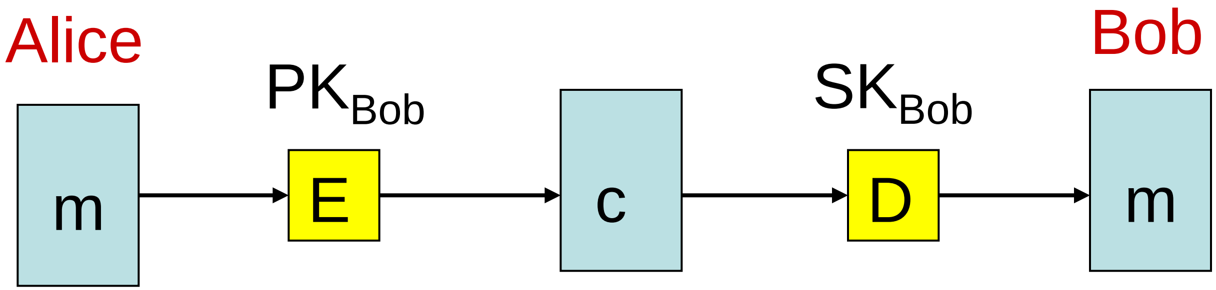
Digital Signatures

Digital Signatures

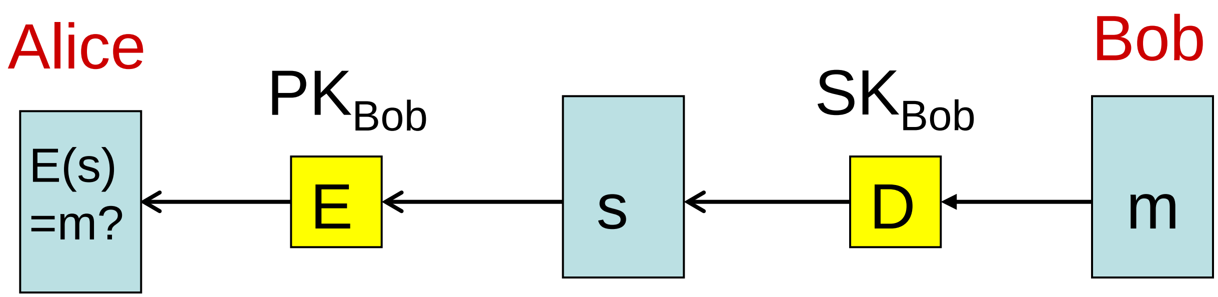
- Digital signature is the same as MAC except that the tag (signature) is produced using the secret key of a public-key cryptosystem.

- Digital signature:
  1. Bob has a key pair (sk, pk).
  2. Bob sends $m \,\|\, \mathrm{Sign}_{sk}(m)$ to Alice.
  3. Alice verifies the received $m' \,\|\, s'$ by checking if $\mathrm{Vrfy}_{pk}(m', s') = 1$?
- $\mathrm{Sign}_{sk}(m)$ is called a signature for m.
- Security requirement: infeasible to produce a valid pair $\langle m,\ \mathrm{Sign}_{sk}(m) \rangle$ without knowing sk.

Encryption (using RSA):



Signing (using RSA^{-1}):



Verify the signature

Sign

Basic RSA Signature

- Keys are generated as for RSA encryption:

  Public key: $pk = (N, e)$. Secret key: $sk = (N, d)$.

- Signing a message $m \in \mathbb{Z}_N^*$:

  $$\sigma = \mathrm{Sign}_{sk}(m) = m^d \bmod N.$$

  That is, $\sigma = \mathrm{RSA}^{-1}(m)$.

- Verifying a signature (m, σ):

  $$\mathrm{Vrfy}_{pk}(m, \sigma) = 1 \text{ if and only if } m = \sigma^e \bmod N$$

  $$\text{or } m = \mathrm{RSA}(\sigma).$$

- Correctness:

$$\mathrm{RSA}_{pk}\left(\mathrm{RSA}_{sk}^{-1}(m)\right)=m.$$

A message m signed with sk will be verified and accepted with the corresponding pk.

- Remarks:
  - Basic RSA signature is the reverse of basic RSA encryption.
  - Because of this, digital signatures are often mistakenly viewed as the reverse of public-key encryption.
  - As will be seen, secure RSA signature is not the reverse of secure RSA encryption. Neither is ElGamal signature.

- Existentially forgeable:
  1. Every message m is a valid signature of its ciphertext c, since $\mathrm{RSA}^{-1}(c) = m$.
  2. If Bob signed m_1 and m_2, then the signature for $m_1 m_2$ can be easily forged: $\sigma(m_1 m_2) = \sigma(m_1)\sigma(m_2)$.
- Remedy: hash then sign:

  $\sigma = \mathrm{Sign}_{sk}(H(m)) = \mathrm{RSA}_{sk}^{-1}(H(m))$, using some hash function H.

- Question:

  Does hash-then-sign make RSA signature secure against chosen-message attacks?

- Answer:

  Yes, if H is a full-domain random oracle, i.e.,

  - H is a random oracle mapping $\{0,1\}^* \rightarrow \mathbb{Z}_N$
  - ($\mathbb{Z}_N$ is the full domain of RSA^{-1})

Theorem: Full-domain hash RSA signature is secure against any chosen-message attack under the random oracle model.

- Problem with full-domain hash:

  In practice, H is not full-domain.

  For instance, the range of SHA-1 is $\{0,1\}^{160}$,

  while $\mathbb{Z}_N = \left\{0,1,\ldots,N-1\right\} \approx \{0,1\}^{1024}$, if $\left\|N\right\| = 1024$.

- Desired: a secure signature scheme that does not require a full-domain hash.

Probabilistic signature scheme

- Hash function $H : \{0,1\}^* \rightarrow \{0,1\}^l \subset \mathbb{Z}_N$ (not full domain).

  $l \ll n = \|N\|$. (E.g., SHA-1, $l = 160$; RSA, $n = 1024$.)

- Idea: $m \xrightarrow{\text{pad}} m \,||\, r \qquad \in \{0,1\}^*$

  $\xrightarrow{\text{hash}} x = H(m \,||\, r) \qquad \in \{0,1\}^l$

  $\xrightarrow{\text{expand}} y = x \,||\, (r \,||\, 0^{n-1-l-k}) \oplus G(x) \qquad \in \{0,1\}^{n-1}$

  $\xrightarrow{\text{sign}} \sigma = \mathrm{RSA}^{-1}(y) \qquad \in \mathbb{Z}_N$

  where $r \in \{0,1\}^k$ for some k

  $G : \{0,1\}^l \rightarrow \{0,1\}^{n-1-l}$ (pseudorandom generator)

- Signing a message $m \in \{0,1\}^*$:

  1. choose a random $r \in \{0,1\}^k$; compute $x = H(m \,||\, r)$;

  2. compute $y = x \,||\, r \oplus G_1(x) \,||\, G_2(x)$; // $G = G_1 \,||\, G_2$ //

  3. The signature is $\sigma = \mathrm{RSA}^{-1}(y)$.

- Verifying a signature (m, σ): compute $\mathrm{RSA}(\sigma) = x \,||\, t \,||\, u$;

  check if $u = G_2(x)$ and $x = H\left(m \,||\, t \oplus G_1(x)\right)$.

Remarks

- PSS is secure against chosen-message attacks in the random oracle model (i.e., if H and G are random oracles).
- PSS is adopted in PKCS #1 v.2.1.
- Hash functions such as SHA-1 are used for H and G.
- For instance,

  let $n = 1024$, and $l = k = 160$

  let H = SHA-1

  $G(x) = H(x \| 0) \; \| \; H(x \| 1) \; \| \; H(x \| 2), \; \ldots$

DLP-based Digital Signatures

Ideas behind DLP-based signature

- $G = \{g^0,\ g^1,\ g^2,\ \ldots, g^x, \ldots,\ g^{q-1}\}$, a cyclic group of order q.
  $\mathbb{Z}_q = \{0,\ 1,\ 2,\ \ldots, x, \ldots,\ q-1\}$.
  $sk = (G, g, q, x),\ pk = (G, g, q, h)$ where $h = g^x$.
- To sign a message m, Alice needs to show that she knows the secret key x. Besides, non-deterministic signature is desired. So, the signature should be a function of (m, x, k), where k is random.
- We have $x \in \mathbb{Z}_q$, suggesting that $m,\ k \in \mathbb{Z}_q$.
- So, let the signature s be a function of (m, x, k) whose validity can be verified using g^m, g^x, g^k.
- The signer needs to send $r := g^k$ along with s.

- The signature s is a function of (m, x, k).
  - $s = km + k'x \mod q$ where $k,\ k' \leftarrow_u \mathbb{Z}_q$
  - $s = km + F(r)x \mod q$ where $r = g^k,\ F : G \to \mathbb{Z}_q$
  - $s = \big(m - F(r)\big)k^{-1} \mod q$ // $m = ks + F(r)x \mod q$ //
- ElGamal signature: $(r, s) \in G \times \mathbb{Z}_q$, $s = \big(m - F(r)x\big)k^{-1} \bmod q$
- To verify a signature (r, s),

  the verifier checks if $g^m = r^s h^{F(r)}$ without knowing x and k:

$$s = \big(m - F(r)x\big)k^{-1} \mod q$$

$$\Leftrightarrow\ m = ks + F(r)x \mod q$$

$$\Leftrightarrow\ g^m = g^{ks} g^{F(r)x} \quad \text{// in } G \text{ //}$$

$$\Leftrightarrow\ g^m = r^s h^{F(r)}$$

- Shnorr signature: $\big(F(r),\ s\big) \in \mathbb{Z}_q \times \mathbb{Z}_q$

$$s = \big(m + F(r)\big)k^{-1} \mod q$$

- To verify a signature $\big(F(r),\ s\big)$,

  the verifier checks if $F(r) = F\left(\left(g^m h^{F(r)}\right)^t\right)$:

$$s = \big(m + F(r)x\big)k^{-1} \mod q$$

$$\Rightarrow\ k = \big(m + F(r)x\big)t \mod q \qquad // \ t = s^{-1} \bmod q \ //$$

$$\Rightarrow\ g^k = \left(g^m g^{F(r)x}\right)^t \qquad // \text{ in } G \ //$$

$$\Rightarrow\ r = \left(g^m h^{F(r)}\right)^t \qquad // \text{ in } G \ //$$

$$\Rightarrow\ F(r) = F\left(\left(g^m h^{F(r)}\right)^t\right) \qquad // \text{ in } \mathbb{Z}_q \ //$$

ElGamal signature in $\mathbb{Z}_p^*$

1. Key generation: same as in ElGamal encryption.
   - a large prime p and a generator $g \in \mathbb{Z}_p^*$.
   - a randomly chosen number $x \in \mathbb{Z}_{p-1}$ and $h = g^x \bmod p$;
   - $sk = (p, g, x)$ and $pk = (p, g, h)$.
2. To sign a message $m \in \mathbb{Z}_{p-1}$,
   - randomly choose $k \in \mathbb{Z}_{p-1}^*$;
   - compute $r = g^k \bmod p$ and $s = (m - rx)k^{-1} \bmod (p-1)$;
   - the signature is $\mathrm{Sign}_{sk}(m) = (r, s) \in \mathbb{Z}_p^* \times \mathbb{Z}_{p-1}$.
3. Verification: $\mathrm{Vrfy}_{pk}(m, r, s) =$ true if and only if

$$(m, r, s) \in \mathbb{Z}_{p-1} \times \mathbb{Z}_p^* \times \mathbb{Z}_{p-1} \text{ and } g^m \equiv r^s h^r \bmod p.$$

Security of ElGamal signature

- Based on the assumed intractability of discrete logarithm.
- Shoud use a new k for each signing, or the adversary can compute k from two signatures

$$s = (m - rx)k^{-1} \text{ and } s' = (m' - rx)k^{-1}$$

$$\Rightarrow \quad s - s' \equiv (m - m')k^{-1} \bmod (p-1)$$

$$\Rightarrow \quad k = (m - m')(s - s')^{-1} \bmod (p-1)$$

- Knowing k, the adversary can compute x with high probability:

$$s = (m - rx)k^{-1} \bmod (p-1)$$

$$\Rightarrow \quad x = (m - sk)r^{-1} \bmod (p-1), \text{ if } r^{-1} \bmod (p-1) \text{ exists.}$$

Security of ElGamal signature (cont'd)

- Existential forgery. Construct a message m and a valid signature (r, s) as follows.

  a) choose $k,\ c \in \mathbb{Z}^{*}_{p-1}$.

  b) set $r = g^{k} h^{c} \bmod p,\ s = -rc^{-1} \bmod (p-1),$ and

  $$m = -rkc^{-1} \bmod (p-1).$$

- Countermeasure: hash then sign.

Digital Signature Algorithm (DSA) - an NIST standard

0. Shnorr's idea: working in a subgroup of $\mathbb{Z}_p^*$ of prime order $q \ll p$ will shorten the signature, desired for Smart Card applications.

- ElGamal signature scheme uses:

  $$\mathbb{Z}_p^* = \left\{\alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{p-2}\right\}. \quad \mathbb{Z}_{p-1} = \left\{0, 1, 2, \ldots, p-2\right\}.$$

  A signature is $(r, s) \in Z_p^* \times Z_{p-1} \approx Z_{p-1} \times Z_{p-1}$.

- DSA uses:

  $$\langle g \rangle = \left\{g^0, g^1, \ldots, g^{q-1}\right\} \subset \mathbb{Z}_p^*, \text{ where } g = \alpha^b, bq = p-1.$$

  A signature is $(\hat{r}, s) \in Z_q \times Z_q$

1. Key generation

- choose two primes p and q such that $q \mid (p-1)$.

  $(q \ll p,$ e.g., $\|q\| = 160,\ \|p\| = 1024.)$
- let $g \in \mathbb{Z}_p^*$ be an element of order q.
- randomly choose $0 \neq x \in \mathbb{Z}_q$ and compute $h = g^x \bmod p$;
- system parameters: (p, q, g)
- $sk = (x)$ and $pk = (h)$.

(Remark: The DLP will be working in $\langle g \rangle$.)

2. Signing: to sign a message m,

- randomly choose $k \in Z_q^*$; compute $\hat{r} = \underbrace{(g^k \bmod p)}_{r} \bmod q$.
- compute $s = (H(m) + \hat{r}x)k^{-1} \bmod q$.

  // choose a different k if $\hat{r} = 0$ or $s = 0$ //
- $(m, \hat{r}, s)$ is the signed message.

3. Verification: accept $(m, \hat{r}, s)$ iff $\hat{r}, s \in Z_q^*$ and

$$\hat{r} = \left((g^{H(m)} h^{\hat{r}})^t \bmod p\right) \bmod q, \text{ where } t = s^{-1} \bmod q.$$

DSA Correctness: if the message is signed correctly, the signature will be verified/accepted.

$$s = (H(m) + \hat{r}x)k^{-1} \bmod q$$

$$\Rightarrow\ k \equiv (H(m) + \hat{r}x)t \ \bmod q \qquad (t = s^{-1} \bmod q)$$

$$\Rightarrow\ g^{k} \equiv \left(g^{H(m)} g^{\hat{r}x}\right)^{t} \bmod p$$

$$\Rightarrow\ g^{k} \equiv \left(g^{H(m)} h^{\hat{r}}\right)^{t} \bmod p$$

$$\Rightarrow\ g^{k} \bmod p = \left(g^{H(m)} h^{\hat{r}}\right)^{t} \bmod p$$

$$\Rightarrow\ \underbrace{(g^{k} \bmod p) \bmod q}_{\hat{r}} = \left((g^{H(m)} h^{\hat{r}})^{t} \bmod p\right) \bmod q$$

Remarks on DSA

- SHA-1 is suggested for use as the hash function.
- $\|q\| = 160$ bits, $\|p\| = 512 + 64t,\ 0 \le t \le 8.$
- Because a hash is used, there is no restriction on *m*.
- Why use $\langle g \rangle$ instead of $\mathbb{Z}_p^*$?
- $\langle g \rangle$ is a subgroup of $\mathbb{Z}_p^*$ of order *q*.
- Shorter message length: $|(\hat{r}, s)| = 2\|q\|$ bits, rather than $2\|p\|$.
- Why not work in $\mathbb{Z}_q^*$? The message length would be $2\|q\|$, too.
- Reason: The Index Calculus method works for $\mathbb{Z}_q^*$ but not for $\langle g \rangle$.