



Protocol Audit Report

Version 1.0

Norbert Orgován

January 15, 2024

Protocol Audit Report

Norbert Orgovan

January 15, 2024

Prepared by: Orgovan & Churros

Lead Auditors: - Norbert Orgován

Table of Contents

- Table of Contents
- Protocol Summary
- Disclaimer
- Risk Classification
- Audit Details
 - Scope
 - Roles
- Executive Summary
 - Issues found
- Findings
 - High
 - * [H-1] Storing the pwd on-chain makes it visible to anyone, and no longer private
 - * [H-2] `PasswordStore::setPassword` has no access control, meaning a non-owner could change the pwd.
 - Informational
 - * [I-1] The `PasswordStore::getPassword` natspec indicates a parameter that does not exist, causing the natspec to be incorrect.

Protocol Summary

PasswordStore is a protocol dedicated to storing and retriving a user’s pwd. The protocol is designed to be used by a single user, not multiple users. Only the owner should be able to set and access this pwd.

Disclaimer

The Orgovan & Churros 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
Likelihood	High	H	H/M	M
	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 de-tails.

Audit Details

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

1 2e8f81e263b3a9d18fab4fb5c46805ffc10a9990

Scope

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

- Solc Version: 0.8.18
- Chain(s) to deploy contract to: Ethereum

Roles

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

Executive Summary

Some notes about how the audit went, types of findings, etc. We spent X hours with Y auditors, using Z tools.

Issues found

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

the user to remember another pwd off-chain to decrypt the pwd. However, you would also likely want to remove the view function as you would not want the user to accidentally send a transaction with the pwd that decrypts your pwd.

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

Description: The `PasswordStore::setPaswword` function is set to be an `external` function, however, the natspec of the function and overall the purpose of the smart contract is that `This` function allows only the owner to set a `new` pwd.

```
1 function setPassword(string memory newPassword) external {
2   @> // @audit - There are no access controls
3     s_password = newPassword;
4     emit SetNetPassword();
5 }
```

Impact: Anyone can set/change the pwd of the contract, severely breaking the contract's intended functionality.

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

Code

```
1 function test_anyone_can_set_password(address randomAddress) public
2 {
3   vm.assume(randomAddress != owner);
4   vm.prank(randomAddress);
5   string memory expectedPassword = "myNewPassword";
6   passwordStore.setPassword(expectedPassword);
7
8   vm.prank(owner);
9   string memory actualPassword = passwordStore.getPassword();
10  assertEq(actualPassword, expectedPassword);
11 }
```

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

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

Informational

[I-1] The PasswordStore::getPassword natspec indicates a parameter that does not 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(), but the natspec says it should be getPassword(string).

Impact: The natspec is incorrect.

Proof of Concept: -

Recommended Mitigation: Remove the incorrect natspec line.

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