Asymmetric key crypto: Digital Signature

2019. 4. 2

Contents

- Introduction to crypto

- Symmetric-key cryptography
  - Stream ciphers
  - Block ciphers
  - Block cypher operation modes

- Public-key cryptography
  - RSA
  - Diffie Hellman, Elgamal
  - ECC
  - Digital signature
  - Public key Infrastructure

- Cryptographic hash function
  - Attack complexity
  - Hash Function algorithm

- Integrity and Authentication
  - Message authentication code
  - GCM
  - Digital signature

- Key establishment
  - server-based
  - Public-key based
  - Key agreement (Diffie-Hellman)

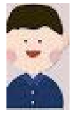
Uses of Public Key Crypto

- Encryption
  - Suppose we **encrypt** M with Bob's public key
  - Bob's private key can **decrypt** to recover M
- Digital Signature
  - **Sign** by encrypting with your private key
  - Anyone can **verify** signature by decrypting with sender's public key
- Key exchange

Digital Signature

- Alice sends a message with her signature which denotes her own unique identity, similar to handwriting.
- However, to prevent forgery of its signature, She should compute the signature that is related to the message being sent.
- In this way, She can ensure that she sends the very message herself.

Alice 

Bob 

Message m

$s = \mathrm{sign}_K(m)$

$\xrightarrow{(m,\ s)}$

$s' = \mathrm{verify}_K(m)$

If $s=s'$, valid

Sign with symmetric key

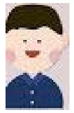
- The symmetric cypto keys can identify the uniqueness of the sender.
- If the symmetric key is used for signature, it verifies:
  - Sender identity
  - Message integrity

Sign with private key

- The private key can also identify the uniqueness of the sender.
- If the private key is used for signature, it verifies
  - Sender identity
  - Message integrity
- Better yet, it also provides
  - Non-repudiation
  - Why can't the symmetric key but the private key do?

RSA Digital Signature

Alice 

Bob 

Message m

$K^{-}=d$

$K^{+} = (n, e)$

$s = sign_{K^{-}}(m)$
$= (m)^{d} \bmod n$

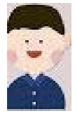
(m, s)

$m' = verify_{K^{+}}(s)$
$= (s)^{e} \bmod n$

If $(m=m')$ then valid

Elgamal Digital Signature

Alice 

Bob 

Message m

select p, $g \in \{2,3,\ldots,p-2\}$

$K^- = d \in \{2,3,\ldots,p-2\}$

$B = g^d \bmod p$

$\xleftarrow{K^+=(B,\, g,\, p)}$

Sign

select $K_E \in \{2,3,\ldots,p-2\}$ s.t.

$\gcd(K_E, p-1)=1$

$r = g^{K_E} \bmod p$

$s = (m-dr)K_E^{-1} \bmod p-1$

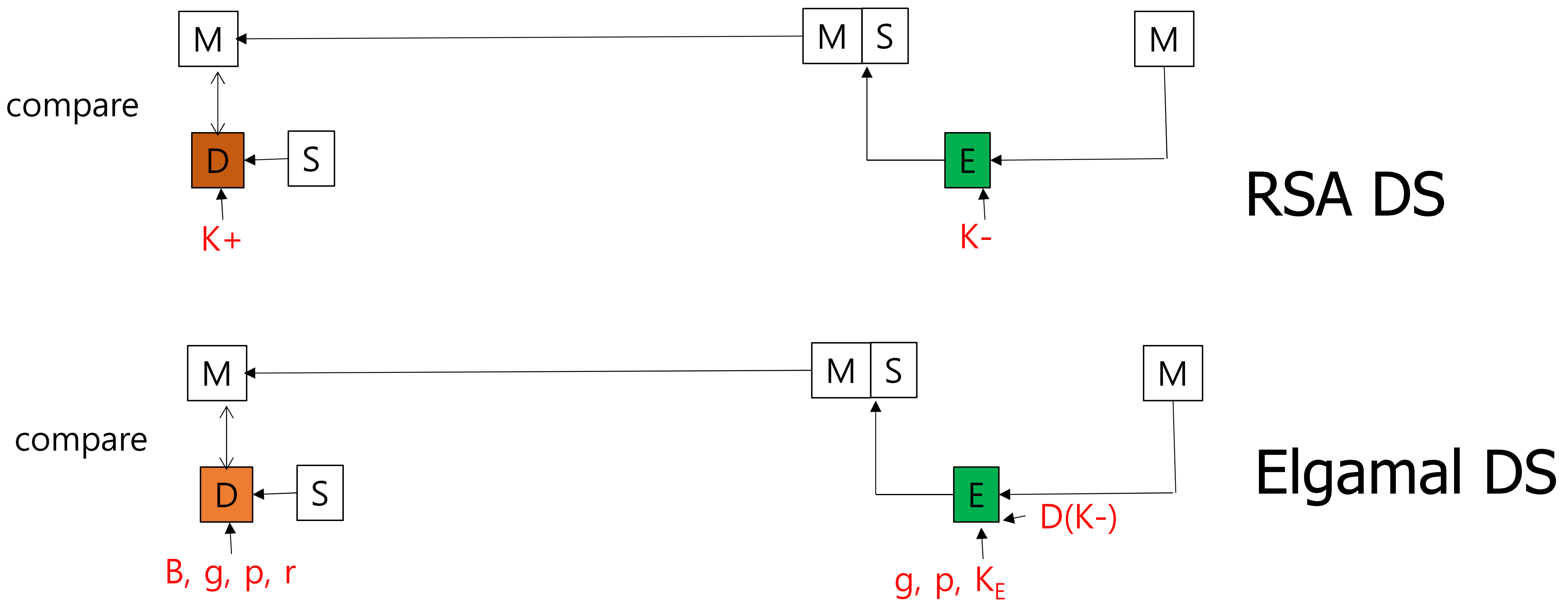
$\xleftarrow{m,\ (r,\, s)}$

Verify

$v = B^r r^s \bmod p$

If $v \equiv g^m \bmod p$, then "valid"

Comparison with RSA DS



Digital Signature Algorithm(DSA)

- US federal government standard for digital signature(DSS)
- DSA is a modification of ElGamal digital signature scheme. It was proposed by NIST in August 1991 and adopted in December 1994.

- The ElGamal DS would lead to signatures with at least 1024bits which is too much for such applications as smart cards.

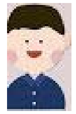
- In DSA a 160 bit message is signed using only 320-bit signature, but computation for verification is done modulo with 512-1024 bits, slower than RSA DS.

DSA key generation

- Generate a prime p with $2^{512} < p < 2^{1024}$ (multiple of 64 bits)
- Find a prime divisor q of $p\text{-}1$ with 160 bits
- Find an integer g such as $g^q = 1 \bmod p$
- Choose a random integer d with $0 < d < q\text{-}1$
- Compute $B \equiv g^d \bmod p$
- $Public = (g, p, q, B)$
- $Private = (d)$

DSA

Alice 

Bob 

Message m

select p, q, s.t., q|(p-1)

Select g, s.t., g^q = 1 mod p

$K^- = d \in \{2,3,\ldots,q-2\}$

$B = g^d \bmod p$

$K^+=(g, p, q, B)$ ←

Sign

H= SHA(m)

select $K \in \{2,3,\ldots,q-1\}$

$r = (g^K \bmod p) \bmod q$

$s = K^{-1}(H+dr) \bmod q$

m, (r, s) ←

Verify

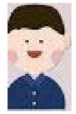
$H'=SHA(m)$, $s'=s^{-1}$

$r' = (B^{s'r} g^{s'H'} \bmod p) \bmod q$

If $r' \equiv r$, then "valid"

DSA example

Alice 

Bob 

Message x

select $p=23$, $g=6 \in \{2,3,\ldots,p-2\}$

$K^- = d=7 \in \{2,3,\ldots,p-2\}$

$B = 6^7 \bmod 23 = 3 \bmod 23$

Select $q=11$, s.t, $11|22$

$\xleftarrow{K^+=(3,\ 6,\ 23,\ 11)}$

Signature

select $K=2 \in \{2,3,\ldots,q-1\}$

$r = 6^2 \bmod 23 \bmod 11$

$= 13 \bmod 11 = 2 \bmod 11$

$H = SHA(m) = 5$

$s = 2^{-1}(5+7\text{x}2) \bmod 11 = 4$

$\xleftarrow{m,\ (2,\ 4)}$

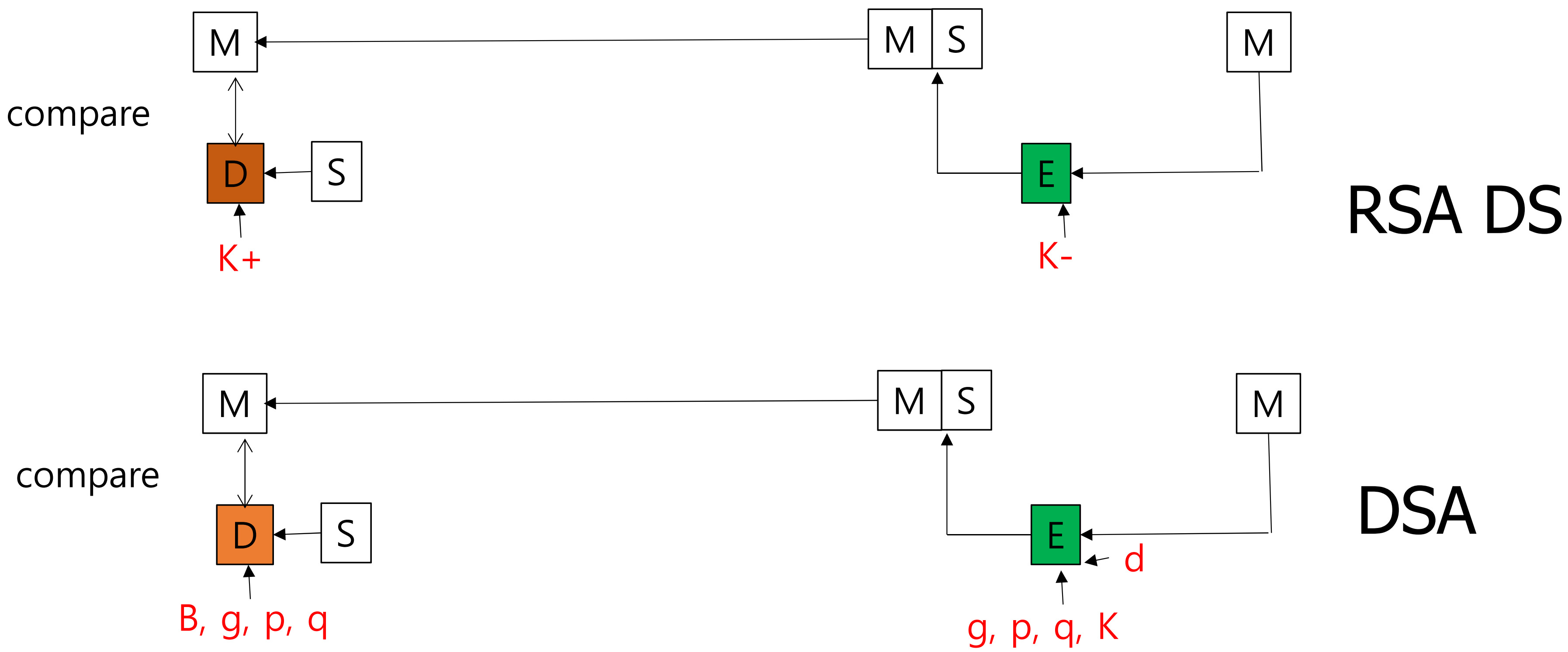
Verify

$H'=5$, $s'=4^{-1} \bmod 11 = 3$

$r' = (3^{3\text{X}2} 6^{3\text{x}5} \bmod 23) \bmod 11 = 2$

If $r' \equiv r$, then "valid"

Comparison with RSA DS



Computation of DSS

- Computationally demanding task of DSS is the exponential calculation g^K mod p.
- Because this is not involved in the message (m), it can be pre-calculated.
- Other demanding task is the determination of a multiplicative inverse K^{-1}, which is also pre-calculated on a number of value K.

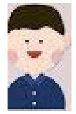
EC DSA

- Bit length of 160-256 can provide the same level of security as 1024-3072 bits RSA.
- The signature is twice the used bit length. (320-512 bits)

EC DSA

Alice 

$E: y^2 = x^3+ax+b \bmod p$, $G= (x_p, y_p)$, $n=|E|$

Bob 

Message m

select $d \in \{2,3,\ldots,n-1\}$
Compute $Q= bG =(x_B, y_B)$

$K^+=(Q)$ ←

Sign
select $k\in\{2,3,\ldots,n-2\}$
$P=kG=(x,y)$
$r = x \bmod n$
$s = k^{-1} (m+dr) \bmod n$

Verify
$w = s^{-1} \bmod n$
$u_1=mw$ and $u_2=rw$
$X=u_1G+u_2Q=(x_1,y_1)$
$v \equiv x_1 \bmod n$
If v=r, then "valid"

m, (r, s) ←