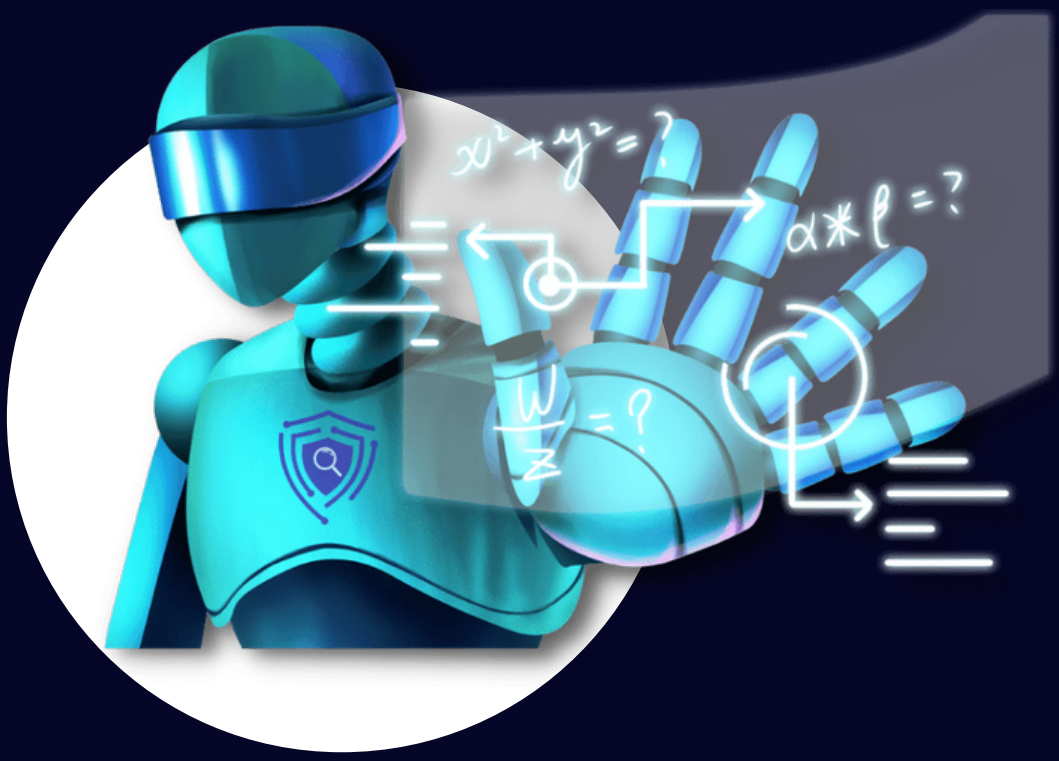
SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



PROJECT: GOALW

DATE: July 16, 2022

GOALW
KICK TO EARN

www.safuaudit.com

INTRODUCTION

Client	GoalW (GLW)
Language	Solidity
Contract address	0xE5ef4A12D42B84c965c65B782230f299f165D640
Owner	0x074E7585A44860ea31789A2a744a0DE267281fc7
Deployer	0x074E7585A44860ea31789A2a744a0DE267281fc7
SHA1-Hash	9b7af77e371b49ca449352f87d8832a052eaeb46
Decimals	9
Supply	100,000,000
Platform	Binance Smart Chain
Compiler	v0.8.7+commit.e28d00a7
Optimization	Yes with 200 runs

TABLE OF CONTENTS

APPROACH

Audit Details

Our comprehensive audit report provides a full overview of the audited system's architecture, smart contract codebase, and details on any vulnerabilities found within the system.

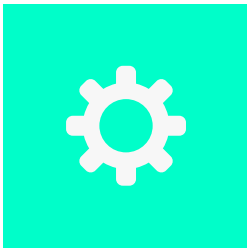
Audit Goals

The audit goal is to ensure that the project is built to protect investors and users, preventing potentially catastrophic vulnerabilities after launch, that lead to scams and rugpulls.

Code Quality

Our analysis includes both automatic tests and manual code analysis for the following aspects:

- Exploits
- Back-doors
- Vulnerability
- Accuracy
- Readability



Tools

- Remix IDE
- Mythril
- Open Zeppelin Code Analyzer
- Solidity Code Complier
- Hardhat

RISK CLASSIFICATION

CRITICAL

Issues on this level are critical to the smart contract's performance/functionality and should be fixed before moving to a live environment.

MEDIUM

Issues on this level could potentially bring problems and should eventually be fixed.

MINOR

Issues on this level are minor details and warning that can remain unfixed but would be better fixed at some point in the future

INFORMATIONAL

Information level is to offer suggestions for improvement of efficacity or security for features with a risk free factor.

OVERVIEW

Fees

- Buy Fees: 3%
- Sell Fees: 5%

Fees privileges

- Can't set buy fees over 10% & sell fees over 10%

Ownership

- Owned

Minting

- No mint function

Max Tx Amount

- Can't set max Tx amount

Pause function

- Can't pause trading

Blacklist

- Can't blacklist

Other privileges

- Can exclude from fees

CONTRACT INSPECTION 🔎

Imported contracts or frameworks used:

SafeMath	Library		|||
Context	Implementation		|||
Ownable	Implementation	Context	|||
IBEP20	Interface		|||

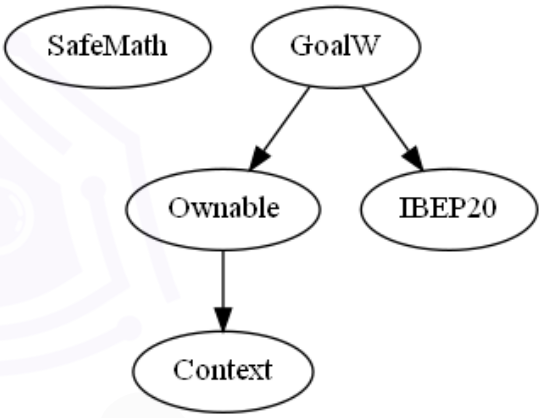
Tested Contract File:

File Name	SHA-1 Hash
GoalW.sol	9b7af77e371b49ca449352f87d8832a052eaeb46

GoalW	Implementation	IBEP20, Ownable	|||
└	<Receive Ether>	External ❗	💵	NO❗
└	<Constructor>	Public ❗	🛑	Ownable
└	totalSupply	External ❗		NO❗
└	decimals	External ❗		NO❗
└	symbol	External ❗		NO❗
└	name	External ❗		NO❗
└	getOwner	External ❗		NO❗
└	balanceOf	Public ❗		NO❗
└	allowance	External ❗		NO❗
└	approve	Public ❗	🛑	NO❗
└	transfer	External ❗	🛑	NO❗
└	transferFrom	External ❗	🛑	NO❗
└	_transferFrom	Internal 🔒	🛑	
└	_basicTransfer	Internal 🔒	🛑	
└	setExcludeTax	External ❗	🛑	onlyOwner
└	setTax	External ❗	🛑	onlyOwner
└	takeTax	Internal 🔒	🛑	
└	transferTax	Internal 🔒	🛑	
└	setTaxers	External ❗	🛑	onlyOwner
└	setPair	External ❗	🛑	onlyOwner
└	setTaxThreshold	External ❗	🛑	onlyOwner
└	getCirculatingSupply	Public ❗		NO❗

Symbol	Meaning
🛑	Function can modify state
💵	Function is payable
🔐	Private function
🔒	Internal function
NO❗	Function has no modifier

INHERITANCE TREE



Inheritance is a feature of the object-oriented programming language. It is a way of extending the functionality of a program, used to separate the code, reduces the dependency, and increases the re-usability of the existing code. Solidity supports inheritance between smart contracts, where multiple contracts can be inherited into a single contract.

MANUAL FUNCTIONS ANALYSIS

The contract is verified to check if functions do and work as they should and malicious code is not inserted.

	Tested	Result
Transfer	Yes	Passed
Total Supply	Yes	Passed
Buy Back	Yes	N/A
Burn	Yes	N/A
Mint	Yes	N/A
Rebase	Yes	N/A
Pause	Yes	N/A
Blacklist	Yes	N/A
Lock	Yes	N/A
Max Transaction	Yes	N/A
Transfer Ownership	Yes	Passed
Renounce Ownership	Yes	Passed

VULNERABILITIES TEST

ID	Description	
V-01	Function Default Visibility	Passed
V-02	Integer Overflow and Underflow	Passed
V-03	Outdated Compiler Version	Passed
V-04	Floating Pragma	Minor
V-05	Unchecked Call Return Value	Passed
V-06	Unprotected Ether Withdrawal	Passed
V-07	Unprotected SELF-DESTRUCT Instruction	Passed
V-08	Re-entrancy	Passed
V-09	State Variable Default Visibility	Minor
V-10	Uninitialized Storage Pointer	Passed
V-11	Assert Violation	Passed
V-12	Use of Deprecated Solidity Functions	Passed
V-13	Delegate Call to Untrusted Callee	Passed
V-14	DoS with Failed Call	Passed
V-15	Transaction Order Dependence	Passed
V-16	Authorization through tx.origin	Passed
V-17	Block values as a proxy for time	Passed

V-18	Signature Malleability	**Passed**
V-19	Incorrect Constructor Name	**Passed**
V-20	Shadowing State Variables	**Passed**
V-21	Weak Sources of Randomness from Chain Attributes	**Passed**
V-22	Missing Protection against Signature Replay Attacks	**Passed**
V-23	Lack of Proper Signature Verification	**Passed**
V-24	Requirement Violation	**Passed**
V-25	Write to Arbitrary Storage Location	**Passed**
V-26	Incorrect Inheritance Order	**Passed**
V-27	Insufficient Gas Griefing	**Passed**
V-28	Arbitrary Jump with Function Type Variable	**Passed**
V-29	DoS With Block Gas Limit	**Passed**
V-30	Typographical Error	**Passed**
V-31	Right-To-Left-Override control character (U+202E)	**Passed**
V-32	Presence of unused variables	**Passed**
V-33	Unexpected Ether balance	**Passed**
V-34	Hash Collisions With Multiple Variable Length Arguments	**Passed**
V-35	Message call with the hardcoded gas amount	**Passed**
V-36	Code With No Effects (Irrelevant/Dead Code)	**Passed**
V-37	Unencrypted Private Data On-Chain	**Passed**

FINDINGS

ID	Category	Issue	Severity
CE-OF	Centralization	Owner Accessible Functions	Minor
V-01	Vulnerabilities	Unlocked Compiler	Minor
V-02	Vulnerabilities	State Variable Default Visibility	Minor
GO-01	Gas Optimization	State Variables that could be declared constant	Informational

CE-OF: Owner Accessible Functions

Description

The owner has the permission through onlyOwner modifier to the following:

1. renounceOwnership()
2. transferOwnership()
3. setExcludeTax()
4. setTax()
5. setTaxers()
6. setPair()
7. setTaxThreshold()

The role OnlyOwner has authority over the above functions that can manipulate the project functionality. Any compromise to the owner account may allow a hacker to take advantage of this authority.

Recommendation

- We advise the client to carefully manage the privilege accounts' private key to avoid any potential risks of being hacked.
- Renounce Ownership at some point in time.

V-01: Unlocked Compiler

Line #235, #260, #335, #356

```
pragma solidity ^0.8.0;
```

Description

Contracts should be deployed with the same compiler version and flags that they have been tested with thoroughly. Locking the pragma helps to ensure that contracts do not accidentally get deployed using, for example, an outdated compiler version that might introduce bugs that affect the contract system negatively.

Recommendation

- Lock the pragma version and also consider known bugs (https://github.com/ethereum/solidity/releases) for the compiler version that is chosen.

V-02: State Variable Default Visibility

Line #369, #370, #371

```
uint256 _totalSupply = 100 * (10**6) * (10 ** _decimals);
mapping (address => uint256) _balances;
mapping (address => mapping (address => uint256)) _allowances;
```

Description

Labeling the visibility explicitly makes it easier to catch incorrect assumptions about who can access the variable.

Recommendation

Variables can be specified as being public, internal or private. Explicitly define visibility for all state variables.

GO-01: State Variables could be declared constant

Line #369

```
uint256 _totalSupply = 100 * (10**6) * (10 ** _decimals);
```

Description

_totalSupply should be declared constant. This is especially important for **taxThreshold** pre-construction variable that is set as **_totalSupply /5000**

Recommendation

- Add the constant attributes to state variables that never change to also save gas.

GOOD PRACTICES

- The owner cannot mint new tokens after deployment
- The owner cannot set sell taxes above 10% and buy taxes above 10%
- The owner cannot stop or pause the contract
- The owner cannot set a transaction limit
- The smart contract utilizes "SafeMath" to prevent overflows

DISCLAIMER

SafuAudit.com is not a financial institution and the information provided on this website does not constitute investment advice, financial advice, trading advice, or any other sort of advice. You should not treat any of the website's content as such. Investing in crypto assets carries a high level of risk and does not hold guarantees for not sustaining financial loss due to their volatility.

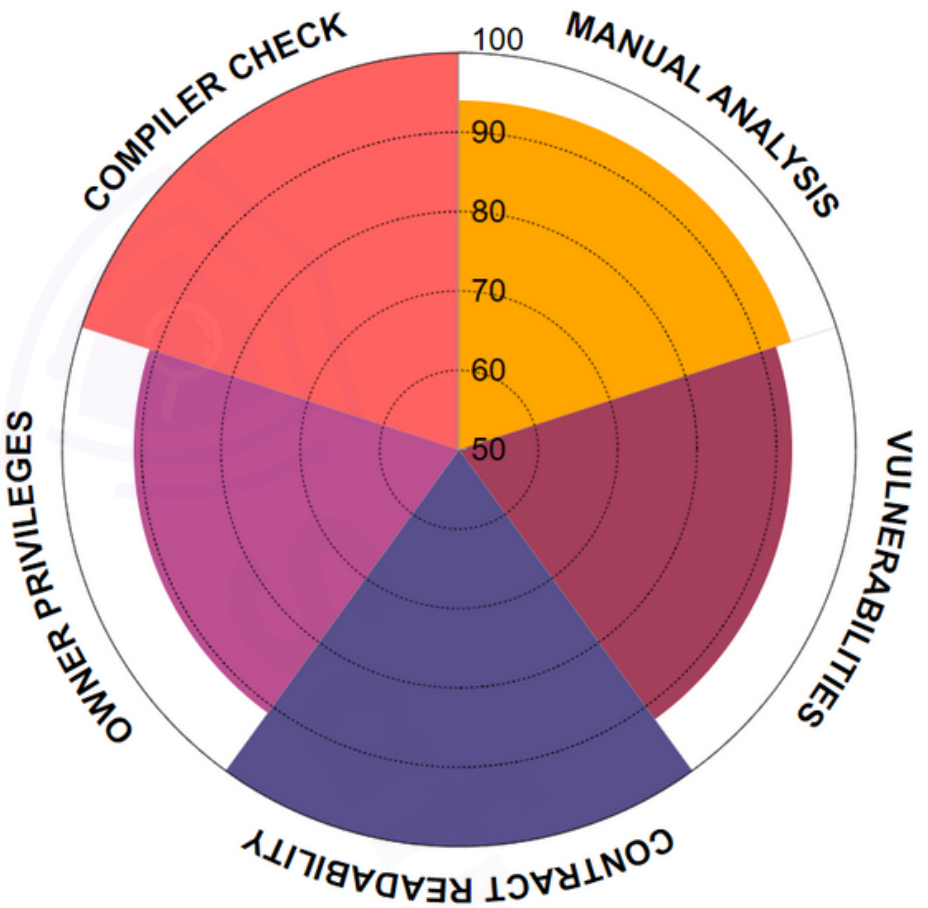
Accuracy of Information

SafuAudit will strive to ensure the accuracy of the information listed on this website although it will not hold any responsibility for any missing or wrong information. SafuAudit provides all information as is. You understand that you are using any and all information available here at your own risk. Any use or reliance on our content and services is solely at your own risk and discretion.

The purpose of the audit is to analyze the on-chain smart contract source code and to provide a basic overview of the project.

While we have used all the information available to us for this straightforward investigation, you should not rely on this report only — we recommend proceeding with several independent audits Be aware that smart contracts deployed on a blockchain aren't secured enough against external vulnerability or a hack. Be aware that active smart contract owner privileges constitute an elevated impact on the smart contract safety and security. Therefore, SafuAudit does not guarantee the explicit security of the audited smart contract. 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.

RATING



Manual Analysis

Vulnerabilities

Contract Readability

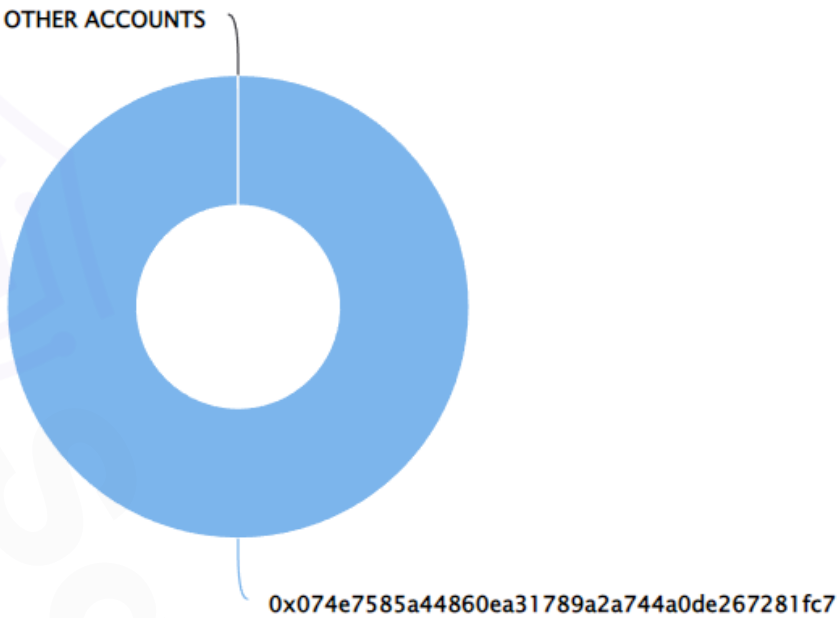
Owner Privileges

Compiler Check

Final Score: **95.4**

SUMMARY

Top 10 holders



Rank	Address	Quantity (Token)	Percentage
1	0x074e7585a44860ea31789a2a744a0de267281fc7	100,000,000	100.0000%

CONCLUSION

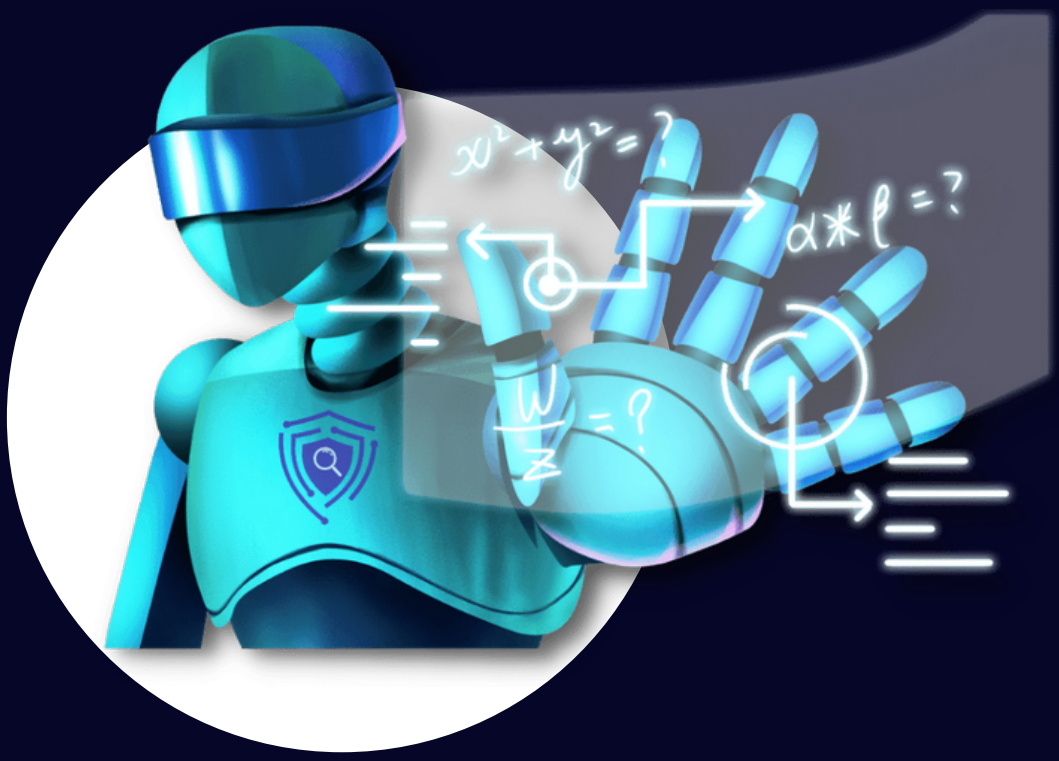
Project GoalW (GLW) does not contain any severe issues or risk characteristics.

SafuAudit has tested the security based on manual and automated tests. Please note that we don't offer any warranties for the business model.

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY

SAFUAUDIT

SMART CONTRACT AUDITS AND BLOCKCHAIN SECURITY



"Only in growth, reform, and change, paradoxically enough, is true security to be found."

www.safuaudit.com