ZERO KNOWLEDGE PROOFS

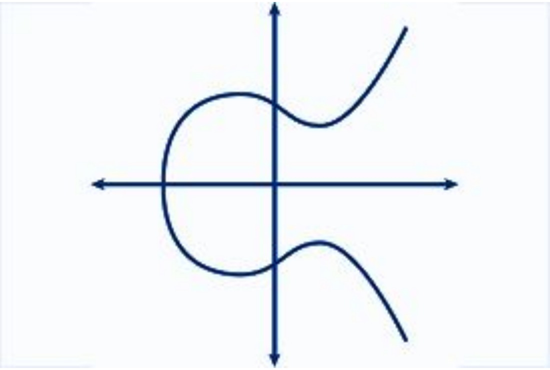
by Marija Mikić

Elliptic curves as cryptographic groups

In mathematics, a finite field is a field that contains a finite number of elements. The order of a finite field is its number of elements, which is either a prime number or a prime power.

Elliptic curves are groups which are defined over finite fields. Equation of elliptic curve is

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



where a and b are from algebraic closure of finite field K. The Equation is called the short Weierstrass equation for elliptic curves.

O - Point at infinity

We will always be working over large prime field.

There's More 👉

Abelian group

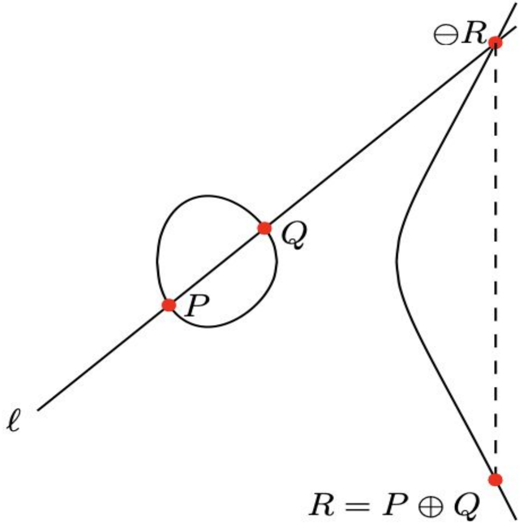
It has been proved that the set of points in ECC always form an Abelian group with the following properties:

Closure: If points P and Q belong to E(K), then P + Q also belongs to E(K);
Associativity: (P + Q) + R = P + (Q + R);
Identity: There exists an identity element 0 such that P + 0 = P;
Inverse: Every element P has an inverse Q such that P + Q = 0;
Commutativity: P + Q = Q + P.

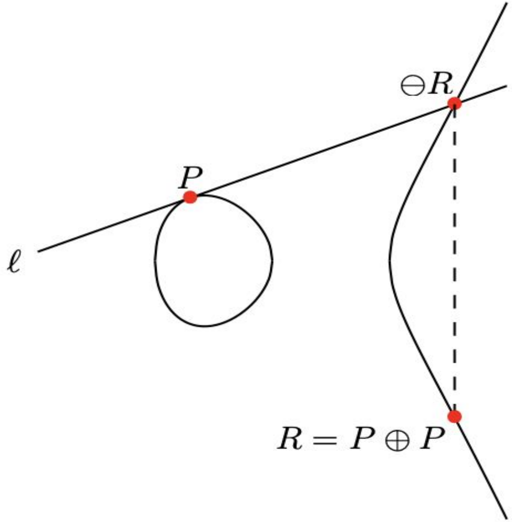
We will assume that elliptic curve over finite field is cyclic.

There's More 👉

Chord rule



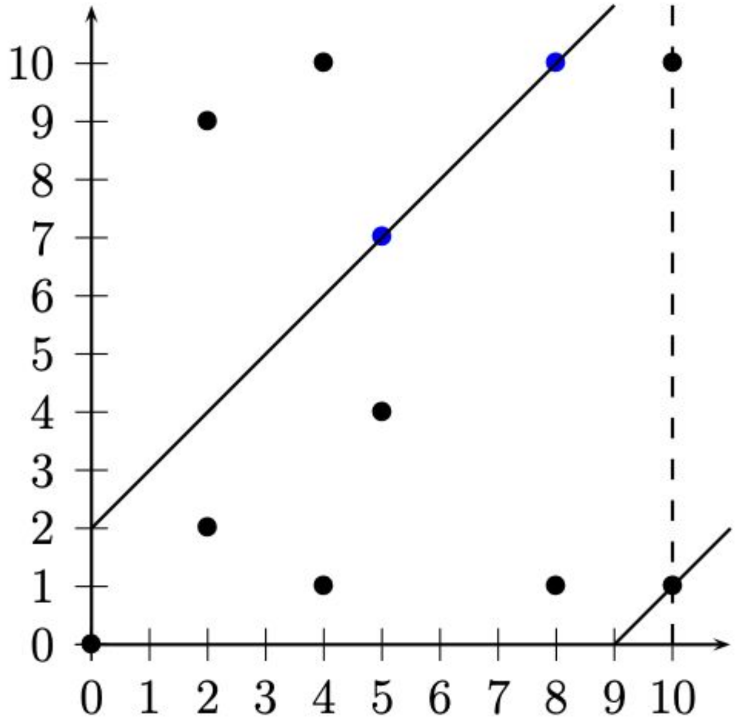
Tangent rule



The point ⊖R is then “flipped” over the x-axis to the point R.

There’s More 👉

Elliptic curve points over finite field F_{11}



There's More 👉

Add and Double algorithm

How can we multiply a point P with a scalar m where $m \geq 0$?

$$m*P = P + P + P + P + \ldots + P \text{ (m times)}$$

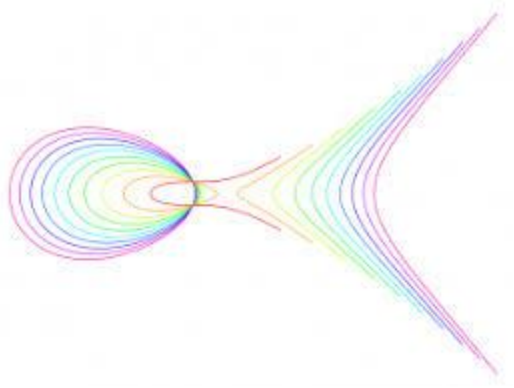
But that method is too slow for large m!

For faster calculation, we represent m as a binary number and get the result in logarithmic time.

For example, to evaluate 79*P, we convert 79 in its binary form.

Thus we can evaluate the sum:

$$79*P = 2^6*P + 2^3*P + 2^2*P + 2^1*P + 2^0*P.$$



There’s More 👉

EC Discrete logarithm problem

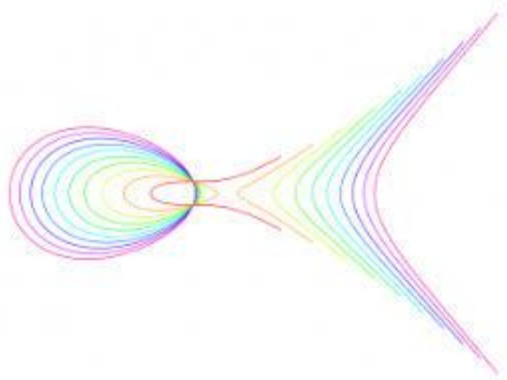
Finding m when P is given and m*P is known as the elliptic curve discrete logarithm problem. Let

$$[m] : E \mapsto E, \qquad P \mapsto [m]P$$

This operation is analogous to exponentiation

$$g \mapsto g^m \text{ in } Z_q^*$$

and is the central operation in ECC, as it is the one-way operation that buries discrete logarithm problems in E(Fq).

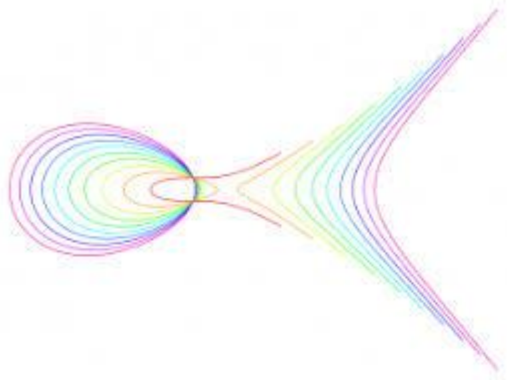


There's More 👉

Security

Elliptic curve groups of a relatively small size achieves the same conjectured security as multiplicative groups in much larger finite fields.

For example, an elliptic curve defined over a 160-bit field currently offers security comparable to a finite field of 1248 bits.



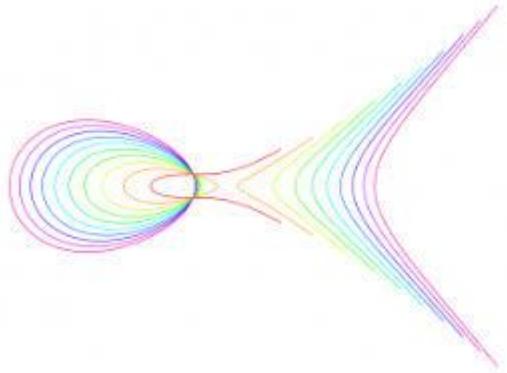
There’s More 👉

Multi-Scalar-Multiplication

The bottleneck in the proving algorithm of most of elliptic-curve-based SNARK proof systems is the Multi-Scalar-Multiplication (MSM) algorithm.

The naive algorithm uses a double-and-add strategy.

The fastest approach is a variant of Pippenger's algorithm, we call it the bucket method.



*

You can find more information on this link:

[Link 1 >]

There's More 👉

Elliptic curve pairings

Definition: Let E be an elliptic curve over a finite field K. Let $\mathbb{G}_1$ and $\mathbb{G}_2$ be additively-written subgroups of order p, where p is a prime number, of elliptic curve E, and let $g_1 \in \mathbb{G}_1, g_2 \in \mathbb{G}_2$ are the generators of groups $\mathbb{G}_1$ and $\mathbb{G}_2$ respectively. A map $e : \mathbb{G}_1 \times \mathbb{G}_2 \to \mathbb{G}_T$, where $\mathbb{G}_T$ is a multiplicatively-written subgroup of K of order p, is called a elliptic curve pairing if satisfies the following conditions:

1. $e(g_1, g_2) \neq 1$,
2. $\forall R, S \in \mathbb{G}_1, \forall T \in \mathbb{G}_2 : e(R + S, T) = e(R, T) * e(S, T)$,
3. $\forall R \in \mathbb{G}_1, \forall S, T \in \mathbb{G}_2 : e(R, S + T) = e(R, S) * e(R, T)$.

The following properties of the elliptic curve pairing can be easily verified:

1. $\forall S \in \mathbb{G}_1, \forall T \in \mathbb{G}_2 : e(S, -T) = e(-S, T) = e(S, T)^{-1}$,
2. $\forall S \in \mathbb{G}_1, \forall T \in \mathbb{G}_2 : e(a * S, b * T) = e(b * S, a * T) = e(S, T)^{a*b}$.

There's More 👉

Types of bilinear map

- **Type 1:** $\mathbb{G}_1 = \mathbb{G}_2$, and we say e is a symmetric bilinear map;
- **Type 2:** $\mathbb{G}_1 \neq \mathbb{G}_2$ and there exists an efficient homomorphism $\phi : \mathbb{G}_2 \to \mathbb{G}_1$, but no efficient one exists in the other direction;
- **Type 3:** $\mathbb{G}_1 \neq \mathbb{G}_2$ and there exists no efficient homomorphism between $\mathbb{G}_1$ and $\mathbb{G}_2$.

*

You can find more information on this link:

Link 1 >

Thank you!