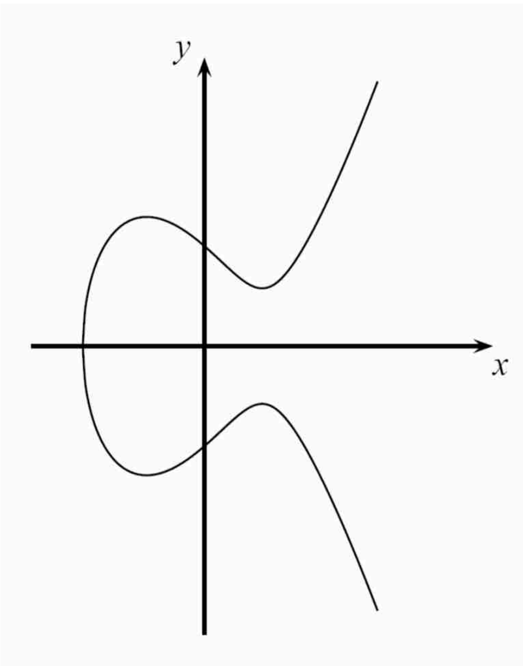
ECC è stata inventata indipendentemente da Neal Koblitz nel 1987 e nel 1986 da Victor Miller

Crittografia basata su Curve Ellitiche



ECC

Elliptic

Curve

Cryptography

Perché ECC

- Schemi asimmetrici come RSA, DEKE, El Gamal si basano sul calcolo di potenze modulari in gruppi o campi finiti con parametri di grande dimensione (più di 1000 bit)
  - Elevata complessità computazionale
  - Problemi nell'utilizzo in sistemi con potenza limitata (e.g., sistemi embedded)
- È possibile utilizzare campi più piccoli conservando lo stesso livello di sicurezza?
  - Si usano punti di una curva quali elementi del gruppo/campo rappresentabili con 160-256 bit

Curve Ellittiche

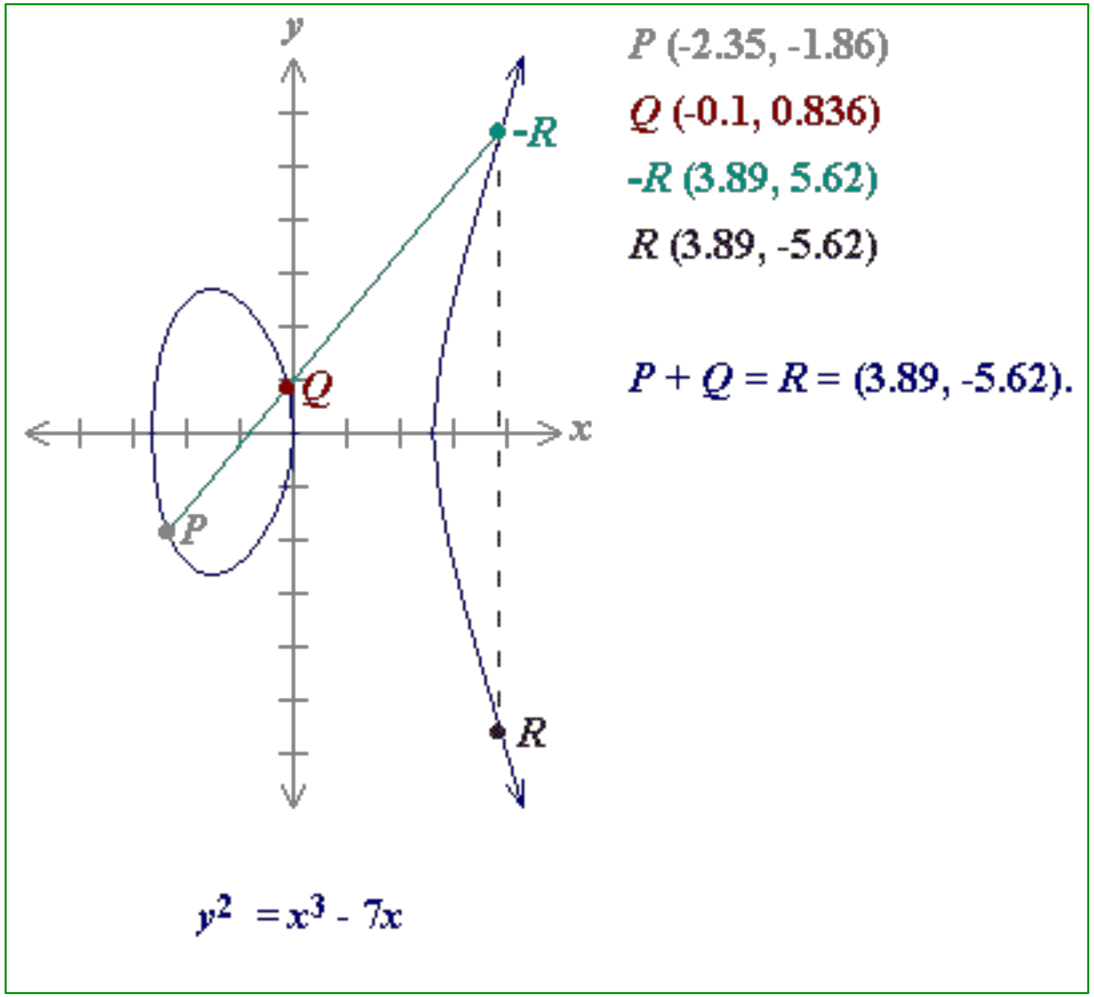
- Le curve ellittiche sono polinomi che definiscono punti basati sull'equazione semplificata di Weierstraß

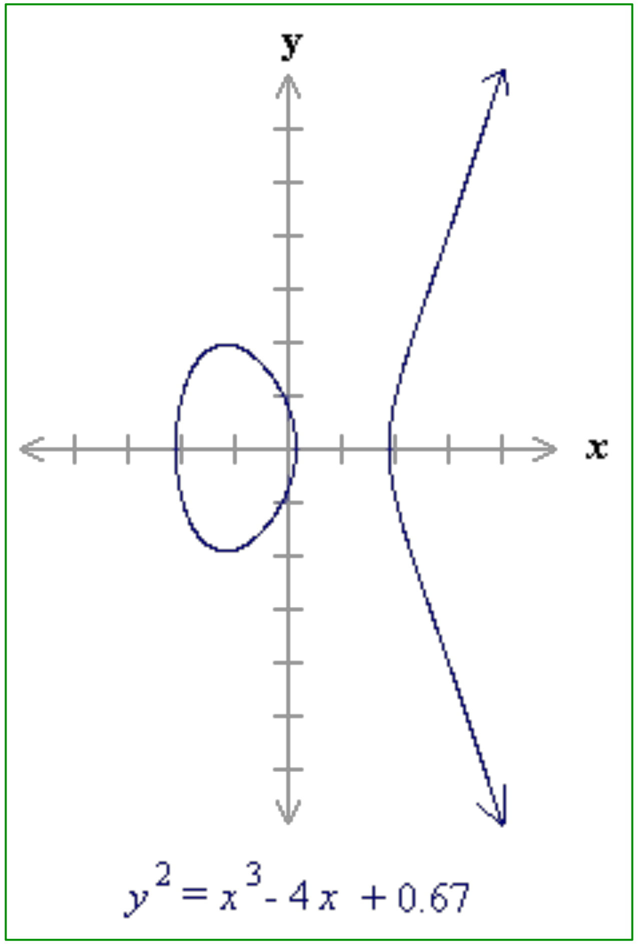
$$y^2 = x^3 + a \cdot x + b$$

- I parametri a e b specificano la forma della curva
- Le curve ellittiche non sono definite solo su R, ma anche su molti tipi di campi finiti

Esempi

La curva a destra è definita dalle coppie $(x,y) \in R^2$ che soddisfano l'equazione $y^2 = x^3 - 4x + 0.67$



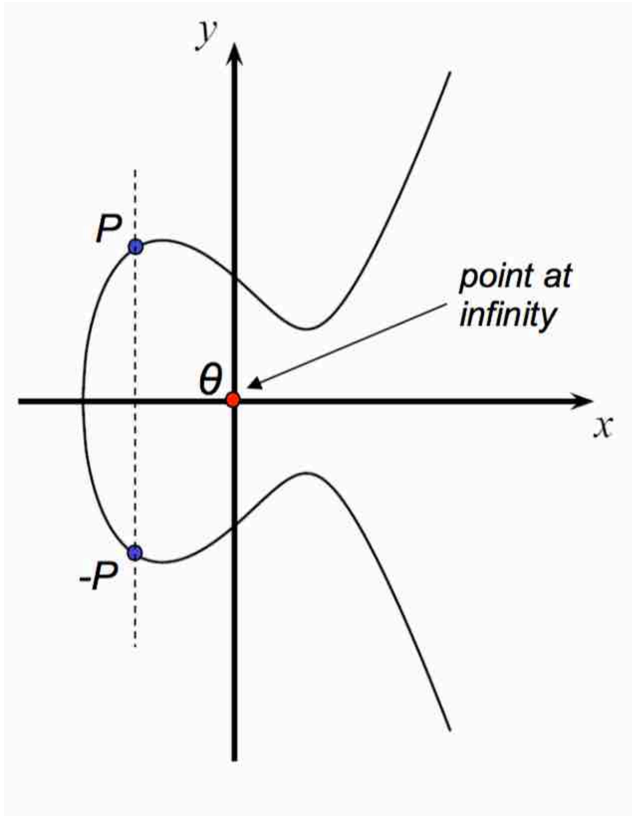


Curve ellittiche in campi finiti

- In crittografia $K=Z_p$ con p primo e maggiore di 3
- Una curva ellittica è l'insieme dei punti di un campo K che soddisfano un'equazione della forma $y^2 = x^3 + ax + b$
- Se $4a^3 + 27b^2 \neq 0 \bmod p$, allora la curva può essere utilizzata per definire un gruppo
  - Nel gruppo ci saranno i punti della curva più un elemento speciale θ detto punto all'infinito
  - θ sarà l'identità del gruppo

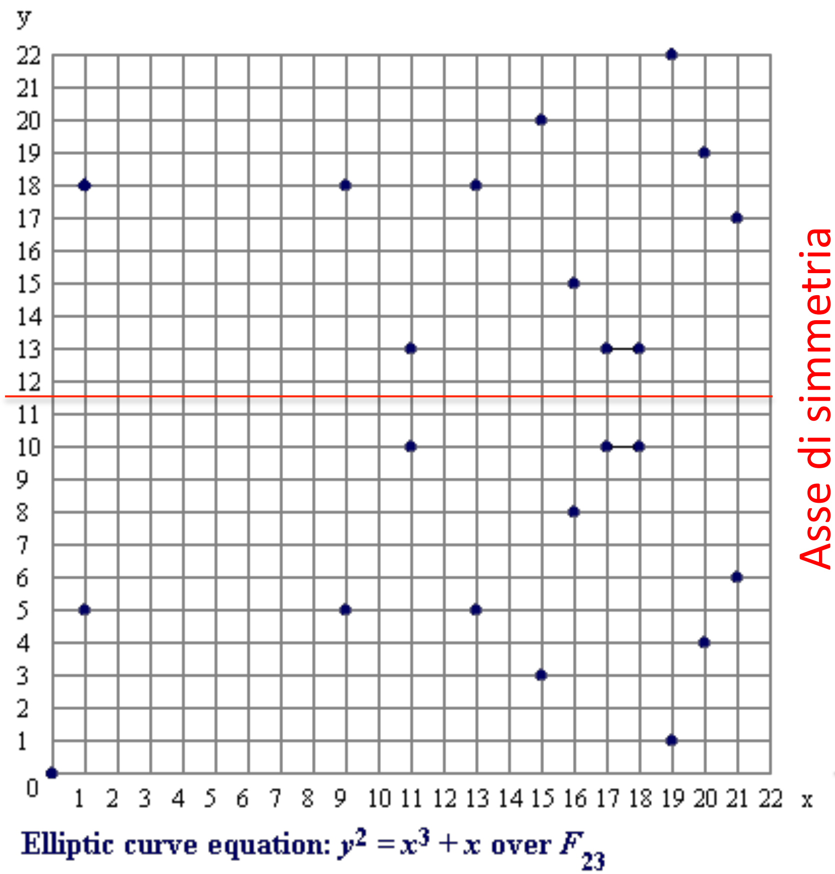
Curve Ellittiche

- La curva ellittica è simmetrica rispetto all'asse x
- Per ogni punto P=(x,y), l'inverso di P è definito come –P=(x, –y)
- Per comodità disegniamo le curve in R anche se sono costruite su Z_p
- Su un campo finito avrebbero un altro aspetto



$y^2 = x^3 - 3 \cdot x + 3$ in R

Esempi



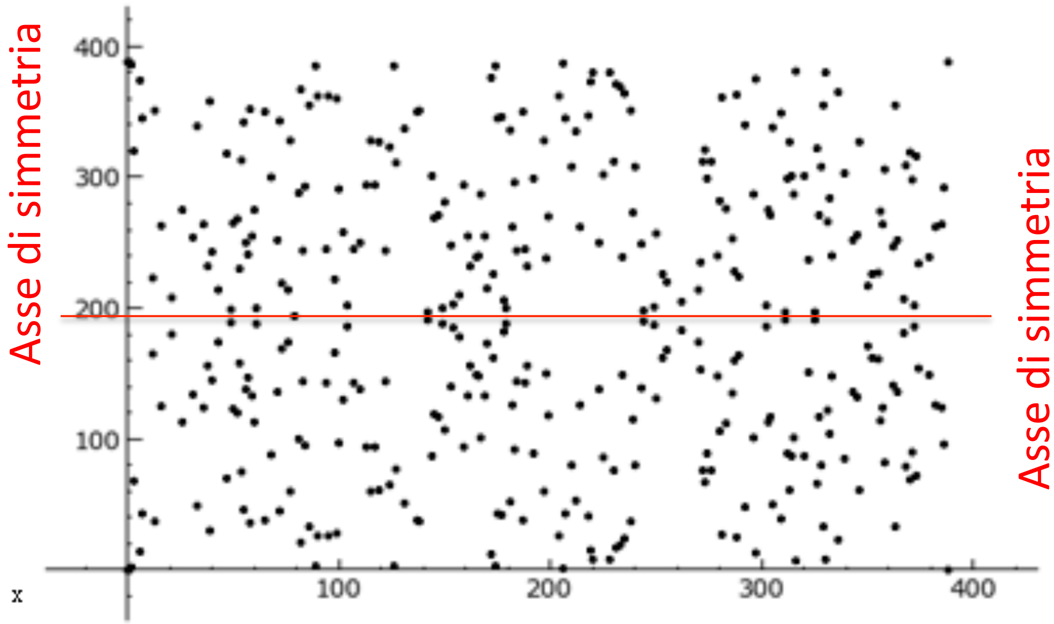
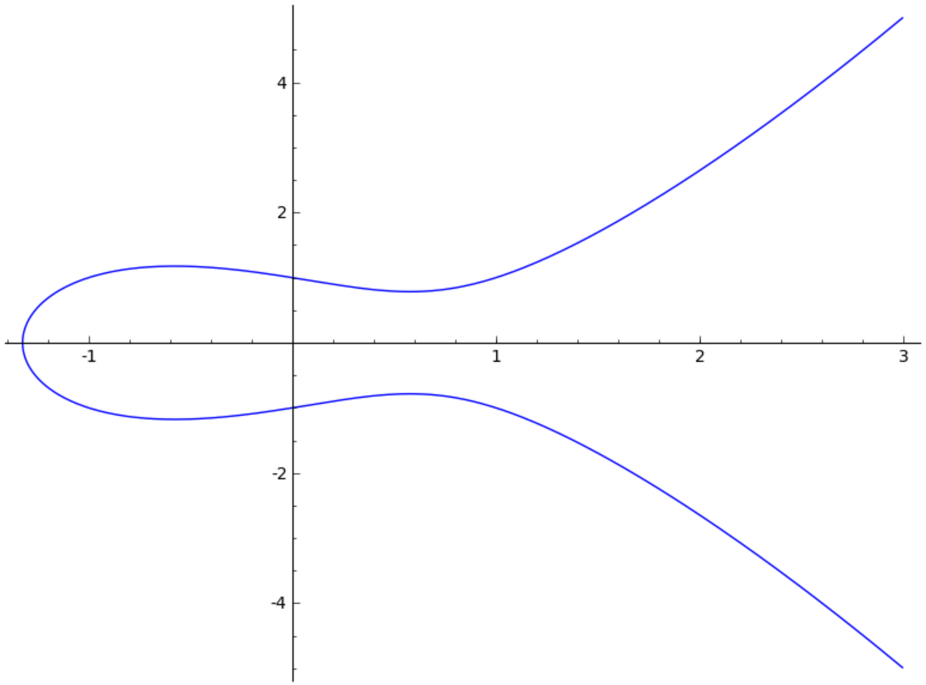
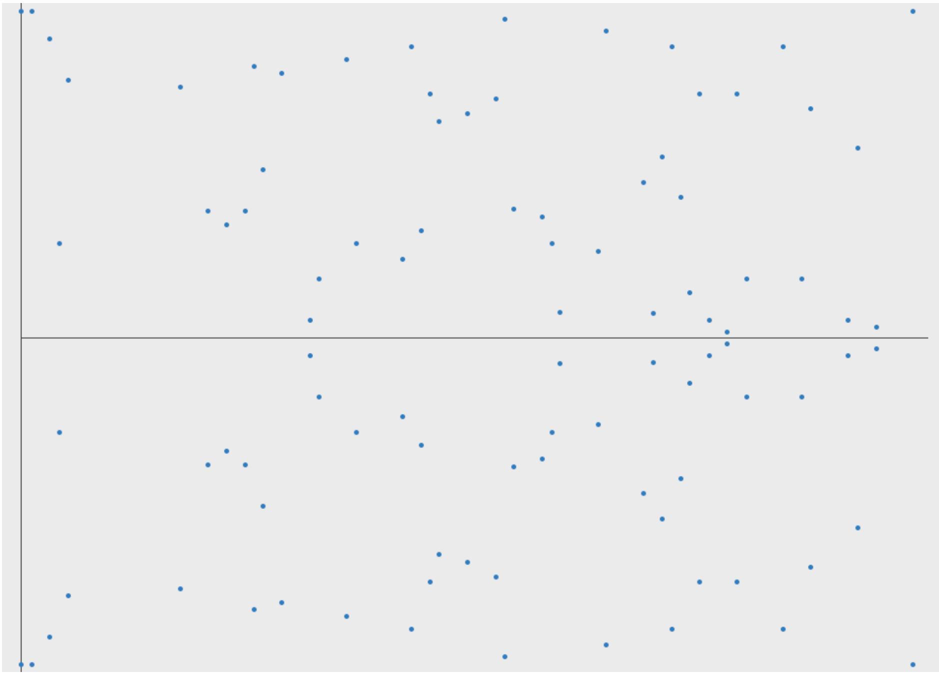


Figura 1.4: La curva $y^2 + y = x^3 - x$ definita sul campo finito F_{389}

F_n campo finito con n elementi

Curva Ellittica $y^2 = x^3 - x + 1$





in R

in F_{97}

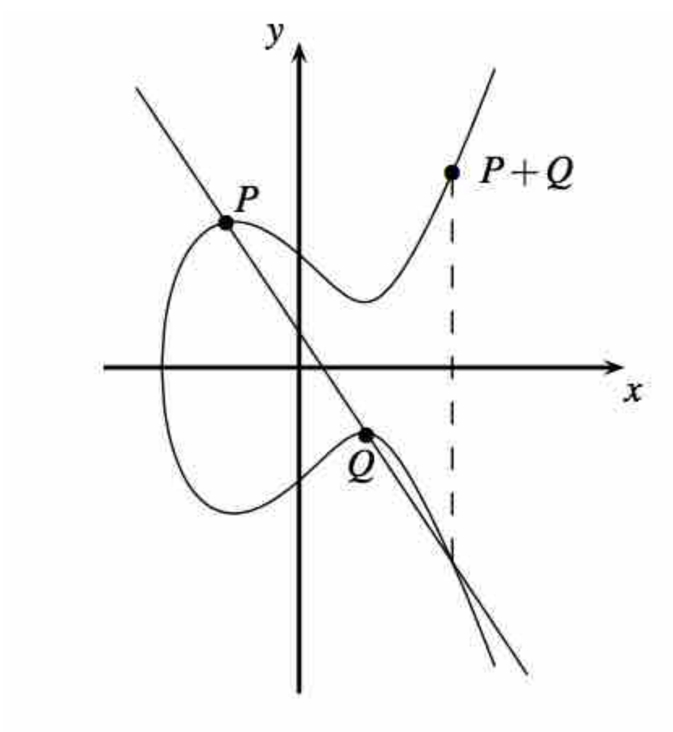
Addizione su Curve Ellittiche

- Per qualsiasi punto P della curva vale che

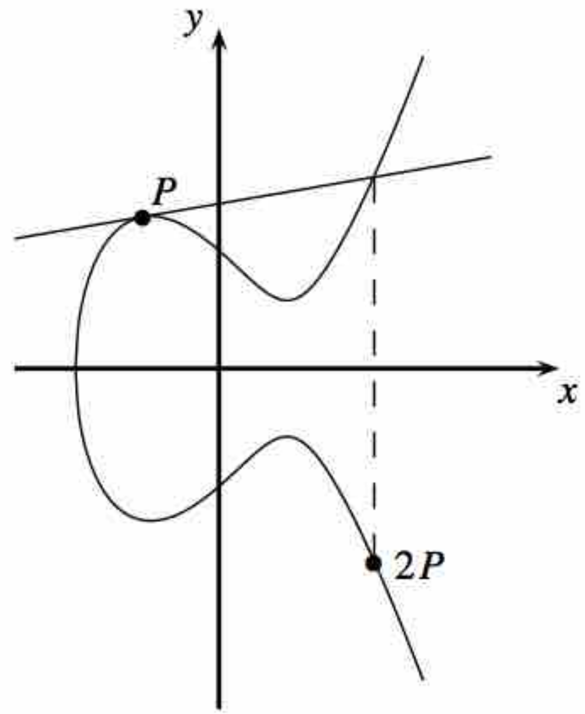
$$P+\theta = \theta + P = P$$

- Dati $P=(x_1,y_1)$ e $Q=(x_2,y_2)$, vogliamo calcolare $R=P+Q$, quindi $(x_3,y_3) = (x_1,y_1) + (x_2,y_2)$
- Se $P \neq Q$ (*point addition*)
  - Si traccia la retta che unisce P e Q. La retta individua un terzo punto X sulla curva. R sarà uguale a –X
- Se $P=Q$ (*point doubling*, $R=P+P = 2P$)
  - Si considera la tangente in P e si prosegue come nel caso precedente

Interpretazione geometrica su R

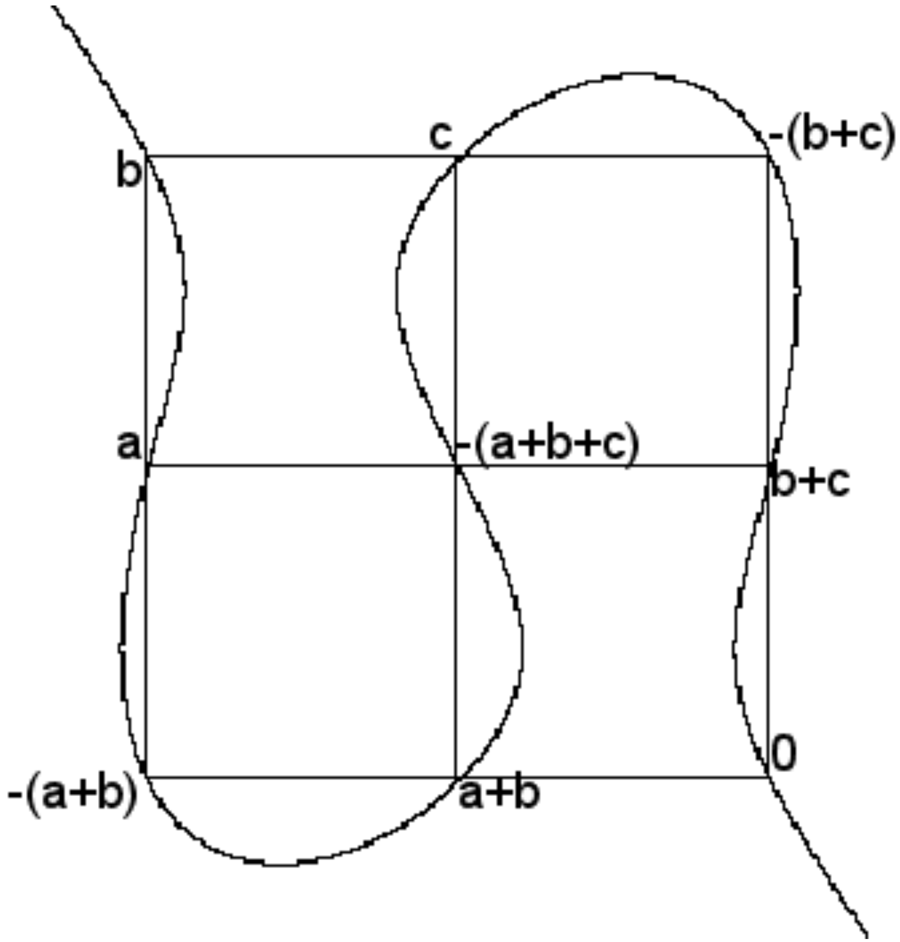


P≠Q



P=Q

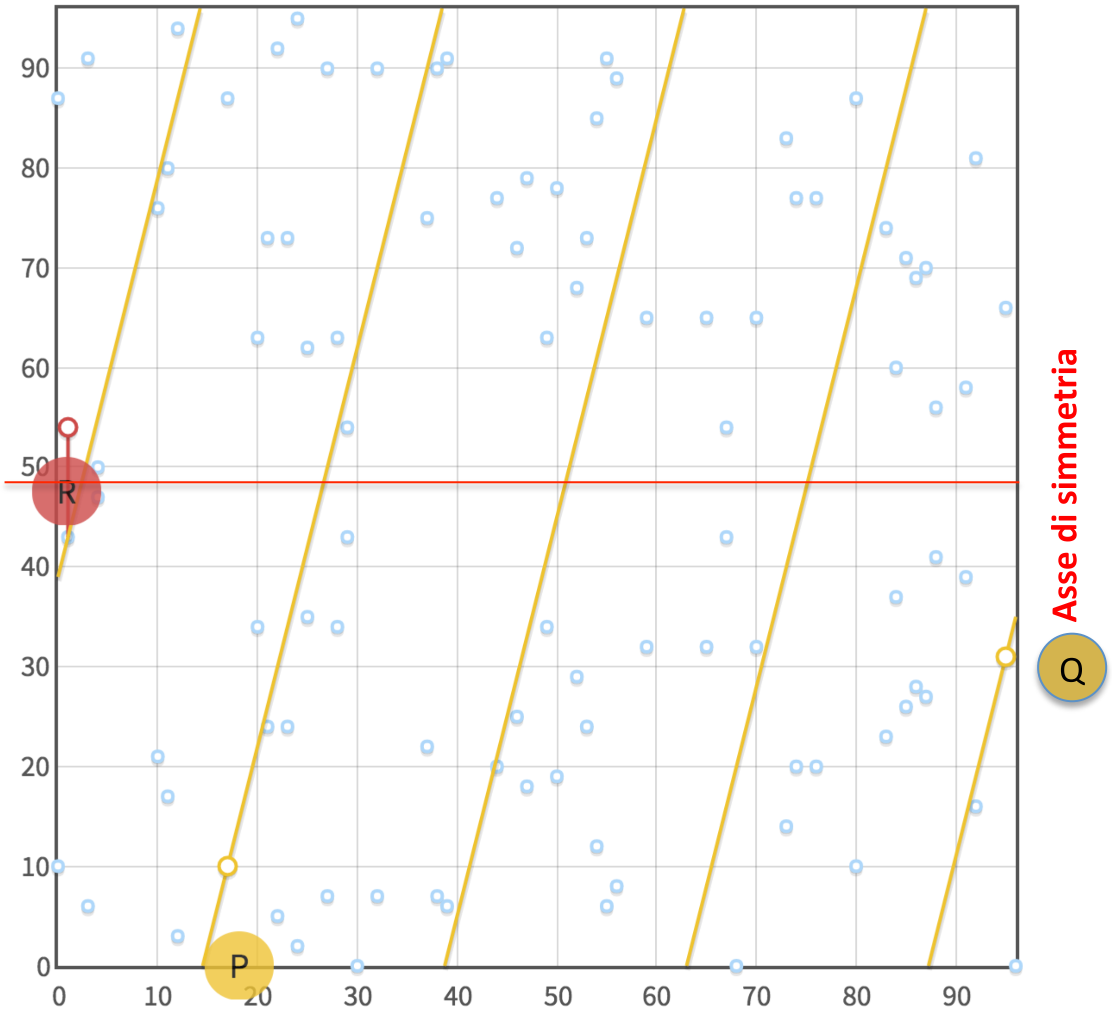
Esempio addizione



Fonte: https://en.wikipedia.org/wiki/File:EllipticGroup.gif gif animata

Addizione in EC su campo finito

Campo finito con 97 elementi F_{97} $y^2 = x^3 + 2x + 3$



P = (17, 10) Q = (95, 31)

R = P + Q R = (1, 54)

Allineamento di tre punti in F_p?

Informalmente, una linea in F_p è l'insieme dei punti (x,y) in $F_p x F_p$ che soddisfano l'equazione $\alpha x+\beta y+\gamma \equiv 0 \mod p$

La curva ha 100 punti compreso il punto all'infinito

Algebricamente

$y^2 = x^3 + ax + b$

- $P=(x_1,y_1)$ e $Q=(x_2,y_2)$, vogliamo calcolare $R= (x_3,y_3) = (x_1,y_1) + (x_2,y_2)$
- Calcoliamo

a coefficiente della x

$$s = \begin{cases} \frac{y_2-y_1}{x_2-x_1} \bmod p \text{ ; if } P \neq Q \text{ (point addition)} \\ \frac{3x_1^2+a}{2y_1} \bmod p \text{ ; if } P = Q \text{ (point doubling)} \end{cases}$$

- $R = (x_3,y_3)$ sarà

$$x_3 = s^2 - x_1 - x_2 \bmod p$$
$$y_3 = s(x_1 - x_3) - y_1 \bmod p$$

Esempio

- Data la curva $y^2 \equiv x^3 + 2 \cdot x + 2 \bmod 17$ e $P = (5, 1)$, vogliamo calcolare $2P = P + P$

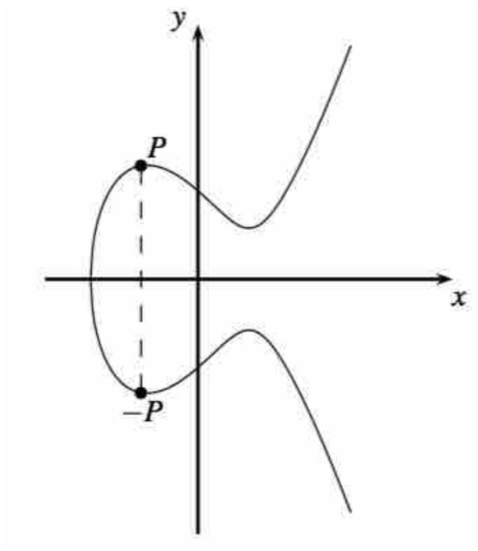
$$2P = P + P = (5,1) + (5,1) = (x_3, y_3)$$
$$s = \frac{3x_1^2 + a}{2y_1} = (2 \cdot 1)^{-1}(3 \cdot 5^2 + 2) = 2^{-1} \cdot 9 \equiv 9 \cdot 9 \equiv 13 \bmod 17$$
$$x_3 = s^2 - x_1 - x_2 = 13^2 - 5 - 5 = 159 \equiv 6 \bmod 17$$
$$y_3 = s(x_1 - x_3) - y_1 = 13(5-6) - 1 = -14 \equiv 3 \bmod 17$$
$$2P = (5,1) + (5,1) = (6,3)$$

(6,3) appartiene alla curva

$$y^2 \equiv x^3 + 2 \cdot x + 2 \bmod 17$$
$$3^2 \equiv 6^3 + 2 \cdot 6 + 2 \bmod 17$$
$$9 = 230 \equiv 9 \bmod 17$$

Identità (elemento neutro)

- Indicato con θ, non appartiene alla curva
- Soddisfa $P = P + \theta$, per ogni punto della curva
- Definiamo l'inverso $-P$ di P come $P + (-P) = \theta$
- Se $P = (x_P, y_P)$, $-P = (x_P, -y_P)$
- Dato che lavoriamo in Z_p,
  $-P = (x_P, p-y_P)$



Risultati

θ

Theorem 9.2.1 *The points on an elliptic curve together with $\mathcal{O}$ have cyclic subgroups. Under certain conditions* all *points on an elliptic curve form a cyclic group.*

Esiste un punto che genera tutti i punti della curva

Theorem 9.2.2 Hasse's theorem
Given an elliptic curve E modulo p, the number of points on the curve is denoted by #E and is bounded by:

$$p+1-2\sqrt{p} \leq \#E \leq p+1+2\sqrt{p}.$$

Per generare una curva con circa 2^{160} punti il primo p deve essere lungo circa 160 bit

Esempio

E: $y^2 \equiv x^3 + 2\cdot x + 2 \bmod 17$

Dato P = (5,1), calcoliamo 2P, 3P, …, (#E)P

P è un generatore della curva

$$
\begin{aligned}
2P &= (5,1)+(5,1) = (6,3) & 11P &= (13,10)\\
3P &= 2P+P = (10,6) & 12P &= (0,11)\\
4P &= (3,1) & 13P &= (16,4)\\
5P &= (9,16) & 14P &= (9,1)\\
6P &= (16,13) & 15P &= (3,16)\\
7P &= (0,6) & 16P &= (10,11)\\
8P &= (13,7) & 17P &= (6,14)\\
9P &= (7,6) & 18P &= (5,16)\\
10P &= (7,11) & 19P &= \mathcal{O}
\end{aligned}
$$

P = (5,1) è l'inverso di 18P = (5, 16)

Abbiamo un gruppo ed un generatore…

Elliptic Curve Discrete Logarithm Problem (ECDLP)

Definition 9.2.1 Elliptic Curve Discrete Logarithm Problem (ECDLP)
Given is an elliptic curve E. We consider a primitive element P and another element T. The DL problem is finding the integer d, where $1 \leq d \leq \#E$, *such that:*

$$\underbrace{P+P+\cdots+P}_{d\ \ times} = dP = T. \tag{9.2}$$

base point

Il punto P è un generatore del gruppo
Invece dell'elevamento a potenza si considera la somma di punti della curva

La somma è solo una notazione

Si ritiene che, per curve ellittiche scelte in maniera opportuna, ECDLP sia un problema difficile da risolvere

Come calcolare d·P?

- Si usa la stessa tecnica dell'elevamento a potenza
  - Square-and-Multiply

Double-and-Add Algorithm for Point Multiplication

Input: elliptic curve E together with an elliptic curve point P

a scalar $d = \sum_{i=0}^{t} d_i 2^i$ with $d_i \in 0, 1$ and $d_t = 1$

Output: $T = d\,P$

Initialization:

$T = P$

Algorithm:

1 FOR $i = t - 1$ DOWNTO 0

1.1 $\quad T = T + T \bmod n$

$\quad$ IF $d_i = 1$

1.2 $\quad\quad T = T + P \bmod n$

2 RETURN (T)

Elliptic Curve e Diffie–Hellman

- Possiamo realizzare DHKE utilizzando come gruppo uno generato da una curva ellittica
- In tal caso si fa riferimento a Elliptic Curve Diffie–Hellman key exchange (ECDH)

- Abbiamo bisogno di una curva ellittica opportuna e di un generatore del gruppo che definiranno i parametri del protocollo

Generazione dei parametri + ECDH

ECDH Domain Parameters

1. Choose a prime p and the elliptic curve

$$E : y^2 \equiv x^3 + a \cdot x + b \bmod p$$

2. Choose a primitive element $P = (x_P, y_P)$

The prime p, the curve given by its coefficients a, b, and the primitive element P are the domain parameters.

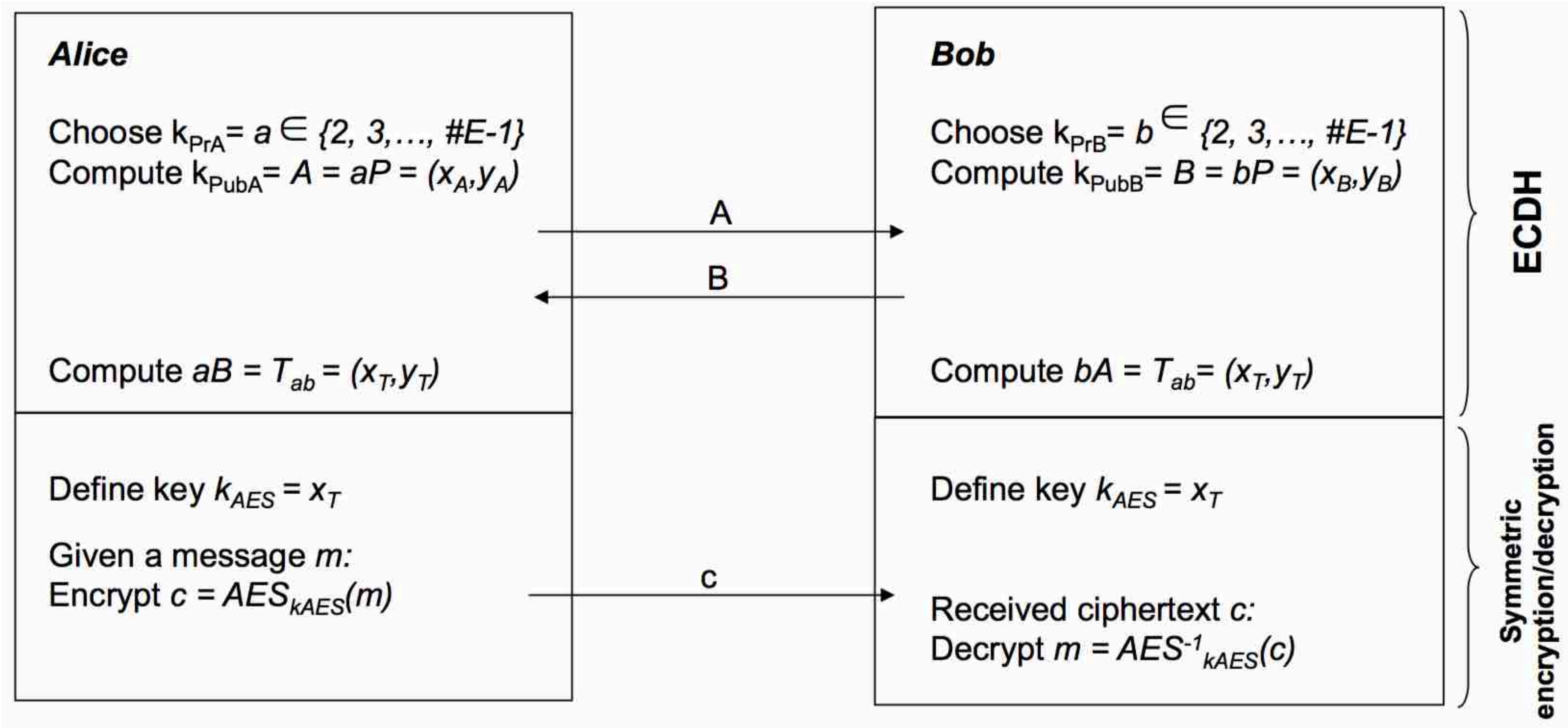
Elliptic Curve Diffie–Hellman Key Exchange (ECDH)

Alice		**Bob**
choose $k_{prA} = a \in \{2, 3, \ldots, \#E - 1\}$		choose $k_{prB} = b \in \{2, 3, \ldots, \#E - 1\}$
compute $k_{pubA} = aP = A = (x_A, y_A)$		compute $k_{pubB} = bP = B = (x_B, y_B)$
	$\xrightarrow{A}$	
	$\xleftarrow{B}$	
compute $aB = T_{AB}$		compute $bA = T_{AB}$

Joint secret between Alice and Bob: $T_{AB} = (x_{AB}, y_{AB})$.

T_{AB} è usato per derivare una chiave di sessione. Dato che y_{AB} dipende da x_{AB}, si usa solo x_{AB} per derivare la chiave

ECDH


Alice
Choose $k_{PrA} = a \in \{2, 3, \ldots, \#E-1\}$
Compute $k_{PubA} = A = aP = (x_A, y_A)$
Bob
Choose $k_{PrB} = b \in \{2, 3, \ldots, \#E-1\}$
Compute $k_{PubB} = B = bP = (x_B, y_B)$
A
B
Compute $aB = T_{ab} = (x_T, y_T)$
Compute $bA = T_{ab} = (x_T, y_T)$
ECDH
Define key $k_{AES} = x_T$
Given a message m:
Encrypt $c = AES_{kAES}(m)$
c
Define key $k_{AES} = x_T$
Received ciphertext c:
Decrypt $m = AES^{-1}_{kAES}(c)$
Symmetric encryption/decryption

Sicurezza ECDH

- Attacchi a gruppi basati su curve ellittiche sono meno efficienti degli attuali attacchi alla fattorizzazione o a DLP
- I migliori attacchi a gruppi basati su curve ellittiche sono i metodi Baby-Step Giant-Step e Pollard-Rho
- Complessità dei metodi indicati: in media, sono necessari circa $p^{1/2}$ passi per risolvere ECDLP

NIST SP 800-57, Part 1, Revision 4

Security Strength	ECC (e.g., ECDSA)
≤ 80	$f = 160\text{-}223$
112	$f = 224\text{-}255$
128	$f = 256\text{-}383$
192	$f = 384\text{-}511$
256	$f = 512+$

- ECC
  - Elliptic Curve Cryptography
- f
  - Intervallo di valori (in bit) per n
    - n è l'ordine del base point della curva
  - Generalmente f è considerata la dimensione della chiave in bit

Confronto *Security Strength*

Security Strength	Symmetric key algorithms	FFC (e.g., DSA, D-H)	IFC (e.g., RSA)	ECC (e.g., ECDSA)
≤ 80	2TDEA[21]	$L = 1024$ $N = 160$	$k = 1024$	$f = 160\text{-}223$
112	3TDEA	$L = 2048$ $N = 224$	$k = 2048$	$f = 224\text{-}255$
128	AES-128	$L = 3072$ $N = 256$	$k = 3072$	$f = 256\text{-}383$
192	AES-192	$L = 7680$ $N = 384$	$k = 7680$	$f = 384\text{-}511$
256	AES-256	$L = 15360$ $N = 512$	$k = 15360$	$f = 512+$

Impronta energetica

- Per visualizzare quanto è difficile rompere un algoritmo crittografico, Lenstra, Kleinjung e Thomé hanno introdotto il concetto di *Global Security*
- Si può calcolare quanta energia è necessaria per rompere un algoritmo crittografico e confrontarla con la quantità di acqua (a 20°) che quell'energia potrebbe portare ad ebollizione
  – Si calcola una sorta di Impronta Energetica (*carbon footprint*)

Impronta energetica

- L’energia necessaria per rompere una chiave RSA di 228 bit è inferiore a quella necessaria per portare ad ebollizione un cucchiaino da the di acqua
- Rompere una chiave EC di 228 bit richiede una quantità di energia sufficiente a portare ad ebollizione tutta l’acqua sulla terra
  - Per ottenere lo stesso livello di sicurezza con RSA è necessaria una chiave di 2.380 bit

Livelli di sicurezza intuitivi

security level	volume of water to bring to a boil	bit-lengths: symmetric key	bit-lengths: cryptographic hash	bit-lengths: RSA modulus
teaspoon security	0.0025 liter	35	70	242
shower security	80 liter	50	100	453
pool security	2 500 000 liter	65	130	745
rain security	0.082 km^3	80	160	1130
lake security	89 km^3	90	180	1440
sea security	3 750 000 km^3	105	210	1990
global security	1 400 000 000 km^3	114	228	2380
solar security	-	140	280	3730

Utilizzare la Security Strength illustrata nelle slide e lezioni precedenti

Parametri NIST

- Descritti in FIPS PUB 186-4

  Digital Signature Standard (DSS), luglio 2013

- Sono descritte curve su GF(p), p primo, e GF(2^m)
  - Per GF(2^m) è descritta anche una curva di Kobliz

$$\mathbb{F}_p = \mathsf{GF}(p) \text{ e } \mathbb{F}_{2^m} = \mathsf{GF}(2^m)$$

- Le curve sono descritte tramite parametri

Parametri curva

- a e b: Coefficienti della curva
- N: Ordine del gruppo della curva
  - #Punti sulla curva + 1 (elemento neutro)
- G: base point (generatore)
  - Generatore del sottogruppo ciclico di E
- n : Ordine del sottogruppo generato da G
- h: Cofattore del sottogruppo
  - $N = h \cdot n$

Serve a far vedere che b non è stato generato ad-hoc dato che SHA-1 non è efficientemente invertibile

Curve su GF(p)

- Curva: $y^2 \equiv x^3 - 3x + b \mod p$
  - p numero primo
  - a fissato al valore -3 motivi di efficienza
  - b generato a partire da un seme casuale (seed)
    - c = PRNG(seed), b soddisfa $b^2c \equiv -27 \mod p$
  - N primo, quindi n=N ed h=1
  - Base point $G=(G_x, G_y)$. A partire da G si possono generare altri base point seguendo le indicazioni in ANSI X9.62 oppure IEEE Standard 1363-2000

PRNG generatore basato su SHA-1 descritto in ANSI X9.62

Esempio

D.1.2.1 Curve P-192

$p =$	6277101735386680763835789423207666416083908700390324961279
$n =$	6277101735386680763835789423176059013767194773182842284081
$SEED =$	3045ae6f c8422f64 ed579528 d38120ea e12196d5
$c =$	3099d2bb bfcb2538 542dcd5f b078b6ef 5f3d6fe2 c745de65
$b =$	64210519 e59c80e7 0fa7e9ab 72243049 feb8deec c146b9b1
$G_x =$	188da80e b03090f6 7cbf20eb 43a18800 f4ff0afd 82ff1012
$G_y =$	07192b95 ffc8da78 631011ed 6b24cdd5 73f977a1 1e794811

Possibili grandezze in bit di p: 192, 224, 256, 384 e 521

521 non è un errore,
non è 512

Curve su GF(2^m)

- Curva non-Kobliz (*curva pseudo-random*)

$$y^2 + xy = x^3 + x^2 + b$$

  - $h = 2$

Possibili valori di m: 163, 233, 283, 409 e 571

- Curva di Kobliz

$$y^2 + xy = x^3 + ax^2 + 1$$

  - Se a=0, allora h=2
  - Se a=1, allora h=4

Navigazione anonima

Dove è usata ECC?

Key Establishment

- IPSec/TLS
- Nei browser, se il server è configurato per supportare ECC
  - Il browser Tor (The Onion Router) utilizza una curva progettata da Daniel J. Bernstein
  - Parametri scelti per efficienza (sicurezza almeno di 128 bit) e per mancanza di fiducia nei parametri del NIST

Firma digitale

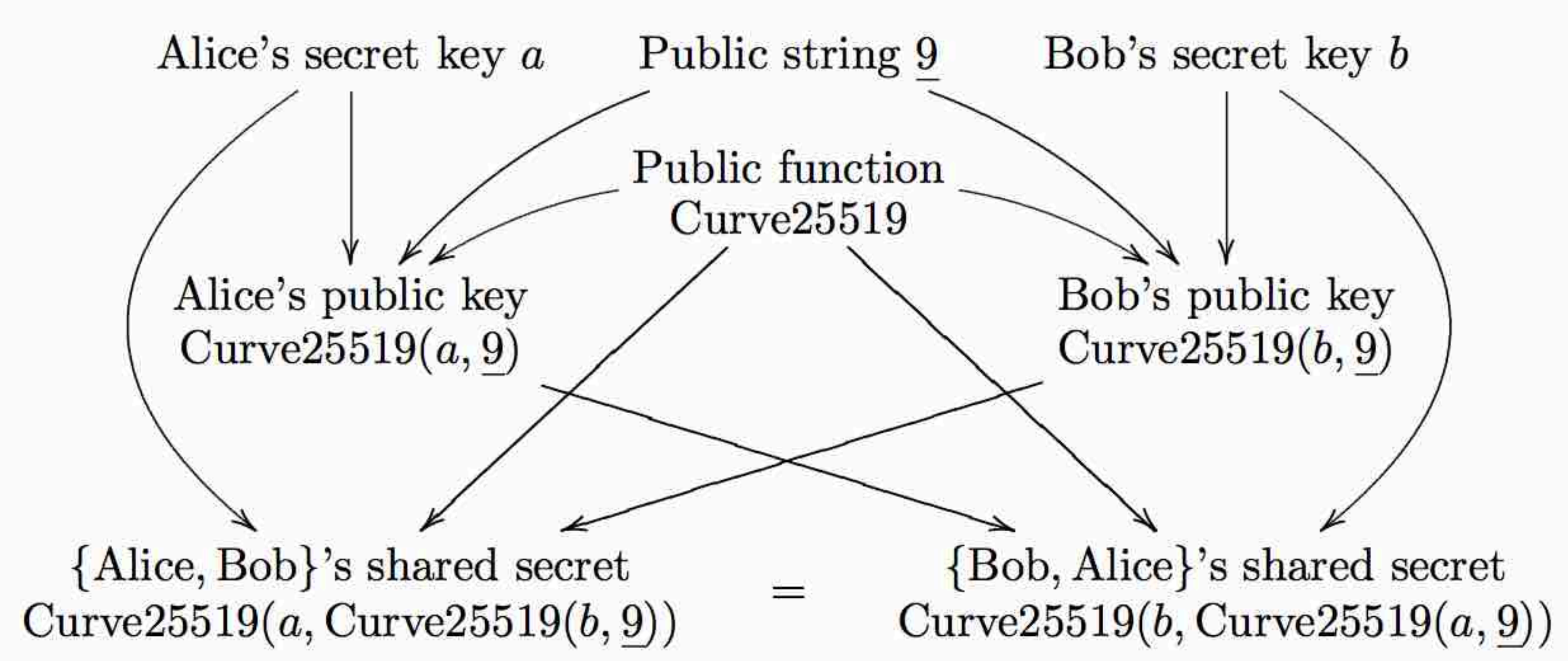
- Bitcoin
  - Per assicurare che le *monete* siamo spese dal legittimo possessore (firma ECDSA)
- iMessage
  - Instant messaging system di Apple
- Playstation 3
  - Software firmato con ECDSA

Dettagli su https://fail0verflow.com/

Chiave segreta estratta nel 2010 per un cattiva implementazione di ECDSA
Stessa randomness riutilizzata

TOR: Curve25519

9 è un parametro fissato della curva



La chiave segreta di Alice, la chiave segreta di Bob e quella condivisa sono di 32 byte

Riferimenti

Christof Paar and Jan Pelzl
Understanding Cryptography
Capitolo 9 Elliptic Curve Cryptosystems

A. K. Lenstra, T. Kleinjung, E. Thomé
Universal security, from bits and mips to pools, lakes – and beyond, 2013