

# Smart Contract Security Audit Report

Passwordstore-Audit

**Auditor:** Owoyemi Yusuf

**Project:** Passwordstore-audit

**Date:** February 1st, 2026

## Executive Summary

This report presents the results of a security assessment of the PasswordStore smart contract. The audit focuses on identifying vulnerabilities that could compromise confidentiality, access control, or deviation from intended contract behavior.

## Findings Summary

ID	Title	Severity
H-1	On-chain password storage exposes private data	High
H-2	Missing access control in setPassword	High
I-1	Incorrect NatSpec documentation	Informational

H-1 Storing the password on-chain makes it visible to anyone

## Description

All data stored on-chain is publicly accessible. The password state variable, although intended to be private, can be read directly from contract storage.

## Impact

Any user can retrieve the password, completely breaking confidentiality guarantees.

# Proof of Concept

anvil

## **Recommended Mitigation**

Avoid storing sensitive data on-chain. If unavoidable, encrypt the password off-chain and store only the encrypted value. Remove any view function that could expose decrypted data.

## H-2 setPassword lacks access control

## Description

The `setPassword` function is externally callable without ownership checks, allowing any address to overwrite the stored password.

# Proof of Concept

```
function test_anyone_can_set_password(address randomAddress) public {
    vm.assume(randomAddress != owner);
    vm.prank(randomAddress);
    passwordStore.setPassword("myNewPassword");

    vm.prank(owner);
    assertEq(passwordStore.getPassword(), "myNewPassword");
}
```

### **Recommended Mitigation**

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

## I-1 Incorrect NatSpec documentation

### Description

The NatSpec documentation for getPassword references a parameter that does not exist in the function signature.

### Impact

Documentation inconsistency only.