Protocol Audit Report

Version 1.0

Cyfrin.io

October 30, 2024

Protocol Audit Report

Iepet

October 29, 2024

Prepared by: [Iepet]

Table of Contents

        * 3. [I-1] The `PasswordStore::getPassword` natspec indicates a parameter that doesn't exist, causing the natspec to be incorrect.
    - Gas

Protocol Summary

PasswordStore is a protocol that leds the owner of the contract to store a password and modify with the function SetPassword and get it back with get Password. And the password should only be setable and retrievable by the owner itself. # Disclaimer

We make 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

Scope

./src/PasswordStore.sol

Roles

- Owner: The user that can set and get the Password
- Outsiders: Anyone that is not the owner, who should not be able to set or read the password

Executive Summary

- Analysis done following UpdrafTCyfrin.io courses.

Issues found

Severity	Number of issues found
High	2
Medium	0
Low	0
Info	1
Total	3

Findings

High

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

Description:

All data stored on-chain is visible to anyone. Therefore the private `PasswordStore::s_password'` variable which is intended to be private and only readble by owner via `PasswordStore::getPassword` function, is actually readble by everyone.

Impact:

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

Proof of Concept:

- Create a locally running chain `Make Anvil`
- Deploy the contract to the chain `Make Deploy`
- Use the Foundry Storage tool `cast storage <ADDRESS_HERE> 1 --rpc-url http://127.0.0.1:8545`
- You will get `0x6d7950617373776f7264000000000000000000000000000000000000000000014`
- You can then parse it with `cast parse-bytes32-string 0x6d7950617373776f72640000000000000`
- And get the 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 stored password. However, you're also likely want to remove the view function as you wouldn't want the user to accidentally send a transaction with this decryption key.

2. [H-2] `PasswordStore::setPassword` has no access controls, meaning a non-owner could change the password

Description: The `PasswordStore::setPassword` function is set to be an `external` function, however the purpose of the smart contract and function's natspec indicate that `This function allows only the owner to set a new password.`

"'js function setPassword(string memory newPassword) external { // @Audit - There are no Access Controls. s_password = newPassword; emit SetNewPassword(); }" '

Impact: Anyone can set/change the stored password, severly breaking the contract's intended functionality

Proof of Concept: Add the following to the PasswordStore.t.sol test file:

' ' 'js 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 conditional to `PasswordStore::setPassword`.

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

Medium

Low

Informational

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

Description: ' ' ' / @notice This allows only the owner to retrieve the password. @> * @param newPassword The new password to set. */ function getPassword() external view returns (string memory) {} ' ' '

```
The `PasswordStore::getPassword` function signature is `getPassword()` while the na
```

Impact: The natspec is incorrect

Recommended Mitigation: Remove the incorrect natspec line.

' ' 'diff / @notice This allows only the owner to retrieve the password. - * @param newPassword The new password to set. */ ' ' '

Gas