Digital Signatures

CS 411 / 507 - Cryptography

Erkay Savaş & Atıl Utku Ay

Faculty of Engineering and Natural Sciences
Sabancı University

November 27, 2023

Digital Signatures

- Digital signatures enable us to personalize 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 analog 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 $z = m$

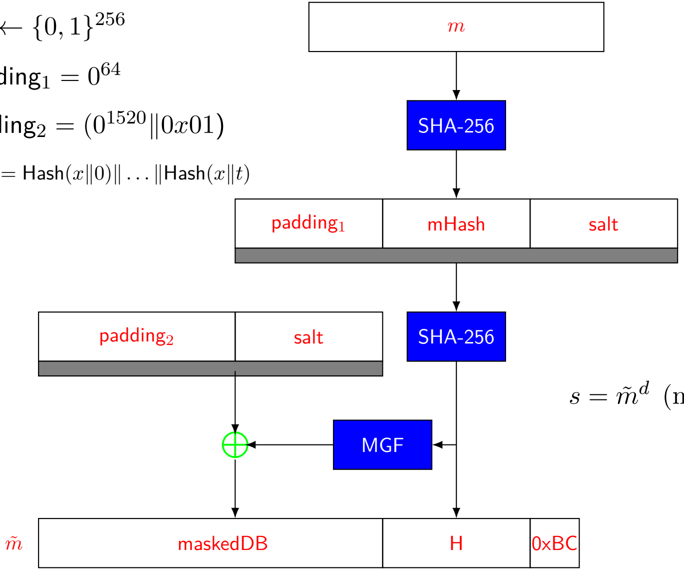
RSA-PSS (Probabilistic Signature Scheme)- 2048 bit

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

padding$_1 = 0^{64}$

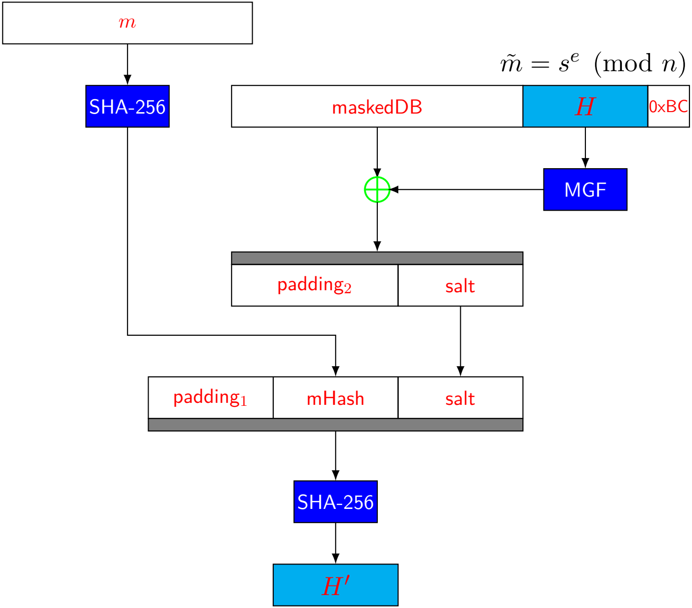
padding$_2 = (0^{1520} \| 0x01)$

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 α be a random number $\bmod p$ and let $g = \alpha^{(p-1)/q} \bmod p$
  - If $g \neq 1 \bmod p$ then use g (otherwise try another α)
- Alice chooses a secret value "a" such that $1 < a < q-1$ and calculates $\beta = g^a \bmod p$
- Alice publishes $\{p, q, g, \beta\}$ and keeps $\{a\}$ secret.

Small DSA Parameters

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

× 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

- 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 $r = (g^k \bmod p)(\mathrm{mod} q)$.
- Computes $s = k^{-1}(h + ar)(\mathrm{mod} q)$.
- 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, β).
- Computes $h = H(m)$
- Computes $u_1 = s^{-1}h \bmod q$.
- Computes $u_2 = s^{-1}r \bmod q$.
- Computes $v \equiv (g^{u_1}\beta^{u_2} \bmod p) \bmod q$.
- Bob accepts the signature if and only if $v = r$.

- Show that the verification really works.

Birthday Attacks on DSA

- Fred is a real estate agent and Alice wants to buy a land in Antalya.
  - She will sign a contract electronically using the DSA (she actually signs the hash value of the contract).
  - Suppose they use a hash function that produces 50-bit hashes instead of SHA-1 (or SHA-2 or SHA-3).
- Can Fred trick Alice to buy a land with no value in Antalya while she thinks otherwise?
- Fred is unlikely to produce a fake contract which produces the hash value as the original contract.
- But he can use a different approach.

Birthday Attacks in DSA

- Fred prepares the original contract for a nice piece of land.
- On the other hand, he also locates other places which have no value whatsoever.
  1. And he prepares 2^{30} different contracts for these junk lands by changing the wording slightly, placing a space at the end of a line, etc.
  2. He also prepares 2^{30} different variations of the original contract using the same tricks in which Alice does not notice the difference.
  3. He searches a match between the two sets of contracts which produces the same hash value.

Birthday Attacks in DSA

- He is pretty sure he can find a match since the birthday paradox tells us the probability that there is match when $k = 2^{30}$ and $n = 2^{50}$ is given by $1 - e^{-1024} \approx 1$. (remember the formula $1 - e^{-k^2/n}$)
- He gives this variation of the original contract to Alice to sign but appends the signature to the fake contract which produces the same hash value.