WHITE HAT DAO

White Hat DAO Gov Token

Audit Report

By tamjid0x01
www.whitehatdao.com

Date: 20/07/2023

WHITE HAT DAO

Table of Contents

WHITE HAT DAO

Disclaimer

This is a limited report on our findings based on our analysis, in accordance with good industry practice as at the date of this report, in relation to cybersecurity vulnerabilities and issues in the framework and algorithms based on smart contracts, the details of which are set out in this report.

In order to get a full view of our analysis, it is crucial for you to read the full report. While we have done our best in conducting our analysis and producing this report, it is important to note that you should not rely on this report and cannot claim against us on the basis of what it says or doesn't say, or how we produced it, and it is important for you to conduct your own independent investigations before making any decisions.

By reading this report or any part of it, you agree to the terms of this disclaimer. If you do not agree to the terms, then please immediately cease reading this report, and delete and destroy any and all copies of this report downloaded and/or printed by you. This report is provided for information purposes only and on a non-reliance basis, and does not constitute investment advice. No one shall have any right to rely on the report or its contents, and White Hat DAO and its affiliates (including holding companies, shareholders, subsidiaries, employees, directors, officers and other representatives)

White Hat DAO owes no duty of care towards you or any other person, nor does White Hat DAOmake any warranty or representation to any person on the accuracy or completeness of the report. The report is provided "as is", without any conditions, warranties or other terms of any kind except as set out in this disclaimer, and White Hat DAOhereby excludes all representations, warranties, conditions and other terms (including, without limitation, the warranties implied by law of satisfactory quality, fitness for purpose and the use of reasonable care and skill) which, but for this clause, might have effect in relation to the report.

Except and only to the extent that it is prohibited by law, White Hat DAOhereby excludes all liability and responsibility, and neither you nor any other person shall have any claim against White Hat DAO, for any amount or kind of loss or damage that may result to you or any other person (including without limitation, any direct, indirect, special, punitive, consequential or pure economic loss or damages, or any loss of income, profits, goodwill, data, contracts, use of money, or business interruption, and whether in delict, tort (including without limitation negligence), contract, breach of statutory duty, misrepresentation (whether innocent or negligent) or otherwise under any claim of any nature whatsoever in any jurisdiction) in any way arising from or connected with this report and the use, inability to use or the results of use of this report, and any reliance on this report. The analysis of the security is purely based on the smart contracts alone. No applications or operations were reviewed for security. No product code has been
reviewed.

WHITE HAT DAO

This report is based on the scope of materials and documentation provided for a limited review at the time provided. Results may not be complete nor inclusive of all vulnerabilities. The review and this report are provided on an as-is, where-is, and as-available basis. You agree that your access and/or use, including but not limited to any associated services, products, protocols, platforms, content, and materials, will be at your sole risk. Blockchain technology remains under development and is subject to unknown risks and flaws. The review does not extend to the compiler layer, or any other areas beyond the programming language, or other programming aspects that could present security risks. A report does not indicate the endorsement of any particular project or team, nor guarantee its security.

No third party should rely on the reports in any way, including for the purpose of making any decisions to buy or sell a product, service or any other asset. To the fullest extent permitted by law, we disclaim all warranties, expressed or implied, in connection with this report, its content, and the related services and products and your use thereof, including, without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

We do not warrant, endorse, guarantee, or assume responsibility for any product or service advertised or offered by a third party through the product, any open source or third-party software, code, libraries, materials, or information linked to, called by, referenced by or accessible through the report, its content, and the related services and products, any hyperlinked websites, any websites or mobile applications appearing on any advertising, and we will not be a party to or in any way be responsible for monitoring any transaction between you and any third-party providers of products or services. As with the purchase or use of a product or service through any medium or in any environment, you should use your best judgment and exercise caution where appropriate.

FOR AVOIDANCE OF DOUBT, THE REPORT, ITS CONTENT, ACCESS, AND/OR USAGE THEREOF, INCLUDING ANY ASSOCIATED SERVICES OR MATERIALS, SHALL NOT BE CONSIDERED OR RELIED UPON AS ANY FORM OF FINANCIAL, INVESTMENT, TAX, LEGAL, REGULATORY, OR OTHER ADVICE.

Executive Summary

White Hat DAO security analyst Tamjid has conducted this security assessment. There was 1 contract reviewed during this audit. The smart contract was manually reviewed and analyzed with static and dynamic analysis tools.

Based on our audit, the contract is Secured.

We found 1 major issue related to the codebase. We found some minor and informational issues. More details have been provided as below.

The code had good comments and documentation. The code uses the natspec standard for comments. Commenting can make the maintenance of the code much easier, as well as helping make finding bugs faster. Also, commenting is very important when writing functions that may be used in other contracts.

Here is a high level overview of the issues found in this report:

Total Issues	2 (1 Resolved)
Critical Risk Issues	0 (0 Resolved)
High Risk Issues	(0 Resolved)
Medium Risk Issues	1(0 Resolved)
Low Risk Issues	1(0 Resolved)
Informational	0(0 Resolved)

Summary of Findings

The most prominent among our findings were a uint64 underflow that gets wrapped around due to the nature of the type in golang(ring) and the usage of pointers in the codebase that would need a more secure approach to ensure maintainability and code health in the future.

Issue ID	Issue Title	Categories of Severity	Status
WHD1	Use `super._safeTransfer` instead of `super._Transfer`	Medium	Fixed
WHD2	Use **`super._safeMint instead of super._Mint`**	Low	Acknowledge

Introduction

This security assessment has been prepared for The White Hat DAO by tamjid0x01 to find any safety concerns, bad practices and vulnerabilities in the source code as well as any contract dependencies in scope that were not part of an officially recognized library. Comprehensive tests have been conducted, utilizing manual code review, static and dynamic analysis and techniques.

The auditing process pays special attention to the following considerations:

- Testing the smart contracts against both common and uncommon attack vectors
- Assessing the codebase to ensure compliance with current best practices and industry standards
- Ensuring contract logic meets the specifications and intentions of the client. Cross referencing contract structure and implementation against similar smart contracts produced by industry leaders
- Thorough line-by-line manual review of the entire codebase by industry experts
- Reviewing unit tests to ensure full coverage of the codebase

The Project Summary, Scope, Audit Details and Methodology of the audit is described in the following sections.

The security assessment resulted in findings that ranged from critical to informational. We recommend addressing these findings to ensure a high level of security standards. These can be found in the Findings section of the report.

WHITE HAT DAO

Project Summary

Project	WHD Token
Description	WhiteHatDao Gov token. Users can use WHD tokens to vote or to delegate their voting power to any address.
Website	https://whitehatdao.com/
Platform	Ethereum
Language used	Solidity
Codebase	./Contracts/Token.sol
Commit	

Project Scope

White Hat DAO was commissioned by The White Hat DAO to perform security assessments on smart contracts as below:

Source Code	Acknowledgement	nSloc
Token.sol	Accepted	168

Audit Details

Delivery Date	20/07/2023
Submission Date	18/07/2023
Key Components	WHD Token

Methodology

White Hat DAO auditing team reviewed the code base of White Hat DAO from 16.07.2023 through 18.07.2023 . The team conducted the assessment based on the Code of Token.sol

The team launched the audit by analyzing the specifications of the project and the key areas of interest, and went through the documentation.

The code was manually reviewed in an attempt to identify potential vulnerabilities and verify adherence to the specification, best practices and proper use of the language itself. The unit tests were examined to ensure full coverage of the code. Automated analysis of the codebase was performed and results were reviewed.

The smart contracts were scanned for commonly known and more specific vulnerabilities. Following is the list of commonly known vulnerabilities that were considered during the audit of the smart contract:

- Access Control
- Arbitrary token minting
- Asset's integrity
- Authority Control attack
- Business Logics Review
- Centralization of power
- Client synchronization

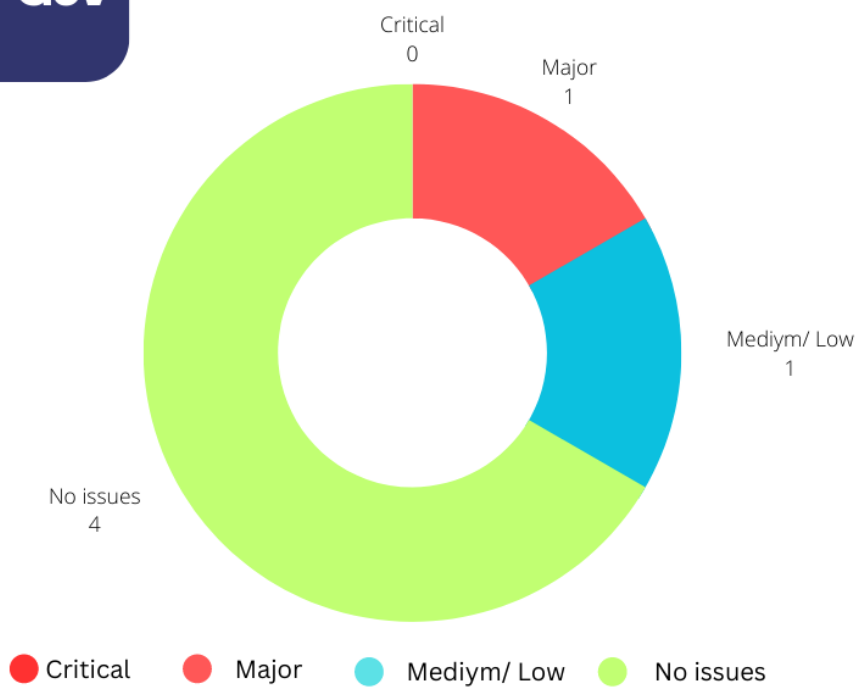
- Code clones, functionality duplication
- Conditional Completion attack
- Consensus splits
- Costly Loop
- Data Consistency
- Data integrity loss
- Denial of service / logical oversights
- Deployment Consistency
- DoS with (Unexpected) Throw
- DoS with Block Gas Limit
- ERC20 API violation
- Escrow manipulation
- Explicit visibility of functions state variables
- False top-up Vulnerability
- Falsified messages
- Floating Points and Numerical Precision
- Functionality Checks
- Gas Usage, Gas Limit and Loops
- Implicit visibility level
- Injection type attacks
- Integer Overflow and Underflow attacks
- Invalid incoming messages
- Kill-Switch Mechanism
- Logic Flaws
- Mishandled exceptions and call stack limits
- Number rounding errors
- Operation Trails & Event Generation
- Outdated data in cache
- Ownership Takeover
- Redundant fallback function
- Reentrancy
- Remote code execution
- Reordering attack
- Replay attacks

- Repository Consistency
- Scoping and Declarations
- Second pre-image attacks on Merkle Trees
- Short address attack
- Style guide violation
- TimeStamp Dependence attack
- Token Supply manipulation
- Transaction Ordering Dependence attack
- tx.origin Authentication
- Unchecked external call
- Unchecked math
- Uninitialized Storage Pointers
- Unsafe external calls
- Unsafe type Inference
- User Balances manipulation

Findings

There have been no major or critical issues related to the codebase and all findings listed here are minor and informational. Additional information on these vulnerabilities is provided in the following sections.

White Hat DAO Gov Token



www.whitehatdao.com

Critical - 0 | Major - 1 | Medium Issue - 1 | Low Issue - 0 | No Issue -0

Severity Definitions

Severity	Definitions
Critical	These vulnerabilities have a catastrophic impact on the security of the project. They can lead to loss, data manipulation, take over, etc. It is strongly recommended to fix these vulnerabilities.
High	These vulnerabilities have a significant impact on the security of the project. They can lead to loss, data manipulation, take over, etc. It is strongly recommended to fix these vulnerabilities.
Medium	These vulnerabilities are important to fix. These vulnerabilities alone can't lead to asset loss or data manipulation. However, medium vulnerabilities can be chained to create a more severe vulnerability. It is highly recommended to review and address these vulnerabilities.
Low	These vulnerabilities are mostly related to outdated, unused code snippets and don't have significant impact on execution. It is suggested that the project party evaluate and consider whether these vulnerabilities need to be fixed.
Informational	These vulnerabilities don't pose an immediate risk, but are relevant to security best practices. They could be code style violations and informational statements that don't affect smart contract execution. They may be able to be ignored.

Critical Vulnerabilities

No Low severity vulnerabilities were found.

Major Vulnerabilities

No Major severity vulnerabilities were found.

Medium Vulnerabilities

WHD1 - `super._safeTransfer` FUNCTION CAN BE CALLED INSTEAD OF `super._transfer` FUNCTION FOR TRANSFERRING GOVERNANCE Token.

Description of Issue:

When the following `WhiteHatDAOToken._transfer` function is called, the `super._transfer` function is called to transfer the WhiteHatDAOToken for tokenId from the from address to the to address. When the address corresponds to a contract, calling the super._transfer function does not check if the receiving contract supports the ERC20 protocol; if not supported, the transferred Token can be locked and cannot be retrieved. To ensure that the receiving contract supports the ERC20 Token protocol, please consider calling the `super._safeTransfer` function instead of the `super._transfer` function in the `WhiteHatDAOToken._transfer` function.

```solidity
    function _transfer(
    address from,
    address to,
    uint256 amount
    ) internal virtual override {
    ...
    super._transfer(from, to, amount); // @audit-issue
    }

```

Recommendation:
My recommendation is to ensure that the receiving contract supports the ERC20 Token protocol, please consider calling the super._safeTransfer.

- `SafeERC20` is a wrapper around the interface that eliminates the need to handle boolean return values.
- Wrappers around ERC20 operations that throw on failure (when the token contract returns false). Tokens that return no value (and instead revert or throw on failure) are also supported, non-reverting calls are assumed to be successful. To use this library you can add a `` `using SafeERC20 for ERC20`; `` statement to your contract, which allows you to call the safe operations as `` `super.safeTransfer(…)` ``, etc.

WHD 2 : `super._safeMint` FUNCTION CAN BE CALLED INSTEAD OF `super._mint` FUNCTION FOR MINTING WhiteHatDAOToken Token.

Description of Issue:

Calling the following **`WhiteHatDAOToken._mint`** functions will mint the WhiteHatDAOToken for amount to the to address. _mint Functions call the **`super._mint`** function. If the to address corresponds to a contract, calling the super._mint function does not check if the receiving contract supports the ERC20 protocol; if not supported, the minted Token can be locked and cannot be retrieved. To make sure that the receiving contract supports the ERC20 protocol, please consider calling the **`super._safeMint`** function instead of the super._mint function in the **`WhiteHatDAOToken._mint`** functions.

```solidity
```

```
function _mint(
    address to,
    uint256 amount
    ) internal override(ERC20, ERC20Votes) {
    super._mint(to, amount);
    }```
```

Recommendation:

My recommendation is to make sure that the receiving contract supports the ERC20 protocol, please consider calling the super._safeMint function instead of the **`super._mint`** function in the **`WhiteHatDAOToken._mint`** functions.

Team comment:

The mint function is available only to contract deployer and an EOA wallet will be used to deploy the contract. Not sure if this is applicable or not sure if this would be concerning. The contract does not have any proxies. So it should be safe enough in my opinion. Please advise if I am wrong. thanks.

Low Vulnerabilities

No Low severity vulnerabilities were found.

Conclusion

White Hat DAO has worked with the White Hat DAO TEAM to perform this audit. There were 1 smart contract reviewed during this audit. The smart contracts were manually reviewed and analyzed with static and dynamic analysis tools. The findings of these reviews were provided in this report.

The code was commented well. Comments are helpful in understanding the overall architecture and the logic flow of the contracts.

The audit has found 1 medium and 1 informational vulnerability.

<Update: *18/07/2023 - The White Hat DAO team has reviewed the necessary recommendations and addressed issues raised in this report.*>

Change Log

- 18-07-2023 - Initial report
- 19-07-2023 - Final report