**Tx Types**

**Introduction**

In this lesson, we will explore the different transaction types within the ZKsync VM and EVM ecosystems.

**/broadcast Folder**

When deploying to a ZKsync local node, a /broadcast folder will be created and it will contain detailed information about the **deployment transactions**. Inside this folder, you will find subfolders named after specific deployment chain IDs, such as **260** for ZKsync and **31337** for Anvil. These subfolders store the data of the transactions executed during the deployment process.

By examining both the run-latest.json file in these folders, we can observe different **transaction types** for each transaction within a chain. For instance, transactions on the Anvil chain might be labeled as type **0x2**, while those on the ZKsync chain will be of type **0x0**. Deploying a smart contract on the EVM without the --legacy flag results in a default transaction type of 0x2. Adding the --legacy flag changes it to type 0x0.

The EVM and ZKsync ecosystems support multiple transaction types to accommodate various Ethereum Improvement Proposals (EIPs). Initially, Ethereum had only one transaction type (0x0 legacy), but as the ecosystem evolved, multiple types were introduced through various EIPs. Subsequent types include type 1, which introduces an *access list* of addresses and keys, and type 2, also known as [EIP 1559](https://eips.ethereum.org/EIPS/eip-1559) transactions.

👀❗**IMPORTANT**  
This 0x2 type is the current default type for the EVM.

Additionally, ZKsync introduces its [unique transaction type](https://docs.zksync.io/zk-stack/concepts/transaction-lifecycle#eip-712-0x71), the type 113 (0x71 in hex), which can enable features like [account abstraction](https://docs.zksync.io/build/developer-reference/account-abstraction/).

💡 **TIP**  
The forge script command will work in some scenarios, but it’s not entirely clear where it might fail. For the purpose of this course, we will assume scripting does not work while working with ZKsync.

**Resources**

* [ZKsync documentation](https://docs.zksync.io/zk-stack/concepts/transaction-lifecycle#transaction-types) about transaction types
* [Cyfrin Blog on EIP-4844](https://www.cyfrin.io/blog/what-is-eip-4844-proto-danksharding-and-blob-transactions)

**Conclusion**

The ZKsync VM and EVM ecosystems support various transaction types to meet different EIP requirements. By examining deployment folders and understanding the use of flags like --legacy, we can effectively distinguish between these transaction types.