VIBRANIUM AUDITS

Security Assessment

Gog and Magog

Vibranium Audits Verified on Oct 30th, 2024

Summary

Finding

Disclimar

Vibranium audits verified on Oct 30th 2024

Gog and Magog

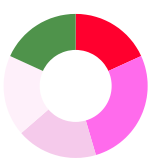
The security assesment was prepared by Vibranium audits , the leader in web3 security

Executive summary

TYPES	ECOSYSTEM	METHODS
DEFI	EVM	Formal Verification, Manual review

LANGUAGE	Codebase
Solidity	GogAndMagog.sol

Vulnerability Summay



4	0	1	3	0	0
Total Findings	Critical	High	Medium	Minor	Informational

High	Aknowledged	Permanent transfer restrictions without a way to disable or modify them can disrupt token utility and adaptability.
Medium	Aknowledged	Relying on block-based timing can unpredictably block legitimate trades due to block time fluctuations.
Medium	Aknowledged	Cumulative transfer limits within 20 minutes of activation can be restrictive and lack a mechanism to pause or stop in emergencies.
Medium	Aknowledged	Direct use of msg.sender may allow bypassing of restrictions in contract-to-contract interactions, limiting restriction effectiveness.

Codebase | Gog and Magog

Code

GogAndMagog.sol

Approach & Methods | Gog and Magog

This report has been prepared for Gog and Magog to discover issues and vulnerabilities in the source code of the project, including smart contracts and the token minting mechanism. A comprehensive examination has been conducted, utilizing Static Analysis and Manual Review techniques to ensure compliance with best security practices.

The auditing process pays special attention to the following considerations:

Testing the smart contracts against common and uncommon attack vectors such as reentrancy attacks, overflow vulnerabilities, and unauthorized access.

Assessing the codebase for compliance with industry best practices, ensuring code security, scalability, and maintainability.

Ensuring contract logic meets the specifications and requirements of the Gog and Magog project, specifically for token minting, access control, and transaction handling.

Cross-referencing contract structure and functionality against similar projects in the blockchain ecosystem to ensure adherence to standard practices.

Performing a line-by-line manual review of the entire codebase to identify any hidden or obscure vulnerabilities that automated tools may miss.

The security assessment identified vulnerabilities across a range of severity levels, from Critical to Informational. We recommend addressing these findings promptly to enhance the overall security of the Gog and Magog project. Key recommendations include:

Testing smart contracts against both common and uncommon attack vectors, ensuring robustness against real-world threats.

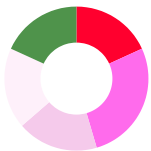
Implementing secure coding practices to enhance the quality and security of the codebase.

Expanding unit tests to cover all possible use cases and edge cases, especially for the token minting functions.

Improving code documentation and comments for better readability, particularly for critical functions involving token minting and transfers.

Ensuring transparency in privileged activities, such as the minting process, and implementing additional safeguards for authorization.

Gog and Magog | Finding



4	0	1	3	0	0
Total Findings	Critical	High	Medium	Minor	Informational

This report provides a detailed audit of the Gog and Magog project's codebase, focusing on the token contract and its associated trading restriction mechanisms. The objective of this audit is to uncover potential security vulnerabilities, optimize code efficiency, and identify areas for improvement to strengthen the project's security and reliability.
In total, five vulnerabilities were identified, categorized by severity as follows: 1 High and 3 Medium.

ID	Title	Category	Severity	Status
Go-M1	Permanent Transfer Restrictions	Liquidity Activation Logic	High	Acknowledged
Go-M2	Dead Block Timing Dependency	Timing Mechanism	Medium	Acknowledged
Go-M3	Restrictive Transfer Limits	Timing Mechanism	Medium	Acknowledged
Go-M4	Direct Use of msg.sender	Access Control	Medium	Acknowledged

Go-M1 | Permanent Transfer Restrictions

Category	Severity	Location	Status
Permanent Transfer Restrictions	High	Gog and Magogfunction activateSniperProtection	Acknowledged

Description

The Gog and Magogcontract's **activateSniperProtection** function activates a mechanism to restrict trading within a specified period after liquidity is added. Once this protection is activated, there is no method to disable it, making the restriction permanent. As a result, this restricts trading flexibility by continuously enforcing the 1% transfer limit and dead block restriction without allowing the contract owner to adapt or lift these restrictions based on changing market or project conditions.

Migitation

Add a Deactivation Function: Introduce a deactivateSniperProtection function that allows the owner to disable the sniper protection, setting isProtectionActivated to false. This would allow for the removal of trading restrictions once they are no longer necessary.

Go-M2 | Dead Block Timing Dependency

Category	Severity	Location	Status
Timing Mechanism	Medium	GogAndMagog activateSniperProtection DEAD_BLOCKS	Acknowledged

Description

The Gog and Magogcontract restricts trading during a specific number of "dead blocks" immediately after liquidity is added. This restriction is intended to prevent frontrunning and early bot activity. However, relying on block-based timing can be unpredictable because block intervals are not consistent. This can lead to legitimate trades being unexpectedly blocked, especially if block production rates change. Consequently, token holders may face unnecessary delays or restrictions on trading due to these timing dependencies.

Mitigation

- Switch to Time-Based Restrictions: Instead of relying solely on block numbers, implement time-based restrictions. Using timestamps (e.g., blocking trades for a set number of minutes) provides more reliable and predictable restrictions, independent of block production speed.
- Hybrid Block-Time Approach: Implement a hybrid solution that includes both block and time checks. For example, if a specific block timing is required, combine it with a minimum timestamp threshold to ensure fair timing regardless of block production speed.

Go-M3 | Restrictive Transfer Limits

Category	Severity	Location	Status
Timing Mechanism	Medium	GogAndMagog Function: _isTradingAllowed	Acknowledged

Description

The Gog and Magogcontract enforces a cumulative transfer limit during the first 20 minutes after liquidity is added, capping transfers to 1% of the total supply. While this restriction is intended to mitigate early, potentially harmful trades (e.g., by bots), it could unintentionally restrict legitimate user transactions. Additionally, there is no emergency mechanism to temporarily pause or adjust the restriction, which could become problematic if market conditions change or unforeseen issues arise.

Migitation

- Introduce a Transfer Limit Adjustment Mechanism: Implement a function allowing the contract owner to adjust or disable the transfer limit during the restriction period. This adjustment can be based on demand or unforeseen issues, providing flexibility for the project owner.
- Emergency Pause (Circuit Breaker): Add an emergency pause feature, also known as a "circuit breaker," which allows the contract owner to halt all transfers temporarily. This would provide a safeguard to prevent unexpected transfer restrictions or abuse without affecting the contract permanently.

Go-M4 | Direct Use of msg.sender

Category	Severity	Location	Status
Access Control	Medium	GogAndMagog _isTradingAllowed	Acknowledged

Description

In the Gog and Magogcontract, the _isTradingAllowed function relies on msg.sender to enforce restrictions like dead blocks and cumulative transfer limits. However, in ERC20 contracts, the transferFrom function enables an approved address (often a contract) to transfer tokens on behalf of a user. In this case, msg.sender would refer to the intermediary contract or approved spender rather than the original token owner. As a result, this logic can be bypassed, allowing certain approved contracts or addresses to circumvent the intended restrictions, reducing the effectiveness of anti-bot protections.

Migitation

Replacing msg.sender with tx.origin in the restriction logic, we can enforce limits based on the true originator of the transaction, even if they use an intermediary contract. In this case, the _isTradingAllowed function would apply dead block restrictions and cumulative transfer limits to the external account that initiated the transaction, rather than allowing an approved contract to bypass these restrictions.

DISCLAIMER | VIBRANIUM AUDITS

This report provides a security assessment of the GogAndMagog smart contract, aiming to identify potential vulnerabilities, coding inefficiencies, and areas for improvement. While every effort has been made to identify possible risks and recommend mitigations, this report does not guarantee the complete safety of the contract, nor does it imply that all possible vulnerabilities have been identified. This audit was conducted based on the code and information available at the time of the review. The GogAndMagog project is advised to implement any recommended changes and conduct further testing to validate the security and functionality of the contract under various conditions.

This report and its contents do not constitute financial or investment advice. Users, investors, and other stakeholders engaging with GogAndMagog's smart contract do so at their own risk and discretion.

Vibranium Audits Securing the Web3 World

Vibranium Audits is a blockchain security company that was founded in 2021 by professors from the University of Greenwich and cyber-security engineers from ITI Capital. As pioneers in the field, Vibranium Audits utilizes best-in-class Formal Verification and AI technology to secure and monitor blockchains, smart contracts, and Web3 apps.