

gruppi ciclici

- Definizione

Un gruppo G con n elementi tale esiste un elemento $g \in G$ con $o(g) = n$ si dice *ciclico*, e g si dice un *generatore* del gruppo

- $U(\mathbb{Z}_9)$ è ciclico – p. es. $U(\mathbb{Z}_8)$ non lo è
- i gruppi $U(\mathbb{Z}_p)$, p primo, sono gruppi ciclici
- se $GF(p^m)$ è un campo finito, il gruppo moltiplicativo $(GF(p^m) - \{0\}, \cdot)$ è un gruppo ciclico
- in questi due esempi, i generatori si chiamano *radici primitive*, o elementi primitivi

logaritmo discreto

- sia G un gruppo ciclico di ordine n , sia g un generatore di G
- dato $y \neq 1 \in G$
- bisogna determinare l'*unico* intero x con $1 \leq x \leq n - 1$ tale che

$$g^x = y$$

- ex: in $U(\mathbb{Z}_9)$ con $g = 2$, se $y = 7$ si ha $x = 4$, perché $2^4 = 7$.
- l'intero x si chiama il *logaritmo discreto* di y in base g , e si denota con $\log_g y$
- ex: in $U(\mathbb{Z}_9)$, $4 = \log_2 7$

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 (2048 bit), $p \approx 2^{2047}$
- 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

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

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

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

protocollo di scambio della chiave

- Alice e Bob non condividono informazioni segrete
- eseguono un protocollo, e alla fine hanno la stessa chiave
- Eve ascolta la comunicazione, ma non ottiene nessuna informazione sulla chiave

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 $k = g^{ab}$

- Esempio: $p = 23, g = 5$
- Alice sceglie $a = 6$ $g^a = 5^6 \equiv 8 \pmod{23}$
- Bob sceglie $b = 15$ $g^b = 5^{15} \equiv 19 \pmod{23}$
- Alice calcola $(g^b)^a = 19^6 \equiv 2 \pmod{23}$
- Bob calcola $(g^a)^b = 8^{15} \equiv 2 \pmod{23}$

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?

How hard is the DH function mod p ?

Suppose prime p is n bits long.

Best known algorithm (GNFS): run time $\exp(\tilde{O}(\sqrt[3]{n}))$

<u>cipher key size</u>	<u>modulus size</u>	<u>Elliptic Curve size</u>
80 bits	1024 bits	160 bits
128 bits	3072 bits	256 bits
256 bits (AES)	<u>15360</u> bits	512 bits

As a result: slow transition away from (mod p) to elliptic curves

- per implementare il protocollo, bisogna essere in grado di produrre numeri primi grandi
- e dato un tale primo p , di trovare una radice primitiva g modulo p
- sicurezza: p almeno 2048 bit, $p - 1$ con un fattore primo grande
- si cerca anche un $p - 1$ a fattorizzazione nota (per trovare facilmente g)
- spesso si sceglie $p = 2q + 1$, q un primo
- la fattorizzazione è $p - 1 = 2q$
- un numero primo q tale che anche $2q + 1$ è primo si chiama **primo di Sophie Germain**
- **osservazione:** la funzione unidirezionale $x \rightarrow g^x \pmod{p}$ **non** ha una trapdoor

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 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) $h \in \{2, \dots, p-2\}$**
- $e_k(x, h) = (y_1, y_2)$
- 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 $(g^h)^a = 58^{30} = 9$
 - 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 2048 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

schemi di firma

- 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 privata k , e calcola $\text{sig}_k(x) = y$
- 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

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

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