da chi proviene un messaggio?

- in un crittosistema simmetrico solo Alice e Bob conoscono la chiave
- se Bob riceve un messaggio di Alice e la decifratura del messaggio ha senso, il messaggio proviene certamente da Alice
- in un crittosistema a chiave pubblica, chiunque può scrivere un messaggio cifrato a Bob affermando di essere Alice
- serve una firma digitale

firma "manuale"

- associa un documento a un utente firmatario
- la firma fa fisicamente parte del documento
- la firma viene verificata confrontandola con una firma campione depositata
- dovrebbe essere difficile da falsificare
- è vincolante dal punto di vista legale (contratti etc.)

firma digitale - differenze

- deve sempre associare un utente a un documento - tramite una stringa digitale
- c'è bisogno di un metodo che leghi la firma al documento
- ci vuole un algoritmo pubblico di verifica – previene la falsificazione
- una copia di un documento digitale è uguale all'originale – bisogna evitare che una firma sia riutilizzabile

signature scheme

- Alice firma un messaggio da mandare a Bob
- ci sono due componenti: un algoritmo sig per firmare e un algoritmo ver per verificare
- quello per firmare dev'essere privato (solo Alice può firmare)
- quello per verificare dev'essere pubblico (Bob - e chiunque altro - può verificare che viene da Alice)
- per firmare il messaggio x Alice usa l'alg sig_k che dipende da una chiave k, e calcola $\mathrm{sig}_k(x) = y$ (lo stesso messaggio può avere diverse firme)
- data una coppia (x, y) dove x è il messaggio e y la firma, l'algoritmo $\mathrm{ver}_k(x, y)$ dà in output vero se y è una firma valida di x, falso altrimenti
- in questo momento, non si chiede che (x, y) sia cifrato

definizione formale

Uno schema di firma è una 5-pla $(\mathcal{P}, \mathcal{A}, \mathcal{K}, \mathcal{S}, \mathcal{V})$ dove

1. $\mathcal{P}$ è un insieme finito di possibili messaggi
2. $\mathcal{A}$ è un insieme finito di possibili firme
3. $\mathcal{K}$, lo spazio delle chiavi, è un insieme finito di possibili chiavi
4. $\forall k \in \mathcal{K}$ c'è un algoritmo di firma $\mathrm{sig}_k \in \mathcal{S}$ e un corrispondente algoritmo di verifica $\mathrm{ver}_k \in \mathcal{V}$.
5. $\mathrm{sig}_k : \mathcal{P} \to \mathcal{A}$ e $\mathrm{ver}_k : \mathcal{P} \times \mathcal{A} \to \{V, F\}$ sono funzioni tali che $\forall x \in \mathcal{P}$ e $\forall y \in \mathcal{A}$ vale

$$\mathrm{ver}_k(x, y) = \begin{cases} V & \text{se } y = \mathrm{sig}_k(x), \\ F & \text{se } y \neq \mathrm{sig}_k(x) \end{cases}$$

procedura di firma usando un PKCS

- l'algoritmo per firmare sig_k dev'essere privato (solo Alice può firmare)
- l'algoritmo per verificare ver_k dev'essere pubblico (Bob - e chiunque altro - può verificare che viene da Alice)
- idea: usare un CS a chiave pubblica (deterministico)
- Alice ha la chiave k_A, e_{k_A} è pubblica e d_{k_A} è privata
- Alice firma il messaggio x ponendo $\mathrm{sig}_{k_A}(x) = y = d_{k_A}(x)$ (è l'unica che può decifrare)
- invia la coppia (x, y)
- Bob (e chiunque altro) calcola $e_{k_A}(y)$
- se $x = e_{k_A}(y)$, allora $\mathrm{ver}_{k_A}(x, y) = V$

schema di firma RSA

- Sia $N = pq$, p, q primi. Sia $\mathcal{P} = \mathcal{A} = \mathbb{Z}_N$.
- Lo spazio delle chiavi è

$$\mathcal{K} = \{(N, p, q, d, e) \mid ed \equiv 1 \pmod{\phi(N)}\}.$$

- N e e sono la chiave pubblica, p, q, d sono la chiave privata
- Se $k = (N, p, q, d, e)$ è una chiave, poniamo
- $\mathrm{sig}_k(x) \equiv x^d \pmod N$
- $\mathrm{ver}_k(x, y) = V \iff x \equiv y^e \pmod N$

esempio

- la chiave RSA di Alice è $k_A = (N_A, p_A, q_A, d_A, e_A) = (2773, 47, 59, 17, 157)$, $\quad \phi(N_A) = 2668$
- Alice vuole firmare e trasmettere il messaggio $x = 920$
- usa la chiave privata $d_A = 17$ e calcola $\mathrm{sig}_{k_A}(920) = 920^{17} \equiv 948 \pmod{2773}$
- la coppia (messaggio, firma) è quindi $(920, 948)$
- Bob riceve $(x, y) = (920, 948)$
- verifica la firma usando la chiave pubblica di Alice $e_A = 157$ $948^{157} \equiv 920 \pmod{2773} \Rightarrow \mathrm{ver}_{k_A}(920, 948) = V$

combinare firma e cifratura

- Alice ha il messaggio x da inviare a Bob
- firma e ottiene $y = \text{sig}_{k_A}(x)$
- cifra (x, y) usando e_{k_B} – ottiene $z = e_{k_B}(x, y)$
- Alice invia a Bob il testo cifrato z
- Bob decifra usando la d_{k_B} e riottiene (x, y)
- poi usa l'algoritmo di verifica ver_{k_A} per controllare se $\text{ver}_{k_A}(x, y) = V$

esempio con lo schema RSA

- la chiave di Alice è $k_A = (N_A, p_A, q_A, d_A, e_A) = (2773, 47, 59, 17, 157), \quad \phi(N_A) = 2668$
- la chiave di Bob è $k_B = (N_B, p_B, q_B, d_B, e_B) = (1073, 29, 37, 25, 121), \quad \phi(N_B) = 1008$
- Alice vuole firmare e trasmettere il messaggio $x = 920$
- usa la chiave privata $d_A = 17$ e firma $\text{sig}_{k_A}(920) = 948$
  la coppia (messaggio, firma) è quindi $(920, 948)$
- cifra con la chiave pubblica di Bob $e_B = 121$
- il testo cifrato è $z = (920^{121}, 948^{121}) = (246, 23) \pmod{1073}$
- Bob riceve $z = (246, 23)$ – decifra usando la sua chiave privata $d_B = 25$
- ritrova $(x, y) = (246^{25}, 23^{25}) = (920, 948)$
- verifica la firma usando la chiave pubblica di Alice $e_A = 157$
  $948^{157} \equiv 920 \pmod{2773} \Rightarrow \text{ver}_{k_A}(920, 948) = V$

prima firmare, poi cifrare

- l'ordine giusto è prima firmare e poi cifrare
- se Alice prima cifra e poi firma, Eve può convincere Bob di essere il mittente
- se x è il messaggio, Alice cifra $z = e_{k_B}(x)$ e firma $y = \mathrm{sig}_{k_A}(z)$
- manda la coppia (z, y) a Bob
- se Eve intercetta la trasmissione, è in grado di firmare il messaggio z, anche se non può decifrarlo
- Eve può calcolare $\tilde{y} = \mathrm{sig}_{k_E}(z)$ e inviare la coppia $(z, \tilde{y})$
- il messaggio passa la verifica di Bob, che lo accetta come proveniente da Eve

falsificazione

- dato $x \in \mathcal{P}$, dev'essere computazionalmente difficile per chi non è Alice calcolare una firma y tale che $\mathrm{ver}_{k_A}(x, y) = V$
- una coppia (x,y) tale che $\mathrm{ver}_{k_A}(x, y) = V$ che non è stata prodotta da Alice (ma da Eve, per esempio) si dice una falsificazione
- usando lo schema RSA (e anche usando un altro PKCS deterministico)
- è difficile, dato il messaggio x produrre una falsificazione (x, y)
- è però facile data y produrre una falsificazione (x, y)

- Eve può falsificare la firma di Alice
- sceglie y e pone $x = e_{k_A}(y)$
- la coppia (x, y) passa la verifica "per costruzione"
- $\text{ver}_k(x, y) = V \iff x = e_{k_A}(y)$
- una falsificazione di questo tipo si chiama falsificazione esistenziale (existential forgery)
- Eve non può scegliere il messaggio x e produrre una firma valida y (una falsificazione di questo tipo si chiama falsificazione scelta (selective forgery))

- Eve vuole ottenere la firma di Alice su un messaggio x da lei scelto
- Alice non firmerebbe mai x
- Eve trova x_1, x_2 tali che $x \equiv x_1 \cdot x_2 \pmod{N}$
- chiede a Alice di firmare x_1 e x_2, e ottiene y_1, y_2
- per le proprietà moltiplicative dell'RSA si ha che

$$\text{ver}_k(x_1 x_2 \bmod N, y_1 y_2 \bmod N) = V$$

funzione hash

- per evitare falsificazioni, si usano schemi di firma insieme a funzioni hash
- molto informalmente, una funzione hash $h : \mathcal{P} \to \mathcal{D}$ è una funzione unidirezionale
- h(x) si dice digest del messaggio x
- può/deve avere molte altre proprietà che non discutiamo

schemi di firma e funzioni hash

- Alice deve firmare il messaggio x
- calcola $h(x)$
- firma il message digest $h(x)$, non x:
  $y = \text{sig}_{k_A}(h(x))$
- Bob riceve (x, y) – per prima cosa, calcola $h(x)$
- poi controlla che $\text{ver}_{k_A}(h(x), y) = V$
- "intuitivamente" questo impedisce le falsificazioni
- Eve sceglie y, calcola $e_{k_A}(y)$ - per produrre una coppia valida, deve trovare $h^{-1}(e_{k_A}(y))$
- lo schema RSA viene sempre usato insieme a una funzione hash

ripetizione: crittosistema Elgamal

- sia p un primo, g un elemento primitivo mod p
- $\mathcal{P} = U(\mathbb{Z}_p)$
- $\mathcal{C} = U(\mathbb{Z}_p) \times U(\mathbb{Z}_p)$
- Lo spazio delle chiavi è

$$\mathcal{K} = \{(p, g, a, \beta) \mid \beta \equiv g^a \pmod{p}\}.$$

- p, g e β sono la chiave pubblica a è la chiave privata

crittosistema Elgamal

- prima di cifrare il messaggio $x \in \mathcal{P}$, Bob sceglie un numero casuale (segreto) $h \in \{2, \ldots, p-2\}$
- $e_k(x, h) = (y_1, y_2)$
- con $y_1 = g^h$, $y_2 = x\beta^h \pmod{p}$
- Alice riceve $(y_1, y_2) \in U(\mathbb{Z}_p) \times U(\mathbb{Z}_p)$ – non conosce h ma conosce a
- calcola $y_1^a = (g^h)^a = (g^a)^h = \beta^h \pmod{p}$
- calcola $(\beta^h)^{-1} \pmod{p}$ e ottiene $x = y_2(\beta^h)^{-1}$
- $d_k((y_1, y_2)) = y_2(y_1^a)^{-1} \pmod{p}$
- notare la somiglianza con DH – non si inverte la funzione $x \to g^x \pmod{p}$
- Bob sceglie un nuovo h a ogni trasmissione

schema di Elgamal

- Alice ha come chiave pubblica Elgamal $k_A = (p, g, \beta)$
  p primo, g elemento primitivo, $\beta = g^a$, a chiave privata
- per firmare $x \in U(\mathbb{Z}_p)$, Alice sceglie h casuale con $(h, p-1) = 1$
- calcola l'inverso l di h modulo $p-1$, quindi $lh \equiv 1 \bmod p-1$
- calcola $z_1 = g^h \bmod p$ e $z_2 = (x - az_1)l \bmod p-1$
- il messaggio firmato è (x, z_1, z_2)
- la verifica funziona – la firma è valida – se $\beta^{z_1} z_1^{z_2} \equiv g^x \bmod p$
- $\beta^{z_1} z_1^{z_2} = g^{az_1} g^{hl(x-az_1)} \bmod p$
- "quindi" $\beta^{z_1} z_1^{z_2} \equiv g^x \bmod p$

esempio

- la chiave pubblica di Alice è $(107, 2, 15)$ e la chiave privata è 11
- quindi $2^{11} \equiv 15 \pmod{107}$
- Alice vuole firmare e trasmettere il messaggio $x = 10$
- sceglie il numero casuale 7
- $(7, 106) = 1$ e $7 \cdot 25 \equiv 1 \ (\bmod 106)$
- la firma di x è (z_1, z_2) con $z_1 = 2^7 \equiv 104 \pmod{107}$
- $z_2 = (10 - 11 \cdot 104) \cdot 25 \equiv 58 \pmod{106}$
- il messaggio firmato è $(10, 104, 58)$
- $\beta^{z_1} z_1^{z_2} = 15^{104} \cdot 104^{58} \equiv 61 \pmod{107}$
- $g^x = 2^{10} \equiv 61 \pmod{107}$
- la verifica ha successo.