Elliptic Curve and its Digital Signature

Elliptic Curve Cryptography (ECC) is used to encrypt and decrypt the information which flows over the internet. It's a key-based technique, which uses pair of public and private keys for encryption and decryption. ECC belongs to public-key cryptography where one key (private key) is used to encrypt the information and another key (public key) is used to decrypt the information. ECC not only provides integrity of data, but it also provides nonrepudiation with the help of Elliptic Curve Digital Signature (ECDSA). ECDSA is a cryptographically secure digital signature scheme which uses Secure Hash (SHA) to calculate the hash of information and further perform digital signature. Mainly this article will talk about ECDSA and how it is used in blockchain technology for cryptocurrencies.

ECDSA performs the same function as any other digital signing signature, but it does it more competently. Because ECDSA uses smaller keys to achieve the same level of security as any other digital signature method. ECDSA signatures and keys are shorter than RSA (Rivest Shamir Adleman) and more complex for cryptanalysis. 256-bit ECDSA can provide the same security level of 3072-bit RSA Signature. ECDSA is used to create ECDSA certificates, which are electronic documents that are used to verify the identity of the certificate's owner. Certificates include information on the key that was used to create the certificate, as well as information about the certificate's owner and the signature of the certificate's issuer, who is a verified trusted entity. This trusted issuer is usually a certificate authority that also possesses a signed certificate that can be traced back to the original issuing certificate authority through the chain of trust. In 1999, ECDSA was accepted as ANSI standard, and in the following year 2000, it was accepted by NIST and IEEE as well.

A little concept that requires your attention here!

ECDSA does not provide encryption or prevent someone from accessing or seeing your data, but it provides the assurance that your data is not tempered.

Working of ECDSA

Just like ECC, ECDSA is a form of public-key cryptography and uses cyclic groups of elliptic curves over finite fields and difficulty of elliptic curve discrete logarithmic problem. While understanding the ECDSA, you will encounter graphs, curves, and points repeatedly. In simple principle, you have a mathematical equation to draw a curve on a graph, and then you will choose a random point on that curve as a point of origin. After that, you will generate a random number and perform a tricky mathematical equation using random numbers and point of origin and you will get the second point on the curve, which will be your private key.

ECDSA basically works on the hash of the message instead of the message itself. Choosing a hash function is up to us and it is advised that a cryptographically secured hash function should be chosen. The hash of the message is ought to be truncated according to the ECDSA chosen parameters so that the bit length of the hash is the same as bits of n (the order of subgroup). ECDSA is based on the following equation:

$$y^2 = (x^3 + Ax + b) \bmod P$$

1. **Key generation in ECDSA:**

The private key (**Pr**) will be a random number depending on the size of the digital signature you want in bits. Generally, 160-bit ECDSA is used. This private key will be multiplied with the reference point or point of origin **G** you have chosen (mentioned above) so that we have

Pu = Pr * G where **G** is the point of reference in curve parameters.

2. Creating Signature:

The digital signature of the elliptic curve is represented by two values, **R** and **S**. In order to generate these two values you need, first you need a random value **K** and calculate **P = K*G** using point multiplication. This point value **P** on the curve will have its **(X, Y)** coordinates. The value to point **P** on coordinate **X** will be your **R**.

To calculate **S**, you need to calculate the secure hash of your message and the result will be donated as **Z.** using the equation below you can calculate **S**:

$S = K^{-1} (z + Pr * R) \bmod P$

Where **K** is a random number, **Z** is the hash of the message of sign, **Pr** is a private key, and **R** is x coordinate.

3. Verifying signature:

To verify the signature, you need public key and curve parameters with the following equation:

$P = S^{-1} * z * G + S^{-1} * R * Pu$

For a signature to be valid, **X** coordinates of point **P** must be equal to **R**.

Let's dig deep into the above equation and see how this equation can lead us to the verification.

We have $P = S^{-1} * z * G + S^{-1} * R * Pu$ …… Equation 1

But $Pu = Pr * G$

Substituting the values in **equation 1**, we get

$P = S^{-1} * z * G + S^{-1} * R * Pr * G$

Taking $\mathbf{S^{-1}}$ and **G** common, we have

$P = = S^{-1} (z + Pr * R) * G$

X coordinates of **P** must match **R** and we know **R** is the coordinate of **K * G**, so we have

$K * G = S^{-1} (z + Pr * R) * G$

By simplifying we will get

$K = S^{-1} (z + Pr * R)$

So, when we will invert the **K** and **S** in the above equation, we will get the same equation that was used for signing the message.

$S = k^{-1} (z + Pr * R) \bmod P$

That is the equation that has been used to generate the signature.

For more in-depth knowledge of math's behind ECDSA, kindly watch the following video: https://www.youtube.com/watch?v=QzUThXGRFBU&t=976s

ECDSA in Block Chain Technology

As we know, blockchain technology is the backbone of cryptocurrencies. Signatures are a vital part of decentralization and blockchain. It's not limited to transactions, but it is needed in decentralized exchanges, interacting, and multi-sig contracts. Digital currencies like Bitcoin and Ethereum use the ECDSA algorithm to ensure that only the rightful owner can spend the funds. Both Bitcoin and Ethereum use the standard elliptic curve of SECP256K1 with SHA256. Table 1 below will list the curves used by other digital currencies in blockchain technology. Further information about the curves used by other digital currencies can be found here.

Name	Signing Algorithm	Curve
Bitcoin	ECDSA	Secp256k1
Ethereum	ECDSA	Secp256k1
Cardano	EdDSA	Curve25519, secp256k1
Polka Dot	ECDSA, Schnorr, EdDSA	Curve25519, secp256k1, ristretto25519
XRP	ECDSA, EdDSA	Curve25519, secp256k1

Table 1: Curves used by various digital currencies

Although there are some researches conducted by cryptographic researchers that ECDSA performs slower than RSA at the lower security level, which means at smaller key length RSA outperforms the ECDSA in terms of verification and signing the signature. Contrastingly, ECDSA outperforms the RSA at a larger key length, and the use of a larger key length to strengthen the security will be the need of time in the future. There are some modified ECDSA as well to enhance their speed performance but it's not the scope of the topic at the moment.

However, ECDSA is now being used by various globally recognized companies and technologies. Bitcoin, websites, certificates, Apple (iMessage), TextSecure, and CrytoCat are top of the list companies and technologies which are using ECDSA. The complexity of ECDSA and its performance at higher key lengths has made it a choice of a security professional.