Staykov Security

Protocol Audit Report

Version 1.0

Cyfrin.io

March 10, 2025

Protocol Audit Report

Staykov

March 10, 2025

Prepared by: Staykov Lead Auditors: - Staykov

Table of Contents

* [I-#] The `PasswordStore::getPassword` netspec indicates a parameter that doens't exist, causing the natspec to be incorect.
*

Protocol Summary

Protocol made for storage and retrival of a user's passwords. Used by miltiple users. Only YOU can access and change your password.

Disclaimer

The YOUR_NAME_HERE 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

Comit Hash

```
2e8f81263b3...
```

Scope

```
.src.
----folder
```

Roles

-Owner -Outsiders # Executive Summary Spend x hours for auditing this protocol. ## Issues found |Severity|Numbers of issues found| |———|———————| |High| | 2 |Medium| | 0 |Low| | 0 |Info| | 1 |Total| | 3 ## Findings ## High ### [H-1] Storing the password on-chain makes it visible to anoyone, an no longer private

Description: All data storend on-chain is visible to aboyone, and can be read directly from the blockchain. The `PasswordStore::s_password` variable is intended to be a private variable and onlt accessed through the `PasswordStore::getPassword` function, whuch 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: Anoyone can read the private password, severly breaking the functionality of the protocol.

Proof of Concept: (Proof of concept/code)

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

1. Creat a localy running chain

make anvil

2. Deploy the smart contract to the chain make deploy
3. Run the storage tool cast storage 0x5FbDB2315678afecb367f032d93F642f64180aa3 1 –rpc-url http://127.0.0.1:8545 we use 1 , because on that sotrage place is stored s_password in the contract

output -> 0x6d7950617373776f726400000000000000000000000000000000000000000014 WE parse that hex value to string with cast oarse-bytes32-string hexcode

and get output myPassword

Recommended Mitigation: The whole contract should be rethinked, because it is not writen up to the good standarts and best practices

–Impact - high -likelihood - High -Critical ### [H-2] `PasswordStore::setPassword` has no accesscontrols, meaning non-owner could change the password

Description: func is set to external, but natspec of the function and overal purpose of the smart contract is that `This function only allow owner to set the new password`

```
javascript -> 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, severely breaking the contract intended functionality

Proof of Concept: add the following to the passwrdStore.t.sol test file

```
1    function test_anyone_can_set_password(address randomAddress) public{
2          vm.assume(randomAddress != owner);
3          vm.prank(randomAddress);
4          string memory expectedPassword = "myNewPassword";
5          passwordStore.setPassword(expectedPassword);
6
7          vm.prank(owner);
8          string memory actualPassword = passwordStore.getPassword();
9          assertEq(actualPassword, expectedPassword);
10     }
```

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

```
1  if(msg.sender =! owner){
2      revert PasswordStore__NotOwner();
3  }
```

–Impact - high -likelihood - High -Critical ## Medium ## Low ## Informational ### [I-#] The `PasswordStore::getPassword` netspec indicates a parameter that doens't exist, causing the natspec to be incorect.

Description:

```
1  /*
2       * @notice This allows only the owner to retrieve the password.
3       * @param newPassword The new password to set.
4       */
5       // @audit
```

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

The `PasswordStore::getPassword` function signature is `getPassword()` which the natspec say it should be `getPassword(String)`

Impact: The natspec is incorect

Recommended Mitigation: Remove the incorrect natspec line

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

–Impact - none -likelihood - High -Severity none