Protocol Audit Report

Prepared by: Ikpong Joseph

Prepared by: Ikpong Joseph

Lead Auditors:

- Ikpong Joseph

Table of Contents

Protocol Summary

The Password Store protocol aims to help contract owners privately set their passwords, and prevent public access to the passwords.

Disclaimer

We make all effort to find as many vulnerabilities in the code in the given time period. We hold no responsibilities for the findings provided in this document. A security audit by the team is not an endorsement of the underlying business or product. The audit was time-boxed and the review of the code was solely on the security aspects of the Solidity implementation of the contracts.

Risk Classification

	Impact

		Impact		
		High	Medium	Low
	High	H	H/M	M
Likelihood	Medium	H/M	M	M/L
	Low	M	M/L	L

We use the CodeHawks severity matrix to determine severity. See the documentation for more details.

Audit Details

The findings in this report correlate with the commit hash

```
7d55682ddc4301a7b13ae9413095feffd9924566
```

From repo PasswordStore

Scope

```
./src/
-- PasswordStore.sol
```

Roles

- Owners: They set private passwords and retain the sole right to view their passwords

Executive Summary

The review was conducted 1 auditor, Ikpong Joseph, on the 19th of June, 2024. We timeboxed ourselves to find vulnerabilities and mitigations for 1 hour using manual review.

Issues found

3 vulnerabilities were discovered in the protocol. Vulnerabilities were classified as either High, Medium or Low. 1 of each was discovered in this audit.

Severity	**Number of Issues Found**
High	1
Medium	1
Low	0
Info	1

Severity	Number of Issues Found
Total	3

Findings

High

[H-1] Lack of access control: Anyone, and not only contract owner, can set password

Description

The `PasswordStore::setPassword` lacks proper access control check to verify that contract owner only should have the access and priviledge to set a new password on the contract.

Impact

This mitigates the protocols very essence of allowing only the owner to set a new password.

Proof of Concepts

Add this test to test suite at `test/PasswordStore.t.sol`

```
function test_non_owner_can_set_password_passes() public {
        console.log("Owner address: ", owner);
        console.log("Non-Owner address: ", OTHER_USER);
@>      vm.prank(OTHER_USER); // @audit non-owner proceeds to successfully
set password
        string memory expectedPassword = "non_user_password";
        passwordStore.setPassword(expectedPassword);
    }
```

Run test with

```
forge test --match-test test_non_owner_can_set_password_passes
```

This test passes with the following output.

```
Running 1 test for test/PasswordStore.t.sol:PasswordStoreTest
[PASS] test_non_owner_can_set_password_passes() (gas: 23458)
Logs:
  Owner address:  0x1804c8AB1F12E6bbf3894d4083f33e07309d1f38
  Non-Owner address:  0xAc567AAce42Cd47ec531A3581773bcE03bbE4118

Traces:
  [23458] PasswordStoreTest::test_non_owner_can_set_password_passes()
    ├─ [0] console::log("Owner address: ", DefaultSender:
[0x1804c8AB1F12E6bbf3894d4083f33e07309d1f38]) [staticcall]
```

```
    │   └─ ← ()
    ├─ [0] console::log("Non-Owner address: ", OTHER_USER:
[0xAc567AAce42Cd47ec531A3581773bcE03bbE4118]) [staticcall]
    │   └─ ← ()
    ├─ [0] VM::prank(OTHER_USER:
[0xAc567AAce42Cd47ec531A3581773bcE03bbE4118])
    │   └─ ← ()
    ├─ [6686] PasswordStore::setPassword("non_user_password")
    │   ├─ emit SetNetPassword()
    │   └─ ← ()
    └─ ← ()
```

It goes to show that `PasswordStore::setPassword` without the proper access control allows random user to set password in the system.

Recommended mitigation

Add access control to `PasswordStore::setPassword` to ensure only contract owner can set new password.

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

Then try running the test with `forge test --match-test test_non_owner_can_set_password_passes`. It should fail this time.

Medium

[M-1] The `PasswordStore::s_password` stored as state variabe is not private on blockchain records, letting non-owner retrieve password.

Description

The `PasswordStore::s_password` state variable, though a "`private`" state variable, is actually not private and can be retrieved on-chain.

Impact

This mitigates the protocols very essence of allowing only the owner to set a new password, storing a password and others should not be able to access the password.

Proof of Concepts

Run an anvil network with `anvil`. Then

```
make deploy
```

Copy contract address, and use as arg in `cast storage <contract address> <storage slot index> --rpc-url <network endpoint>`

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

This will return a hex value as `0x6d7950617373776f726400000000000000000000000000000000000000000014`

Use `cast parse-bytes32-string 0x6d7950617373776f726400000000000000000000000000000000000000000014` to decipher.

This will be received `"myPassword"`

This is equal to password as set in the deploy script by contract owner.

```
 function run() public returns (PasswordStore) {
        vm.startBroadcast();
        PasswordStore passwordStore = new PasswordStore();
@>      passwordStore.setPassword("myPassword");
        vm.stopBroadcast();
        return passwordStore;
    }
```

Recommended mitigation

Passwords can be stored off-chain, with proper encryption and salting techniques.

Informational

[I-1] Wrong natspec documentation can lead to misguided use of PasswordStore::getPassword function

Description

The natspec documentation describes a param to be required to interact with the `PasswordStore::getPassword` function. The function takes no such parameter.

```
 /*
     * @notice This allows only the owner to retrieve the password.
@>   * @param newPassword The new password to set.
     */
    function getPassword() external view returns (string memory) {
        if (msg.sender != s_owner) {
            revert PasswordStore__NotOwner();
        }
        return s_password;
    }
```

Impact

This wrong natspec documentation can lead to wrong interaction with this function, where they should have been none

Recommended mitigation

The following lines should be removed from the `PasswordStore::getPassword` natspec documentation.

```diff
- @param newPassword The new password to set.
```