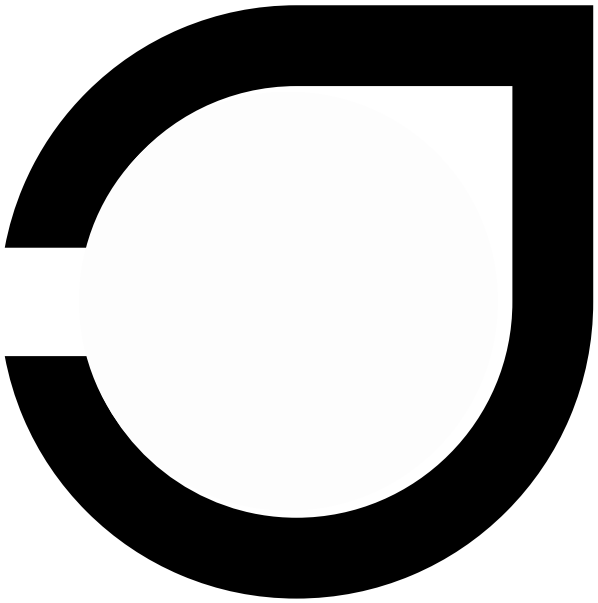

Protocol Audit Report

Version 1.0

mahadihassanriyadh

May 21, 2024

Protocol Audit Report

Md. Mahadi Hassan Riyadh

May 21, 2024

Prepared by: Md. Mahadi Hassan Riyadh Lead Security Researcher: - Md. Mahadi Hassan Riyadh

Table of Contents

- Table of Contents
- Protocol Summary
- Disclaimer
- Risk Classification
- Audit Details
  - Scope
  - Role
- Executive Summary
  - Issues found
- Findings
  - High
    - [H-1] Storing the password on-chain makes it visible to anyone, and no longer private
    - [H-2] `PasswordStore::setPassword()` function has no access control, meaning anyone can change the password
  - Medium
  - Low
  - Informational
    - [I-1] The `PasswordStore::getPassword()` netspec indicates a parameter that doesn't exist, causing the netspec to be incorrect
  - Gas

Protocol Summary

PasswordStore is a protocol dedicated to storage and retrieval of a user's passwords. The protocol is designed to be used by a single user, and is not designed to be used by multiple users. Only the owner should be able to set and access this password.

Disclaimer

The Perspectree team makes all effort to find as many vulnerabilities in the code in the given time period, but holds 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		
		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

Commit Hash:

```
7d55682ddc4301a7b13ae9413095feffd9924566
```

Scope

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

Role

- Owner: The user who can set the password and read the password.
- Outsiders: No one else should be able to set or read the password.

Executive Summary

Add some notes about how the audit went, types of things you found, etc. We spent X hours with Z auditors using Y tools. etc

Issues found

Severity	Number of issues found
High	2
Medium	0
Low	1
Info	1
Gas Optimizations	0
Total	0

Findings

High

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

Likelihood & Impact: - Impact: High - Likelihood: High - Severity: High / Critical

Description:

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

We show one such method of reading any data off chain below.

Impact:

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

Proof of Concept:

The below test case shows how anyone can read the password directly from the blockchain.

1. Create a locally running chain `bash make anvil`
2. Deploy the contract to the chain `bash make deploy`
3. Run the storage tool `bash cast storage <contract_address>` or `bash cast storage <contract_address> <storage_slot>` You should get an output similar to the below: `bash 0x6d7950617373776f726400000000000000000000000000000000000000000000`
4. Copy the output and run the below command to convert it to readable text `bash cast parse-bytes32-string <output>` You should get the password in plain text: `bash 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 accidentally send a transaction with the password that decrypts your password.

[H-2] `PasswordStore::setPassword()` function has no access control, meaning anyone can change the password

Likelihood & Impact: - Impact: High - Likelihood: High - Severity: High / Critical

Description:

The natspec comment for the `PasswordStore::setPassword()` function states `This function allows only the owner to set a new password` but there is no access

control in the function to enforce this. This means that anyone can call this function and change the password.

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

Impact:

Anyone can set/change the password of the contract, severely breaking the functionality of the contract.

Proof of Concept:

Add the following to the `PasswordStore.t.sol` file:

Code

```
    function test_anyone_can_set_password(address _randomAddress)
        public {
        vm.assume(_randomAddress != owner);
        vm.startPrank(_randomAddress);
        string memory expectedPassword = "myNewPassword";
        passwordStore.setPassword(expectedPassword);

        vm.startPrank(owner);
        string memory actualPassword = passwordStore.getPassword();

        assertEq(actualPassword, expectedPassword);
    }
```

Recommended Mitigation:

Add an access control modifier to the `setPassword()` function to ensure only the owner can change the password.

```
    modifier onlyOwner() {
        if(msg.sender != owner) {
            revert PasswordStore__NotOwner();
        }
        _;
    }
```

Medium

Low

Informational

[I-1] The `PasswordStore::getPassword()` netspec indicates a parameter that doesn't exist, causing the netspec to be incorrect

Likelihood & Impact: - Impact: None - Likelihood: Low - Severity: Informational / Gas / Non-critical

Description:

```
1      /**
2       * @notice This allows only the owner to retrieve the password.
3       * @audit-doc there is no parameter needed for this function, so
           the below comment is incorrect and should be removed
4       * @param newPassword The new password to set.
5       */
6      function getPassword() external view returns (string memory) {
```

Impact:

The natspec comment is incorrect and could cause confusion for developers reading the code.

Recommended Mitigation:

Remove the incorrect natspec comment.

```
1  -    * @param newPassword The new password to set.
2  +
```

Gas