



PasswordStore Initial Audit Report

Version 0.1

mhng

August 20, 2025

PasswordStore Audit Report

mhng

August 18, 2025

PasswordStore Audit Report

Prepared by: mhng

Lead Auditors: mhng

- mhng

Assisting Auditors:

- None

Table of contents

See table

- PasswordStore Audit Report
- Table of contents
- About mhng
- Disclaimer
- Risk Classification
- Audit Details
 - Scope
- Protocol Summary
 - Roles

- Executive Summary
 - Issues found
- Findings
 - High
 - * [H-1] Passwords stored on-chain are visible to anyone, not matter solidity variable visibility
 - * [H-2] `PasswordStore::setPassword` is callable by anyone
 - Informational Findings
 - * [I-1] The `PasswordStore::getPassword` natspec indicates a parameter that doesn't exist, causing the natspec to be incorrect

About mhng

Disclaimer

We are making all efforts to find as many vulnerabilities in the code in the given time period, but we are not holding any responsibilities for the the findings provided in this document. A security audit 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

Audit Details

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

```
1 2e8f81e263b3a9d18fab4fb5c46805ffc10a9990
```

Scope

```
1 src/  
2 --- 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
Total	3

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 is callable by anyone

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

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

Impact: Anyone can set/change the password of the contract.

Proof of Concept:

Add the following to the `PasswordStore.t.sol` test suite.

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

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

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

Informational Findings

[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 natspec for the function `PasswordStore::getPassword` indicates it should have a parameter with the signature `getPassword(string)`. However, the actual function signature is `getPassword()`.

Impact: The natspec is incorrect.

Recommended Mitigation: Remove the incorrect natspec line.

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