Digital Signatures

Objectives

- Features of hand-written signatures in Digital World
- Ensure hardness of forgery

Hand-written Signatures

- Function: bind a statement/message to its authors.
- Verification is public. (against a prior authenticated one)

Hand-written Signatures

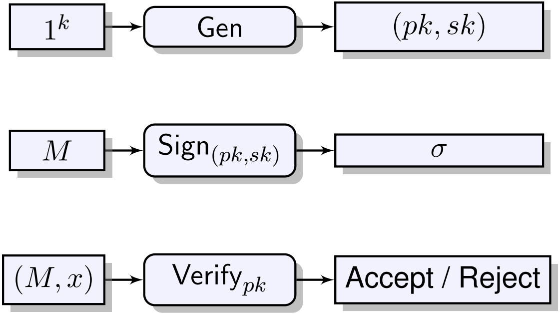
- Function: bind a statement/message to its authors.
- Verification is public. (against a prior authenticated one)
- Properties:
  - Correctness: A correct signature should always be verified true.

Hand-written Signatures

- Function: bind a statement/message to its authors.
- Verification is public. (against a prior authenticated one)
- Properties:
  - Correctness: A correct signature should always be verified true.
  - Security: Hard to forge.

Signature Schemes

Signature Scheme (Gen, Sign, Verify)

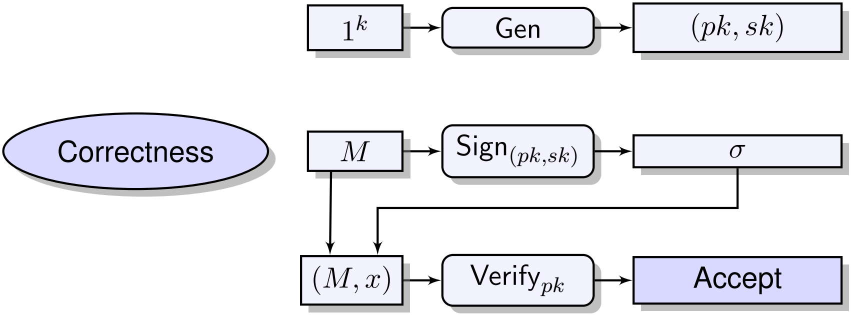


Required Properties:

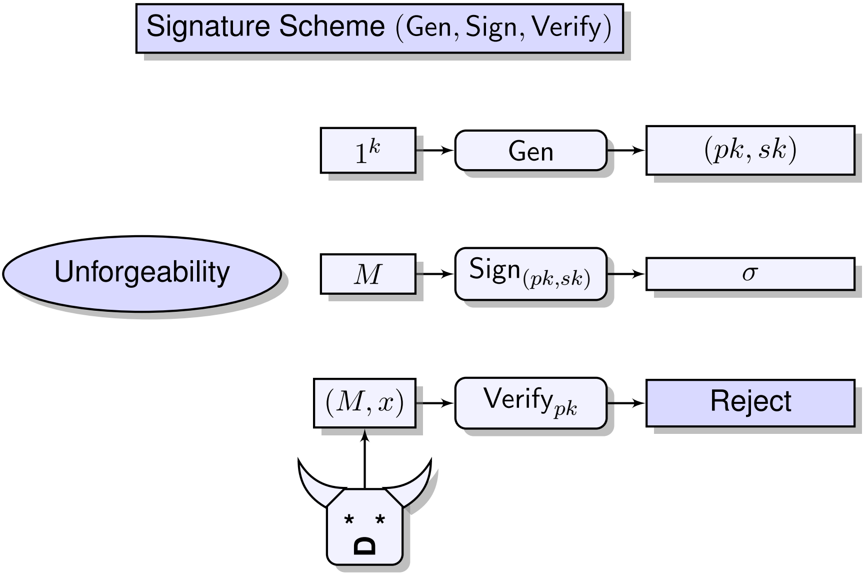
- Correctness
- Unforgeability

Signature Schemes

Signature Scheme $(\mathsf{Gen}, \mathsf{Sign}, \mathsf{Verify})$



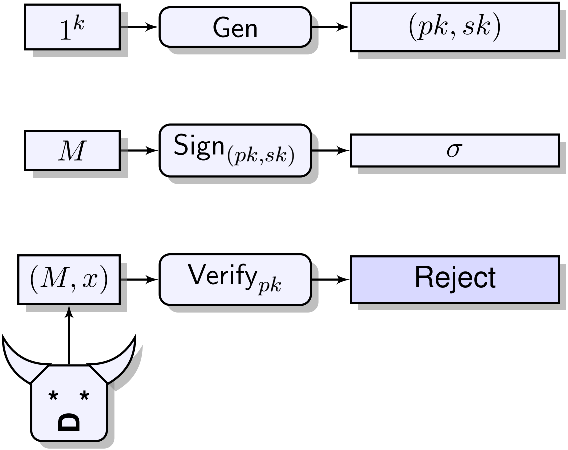
Signature Schemes



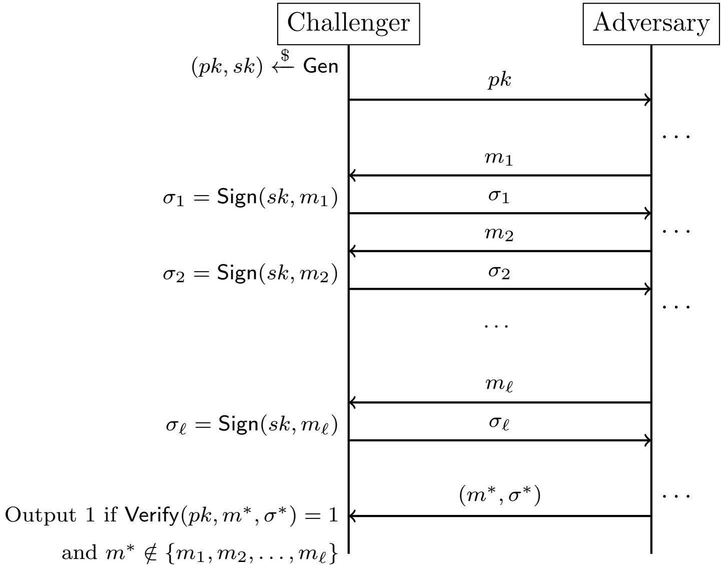
Signature Schemes

Signature Scheme (Gen, Sign, Verify)

Unforgeability: Must output forgery for a message for which the attacker did not request the signature.



Unforgeability against Chosen Message Atttack



Signature Schemes Designs: RSA Full Domain Hash

- **Public Functions** A hash function $H : \{0,1\}^* \to \mathbb{Z}_N^*$
- **Keygen:** Run RSA.Keygen. $pk = (e, N)$, $sk = (d, N)$.
- **Sign**: Input: sk, M. Output
  $\sigma =$ RSA.Dec$(sk, H(M)) = H(M)^d \mod N$

Correctness

Suppose $\sigma =$ **Sign**(sk, M). This implies $\sigma = H(M)^d \mod N$. This implies

$$\sigma^e \mod N = (H(M)^d \mod N)^e \mod N = H(M)^{ed} \mod N$$

As $ed \equiv 1 \mod \phi(N)$ and H maps to $\mathbb{Z}_N^*$, we have

$$\sigma^e \mod N = H(M) \mod N = H(M)$$

which is the acceptance condition in the verification algorithm.

Signature Schemes Designs: RSA Full Domain Hash

- **Public Functions** A hash function $H : \{0,1\}^* \to \mathbb{Z}_N^*$
- **Keygen:** Run RSA.Keygen. $pk = (e, N)$, $sk = (d, N)$.
- **Sign**: Input: sk, M. Output $\sigma = \text{RSA.Dec}(sk, H(M)) = H(M)^d \mod N$
- **Verify**: Input: pk, M, σ. If $\text{RSA.Enc}(pk, \sigma) = H(M)$ output accept, else reject

Correctness

Suppose $\sigma =$ **Sign**(sk, M). This implies $\sigma = H(M)^d \mod N$. This implies

$$\sigma^e \mod N = (H(M)^d \mod N)^e \mod N = H(M)^{ed} \mod N$$

As $ed \equiv 1 \mod \phi(N)$ and H maps to $\mathbb{Z}_N^*$, we have

$$\sigma^e \mod N = H(M) \mod N = H(M)$$

which is the acceptance condition in the verification algorithm.

Signature Schemes Designs: RSA Full Domain Hash

- **Public Functions** A hash function $H : \{0,1\}^* \to \mathbb{Z}_N^*$
- **Keygen:** Run RSA.Keygen. $pk = (e, N)$, $sk = (d, N)$.
- **Sign**: Input: sk, M. Output
  $\sigma =$ RSA.Dec$(sk, H(M)) = H(M)^d \mod N$
- **Verify**: Input: pk, M, σ. If RSA.Enc$(pk, \sigma) = H(M)$ output accept, else reject
- If $\sigma^e \mod N = H(M)$, output accept, else reject.

Correctness

Suppose $\sigma =$ **Sign**(sk, M). This implies $\sigma = H(M)^d \mod N$. This implies

$$\sigma^e \mod N = (H(M)^d \mod N)^e \mod N = H(M)^{ed} \mod N$$

As $ed \equiv 1 \mod \phi(N)$ and H maps to $\mathbb{Z}_N^*$, we have

$$\sigma^e \mod N = H(M) \mod N = H(M)$$

which is the acceptance condition in the verification algorithm.

Signature Schemes Designs: Digital Signature Algorithm (DSA) 1991

DSA is adopted in standard FIPS 186-1 to FIPS 186-4

Keygen

- Choose a 2048 bit prime p, a 224 bit prime q where q divides $p-1$. Choose a number $g < q$ such that gcd$(g, p-1) = 1$
- A cryptographic hash function $H : \{0,1\}^* \to \mathbb{Z}_q$
- Choose a random $x \xleftarrow{\$} \mathbb{Z}_q$. Compute $y = g^x \mod p$
- $pk = (p, q, g, y, H)$. $sk = x$.

Signature Schemes Designs: Digital Signature Algorithm (DSA) 1991

$\mathsf{Sign}(sk = x, pk, M)$

- $r \xleftarrow{\$} \mathbb{Z}_q$
- compute $s = (g^r \mod p) \mod q$
- compute $t = (H(M) + x \cdot s)r^{-1} \mod q$
- output $\sigma = (s, t)$

$\mathsf{Verify}(pk, M, \sigma)$

- Compute $u_1 = H(M)t^{-1} \mod q$
- Compute $u_2 = s \cdot t^{-1} \mod q$
- If $g^{u_1} \cdot y^{u_2} \mod p \mod q = s$, accept
- otherwise reject

Correctness of signature

Correctness condition

$g^{u_1} \cdot y^{u_2} \mod p \mod q = s = (g^r \mod p) \mod q$

Derivations

$t^{-1} = (H(M) + x \cdot s)^{-1} r \mod q \implies t^{-1}(H(M) + x \cdot s) = r \mod q$

- $y^{u_2} \mod p = (g^x)^{u_2} \mod p = g^{x \cdot u_2} \mod p = g^{xst^{-1}} \mod p$
- $g^{u_1} \cdot y^{u_2} \mod p = g^{u_1 + xst^{-1}} \mod p = g^{t^{-1}(H(M)+xs)} \mod p = g^r \mod p$

Schnorr Signature

Keygen

- Choose a 3072 bit prime p, a 256 bit prime q where q divides $p - 1$. Choose a number $g < q$ such that $\gcd(g, p - 1) = 1$.
- A cryptographic hash function $H : \{0, 1\}^* \to \mathbb{Z}_q$
- Choose a random $x \stackrel{\$}{\leftarrow} \mathbb{Z}_q$. Compute $y = g^x \mod p$
- $pk = (p, q, g, y, H)$. $sk = x$.

Schnorr

$\mathsf{Sign}(sk = x, pk, M)$

- $r \xleftarrow{\$} \mathbb{Z}_q$
- compute $s = H(M||g^r)$
- compute $t = (r + x \cdot s) \mod q$
- output $\sigma = (s, t)$

$\mathsf{Verify}(pk, M, \sigma = (s, t))$

- If $H(M||g^t y^{-s}) = s$, accept
- otherwise reject

Schnorr

Sign$(sk = x, pk, M)$

- $r \xleftarrow{\$} \mathbb{Z}_q$
- compute $s = H(M||g^r)$
- compute $t = (r + x \cdot s) \mod q$
- output $\sigma = (s, t)$

Verify$(pk, M, \sigma = (s, t))$

- If $H(M||g^t y^{-s}) = s$, accept
- otherwise reject

Correctness

$$
\begin{aligned}
g^t y^{-s} \mod p = g^t g^{-x \cdot s} \mod p &= g^{r + x \cdot s} g^{-x \cdot s} \mod p \\
&= g^r \mod p
\end{aligned}
$$