[H-1] Storing the password on-chain makes it visible to anyone, and no longer private.

Description:

All data stored on-chain is visible to anyone, and can be read directly from the blockchain. The 's_password' variable is intended to be a private variable and only accessed through the getPassword function, which is intended to be only call by the owner of the contract.

Impact:

Anyone can read the private password, severly breaking the functionality of the protocol.

Proof of Concept:

Create a locally running chain [cmd]

anvil

Deploy the contract to the chain [cmd]

make deploy

[⠒] Compiling... No files changed, compilation skipped EIP-3855 is not supported in one or more of the RPCs used. Unsupported Chain IDs: 31337. Contracts deployed with a Solidity version equal or higher than 0.8.20 might not work properly. For more information, please see https://eips.ethereum.org/EIPS/eip-3855 Script ran successfully.

== Return == 0: contract PasswordStore 0x5FbDB2315678afecb367f032d93F642f64180aa3

Setting up 1 EVM.

==========

Chain 31337

Estimated gas price: 2.000000001 gwei

Estimated total gas used for script: 450223

Estimated amount required: 0.000900446000450223 ETH

==========

anvil-hardhat

✅ [Success]Hash: 0xc376757e1c78caf0e7c09709ac6231f1447098d3d784f06820a392c5a32d9fc8 Block: 2 Paid: 0.000039817434378978 ETH (45378 gas * 0.877461201 gwei)

anvil-hardhat

✅ [Success]Hash: 0xcef19463115dfbbd860b85ad14567ad772f9454fddcc9d9627b7479fd16c8966 Contract Address: 0x5FbDB2315678afecb367f032d93F642f64180aa3 Block: 1 Paid: 0.000295344000295344 ETH (295344 gas * 1.000000001 gwei)

✅ Sequence #1 on anvil-hardhat | Total Paid: 0.000335161434674322 ETH (340722 gas * avg 0.938730601 gwei)

==========

ONCHAIN EXECUTION COMPLETE & SUCCESSFUL.

Transactions saved to: /home/monish/3-passwordstore-audit/broadcast/DeployPasswordStore.s.sol/31337/run-latest.json

Sensitive values saved to: /home/monish/3-passwordstore-audit/cache/DeployPasswordStore.s.sol/31337/run-latest.json

Run the storage tool [cmd]

cast storage 0x5FbDB2315678afecb367f032d93F642f64180aa3 1 --rpc-url http://127.0.0.1:8545

output

0x6d7950617373776f726400000000000000000000000000000000000000000014

Convert the bytes-to-string[cmd]

cast parse-bytes32-string 0x6d7950617373776f726400000000000000000000000000000000000000000014

output :

myPassword

Recommended Mitigation:

Due to this , the overall architecture of the contract should be rethought. One could encrypt the password off-chain, and then store the encrypted password on-chain. This would require the user to remember another password off-chain to decrypt the password. However, you'd also likely want to remove the view function as you wouldn't want the user to accendentally send the transaction with
the password that decripts your password.

Likelihood & Impact:

-Impact : HIGH

-Likelihood : HIGH

-Severity : HIGH

[H-2] setPassword function has no access control , meaning that non-owner could call and change the password

Description:

The setPassword function is set to be an external function, however the natspec of the function and overall purpose of the smart contract is that

' This function allows only the owner to set the new password '.

```
 function setPassword(string memory newPassword) external {
    // @audit - There are no access controls
    s_password = newPassword;
    emit SetNetPassword();
}
```

Impact:

Anyone can set/change the password of the contract.

Proof of Concept:

Testcase :

```
function test_anyone_can_chg_password(address randomAddress) public {

    vm.assume(randomAddress != owner);
    vm.prank(randomAddress);
    string memory expectedPassword = "monish@2203";
    passwordStore.setPassword(expectedPassword);

    vm.prank(owner);
    string memory actualPassword = passwordStore.getPassword();
    assertEq(actualPassword, expectedPassword);
}
```

Recommended Mitigation:

Add an access control conditional to the 'setPassword' function.

```
    if (msg.sender != s_owner) {
        revert PasswordStore__NotOwner();
    }
```

Likelihood & Impact:

-Impact : HIGH

-Likelihood : HIGH

-Severity : HIGH

[I-1] The getPassword natspec indicates a parameter that doesn't exist, causing the natspec to the incorrect

Description:

@audit : here no parameter pass : the netspec is incorrect

/* * @notice This allows only the owner to retrieve the password. * @param newPassword The new password to set. [Correct here] */

```
function getPassword() external view returns (string memory) {
}
```

Likelihood & Impact:

-Impact : NONE

-Likelihood : NONE

-Severity : Informational/Gas/non-crit

Informational :hey this isn't a bug, but you should know...