

## logaritmo discreto come funzione unidirezionale

- in generale, lavoreremo con il gruppo  $U(\mathbb{Z}_p) = \mathbb{Z}_p^*$
- dati  $g$  generatore di  $\mathbb{Z}_p^*$  e  $x$  tale che  $1 \leq x \leq p-1$ , calcolare  $y = g^x$  è computazionalmente facile
- ( $y \equiv g^x \pmod{p}$ ) – si usa l'algoritmo square-and-multiply )
- si ritiene che, dati  $g$  generatore di  $\mathbb{Z}_p^*$  e  $y \in \mathbb{Z}_p^*$ , determinare  $x = \log_g y$  sia difficile (sotto opportune ipotesi su  $p$ )
- in particolare,  $p$  devessere grande (1024 bit),  $p \approx 2^{1023}$
- dati  $g$  e  $y$  posso trovare  $x$  tale che  $g^x = y$  per tentativi – calcolando  $g^x$  per tutti gli  $x$ ,  $1 \leq x \leq p-1$
- ma il numero di tentativi è enorme

## cifratura RSA e logaritmo discreto

- nella cifratura RSA, la funzione è del tipo

$$x \longrightarrow x^e \pmod{N}$$

$$f(x_1 \cdot x_2) = f(x_1) \cdot f(x_2)$$

- nel problema del logaritmo discreto, la funzione è del tipo

$$x \longrightarrow g^x \pmod{p}$$

$$f(x_1 + x_2) = f(x_1) \cdot f(x_2)$$

## scambio della chiave di Diffie-Hellman

- Alice e Bob scelgono pubblicamente un primo  $p$  e un elemento primitivo  $g \pmod{p}$
- Alice sceglie casualmente  $a \in \{2, \dots, p-2\}$ ; calcola  $g^a \pmod{p}$  e invia il risultato a Bob
- Bob sceglie casualmente  $b \in \{2, \dots, p-2\}$ ; calcola  $g^b \pmod{p}$  e invia il risultato a Alice
- Alice calcola  $(g^b)^a \pmod{p}$
- Bob calcola  $(g^a)^b \pmod{p}$
- la chiave è  $k = g^{ab}$

## DH problem

- se Eve sa risolvere il problema del logaritmo discreto, sa ricavare la chiave comune di Bob e Alice
- dall'osservazione di  $g^a, g^b \pmod{p}$  ricava  $a$  e  $b$ , quindi calcola  $k = g^{ab}$
- **DH problem:** dato un gruppo ciclico  $G$ ,  $g$  un generatore e dati  $g^a, g^b$  trovare  $g^{ab}$
- basta che sappia risolvere il DH problem per trovare la chiave
- equivalenza DH - DL?

## crittosistema Elgamal (ca 1985)

- 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  $\beta$  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)  $h \in \{2, \dots, p-2\}$**
- $e_k(x, h) = (y_1, y_2)$
- 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  $x \rightarrow g^x \pmod{p}$
- Bob sceglie un nuovo  $h$  a ogni trasmissione (randomized encryption)

## esempio

- Alice sceglie  $p = 83$ ,  $g = 2$  e la chiave privata  $a = 30$
- $\beta = 2^{30} \equiv 40 \pmod{83}$
- la chiave pubblica di Alice è  $(83, 2, 40)$
- il messaggio di Bob è  $x = 54$  – il numero scelto per la cifratura è  $h = 13$
- Bob invia  $(g^h, x\beta^h) = (2^{13}, 54 \cdot 40^{13}) \equiv (58, 71) \pmod{83}$
- per decifrare, Alice calcola  $(g^a)^h = 58^{30} = 9$
- l'inverso di 9 mod 83 è 37 – il messaggio è quindi  $37 \cdot 71 = 54 \pmod{83}$

- la “cifratura” è una moltiplicazione:  $x \rightarrow xg^{ha}$ 
  - si può invece usare un cifrario a blocchi (AES) e cifrare  $x \rightarrow e_{g^{ha}}(x)$
- anche per Elgamal, si usa  $p$  di almeno 1024 bit,  $p - 1$  con un fattore primo grande e a fattorizzazione nota
- anche per violare Elgamal, basta che Eve sappia risolvere il DH problem
- se dati  $g^h$  e  $\beta = g^a$  sa trovare  $g^{ah} = \beta^h$ , può leggere il messaggio
- se Eve sa risolvere il logaritmo discreto può ricavare l'esponente  $h$  e quindi ricavare direttamente  $x$
- oppure ricavare  $a$  e decifrare come Alice (conviene –  $h$  cambia in ogni trasmissione)

## 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 **leggi** 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  $\text{sig}_k$  che dipende da una chiave  $k$ , e calcola  $\text{sig}_k(x) = y$  (lo stesso messaggio può avere diverse firme)
- data una coppia  $(x, y)$  dove  $x$  è il messaggio e  $y$  la firma, l'algoritmo  $\text{ver}_k(x, y)$  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  $\text{sig}_k \in \mathcal{S}$  e un corrispondente algoritmo di verifica  $\text{ver}_k \in \mathcal{V}$ .
- ⑤  $\text{sig}_k : \mathcal{P} \rightarrow \mathcal{A}$  e  $\text{ver}_k : \mathcal{P} \times \mathcal{A} \rightarrow \{V, F\}$  sono funzioni tali che  $\forall x \in \mathcal{P}$  e  $\forall y \in \mathcal{A}$  vale

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

## procedura di firma usando un PKCS

- l'algoritmo per firmare  $\text{sig}_k$  dev'essere **privato** (solo Alice può firmare)
- l'algoritmo per verificare  $\text{ver}_k$  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  $\text{sig}_{k_A}(x) = y = d_{k_A}(x)$  (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  $\text{ver}_{k_A}(x, y) = V$

## 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
- $\text{sig}_k(x) \equiv x^d \pmod{N}$
- $\text{ver}_k(x, y) = V \iff x \equiv y^e \pmod{N}$

## esempio

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



## 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  $\text{ver}_{k_A}$  per controllare se  $\text{ver}_{k_A}(x, y) = V$

## esempio con lo schema RSA

- la chiave di Alice è  $k_A = (N_A, p_A, q_A, d_A, e_A) = (2773, 47, 59, 17, 157)$ ,  $\phi(N_A) = 2668$
- la chiave di Bob è  $k_B = (N_B, p_B, q_B, d_B, e_B) = (1073, 29, 37, 25, 121)$ ,  $\phi(N_B) = 1008$
- 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  $y = \text{sig}_{k_A}(z)$
- 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  $\tilde{y} = \text{sig}_{k_E}(z)$  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  $\text{ver}_{k_A}(x, y) = V$
- una coppia  $(x, y)$  tale che  $\text{ver}_{k_A}(x, y) = V$  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

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