Cavit Çakır 23657

CS480001 / SEC532
Blockchain: Security and Applications
Homework 2
A Secure Token Exchanger Contract

Table of Contents:

Information about Coins
- CavitCakirCoin
- CakirCavitCoin
  - Screenshot of the coins in the wallet as follows

Exchanger Contract
- Transfer of 10 coins
  - Screenshot of the coins in the wallet as follows

Appendix
- Code
  - contract CavitCakirCoin is ERC20PresetMinterPauser {
  - contract CakirCavitCoin is ERC20PresetMinterPauser {
  - contract Exchanger {

Information about Coins

- For both of the following coins, I used **ERC20PresetMinterPauser.sol from open-zeppelin**. I choose this particular preset in order to mint after the contract deployed.

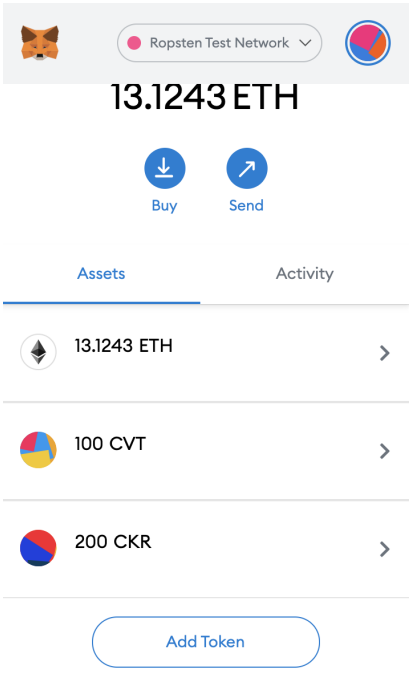
CavitCakirCoin

- Implemented CavitCakirCoin with minting 100 coins to my account at the constructor.
- Token symbol: CVT
- Decimals: 0
- Transaction Hash of deployment of CavitCakirCoin's Contract: 0x23198243e7b3a335060375a015874c3bc5c0745aa17c13cf0f20e8fc72876d58
- **Address of CavitCakirCoin:** 0xDA12c93F37921206Bb4fF1d5660bC93a2016B243

CakirCavitCoin

- Implemented CakirCavitCoin with minting 200 coins to my account at the constructor.
- Token symbol: CKR
- Decimals: 0
- Transaction Hash of deployment of CakirCavitCoin's Contract: 0x49a4ec0cb552fbd97d74a1988b536f10aa0938573e94bb40e59988e73e2e51ae
- **Address of CakirCavitCoin**: 0xd7fe2432BD1528B01312C486F256E9E06Af209E1

Screenshot of the coins in the wallet as follows



Exchanger Contract

- I implemented Exchanger Contract with one public function:
  transfer(address from_addrs, address to_addrs, uint256 amount)

  - This function requires:
    - The Exchanger has enough allowance in from_addrs
    - The Exchanger has enough balance in to_addrs
    - The Message Sender has enough balance in from_addrs.
  - When the function called, it first checks whether the requirements are satisfied. Afterward, transfers "amount" of from_addrs coins from msg.sender to own address. Then transfers "amount" of to_addrs coins to msg.sender.

- Transaction Hash of deployment of Exchanger Contract:
  0xb28a81d9fc87c740465553b2f108760a740881d58b3151acb3b2dea0ba44b212
- **Address of Exchanger:** 0xE447f069EA6B1c077823d16a8eb5C16461330790

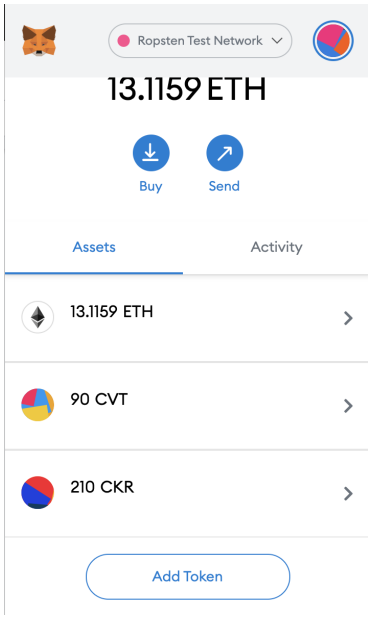
- In order to call the transfer function, the Exchanger has to have some coins in CavitCakirCoin and CakirCavitCoin. For that reason, I made 2 transactions by minting 1000 coins from each to the Exchanger.
  - Transaction of add 1000 coins to Exchanger in CavitCakirCoin Contract:
    0xb9f5b8efb6e180463878bcf2c7675745bf3fd5465f7ad84bd2b8e5ea536cc15a
  - Transaction of add 1000 coins to Exchanger in CakirCavitCoin Contract:
    0xe8e563c55b94b1526e654373d623787775b25cb8157cc3a49db431f9a8d430ed

Transfer of 10 coins

- In order to Exchanger transfer user's coins in the transfer function defined in the Exchanger Contract; the user should give allowance to Exchanger Contract's address to transfer his/her 10 coins.
  - Transaction Hash of Approve:
    0xf3ebfe6a401b695d1d316a27cc1fc46afa8d14c512b7459c32db58d76c36acc2

- By calling the transfer function defined in the Exchanger Contract, 10 coins are transferred from CVT coin to CKR coin.
  - Transaction Hash of transfer function call:
    0x68d33db0dd28ae7da118a1c2ef49867dacfd3a23b199e57ec26ce79683fa23c3

Screenshot of the coins in the wallet as follows



Appendix

Code

```
pragma solidity ^0.8.0;

import
"https://github.com/OpenZeppelin/openzeppelin-contracts/blob/master/contracts/token/ERC20/
presets/ERC20PresetMinterPauser.sol";


contract CavitCakirCoin is ERC20PresetMinterPauser {
    constructor() ERC20PresetMinterPauser("CavitCakirCoin", "CVT") {
        _mint(msg.sender, 100);
    }
    function decimals() public view virtual override returns (uint8) {
        return 0;
    }
}
```

```
contract CakirCavitCoin is ERC20PresetMinterPauser {
    constructor() ERC20PresetMinterPauser("CakirCavitCoin", "CKR") {
        _mint(msg.sender, 200);
    }
    function decimals() public view virtual override returns (uint8) {
        return 0;
    }
}


contract Exchanger {


    ERC20PresetMinterPauser public from_coin;
    ERC20PresetMinterPauser public to_coin;

    // transfer function requires that the Exchanger has enough balance and allowance in
from_addrs, and the Message Sender has enough balance in from_addrs.
    function transfer(address from_addrs, address to_addrs, uint256 amount) public {
        from_coin = ERC20PresetMinterPauser(from_addrs);
        to_coin = ERC20PresetMinterPauser(to_addrs);

        require(from_coin.balanceOf(msg.sender) >= amount, "ERC20: You dont have enough
balance in to_coin");
        require(from_coin.allowance(msg.sender, address(this)) >= amount, "ERC20: Exchanger
do not have enough allowance.");

        require(to_coin.balanceOf(address(this)) >= amount, "ERC20: Exchanger do not have
enough balance in to_coin.");

        from_coin.transferFrom(msg.sender, address(this), amount);
        to_coin.transfer(msg.sender, amount);
    }
}
```