PasswordStore Initial Audit Report

Version 0.1

Cyfrin.io

April 16, 2024

PasswordStore Audit Report

Kostiantyn Osadchii

April 16, 2024

PasswordStore Audit Report

Prepared by: Kostiantyn Osadchii Lead Auditors:

- Kostiantyn Osadchii

Assisting Auditors:

- None

Table of contents

See table

    - Issues found
- Findings
    - High
        * [H-1] Storing he password on-chain makes it visible to anyone, and no longer private
        * [H-2] `PasswordStore::setPassword` has no access control, meaning that no-owner can change the password
    - Informational
        * [I-1] The `PasswordStore::getPassword` natspec indicates a parameter that doesn‘t exist, causing the natspec to be incorrect.

About Kostiantyn Osadchii

The Smart Contract Security Researcher on the start of his journey!!!

Disclaimer

The Kostiantyn Osadchii team makes all effort to find as many vulnerabilities in the code in the given time period, but holds no responsibilities for the 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

Audit Details

The findings described in this document correspond the following commit hash:

```
2e8f81e263b3a9d18fab4fb5c46805ffc10a9990
```

Scope

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

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.

Roles

- Owner: Is the only one who should be able to set and access the password.

For this contract, only the owner should be able to interact with the contract.

Executive Summary

Issues found

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

Severity	Number of issues found
Total	3

Findings

High

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

Description: All data stored on-chain can be read directly from the blockchain by anybody. The `PasswordStore::s_password` variable is intended to be private and only accessed through the `PasswordStore::getPassword` function.

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

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

Proof of Concept: The bellow test shows anyone can read the password directly rom the blockchain.

1. Create a locally running chain

```
make anvil
```

2. Deploy the contract to the chain

```
make deploy
```

3. Run the storage tool

We use 1 because that's the storage slot of `s_password` in the contract.

```
cast storage <ADDRESS_HERE> 1 --rpc-url http://127.0.0.1:8545
```

You'll get an output that looks like this:

`0x6d7950617373776f726400000000000000000000000000000000000000000014`

You can then parse that hex to a string with:

```
cast parse-bytes32-string 0
    x6d7950617373776f726400000000000000000000000000000000000000000014
```

And get an output of:

```
myPassword
```

Recommended Mitigation: You need somehow to encrypt your password so it become unreadable for person who don‘t know the encryption keys.

[H-2] `PasswordStore::setPassword` has no access control, meaning that no-owner can change the password

Description: The `PasswordStore::setPassword` must be executable only by the owner of a contract, as it written in natspec: `This function allows only the owner to set a new password.`

```
function setPassword(string memory newPassword) external {
@>      //@audit - access control must be in this line.
        s_password = newPassword;
        emit SetNetPassword();
    }
```

Impact: Anyone can set or change the password of the contract, breaking the contract functionality.

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

Code

```
function test_anyone_can_set_password(address randomAddress) public {
        vm.assume(randomAddress != owner);
        vm.prank(randomAddress);
        string memory expectedPassword = "myNewPassword";
        passwordStore.setPassword(expectedPassword);
        vm.prank(owner);
        string memory actualPassword = passwordStore.getPassword();
        assertEq(actualPassword, expectedPassword);
    }
```

Recommended Mitigation: Add the access control conditional to the `PasswordStore::setPassword` function

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

Informational

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

Description:

```
1   /*
2        * @notice This allows only the owner to retrieve the password.
3   @>   * @param newPassword The new password to set.
4        */
5       function getPassword() external view returns (string memory) {
```

The `PasswordStore::getPassword` function signature is `getPassword()` and according to natspec it should be `getPassword(string)`

Impact: The natspec is incorrect

Recommended Mitigation: Remove the incorrect natspec line.

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