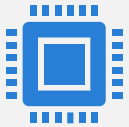
Algoritmo di firma digitale ECDSA

Cos'è l'ECDSA?

L'ECDSA (Elliptic Curve Digital Signature Algorithm) è un algoritmo di firma digitale che nasce come variante dello standard DSA.



Viene proposto per la prima volta nel 1992 e diventa uno standard ISO nel 1998 e IEEE nel 2000.

Offre come particolarità, rispetto al DSA, l'utilizzo di crittografia ellittica, un tipo di crittografia a chiave pubblica.

Cos'è un algoritmo di firma digitale?

La firma digitale è un metodo matematico impiegato per analizzare un messaggio o un documento digitale inviato su un canale di comunicazione non sicuro, con lo scopo di verificare la sua autenticità.

Esso infatti, rispetto a un determinato messaggio, garantisce:

- L'autenticazione, per verificare l'identità del mittente;
- Il non ripudio, affinché il mittente non possa negare di averlo inviato;
- L'integrità, col fine di essere sicuri che il messaggio non sia stato alterato lungo il percorso.

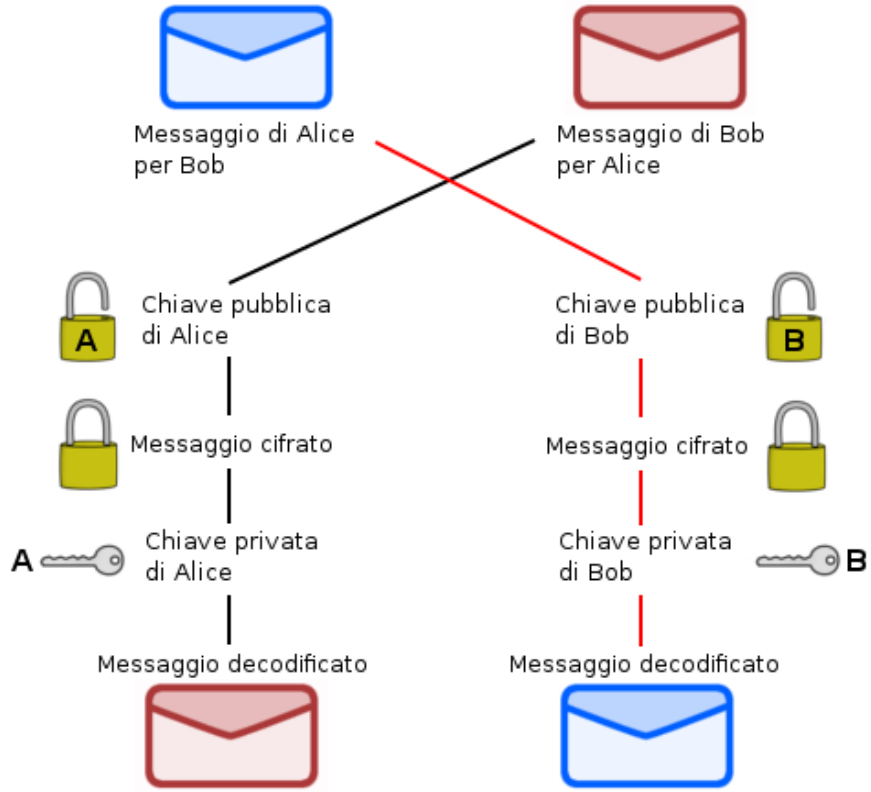
Cos'è la crittografia a chiave pubblica?

La crittografia a chiave pubblica (o asimmetrica) è un tipo di crittografia che associa a ogni attore coinvolto nella comunicazione una coppia di chiavi: quella pubblica e quella privata.

La prima, di cui chiunque è a conoscenza, permette di cifrare un messaggio che può essere decifrato solo dall'attore che possiede la seconda, che invece non viene distribuita.

Questo meccanismo di crittografia riesce così a evitare qualsiasi problema legato alla ricerca di un canale sicuro al fine di scambiare la chiave.

La crittografia ellittica possiede come particolarità l'impiego di curve ellittiche definite su campi finiti (o campi di Galois).



Campi finiti

Sono strutture algebriche $<A, +, *>$ (definite a partire dagli anelli) in cui:

- A rappresenta il sostegno della struttura, cioè un insieme (in questo caso finito);
- + e * rappresentano due operazioni binarie sul sostegno della struttura.

Inoltre:

- $<A, +>$ e $<A \setminus \{0\}, *>$ sono gruppi commutativi;
- Vale la proprietà distributiva di * rispetto a +.

L'ordine di un campo finito è definito dal numero di elementi del campo. Ogni campo finito K ha ordine p^n, dove p è la caratteristica di K ed n un intero positivo.

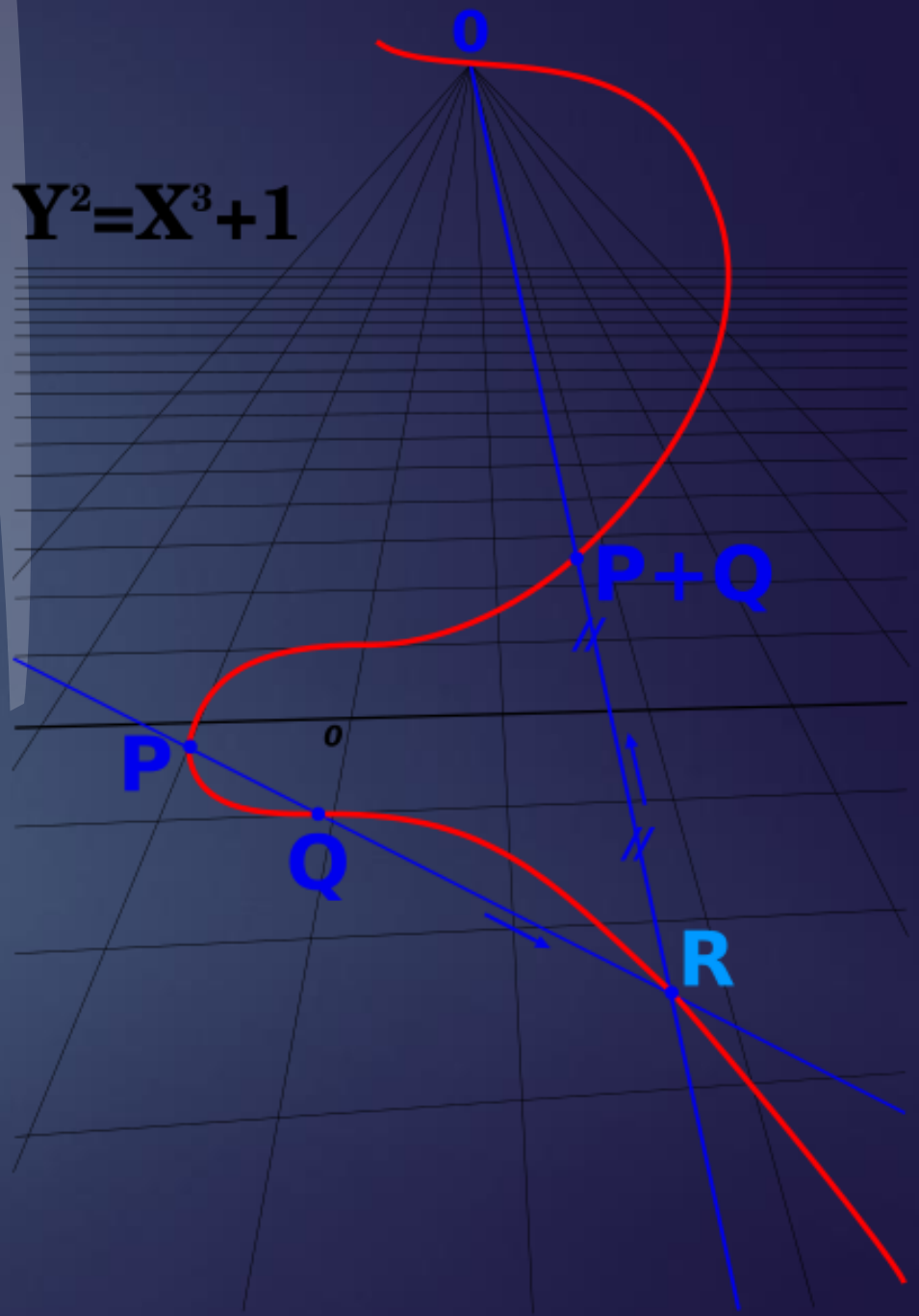
Notoriamente, in crittografia ellittica, ci si limita a utilizzare campi finiti di ordine p (F_p) e di ordine 2^m (F_{2^m}).

Curve ellittiche

- Le curve ellittiche sono curve piane definite su un campo F_p, sulle quali viene specificato un punto O (chiamato punto all'infinito).

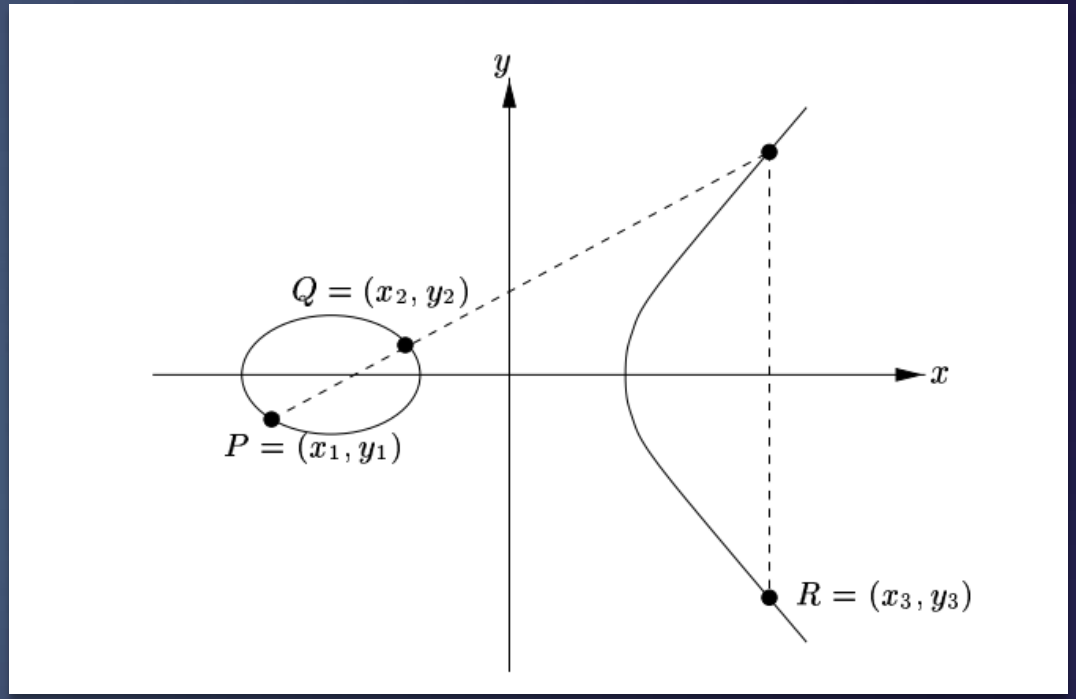
- Ogni curva ellittica E sul campo F_p (con caratteristica diversa da 2 e da 3) può essere scritta come una curva algebrica piana in base all'equazione di Weierstrass:
$y^2 = x^3 + ax + b$,
con a, b ∈ F_p e $4a^3 + 27b^2 \neq 0 \pmod p$.

- L'insieme $E(F_p)$ consiste di tutti i punti (x,y), con x, y ∈ F_p, che soddisfano l'equazione di Weierstrass, oltre al punto O.



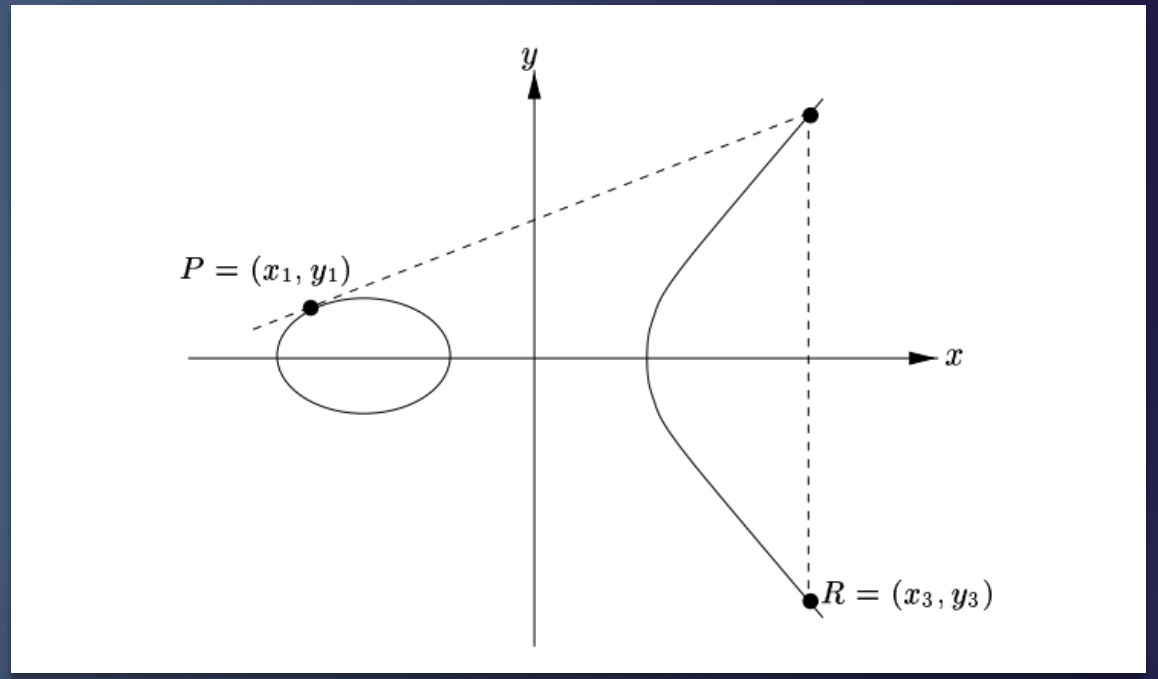
Attraverso la regola chord-and-tangent (corda e tangente) è possibile sommare due punti sulla curva ellittica $E(F_p)$….

Dati due punti distinti $P=(x_1, y_1)$ e $Q=(x_2, y_2)$ sulla curva ellittica $E(F_p)$, la somma di P e Q sarà denominata $R=(x_3, y_3)$ e sarà determinata dal riflesso, rispetto all'asse x, del punto intersezione della retta PQ con la curva ellittica.



…e trovare il doppio di un punto P

Dato il punto $\mathrm{P} = (\mathrm{x}_1, \mathrm{y}_1)$, il doppio di P, denotato come $\mathrm{R} = (\mathrm{x}_3, \mathrm{y}_3)$, sarà determinato dal riflesso, rispetto all'asse x, dell'intersezione tra la tangente alla curva in P e la curva ellittica.



Come funziona l'algoritmo ECDSA?

Si suppone che Alice voglia mandare a Bob un messaggio protetto da firma digitale.

Innanzitutto, ci si deve accordare su tre parametri:

- *CURVE*: campo ed equazione della curva ellittica usata;
- *G*: punto base della curva, generatore della curva ellittica avente come ordine un primo grande n;
- *n*: ordine intero di G, tale per cui $\mathrm{n} \times \mathrm{G} = \mathrm{O}$.

Successivamente non resta che generare le chiavi:

- Chiave privata $\mathrm{d_A}$, scelta casualmente nell'intervallo [1, n-1];
- Chiave pubblica $\mathrm{Q_A} = \mathrm{d_A} \times \mathrm{G}$.

Generazione della firma

A questo punto, Alice ha tutte le informazioni necessarie per la generazione della firma del messaggio m:

1. Si computa $e = \mathrm{HASH}(\mathrm{m})$;
2. Si ottiene la stringa z prendendo gli $\mathrm{L_n}$ bit più a sinistra di e, sapendo che $\mathrm{L_n}$ è la lunghezza in bit del gruppo di ordine n;
3. Si seleziona casualmente, ma in modo crittografico-sicuro, un intero k dall'intervallo [1, n-1];
4. Si calcola il punto della curva $(\mathrm{x_1}, \mathrm{y_1}) = \mathrm{k\ x\ G}$;
5. Si calcola $\mathrm{r} \equiv \mathrm{x_1} \bmod \mathrm{n}$.
   Se r = 0, l'assegnamento non è valido e occorre tornare al passo 3;
6. Si calcola $\mathrm{s} \equiv \mathrm{k^{-1}}\ (\mathrm{z} + \mathrm{r}\ \mathrm{d_A}) \bmod \mathrm{n}$.
   Se s = 0, l'assegnamento non è valido e occorre tornare al passo 3;
7. La firma è la coppia (r, s).

Attacco del nonce ripetuto

Nella generazione di firme diverse, è cruciale fare attenzione alla scelta dell'intero k (passo 3).

Infatti, date due firme (r, s) ed (r, s'), utilizzare lo stesso k per due messaggi differenti m ed m' significa aprirsi ad una vulnerabilità ad attacchi. Questo accade perché, nella condizione di nonce non ripetuto, un eventuale aggressore può facilmente ricondursi dal calcolo di z e z' alla chiave privata d_A.

Infatti, poiché $s - s' = k^{-1} (z - z') \bmod n$, di conseguenza $k = \frac{z - z'}{s - s'} \bmod n$. E dato che $s = k^{-1} (z + r\, d_A)$ [passo 6 della slide precedente], a questo punto per l'aggressore diventa semplice calcolare la chiave privata $d_A = \frac{sk - z}{r}$.

Verifica della firma

Una volta ricevuta la firma di Alice, Bob può decidere di autenticarla, e per farlo ha bisogno di una copia della chiave pubblica Q_A.

La prima cosa da fare per Bob è verificare che Q_A sia un punto valido della curva, controllando:

1. che Q_A abbia coordinate valide e non sia uguale all'elemento neutro O;
2. che Q_A appartenga alla curva;
3. che $n \times Q_A = O$;

A questo punto, non resta che verificare la validità della firma attraverso i seguenti passi:

1. Si verifica che r e s siano interi $\in [1, n-1]$.
   Altrimenti la firma non è valida.
2. Si computa $e = HASH(m)$, dove HASH è la stessa funzione impiegata nella generazione della firma;
3. Si ottiene la stringa z prendendo gli L_n bit più a sinistra di e;
4. Si calcola $w = s^{-1} \bmod n$;
5. Si calcolano $u_1 = zw \bmod n$ e $u_2 = rw \bmod n$;
6. Si calcola il punto della curva $C = (x_1, y_1) = u_1 \times G + u_2 \times Q_A$;
7. Si verifica che la firma sia valida controllando se vale $r \equiv x_1 \pmod n$.
   In caso contrario, la firma non è accettata.

Correttezza dell'algoritmo di verifica della firma

Al fine di verificare l'effettivo funzionamento dell'algoritmo di firma, si seguono i seguenti passi:

Sapendo che C è il punto calcolato al passo 6 della verifica

$$C = u_1 \text{ x } G + u_2 \text{ x } Q_A;$$

Sostituendo la definizione della chiave pubblica $Q_A = d_A \text{ x } G$

$$C = u_1 \text{ x } G + u_2 \, d_A \text{ x } G;$$

Applicando la proprietà distributiva alla moltiplicazione di un punto della curva ellittica per uno scalare

$$C = (u_1 + u_2 \, d_A) \text{ x } G;$$

Espandendo la definizione di u_1 e u_2 dal passo 5 dell'algoritmo di verifica

$$C = (zs^{-1} + r \, d_A \, s^{-1}) \text{ x } G;$$

Raccogliendo s^{-1}:

$$C = (z + r\,d_A)\,s^{-1}\,x\,G;$$

Espandendo la definizione di s dal passo 6 dell'algoritmo di generazione della firma

$$C = (z + r\,d_A)\,(z + r\,d_A)^{-1}\,(k^{-1})^{-1}\,x\,G;$$

Da cui

$$C = k\,x\,G,$$

che coincide con il punto (x_1, y_1) calcolato nella generazione della firma, pertanto si può affermare che un messaggio firmato correttamente supererà la verifica.

Considerazioni sulla sicurezza dell'algoritmo di firma

- L'impiego delle curve ellittiche permette all'algoritmo ECDSA di ridurre di molto la dimensione di una chiave pubblica sicura per cifrare rispetto al DSA. Infatti, l'ECDSA ha bisogno di una chiave che abbia come dimensione circa il doppio del livello di sicurezza in bit: per esempio, con un livello di sicurezza di 80 bit, la dimensione di una chiave pubblica ECDSA sarebbe di 160 bit, mentre per il DSA avrebbe una lunghezza di almeno 1024 bit.
- La dimensione della firma, invece, è la stessa per entrambi gli algoritmi: 4t bit, dove t è il livello di sicurezza in bit.
- L'obiettivo dell'algoritmo ECDSA è di risultare infalsificabile rispetto a un attacco a un determinato messaggio. In questa circostanza, l'avversario ha come obiettivo quello di ottenere una firma valida per una certa entità A su un singolo messaggio m, dopo aver ottenuto la firma della stessa entità A su una collezione di messaggi (tra cui non è presente m).

Possibili attacchi all'ECDSA

- Attacco al problema del logaritmo discreto per le curve ellittiche (ECDLP), attraverso alcuni algoritmi di risoluzione di questo problema:
  a. Semplice ricerca esaustiva;
  b. Algoritmo di Polig-Hellman;
  c. Algoritmo Baby-Step Giant-Step;
  d. Algoritmo di Pollard parallelizzato (opzione più efficace).
- Attacco alla funzione di hash impiegata, cercando di sfruttare le eventuali debolezze dell'algoritmo di hashing SHA-1, come:
  a. Mancata resistenza all'immagine;
  b. Mancata resistenza alle collisioni.
- Altri attacchi, come l'attacco del nonce ripetuto.