Principles of Cryptocurrency Design

Appunti ed Esercizi

Francesco Pasquale

7 maggio 2024

Possiamo descrivere una curva ellittica come l'insieme dei punti (x, y), a coordinate in un opportuno campo, che soddisfano l'equazione

$$y^2 = x^3 + ax + b \tag{1}$$

dove a e b sono parametri che definiscono la curva, con l'aggiunta di un punto che chiamiamo *punto all'infinito* $\{\infty\}$.

Esercizio 1. Scrivere un programma che prenda in input i parametri a e b e gli estremi di un intervallo di numeri reali, $[x_0, x_1] \subseteq \mathbb{R}$ e disegni il grafico della curva in (1) al variare di $x \in [x_0, x_1]$.

Per le applicazioni in crittografia non si considera il campo dei numeri reali, ma i campi finiti $\mathbb{F}_p$, dove p è un numero primo, ossia $\mathbb{F}_p$ è l'insieme dei numeri interi $\{0, 1, \ldots, p-1\}$ con le usuali operazioni di somma e prodotto, modulo p.

La curva ellittica che viene usata in Bitcoin si chiama secp256k1 ed è definita dai parametri $a = 0$, $b = 7$, $p = 2^{256} - 2^{32} - 2^9 - 2^8 - 2^7 - 2^6 - 2^4 - 1$.

Esercizio 2. Scrivere un programma per verificare che il numero p definito qui sopra è un numero primo.

Su una curva ellittica

$$\mathcal{C} = \{(x, y) \in \mathbb{F}_p^2 \,:\, y^2 = x^3 + ax + b\} \cup \{\infty\}$$

si può definire un'operazione, che chiamiamo *somma*, che a due punti sulla curva $P, Q \in \mathcal{C}$ associa un terzo punto sulla curva $P + Q \in \mathcal{C}$, in modo tale che $(\mathcal{C}, +)$ sia un gruppo.

È facile vedere che, siccome $(\mathcal{C}, +)$ è un gruppo, c'è un algoritmo polinomiale per il seguente problema computazionale
INPUT: Un punto sulla curva $G \in \mathcal{C}$ e un intero $k \in \mathbb{N}$
OUTPUT: Il punto $P \in \mathcal{C}$ tale che $P = kG$
Dove con kP intendiamo $P + P + \cdots + P$, k volte.

Esercizio 3. Descrivere un algoritmo che, assumendo che $P + P$ si possa calcolare in tempo $\mathcal{O}(1)$, calcoli kP in tempo $\mathcal{O}(\log k)$.

D'altra parte, nessuno conosce un algoritmo polinomiale per il problema computazionale inverso (Discrete Logarithm Problem)
INPUT: Due punti sulla curva $G, P \in \mathcal{C}$
OUTPUT: Un intero $k \in \mathbb{N}$ tale che $P = kG$, oppure `None` se un tale intero non esiste.

Perciò, dato un *punto base* $G \in \mathcal{C}$ si può definire una coppia di chiavi $(\texttt{sk}, \texttt{pk})$ dove la chiave segreta `sk` è un intero positivo minore di p, e la chiave pubblica `pk` è il punto sulla curva $\texttt{sk} \cdot G$

Il punto base della curva SECP256K1 è $G = (x, y)$, dove

$x =$ `0x 79BE667E F9DCBBAC 55A06295 CE870B07 029BFCDB 2DCE28D9 59F2815B 16F81798`

$y =$ `0x 483ADA77 26A3C465 5DA4FBFC 0E1108A8 FD17B448 A6855419 9C47D08F FB10D4B8`

Esercizio 4. Verificare che le coordinate (x, y) qui sopra soddisfano l'equazione $y^2 = x^3 + 7 \mod p$.

Un *indirizzo* Bitcoin relativo a uno script del tipo P2PKH è semplicemente l'hash, opportunamente calcolato, di un punto kG sulla curva ellittica SECP256K1. Nella prossima lezione vedremo i dettagli per calcolarlo.