3.5 **Transactions**

**Transactions are the signed messages** which are generated by **External Accounts (EOA**). The messages are signed by the **Private keys of External Owned Accounts** then they are broadcasted to the network and through **consensus** they are stored in the blockchain of **Ethereum.**

In Ethereum Transaction is the only thing through which **state is changed** or any contract is executed.

If we think Ethereum is World computer and it manages the **singleton state**, transaction is the only input through which any **event is generated** or state is updated.

Transaction can only be generated by **Originated Account**; **Contract Account can’t generate it by self.**

Ethereum does not **run autonomously** (Contract don’t run on their own). Everything starts with Transaction.

It may possible in the reaction or output contract may be execute the transaction which are said by the Originated Account Transaction Execution.

Ethereum have 4 Types of Transactions.

**Signed Transaction**

Code execution

Value transfer

Transaction

CONTRACT

Transaction

EOA

EOA

EOA

**Unsigned Transaction or call**

CONTRACT

Internal TXN

Contract

EOA

Contract

Internal TXN

**Transaction Structure:**

* **Nonce 🡺** Ethereum have 2 Nonce 1 is for **proof-of-work** which maintain the difficulty level and 2 is present in the transactions which maintain the **sequence of transactions**. And also tells about the **confirm transactions.**
* **Gas Price 🡺** Gas price is the exchange rate of Ether
* **Gas Limit 🡺** It is the limit of gas required for transaction
* **Value 🡺** Amount of Ether
* **Recipient 🡺** receiver address (EOA or Contract)
* **Data 🡺** when the code execute the data (byte format) go in this field
* **V,R,S 🡺** These are the signatures field and the signatures execute(come) in this field.

**Recursive Length Prefix (RLP Encoding**) is the method which separate each and every field of **Transaction Structure.**

All numbers in Ethereum are encode as **big-endian integers (**It’s a scheme to store integers. in Ethereum floating or decimal numbers are not available)**, of lengths that are multiple of 8 bits.**

The data are separated by fixed length. No label is need for any field

**There is no “From” address in Transaction Structure**. The clients of Ethereum generated the address from the signatures (The address is generated through V,R,S fields) because to generate the address public key is required and by applying hash on public key address is generated.

**Transactions Nonce:**

Are the confirmed transactions. It will be the next one after the confirm transaction or contract.

In Bitcoin the model is **Unspent Transaction** is used (there is no issue of sequencing and gaps of transactions, therefor no nonce is required) but in Ethereum there is a different modal i.e. **Account-based modal.**

In bitcoin when the transaction propagates to the network then the client picks the unspent transaction of and verify through it. But in Ethereum this is not happening there is **account-base** state is mange so nonce is required to keep the sequence.

**What will be the benefit of this Nonce in Ethereum?**

1. **Sequence is maintained**

If I want to send a transaction in network without this nonce, I have to wait every transaction confirmation then I can do next transaction but with this Nonce I can do more than one transaction at the same time.

If some clients receive the 4 transaction before the 3 transaction then the clients keep the 4 transaction in the memory pool and wait the 3 transaction when they receive both they confirmed them so in this way both sequencing and gap issue is solved.

**None Prevent the Double Spending?**

If I made a transaction 1 ether to an account and also made a false intention to send back the 1 ether to my account and I assign the higher fee for 2 transaction then in the network the miner pick the higher transaction fee first and here nonce play the role to prevent double-spending .Nonce assign the every transaction a number if the 2 transaction first reach to the miner the miner will wait for the 1st transaction then execute according to nonce.

Nonce prevent the gaps btw Transactions and also the duplication.