



Cyberscope

# Audit Report

## Hide Coin

April 2024

Repository <https://github.com/secretblock/hidecoin-smart-contracts>

Commit 88a843aa5ff46559ceebda0180b06e76c8e8b5de

Audited by © cyberscope

# Table of Contents

|                                |           |
|--------------------------------|-----------|
| <b>Table of Contents</b>       | <b>1</b>  |
| <b>Review</b>                  | <b>2</b>  |
| Audit Updates                  | 2         |
| Source Files                   | 2         |
| <b>Findings Breakdown</b>      | <b>3</b>  |
| <b>Diagnostics</b>             | <b>4</b>  |
| BC - Blacklists Addresses      | 5         |
| Description                    | 5         |
| Recommendation                 | 5         |
| ST - Stops Transactions        | 7         |
| Description                    | 7         |
| Recommendation                 | 7         |
| MT - Mints Tokens              | 8         |
| Description                    | 8         |
| Recommendation                 | 8         |
| RSW - Redundant Storage Writes | 10        |
| Description                    | 10        |
| Recommendation                 | 10        |
| L19 - Stable Compiler Version  | 12        |
| Description                    | 12        |
| Recommendation                 | 12        |
| <b>Functions Analysis</b>      | <b>13</b> |
| <b>Inheritance Graph</b>       | <b>14</b> |
| <b>Flow Graph</b>              | <b>15</b> |
| <b>Summary</b>                 | <b>16</b> |
| <b>Disclaimer</b>              | <b>17</b> |
| <b>About Cyberscope</b>        | <b>18</b> |

## Review

|                |   |
|----------------|---|
| Contract Name  | HideCoin  |
| Repository     | <a href="https://github.com/secretblock/hidecoin-smart-contracts">https://github.com/secretblock/hidecoin-smart-contracts</a>                               |
| Commit         | 88a843aa5ff46559ceebda0180b06e76c8e8b5de  |
| Testing Deploy | <a href="https://bscscan.com/address/0xbc4d19d6ab09fafe720db304fac0e9ec941a932b">https://bscscan.com/address/0xbc4d19d6ab09fafe720db304fac0e9ec941a932b</a> |
| Symbol         | HIDE  |
| Decimals       | 18  |

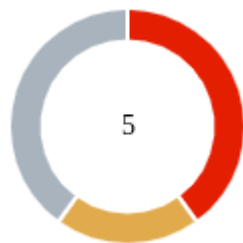
## Audit Updates

|               |            |
|---------------|------------|
| Initial Audit | 1 May 2024 |
|---------------|------------|

## Source Files

|                        |  |
|------------------------|--|
| Filename               | SHA256   |
| contracts/HideCoin.sol | 4503d12c480e8b4ce1355728f3a3ed2a4fc6dd83dc57f482aee785b96691f9f8 |

## Findings Breakdown



|                       |   |
|-----------------------|---|
| ● Critical            | 2 |
| ● Medium              | 1 |
| ● Minor / Informative | 2 |

| Severity              | Unresolved | Acknowledged | Resolved | Other |
|-----------------------|------------|--------------|----------|-------|
| ● Critical            | 0          | 2            | 0        | 0     |
| ● Medium              | 0          | 1            | 0        | 0     |
| ● Minor / Informative | 0          | 2            | 0        | 0     |

# Diagnostics

● Critical   ● Medium   ● Minor / Informative

| Severity | Code | Description              | Status       |
|----------|------|--------------------------|--------------|
| ●        | BC   | Blacklists Addresses     | Acknowledged |
| ●        | ST   | Stops Transactions       | Acknowledged |
| ●        | MT   | Mints Tokens             | Acknowledged |
| ●        | RSW  | Redundant Storage Writes | Acknowledged |
| ●        | L19  | Stable Compiler Version  | Acknowledged |

## BC - Blacklists Addresses

|             |                            |
|-------------|----------------------------|
| Criticality | Critical                   |
| Location    | contracts/HideCoin.sol#L81 |
| Status      | Acknowledged               |

### Description

The contract owner has the authority to massively blacklist addresses thereby preventing these addresses from transactions. The owner may take advantage of it by calling the `setBots` function.

```
function setBots(address[] calldata accounts, bool value)
public onlyOwner {
    for (uint256 i = 0; i < accounts.length; i++) {
        if (
            (accounts[i] != address(this)) &&
            (!_isExcludedFromLimits[accounts[i]])
        ) _setBots(accounts[i], value);
    }
}
```

### Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

## ST - Stops Transactions

|                    |                            |
|--------------------|----------------------------|
| <b>Criticality</b> | Critical                   |
| <b>Location</b>    | contracts/HideCoin.sol#L52 |
| <b>Status</b>      | Acknowledged               |

### Description

The transactions are initially disabled for all users excluding the authorized addresses. The owner can enable the transactions for all users. Once the transactions are enable the owner will not be able to disable them again.

```
require(  
    launched || from == ownr || to == ownr || to == address(0xdead),  
    "HideCoin: Not launched."  
);
```

### Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions. Some suggestions are:

- Introduce a multi-sign wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.



## MT - Mints Tokens

|             |                            |
|-------------|----------------------------|
| Criticality | Medium                     |
| Location    | contracts/HideCoin.sol#L43 |
| Status      | Acknowledged               |

### Description

The contract owner has the authority to mint tokens until the `maxSupply` is reached. The owner may take advantage of it by calling the `mint` function. As a result, the contract tokens will be highly inflated.

```
function mint(address to, uint256 amount) public onlyOwner {
    uint256 supply = _currentSupply + amount;

    require(supply <= maxSupply, "HideCoin: Exceeds max supply");
    _mint(to, amount);
    _currentSupply += amount;
    emit Mint(to, amount);
}
```

### Recommendation

The team should carefully manage the private keys of the owner's account. We strongly recommend a powerful security mechanism that will prevent a single user from accessing the contract admin functions.

Temporary Solutions:

These measurements do not decrease the severity of the finding

- Introduce a time-locker mechanism with a reasonable delay.
- Introduce a multi-signature wallet so that many addresses will confirm the action.
- Introduce a governance model where users will vote about the actions.

Permanent Solution:

- Renouncing the ownership, which will eliminate the threats but it is non-reversible.

## RSW - Redundant Storage Writes

|             |                               |
|-------------|-------------------------------|
| Criticality | Minor / Informative           |
| Location    | contracts/HideCoin.sol#L72,81 |
| Status      | Acknowledged                  |

### Description

The contract modifies the state of the following variables without checking if their current value is the same as the one given as an argument. As a result, the contract performs redundant storage writes, when the provided parameter matches the current state of the variables, leading to unnecessary gas consumption and inefficiencies in contract execution.

```
function excludeFromLimits (
    address[] calldata accounts,
    bool value
) public onlyOwner {
    for (uint256 i = 0; i < accounts.length; i++) {
        _excludeFromLimits(accounts[i], value);
    }
}

function setBots(address[] calldata accounts, bool value) public
onlyOwner {
    for (uint256 i = 0; i < accounts.length; i++) {
        if (
            (accounts[i] != address(this)) &&
            (!_isExcludedFromLimits[accounts[i]])
        ) _setBots(accounts[i], value);
    }
}
```

### Recommendation

The team is advised to implement additional checks within to prevent redundant storage writes when the provided argument matches the current state of the variables. By incorporating statements to compare the new values with the existing values before

proceeding with any state modification, the contract can avoid unnecessary storage operations, thereby optimizing gas usage.

## L19 - Stable Compiler Version

|             |                           |
|-------------|---------------------------|
| Criticality | Minor / Informative       |
| Location    | contracts/HideCoin.sol#L2 |
| Status      | Acknowledged              |

### Description

The `^` symbol indicates that any version of Solidity that is compatible with the specified version (i.e., any version that is a higher minor or patch version) can be used to compile the contract. The version lock is a mechanism that allows the author to specify a minimum version of the Solidity compiler that must be used to compile the contract code. This is useful because it ensures that the contract will be compiled using a version of the compiler that is known to be compatible with the code.

```
pragma solidity ^0.8.20;
```

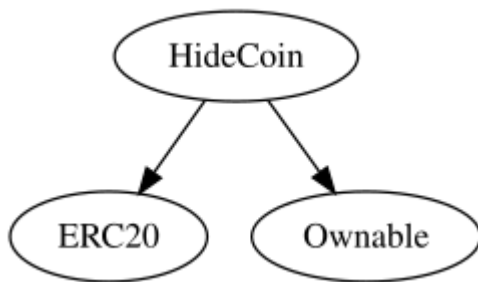
### Recommendation

The team is advised to lock the pragma to ensure the stability of the codebase. The locked pragma version ensures that the contract will not be deployed with an unexpected version. An unexpected version may produce vulnerabilities and undiscovered bugs. The compiler should be configured to the lowest version that provides all the required functionality for the codebase. As a result, the project will be compiled in a well-tested LTS (Long Term Support) environment.

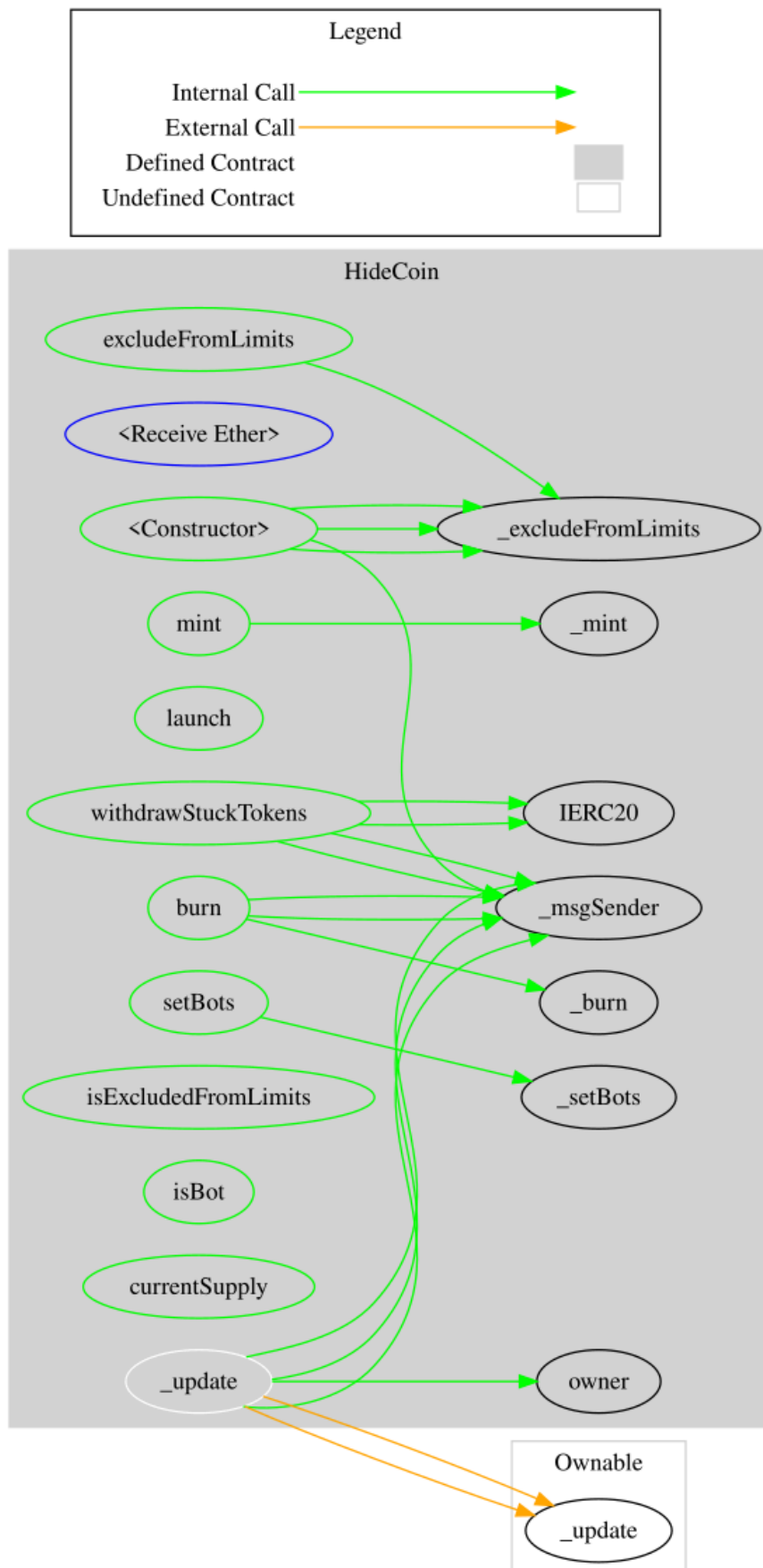
## Functions Analysis

| Contract | Type                 | Bases             |            |                  |
|----------|----------------------|-------------------|------------|------------------|
|          | Function Name        | Visibility        | Mutability | Modifiers        |
|          |                      |                   |            |                  |
| HideCoin | Implementation       | ERC20,<br>Ownable |            |                  |
|          |                      | Public            | ✓          | ERC20<br>Ownable |
|          |                      | External          | Payable    | -                |
|          | burn                 | Public            | ✓          | -                |
|          | mint                 | Public            | ✓          | onlyOwner        |
|          | launch               | Public            | ✓          | onlyOwner        |
|          | withdrawStuckTokens  | Public            | ✓          | onlyOwner        |
|          | excludeFromLimits    | Public            | ✓          | onlyOwner        |
|          | setBots              | Public            | ✓          | onlyOwner        |
|          | isExcludedFromLimits | Public            |            | -                |
|          | isBot                | Public            |            | -                |
|          | currentSupply        | Public            |            | -                |
|          | _update              | Internal          | ✓          |                  |
|          | _excludeFromLimits   | Internal          | ✓          |                  |
|          | _setBots             | Internal          | ✓          |                  |

## Inheritance Graph



## Flow Graph





## Summary

Hide Coin contract implements a token mechanism. This audit investigates security issues, business logic concerns and potential improvements. There are some functions that can be abused by the owner like stop transactions, mint tokens and massively blacklist addresses. if the contract owner abuses the mint functionality, then the contract will be highly inflated. A multi-wallet signing pattern will provide security against potential hacks. Temporarily locking the contract or renouncing ownership will eliminate all the contract threats. The team has acknowledged the findings.

## Disclaimer

The information provided in this report does not constitute investment, financial or trading advice and you should not treat any of the document's content as such. This report may not be transmitted, disclosed, referred to or relied upon by any person for any purposes nor may copies be delivered to any other person other than the Company without Cyberscope's prior written consent. This report is not nor should be considered an "endorsement" or "disapproval" of any particular project or team. This report is not nor should be regarded as an indication of the economics or value of any "product" or "asset" created by any team or project that contracts Cyberscope to perform a security assessment. This document does not provide any warranty or guarantee regarding the absolute bug-free nature of the technology analyzed, nor do they provide any indication of the technologies proprietors' business, business model or legal compliance. This report should not be used in any way to make decisions around investment or involvement with any particular project. This report represents an extensive assessment process intending to help our customers increase the quality of their code while reducing the high level of risk presented by cryptographic tokens and blockchain technology.

Blockchain technology and cryptographic assets present a high level of ongoing risk. Cyberscope's position is that each company and individual are responsible for their own due diligence and continuous security. Cyberscope's goal is to help reduce the attack vectors and the high level of variance associated with utilizing new and consistently changing technologies and in no way claims any guarantee of security or functionality of the technology we agree to analyze. The assessment services provided by Cyberscope are subject to dependencies and are under continuing development. You agree that your access and/or use including but not limited to any services reports and materials will be at your sole risk on an as-is where-is and as-available basis. Cryptographic tokens are emergent technologies and carry with them high levels of technical risk and uncertainty. The assessment reports could include false positives, false negatives and other unpredictable results. The services may access and depend upon multiple layers of third parties.

# About Cyberscope

Cyberscope is a blockchain cybersecurity company that was founded with the vision to make web3.0 a safer place for investors and developers. Since its launch, it has worked with thousands of projects and is estimated to have secured tens of millions of investors' funds.

Cyberscope is one of the leading smart contract audit firms in the crypto space and has built a high-profile network of clients and partners.



**The Cyberscope team**

<https://www.cyberscope.io>