



**WSB Token (wsbchef.com)**

**SMART CONTRACT AUDIT**

**25.02.2021**

**Made in Germany by Chainsulting.de**



## Table of contents

1. Disclaimer.....	3
2. About the Project and Company .....	4
2.1 Project Overview.....	5
3. Vulnerability & Risk Level .....	6
4. Auditing Strategy and Techniques Applied.....	7
4.1 Methodology .....	7
4.2 Used Code from other Frameworks/Smart Contracts .....	8
4.3 Tested Contract Files .....	9
4.4 Metrics / CallGraph.....	10
4.5 Metrics / Source Lines .....	11
4.6 Metrics / Capabilities .....	12
4.7 Metrics / Source Unites in Scope .....	12
5. Scope of Work .....	13
5.1 Manual and Automated Vulnerability Test.....	14
5.2 Verify Claims .....	15
6. Executive Summary.....	16
7. Deployed Smart Contract .....	17



## 1. Disclaimer

The audit makes no statements or warranties about utility of the code, safety of the code, suitability of the business model, investment advice, endorsement of the platform or its products, regulatory regime for the business model, or any other statements about fitness of the contracts to purpose, or their bug free status. The audit documentation is for discussion purposes only.

The information presented in this report is confidential and privileged. If you are reading this report, you agree to keep it confidential, not to copy, disclose or disseminate without the agreement of wsbchef.com. If you are not the intended receptor of this document, remember that any disclosure, copying or dissemination of it is forbidden.

Major Versions / Date	Description
0.1 (22.02.2021)	Layout
0.5 (23.02.2021)	Automated Security Testing Manual Security Testing
0.8 (24.02.2021)	Testing SWC Checks
1.0 (25.02.2021)	Summary and Recommendation

## 2. About the Project and Company

**Company address:** Anon

**Website:** <https://wsbchef.com>

**Twitter:** <https://twitter.com/BetsPodcast>

**Youtube:** <https://www.youtube.com/channel/UCuvw3AIJmW-db0OTWNd1I-Q>

**Telegram:** <https://t.me/satoshistreetbets>

## 2.1 Project Overview

The WSB movement for financial democracy started on reddit and it became a global phenomenon. This is why decided to create a token to give our community a tool to grow their personal assets by farming the WSB token.



### 3. Vulnerability & Risk Level

Risk represents the probability that a certain source-threat will exploit vulnerability, and the impact of that event on the organization or system. Risk Level is computed based on CVSS version 3.0.

Level	Value	Vulnerability	Risk (Required Action)
Critical	9 – 10	A vulnerability that can disrupt the contract functioning in a number of scenarios, or creates a risk that the contract may be broken.	Immediate action to reduce risk level.
High	7 – 8.9	A vulnerability that affects the desired outcome when using a contract, or provides the opportunity to use a contract in an unintended way.	Implementation of corrective actions as soon as possible.
Medium	4 – 6.9	A vulnerability that could affect the desired outcome of executing the contract in a specific scenario.	Implementation of corrective actions in a certain period.
Low	2 – 3.9	A vulnerability that does not have a significant impact on possible scenarios for the use of the contract and is probably subjective.	Implementation of certain corrective actions or accepting the risk.
Informational	0 – 1.9	A vulnerability that have informational character but is not effecting any of the code.	An observation that does not determine a level of risk

## 4. Auditing Strategy and Techniques Applied

Throughout the review process, care was taken to evaluate the repository for security-related issues, code quality, and adherence to specification and best practices. To do so, reviewed line-by-line by our team of expert pentesters and smart contract developers, documenting any issues as there were discovered.

### 4.1 Methodology

The auditing process follows a routine series of steps:

1. Code review that includes the following:
  - i. Review of the specifications, sources, and instructions provided to Chainsulting to make sure we understand the size, scope, and functionality of the smart contract.
  - ii. Manual review of code, which is the process of reading source code line-by-line in an attempt to identify potential vulnerabilities.
  - iii. Comparison to specification, which is the process of checking whether the code does what the specifications, sources, and instructions provided to Chainsulting describe.
2. Testing and automated analysis that includes the following:
  - i. Test coverage analysis, which is the process of determining whether the test cases are actually covering the code and how much code is exercised when we run those test cases.
  - ii. Symbolic execution, which is analysing a program to determine what inputs causes each part of a program to execute.
3. Best practices review, which is a review of the smart contracts to improve efficiency, effectiveness, clarify, maintainability, security, and control based on the established industry and academic practices, recommendations, and research.
4. Specific, itemized, actionable recommendations to help you take steps to secure your smart contracts.

## 4.2 Used Code from other Frameworks/Smart Contracts

1. SafeMath.sol (0.5.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v2.3.0/contracts/math/SafeMath.sol>

2. IERC20.sol (0.5.0)

<https://github.com/OpenZeppelin/openzeppelin-contracts/blob/v2.3.0/contracts/token/ERC20/IERC20.sol>



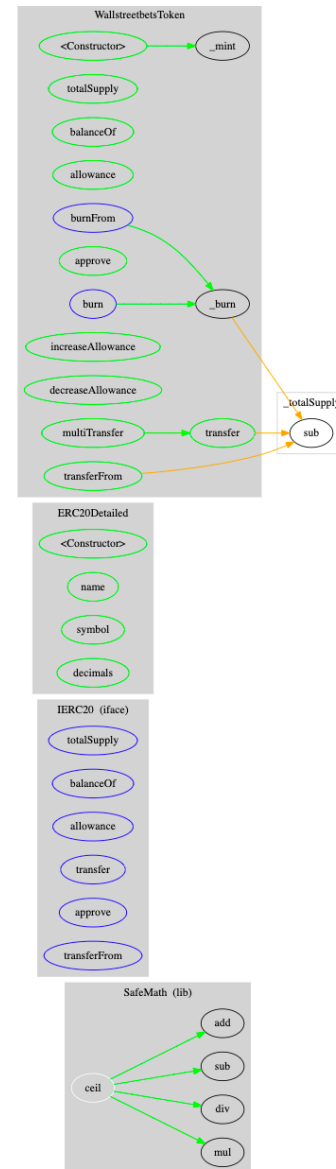
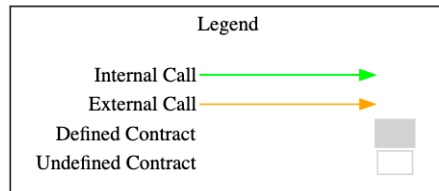


## 4.3 Tested Contract Files

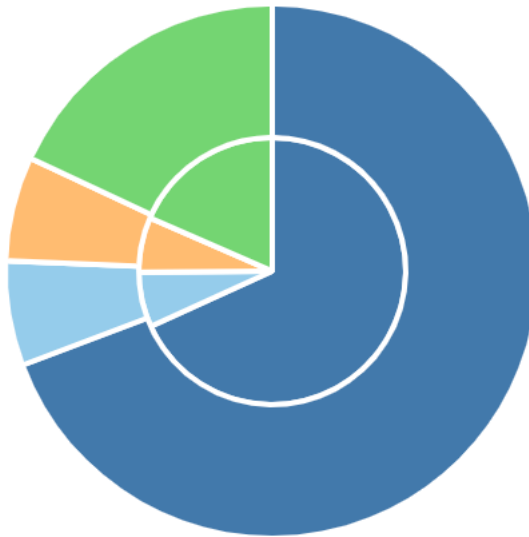
The following are the MD5 hashes of the reviewed files. A file with a different MD5 hash has been modified, intentionally or otherwise, after the security review. You are cautioned that a different MD5 hash could be (but is not necessarily) an indication of a changed condition or potential vulnerability that was not within the scope of the review

File	Fingerprint (MD5)
wsb_token.sol	f7edf216c8c1c961b87c1962c680af8c











## 4.4 Metrics / CallGraph







## 4.5 Metrics / Source Lines



## 4.6 Metrics / Capabilities

<b>Solidity Versions observed</b>	 <b>Experimental Features</b>	 <b>Can Receive Funds</b>	 <b>Uses Assembly</b>	 <b>Has Destroyable Contracts</b>	
<div>^0.5.0</div>		<div>yes</div>	**** (0 asm blocks)		
 <b>Transfers ETH</b>	 <b>Low-Level Calls</b>	 <b>DelegateCall</b>	 <b>Uses Hash Functions</b>	 <b>ECRecover</b>	 <b>New/Create/Create2</b>

## 4.7 Metrics / Source Unites in Scope

Type	File	Logic Contracts	Interfaces	Lines	nLines	nSLOC	Comment Lines	Complex. Score	Capabilities
	wsb_token.sol	3	1	192	182	133	13	121	
	<b>Totals</b>	<b>3</b>	<b>1</b>	<b>192</b>	<b>182</b>	<b>133</b>	<b>13</b>	<b>121</b>	

Legend:

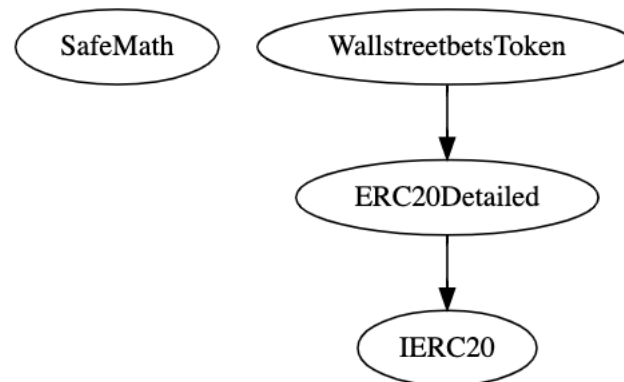
- **Lines**: total lines of the source unit
- **nLines**: normalized lines of the source unit (e.g. normalizes functions spanning multiple lines)
- **nSLOC**: normalized source lines of code (only source-code lines; no comments, no blank lines)
- **Comment Lines**: lines containing single or block comments
- **Complexity Score**: a custom complexity score derived from code statements that are known to introduce code complexity (branches, loops, calls, external interfaces, ...)

## 5. Scope of Work

The WSB team provided us with the files that needs to be tested. The scope of the audit is the ERC20 Token contract (WSB)

Verify claims:

1. ERC-20 Token implementation
2. Deployer cannot mint any new tokens.
3. Deployer cannot burn or lock user funds
4. Deployer cannot pause the contract



## 5.1 Manual and Automated Vulnerability Test

### **CRITICAL ISSUES**

During the audit, Chainsulting's experts found **no Critical issues** in the code of the smart contract.

### **HIGH ISSUES**

During the audit, Chainsulting's experts found **no High issues** in the code of the smart contract.

### **MEDIUM ISSUES**

During the audit, Chainsulting's experts found **no Medium issues** in the code of the smart contract.

### **LOW ISSUES**

During the audit, Chainsulting's experts found **no Low issues** in the code of the smart contract.

## 5.2 Verify Claims

### 5.2.1 ERC-20 Token implementation

**Status:** tested and verified

### 5.2.2 Deployer cannot mint any new tokens.

**Status:** tested and verified

**Code:** Ln 169 – 173 wsb\_token.sol

### 5.2.3 Deployer cannot burn or lock user funds

**Status:** tested and verified

**Code:** Ln 187 – 191 wsb\_token.sol

 To:

Contract [0x268b15e9aed9900489c20c748636b07ce4a436a6](#)  

 Warning! Error encountered during contract execution [**Reverted**] 

### 5.2.4 Deployer cannot pause the contract

**Status:** tested and verified

## 6. Executive Summary

Two (2) independent Chainsulting experts performed an unbiased and isolated audit of the smart contract codebase. The final debrief took place on the February 25, 2021. The overall code quality of the project is very good, not overloaded with unnecessary functions, these is greatly benefiting the security of the contract. It correctly implemented widely-used and reviewed contracts from OpenZeppelin and for safe mathematical operations.

The main goal of the audit was to verify the claims regarding the security of the smart contract and the claims inside the scope of work. During the audit, no issues were found after the manual and automated security testing.





## 7. Deployed Smart Contract

PENDING

WSB Token  
0x9F....

