Prepared by: Idriis Perrin Lead Auditors:

- xxxxxxx

Table of Contents

Protocol Summary

Protocol does X, Y, Z \what does the protocol do When a user creates or updates their password, the protocol hashes the password using a strong cryptographic hash function (e.g., SHA-256). Hashing transforms the password into a fixed-length string of characters, making it computationally infeasible to reverse the process and obtain the original 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

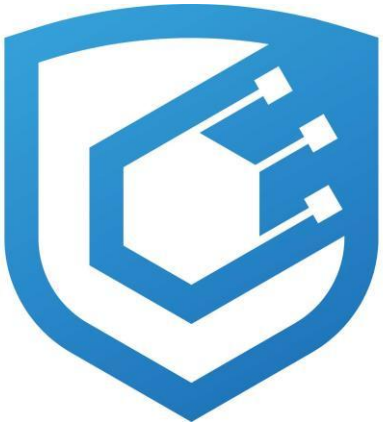
		Impact		
		High	Medium	Low
Likelihood	High	H	H/M	M
	Medium	H/M	M	M/L
	Low	M	M/L	L

We use the CodeHawks severity matrix to determine severity. See the documentation for more details.

Audit Details

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

`7d55682ddc4301a7b13ae9413095feffd9924566`



Scope

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

Roles

- Owner : The user who can read the password and use the password
- Outsiders: no one else should be able to read or use the password.

Executive Summary

*add some notes about how the audit went. *we spent x hours with z auditors using y tools. etc*

Issues found

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

Findings

High

[H-1] Title (Storing the password on-chain makes it visible to anyone and no longer private(root cause + impact)

) Storing the password on-chain makes it visible to anyone and no longer private(root cause + impact)

Description: All data stored on-chain is visible to anyone, and can be read directly from the blockchain. The 'PasswordStore::s_password' is intended to be a private variable and only accessed through the 'PasswordStore::get_password function', which is intended to be only called bythe owner of the contract.

We show one method of reading any data bekow Impact:Anyone can read the private password, severely breaking the functionality of the protocol Proof of concept: (Proof of code) The below test case shows how anyone can read from the blockchain.

1. Create a locally running blockchain with anvil
2. Deploy the contract to the local chain (anvil) --make deploy--
3. RUn the storage tool We run 1 because that is the slot of s_password in the contract -- cast storage <address_here> 1 --rpc-url http://127.0.0.1:8545

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 passwogd 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-1] Title (Storing the password on-chain makes it visible to anyone and no longer private(root cause + impact)

) Storing the password on-chain makes it visible to anyone and no longer private(root cause + impact)

Description: All data stored on-chain is visible to anyone, and can be read directly from the blockchain. The 'PasswordStore::s_password' is intended to be a private variable and only accessed through the 'PasswordStore::get_password function', which is intended to be only called bythe owner of the contract.

We show one method of reading any data bekow Impact:Anyone can read the private password, severely breaking the functionality of the protocol Proof of concept: (Proof of code) The below test case shows how anyone can read from the blockchain.

4. Create a locally running blockchain with anvil
5. Deploy the contract to the local chain (anvil) --make deploy--
6. RUn the storage tool We run 1 because that is the slot of s_password in the contract -- cast storage <address_here> 1 --rpc-url http://127.0.0.1:8545

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 passwogd 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` (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 natspec of the function and overall purpose of the smart contract is that `This function allows only the owner to set a new password.`

```
javascript

function setPassword(string memory newPassword) external {
@>    //@audit there are no access controls
                s_password = newPassword;
                emit SetNetPassword();
                }
```

Impact: anyone can change the password and this severely breaks the intended functionality of the contract. **Proof of concept: Recomended mitigation:**

Medium

Low

Informational

[I-1] `Password::getPassword` - Incorrect Parameter in NatSpec

Root Cause + Impact: The NatSpec documentation indicates a parameter that does not exist, leading to incorrect documentation.

Description:

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

The function signature of `Passwordstore::getpassword` is `getpassword`, while the NatSpec documentation specifies it should be `getPassword(string)`.

Impact: The NatSpec documentation is incorrect, potentially causing confusion for developers.

Proof of Concept:
(If applicable, provide any relevant proof of concept)

Recommended Mitigation: Remove the incorrect line:
```diff

- *@param newpasword the new password the set.
```