Elliptical curve cryptography - Elliptic curve arithmetic

Elliptical curve cryptography is a public key encryption technique based on elliptic curve theory that can be used to create faster, smaller and more efficient cryptographic keys

ECC is an alternative to the Rivest-Shamir-Adleman (RSA) cryptographic algorithm and is most often used for digital signatures in cryptocurrencies, such as Bitcoin and Ethereum, as well as one-way encryption of emails, data and software

An elliptic curve is not an ellipse or oval shape, but it is represented as a looping line intersecting two axes, which are lines on a graph used to indicate the position of a point. The curve is completely symmetric or mirrored, along the x-axis of the graph.

Public key cryptographic systems, like ECC use a mathematical process to merge two distinct keys and then use the output to encrypt and decrypt data. one is a public

key that is known to anyone and other is a private key that is only known by the sender and receiver of the data.

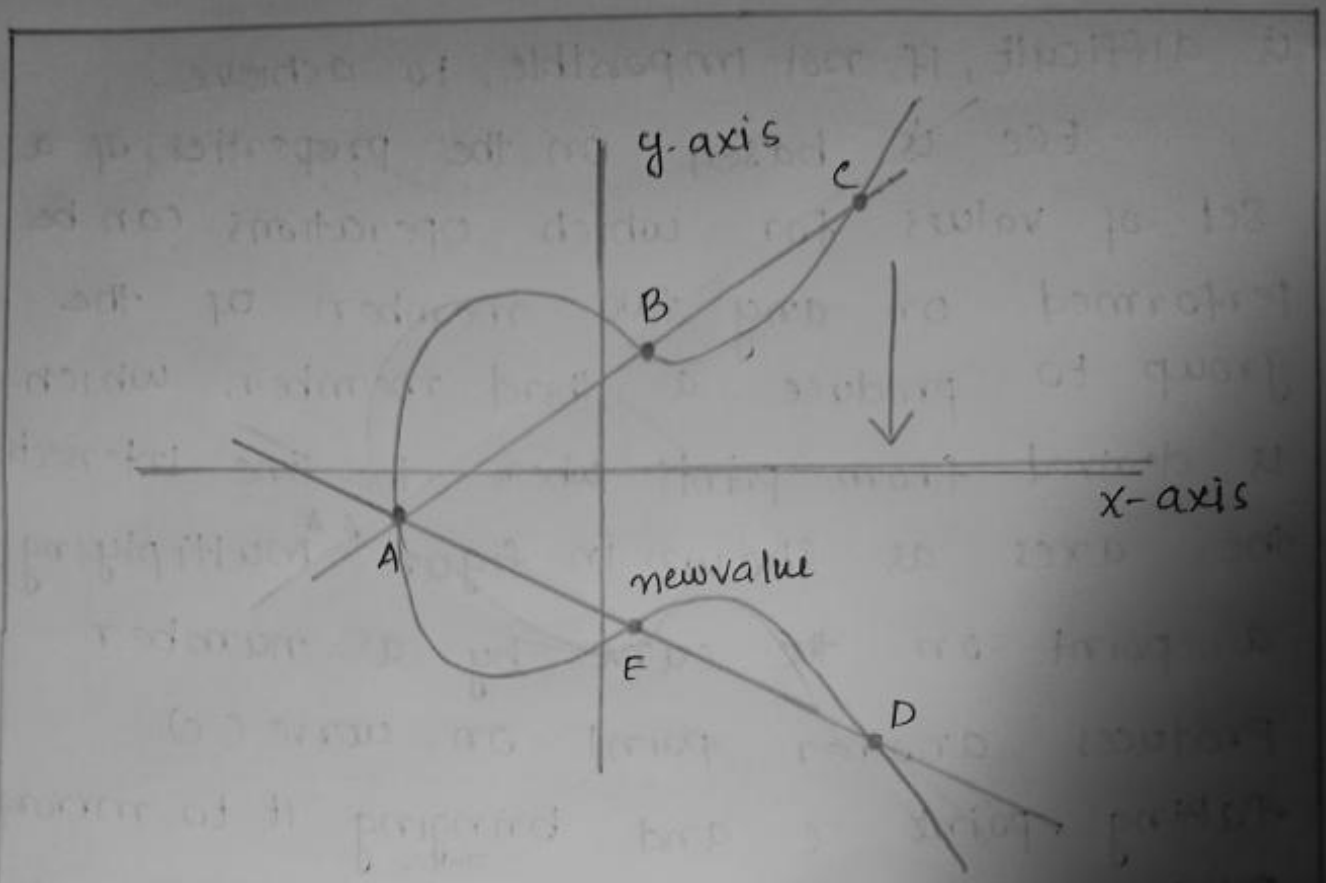
Ecc generates keys through the properties of an elliptic curve equation instead of traditional method of generation as product of large prime numbers. from a cryptographic perspective, the points along the graph can be formulated using the following equation:

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

ECC is like most other public key encryption methods such as RSA algorithm and Diffie-Hellman. Each of these cryptography mechanisms uses the concept of a one-way or trapdoor function. This means that a mathematical equation with a public and private key can be used to easily get from point A to point B. But without knowing private key and depending on key size used, getting from B to A is

is difficult, if not impossible, to achieve

ECC is based on the properties of a set of values for which operations can be performed on any two members of the group to produce a third member, which is derived from points where the line intersects the axes as shown in figure. Multiplying a point on the curve by a number produces another point on curve (C). Taking point C and bringing it to mirror point on the opposite side of the x-axis produces point D. From here, a line is drawn back to our original point A, creating an intersection at point E. This process can be completed n number of times within a defined max value. The n is the private key value which indicates how many times the equation should be run, ending on the final value that is used to encrypt and decrypt data. The maximum defined value of the equation relates to keysize used.



ECC Algorithm.

ECC key exchange

Global public elements -

1) $E_q(a,b)$ - Elliptic curve with parameters a, b & q (prime number or an integer of form 2^m)

2) G - point on elliptic curve

User A key generation

Select private key n_A $n_A < n$.

calculate public key P_A. $P_A = n_A \times G$.

User B key generation.

Select private key n_B $n_B < n$

Calculate publickey P_B $P_B = n_B \times G$

Calculation of secret key by User A

$K = n_A \times P_B$

Calculation of secret key by User B

$K = n_B \times P_A$

ECC Encryption

Let the message be M

First encode this message M into a point on elliptic curve

Let this point be Pm

For encryption, choose a random positive integer K

The cipher point will be

$Cm = \{ KG, Pm + KP_B \}$

This point will be sent to receiver.

ECC decryption.

For decryption multiply x-cordinate with receiver's secret key

$KG \times n_B$

then subtract $(KG \times nB)$ from y-cordinate of cipher point

$$Pm + KP_B - (KG \times nB)$$

we know that $P_B = n_B \times G$

$$\therefore Pm + KP_B - KP_B$$

$$= Pm$$

So receiver get the same point

Key sizes with equivalent security levels.

minimum size (bits) of public keys.

DSA / DH	RSA	ECC
1024	1024	160
2048	2048	224
3072	3072	256
7680	7680	384
15360	15360	512