pmod n = m$
    - Then (m, s) *is a valid message-signature pair.*
  - *Example 2*
    - Let m be the message of which you want to forge a signature.
    - Choose two messages x *and* y such that $xy = m \pmod n$
    - If you could obtain the signatures of x *and* y, *then,* you can easily forge a signature of m by $S(m) = S(y)S(x) \pmod n$

Digital Signature Overview - Proper way

- Use "hash-and-sign" paradigm: signing the hash value of a message.

Digital Signature using RSA - Proper Way

What *Bob* knows/does	What public sees	What Alice knows/does
Alice's public key $PU_a=\{e, n\}$ Hash function H( )		$A's$ public key $PU_a=\{e, n\}$ $A's$ private key $PR_a=\{d, n\}$ Hash function H( )
		Step 1: Signature generation for *M* $h=H(M)$ $S=h^d \bmod n$
Step 3: Signature verification Compute $h'=H(M')$ *and* $h=S^e \bmod n$ See if $h=h'$	$M' \| S$	**Step 2: send *M and S to Bob.***

Sometimes, we write in this format: $S=E_{PRa}[H(M)]$ for signature generation, and $h= D_{PUa}[S]=D_{PUa}[E_{PRa}[H(M)]]$ for signature verification.

Part 2 Overview

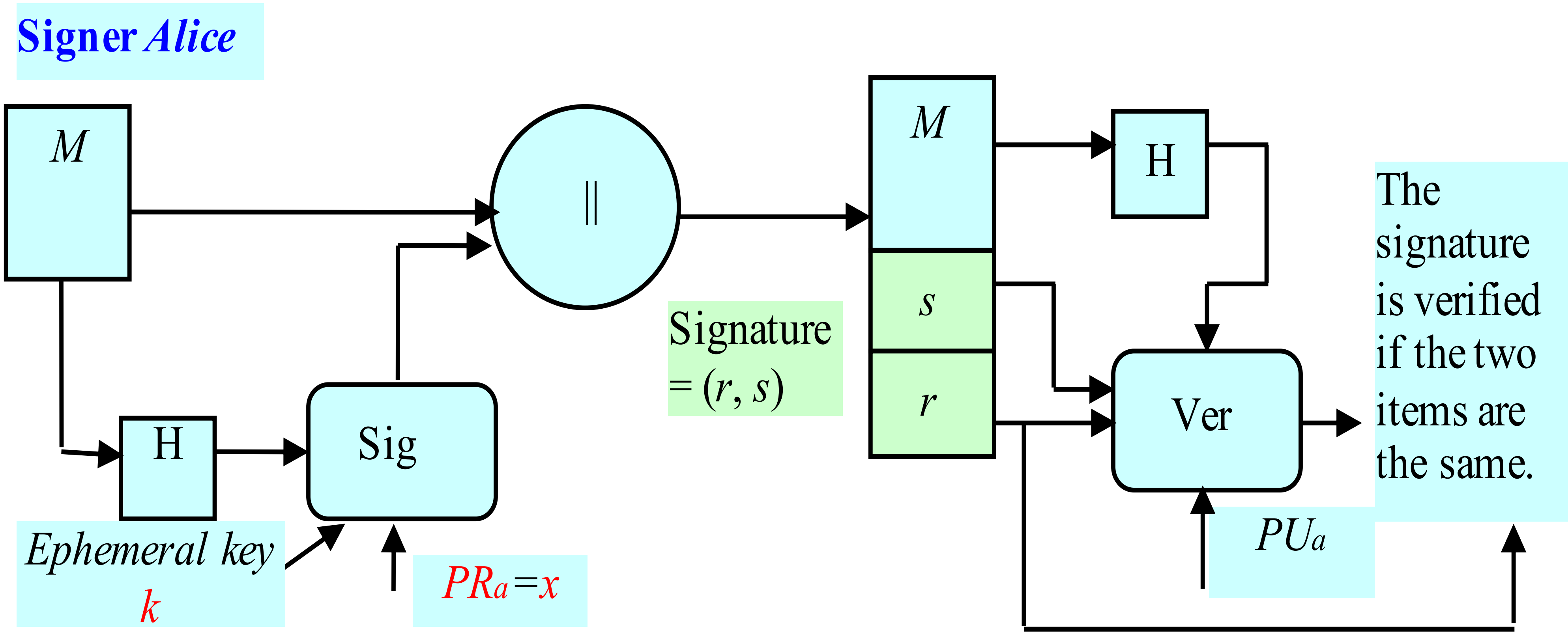
- DSS (Digital Signature Standard, also called DSA - digital signature algorithm)
- RSA vs DSA
- Conclusion

DSS/DSA - Background

- In 1991, the NIST (National Institute of Standards and Technology) proposed the **DSA** (Digital Signature Algorithm) for use in their **DSS** (Digital Signature Standard).
- Unlike RSA, DSA is a digital-signature-only algorithm.

DSS/DSA - Algorithm Overview

- k is a random secret number just generated for this signature.
  - (k, r) = (ephemeral private key, ephemeral public key); only used for this signature.
- (PR_a, PU_a) = sender's (private key, public key).



DSS/DSA - Key Generation

- Have shared global public key values (p, q, g)
  - choose a large prime p with $2^{L-1} < p < 2^L$, where L= 512 to 1024 bits.
  - choose 160-bit prime number q, such that q is a divisor of (p-1).
  - choose $g = h^{(p-1)/q} \bmod p$, where $1<h<p-1$ and $h^{(p-1)/q} \bmod p > 1$

- Users choose (long-term) private and compute public key
  - Choose random private key, x, i.e. $PR_a = x$, with $0< x <q$.
  - Compute public key: $PU_a = y = g^x \bmod p$.

DSS/DSA - Signature Generation

- to **sign** a message `M` the sender
  - Chooses a random number (the ephemeral key), k with $0 < k < q$; `k` must be destroyed after use, and must never be reused.
  - Computes the signature pair:
    - $r = (g^k \bmod p) \bmod q$.
    - $s = [k^{-1}\,(H(M)+xr)] \bmod q$.
    - $Signature = (r, s)$.
- Sends $M||Signature$ to the receiver.

DSS/DSA - Signature Verification

- The receiver has got $PU_a = \{p, q, g, y\}$, and $\{M', r', s'\}$.
- To verify the signature, he computes
  - message hash $H(M')$.
  - mod q inverse of s': $w=(s')^{-1} \bmod q$.
  - $u_1 = [H(M')w] \bmod q$.
  - $u_2 = (r')w \bmod q$.
  - $v = [(g^{u1} y^{u2}) \bmod p] \bmod q$.
- And check if: $v = r'$; if true, then the signature is verified (U prove!).
- Here, M = message to be signed; H(M) = hash of M using a hash function; M', r', s' = received versions of M, r, s.

DSS/DSA - a Baby Example

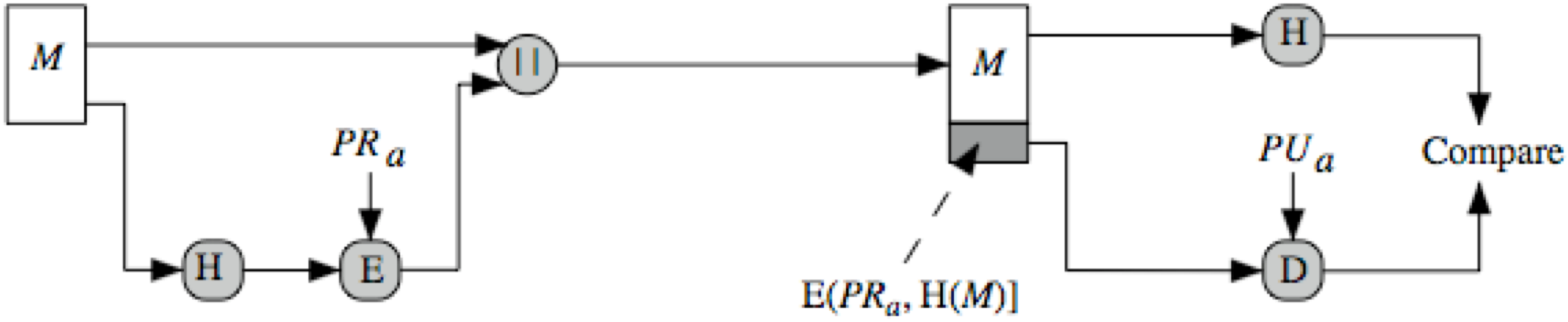
- Key generation
  - Generate q, p and g
    - q=13; p=4q+1=53; g=16;
  - Generate private key: x=3;
  - Compute public key: $y=g^3 \pmod{p} = 15$;

- Signature Signing
  - assuming H(M)=5;
  - choose k=2;
  - $r = (g^k \bmod p) \bmod q = (16^2 \bmod 53) \bmod 13 = 5$;
  - $s = [\,(H(M)+xr)*k^{-1}] \bmod q = [(5+3*5)*2^{-1}] \bmod 13 = 10$.

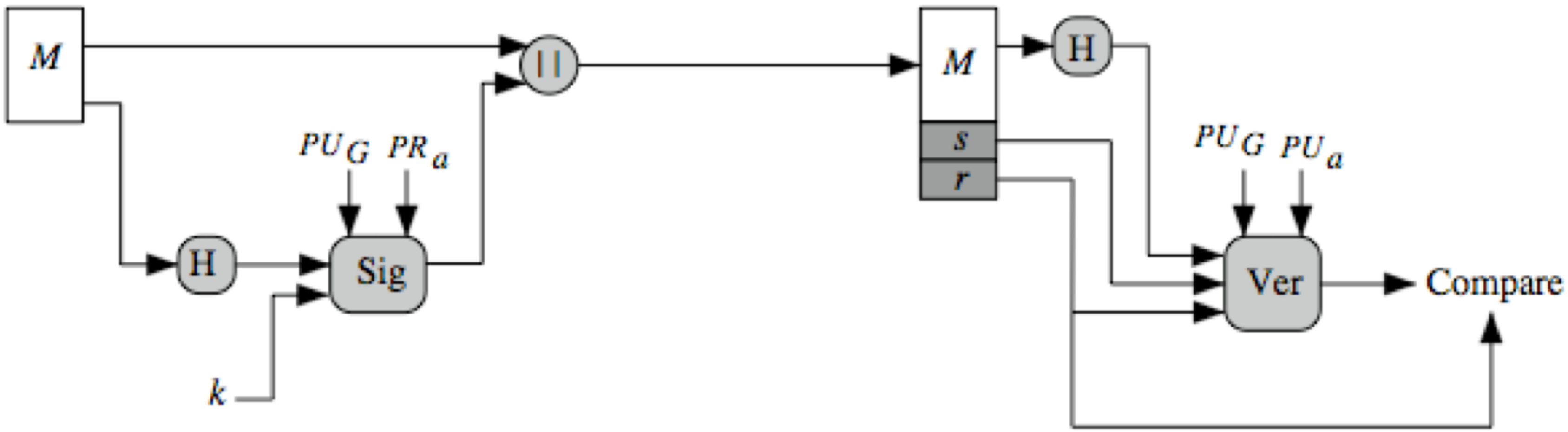
DSS/DSA - a baby example

- Signature Verification
  - computes
    - message hash $H(M')$ =5.
    - mod q inverse of s':
      $w=(s')^{-1} \bmod q = (10)^{-1} \bmod 13 = 4$.
    - $u_1 = [H(M')w] \bmod q = 5*4 \bmod 13 = 7$.
    - $u_2 = (r')w \bmod q = 5*4 \bmod 13 = 7$.
    - $v = [(g^{u1}\, y^{u2}) \bmod p] \bmod q$
      $= [(16^7 * 15^7) \bmod 53] \bmod 13 = 5$.
  - Note $v = r'$, so the signature is verified.

RSA vs DSA



(a) RSA Approach



(b) DSS Approach

RSA vs DSA

RSA	DSA
Security is based on difficulty of factoring large numbers.	Security is based on difficulty of taking discrete logarithms.
Can encrypt and sign.	Can only sign messages.
	Some signature computations can be computed a priori, so generally faster.
A RSA signature is about 1k - 2k bits long, depending on the size of the modulus.	A DSA signature is 320 bit long, desirable for applications requiring smaller signature footprints.
Can recover the message digest from the signature.	Cannot recover the message digest from the signature.
	Need to choose a unique secret number *k* for each message.

Exercise Question – E6.1

(i) Discuss, at the generic level, what are the factors that impact on the security of a digital signature.

(ii) Assuming that the RSA algorithm is used for signature signing, identify all possible ways of forging a signature.

Exercise Question – E6.2

A digital signature scheme may also be implemented using a symmetric-key cipher, but with the assistance of a trusted third party, an Arbitrator.

(i) Design a digital signature protocol using symmetric-key encryption and an arbitrator, but do not expose the content of a message to be signed to the arbiter.

(ii) Compare the signature protocol designed in (i) with the RSA based signature scheme.

Conclusions

- Digital signatures provide message authentication (integrity & origin authentication) and non-repudiation security services.
- There are *two* well-known signature schemes
  - RSA encryption algorithm can be used in reverse to produce a signature.
  - DSA is a signature algorithm based on discrete logarithms and can only be used for signature purposes.
- A signature scheme should be used in conjunction with a hash function to obtain security in an efficient way.