# Digital Signatures

## Cryptography - CS 411 / CS 507

Erkay Savaş

Department of Computer Science and Engineering
Sabancı University

November 15, 2019

# Digital Signatures

- Digital signatures enable us to <u>personalize</u> electronic documents, i.e., to associate our identities to them.
- The assumption is that no one else can fake our signature for a given message.
- Why don't we just digitize our <u>analog</u> signature and append it to a document?
- While classical signatures cannot be cut from a document and pasted into another document, the digitized analog signatures can easily be forged.
- *We need digital signature that cannot be separated from a message and attached to another.*

# Digital Signatures

- A digital signature is not only tied to the signer but also to the message that is being signed.
- Digital signatures must be easily verified by the others.
- Therefore, digital signature schemes consist of two distinct steps:
  1. The signing process (signature generation)
  2. The verification process (signature verification)

# RSA Signatures

## Alice (signer)

### RSA Setup

1. generates
   public key: $(e_A, n)$ and
   private key: $(d_A, p, q)$
2. generates signature for $m$
   $s = m^{d_A} \bmod n$
3. Sends $(m, s)$ to Bob

## Bob (verifier)

1. receives $(m, s)$
2. download $(e_A, n)$
3. Computes $z = s^{e_A} \bmod n$
4. Checks $z = m$

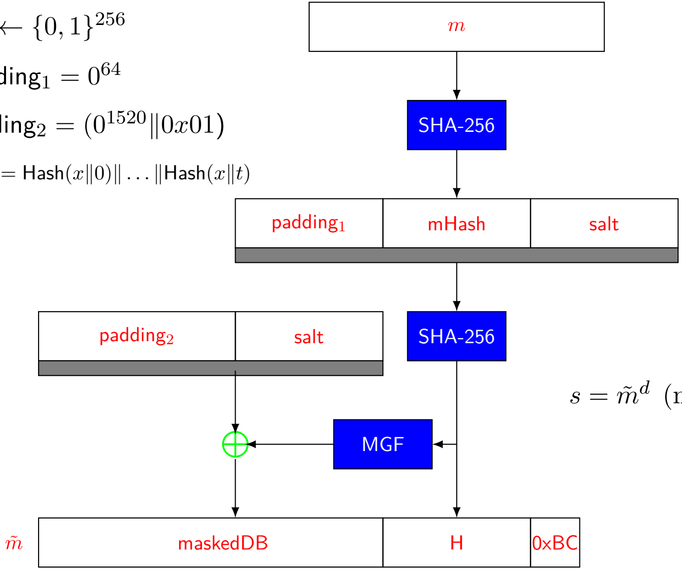
# RSA-PSS (Probabilistic Signature Scheme)- 2048 bit

salt $\leftarrow \{0,1\}^{256}$

padding$_1 = 0^{64}$

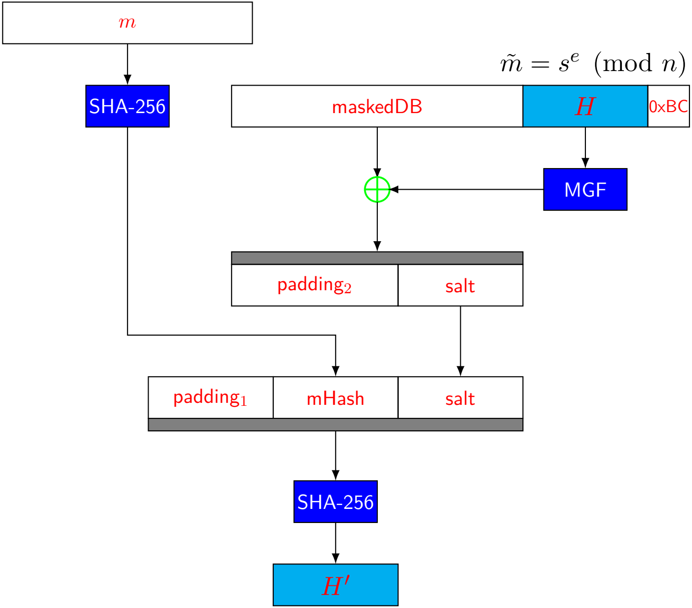
padding$_2 = (0^{1520} \| 0x01)$

MGF$(x) =$ Hash$(x\|0)\| \ldots \|$Hash$(x\|t)$




$s = \tilde{m}^d \pmod{n}$

# RSA-PSS - Signature Verification



# The Digital Signature Algorithm

- NIST proposed the DSA in 1991 and adopted it as a standard in 1993.
- It is similar to the ElGamal method.
- It uses a hash value (message digest) that is signed.
- The original standart (DSS) utilizes SHA-1 hash function which produces 160-bit hash values.
  - SHA-2 variants are approved for use
- We are trying to sign a 256-bit hash values.
  - 384-bit, or 512-bit

# DSA Setup

- Alice finds a prime $q$ that is $256$ bits long and chooses a prime $p$ that satisfies $q|p-1$ ($p$ is $3072$ bits)
  - Options: $(1024, 160)$, $(2048, 224)$, $(2048, 256)$, and $(3072, 256)$
- Let $g$ be a primitive root in group $G_q$.
- Let $\alpha$ be a random number $\pmod{p}$ and $g = \alpha^{(p-1)/q} \pmod{p}$
  - If $g \neq 1 \bmod p$ then use $g$ (otherwise try another $\alpha$)
- Alice chooses a secret value "$a$" such that $1 < a < q-1$ and calculates $\beta = g^a \pmod{p}$
- Alice publishes $\{p, q, g, \beta\}$ and keeps $\{a\}$ secret.

# Small DSA Parameters

- $p = 23$; $q = 11$; $g = 3$
  - $G_q = \{g^0, g^1, g^2, g^3, g^4, g^5, g^6, g^7, g^8, g^9, g^{10}\} \bmod p$
  - $G_q = \{1, 3, 9, 4, 12, 13, 16, 2, 6, 18, 8\}(3^{11} \bmod 23 = 1)$

| $\times$ mod 23 | 1 | 3 | 9 | 4 | 12 | 13 | 16 | 2 | 6 | 18 | 8 |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 1 | 3 | 9 | 4 | 12 | 13 | 16 | 2 | 6 | 18 | 8 |
| 3 | 3 | 9 | 4 | 12 | 13 | 16 | 2 | 6 | 18 | 8 | 1 |
| 9 | 9 | 4 | 12 | 13 | 16 | 2 | 6 | 18 | 8 | 1 | 3 |
| 4 | 4 | 12 | 13 | 16 | 2 | 6 | 18 | 8 | 1 | 3 | 9 |
| 12 | 12 | 13 | 16 | 2 | 6 | 18 | 8 | 1 | 3 | 9 | 4 |
| 13 | 13 | 16 | 2 | 6 | 18 | 8 | 1 | 3 | 9 | 4 | 12 |
| 16 | 16 | 2 | 6 | 18 | 8 | 1 | 3 | 9 | 4 | 12 | 13 |
| 2 | 2 | 6 | 18 | 8 | 1 | 3 | 9 | 4 | 12 | 13 | 16 |
| 6 | 6 | 18 | 8 | 1 | 3 | 9 | 4 | 12 | 13 | 16 | 2 |
| 18 | 18 | 8 | 1 | 3 | 9 | 4 | 12 | 13 | 16 | 2 | 6 |
| 8 | 8 | 1 | 3 | 9 | 4 | 12 | 13 | 16 | 2 | 6 | 18 |

# Small DSA Parameters

- $p = 23$; $q = 11$; $g = 5$
  - $\mathbb{Z}_{23}{}^{*} = \{5^0, 5^1, 5^2, 5^3, 5^4, 5^5, 5^6, 5^7, 5^8, 5^9, 5^{10}, 5^{11}, 5^{12}, 5^{13}, 5^{14}, 5^{15}, 5^{16}, 5^{17}, 5^{18}, 5^{19}, 5^{20}, 5^{21}, 5^{22}\} \bmod 23$
  - $\mathbb{Z}_{23}{}^{*} = \{1, 5, 2, 10, 4, 20, 8, 17, 16, 11, 9, 22, 18, 21, 13, 19, 3, 15, 6, 79, 12, 14\} \bmod 23$
  - $5^{22} \equiv 1 \bmod 23$
- $g = 22$
  - $22^0 \equiv 1 \bmod 23$
  - $22^1 \equiv 22 \bmod 23$
  - $22^2 \equiv 1 \bmod 23$

# Small DSA Parameters

- Pick a random $\alpha = 22$,
- Compute $\alpha^{(p-1)/q} \bmod p = 22^2 \bmod 23 = 1$ No good!
- Pick another random $\alpha = 4$,
- Compute $\alpha^{(p-1)/q} \bmod p = 4^2 \bmod 23 = 16$
- Compute $16^i \bmod 23$ for $i = 0, 1, ..., 11$:
  $1, 16, 3, 2, 9, 6, 4, 18, 12, 8, 13, 1$

# Small DSA Parameters: Another Example

- $p = 31$, $q = 5$ then $(p-1)/q = 6$
- Pick a random $\alpha = 25$,
- Compute $\alpha^{(p-1)/q} \bmod p = 25^6 \bmod 31 = 1$ No good!
- Pick another random $\alpha = 17$,
- Compute $\alpha^{(p-1)/q} \bmod p = 17^6 \bmod 31 = 8$
- Compute $8^i \bmod 31$ for $i = 0, 1, ..., 5$: $1, 8, 2, 16, 4, 1$

# DSA - Signature Scheme

- Message $m$
- Computes $h = H(m)$
- She selects a random, secret integer $k$ such that $1 < k < q$.
- Computes $r = (g^k \pmod p) \pmod q$.
- Computes $s = k^{-1}(h + ar) \pmod q$.
- Alice's signature for $m$ is $(r, s)$.
- Alice sends $(r, s)$ and $m$ to Bob to verify.

# DSA - Verification Scheme

- Bob downloads Alice's public information $(p, q, g, \beta)$.
- Computes $h = H(m)$
- Computes $u_1 = s^{-1}h \pmod{q}$.
- Computes $u_2 = s^{-1}r \pmod{q}$.
- Computes $v = (g^{u_1}\beta^{u_2} \pmod{p}) \pmod{q}$.
- Bob accepts the signature if and only if
  $v = r$.

- Show that the verification really works.

# Session key $k$ must be unique and randomly chosen

- What if $k$ is used twice?
- Then, we have two signatures $(m_i, r, s_i)$ and $(m_j, r, s_j)$ for $m_i \neq m_j$
- $s_i = k^{-1}(h_i + ar) \pmod q$ and $s_j = k^{-1}(h_j + ar) \pmod q$.
- $k = s_i^{-1}(h_i + ar) \pmod q = s_j^{-1}(h_j + ar) \pmod q$.
- $s_j(h_i + ar) = s_i(h_j + ar) \pmod q$.
- $ar(s_j - s_i) = s_i h_j - s_j h_i \pmod q$.
- $a = (s_i h_j - s_j h_i)(r(s_j - s_i))^{-1} \pmod q$

# Session keys $k$ must be independent

- What if $k_j = xk_i$ for a relatively small integer $x$?
- Then, we have two signatures $(m_i, r_i, s_i)$ and $(m_j, r_j, s_j)$ for $m_i \neq m_j$ .
- $s_i = k_i^{-1}(h_i + ar_i) \pmod q$
- $s_j = k_j^{-1}(h_j + ar_j) = k_i^{-1}x^{-1}(h_j + ar_j) \pmod q$.
- $k_i = s_i^{-1}(h_i + ar_i) \pmod q = s_j^{-1}x^{-1}(h_j + ar_j) \pmod q$.
- $s_j x(h_i + ar_i) = s_i(h_j + ar_j) \pmod q$.
- $a(s_j r_i x - s_i r_j) = s_i h_j - s_j h_i x \pmod q$.
- $a = (s_i h_j - s_j h_i x)(s_j r_i x - s_i r_j)^{-1} \pmod q$