CREATE2: Predictable Address for Multi-Chain Contract

Selama ini kita deploy smart contract, contract address-nya random kan? Gimana kalau kita bisa menentukan address yang kita mau untuk contract kita? Atau bahkan bikin smart contract address-nya sama di berbagai chain? Deploy contract yang biasa kita lakukan itu menggunakan opcode (operation code) **CREATE**. Opcode itu simpelnya adalah perintah buat EVM (Ethereum Virtual Machine). Di sini akan dijelaskan sesimpel mungkin tentang kedua opcode ini.

Contents

Understanding CREATE

CREATE adalah opcode paling umum digunakan untuk deploy contract dengan mudah dan murah. Umumnya ini digunakan jika address contract tidak perlu diatur berdasarkan suatu kriteria. Cara CREATE menentukan address contract yang akan di-deploy adalah mengambil 20 bytes terakhir hash keccak256 dari:

- senderAddress (address yang deploy)
- nonce (angka yang menunjukkan jumlah transaksi dari sender)

Simplified syntax

```
keccak256(senderAddress, nonce);
```

Testing dengan JS/TS

```
import { ethers } from 'ethers';

const deployer = await ethers.getNamedSigner("deployer");
const nonce = await deployer.getTransactionCount();
const computedAddress = ethers.utils.getContractAddress({
    from: deployer.address,
    nonce: nonce, // modify nonce to get different contract address
});
console.log(`computed address: ${computedAddress}`);

const factory = await ethers.getContractFactory("KtbArbitrageur", deployer);
const contract = await factory.deploy(constructorParams);
console.log(`deployed address: ${await contract.getAddress()}`);
```

Untuk single-chain (hanya di 1 blockchain), nonce bisa diubah-ubah agar address sesuai keinginan (brute force looping angka nonce) dengan cek hasil contract address menggunakan ethers.utils.getContractAdress.

Question: Berdasarkan penjelasan di atas, apakah memungkinkan untuk menghasilkan contract address yang sama di EVM chain berbeda? (Penjelasan ada di bagian terakhir)

Fun fact: Di Metamask, ada fitur untuk customize transaction nonce. Boleh dicoba kalau udah paham.

CREATE2, Better than CREATE

CREATE2 adalah opcode yang menentukan contract address tanpa bergantung pada kondisi blockchain saat dan deployment. CREATE2 berguna untuk membuat contract multi-chain token atau smart wallet (wallet contract) sehingga contract address-nya sama semua. Kalau contract hanya akan ada di 1 chain, maka sebenarnya CREATE2 hanya memberikan kustomisasi contract address yang lebih flexible. Karena itu, gas cost untuk deployment dengan CREATE2 lebih mahal.

Cara CREATE2 menentukan address contract yang akan di-deploy adalah mengambil 20 bytes terakhir dari hash keccak256 dari:

- 0xFF (constant value untuk menghindari address sama dengan yang dihasilkan CREATE)
- senderAddress (address yang deploy)
- salt (nilai apa aja selama ngga lebih dari 32 bytes/32 huruf ASCII, bisa diubah agar contract address sesuai harapan)
- bytecode (bytecode dari contract yang mau di-deploy)

Simplified syntax

```
keccak256(0xFF, senderAddress, salt, bytecode);
```

CREATE2 biasa digunakan dalam suatu function smart contract (**factory contract**) yang bertugas untuk men-deploy contract lain. Berikut Solidity code-nya:

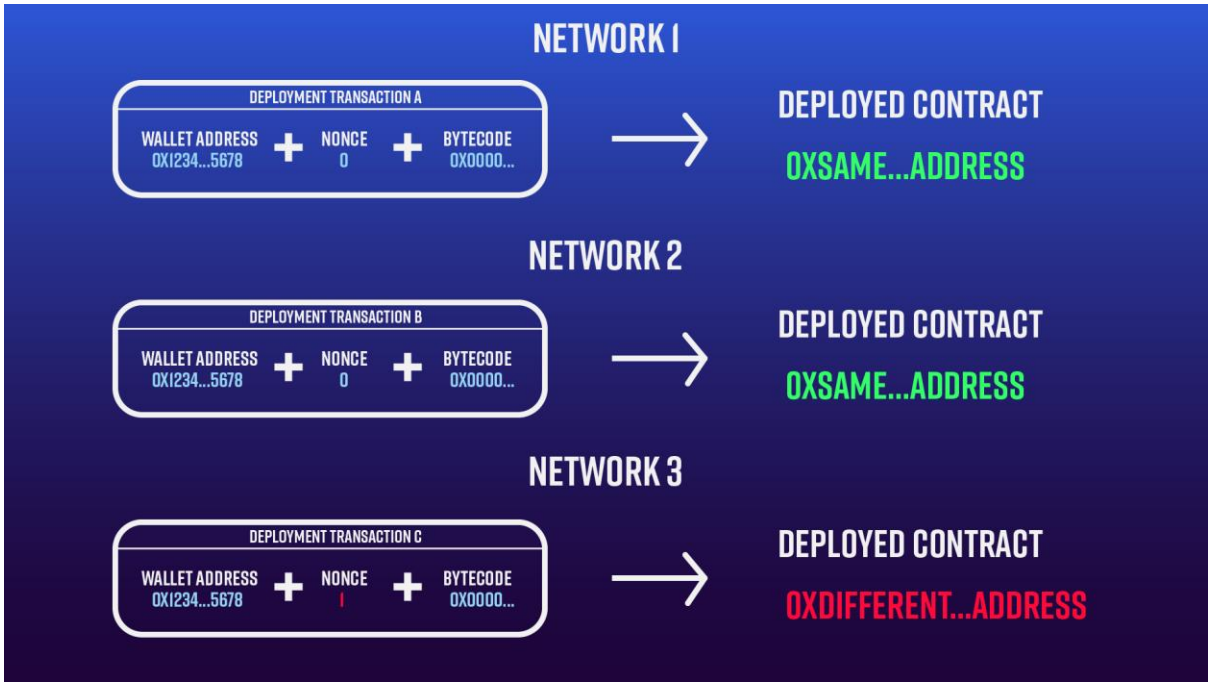
```
// 'ContractName' should be changed to the contract name that is going to be deployed
function deploy(uint _salt) external {
        ContractName _contract = new ContractName{
            salt: bytes32(_salt)
        }(msg.sender);
        emit Deploy(address(_contract));
    }

function getAddress(bytes memory bytecode, uint _salt) public view returns (address) {
        bytes32 hash = keccak256(
            abi.encodePacked(
                bytes1(0xff), address(this), _salt, keccak256(bytecode)
          )
        );
        // get last 20 bytes
        return address(uint160(uint256(hash)));
    }

function getBytecode(address _owner) public pure returns (bytes memory) {
        bytes memory bytecode = type(ContractName).creationCode;
        return abi.encodePacked(bytecode, abi.encode(_owner));
    }
```

Code di atas bisa dicoba di **Remix**. Kalau CREATE2 mau dicoba deploy dengan JS/TS, bisa pake baseContract.getDeployedCode dari ethers untuk dapat bytecode dari contract.

Fun fact: '0xFF' hanya untuk membedakan byte awal dari CREATE2 generated addresses, soalnya CREATE pakai '0x'.

Answer: CREATE multi-chain contract address

CREATE opcode bisa menghasilkan contract address yang sama di EVM chain berbeda hanya jika nonce dan deployer address sama. Jika suatu contract A sudah di deploy di network 1 dengan nonce 3 dan ingin ada contract address yang sama di network 2 TAPI nonce wallet deployer di network 2 sudah lebih dari 3, maka hal ini tidak memungkinkan.



CREATE3?

Selain CREATE dan CREATE2, ada juga CREATE3 tapi itu lebih advance lagi. Yang umum digunakan adalah CREATE dan CREATE2. Memahami CREATE2 opcode bisa menambah estetika contract address dan cross-chain functionality, juga bisa dimanfaatkan untuk serangan hack seperti Tornado cash hack 2023. Happy learning (or hacking).